Automatic Induction of Lexical Features

Inauguraldissertation

zur Erlangung des Akademischen Grades

eines Dr. phil.,

vorgelegt dem Fachbereich 05

Philosophie und Philologie

der Johannes Gutenberg-Universität

Mainz

von

Peter Jäger

aus Wiesbaden

Peter Jäger

2011

Referent:	Professor Dr. Walter Bisang
Koreferent:	Professor Dr. Ulrich Heid

Tag der mündlichen Prüfung: 04.10.2010

Abstract

This thesis concerns artificially intelligent natural language processing systems that are capable of learning the properties of lexical items (properties like verbal valency or inflectional class membership) autonomously *while* they are fulfilling their tasks for which they have been deployed in the first place. Many of these tasks require a deep *analysis* of language input, which can be characterized as a mapping of utterances in a given input C to a set S of linguistically motivated structures with the help of linguistic information encoded in a grammar G and a lexicon L:

$$G \ + \ L \ + \ C \ \rightarrow \ S \tag{1}$$

The idea that underlies intelligent lexical acquisition systems is to modify this schematic formula in such a way that the system is able to exploit the information encoded in S to create a new, *improved* version of the lexicon:

$$G \ + \ L \ + \ S \ \rightarrow \ L' \tag{2}$$

Moreover, the thesis claims that a system can only be considered intelligent if it does not just make maximum usage of the *learning opportunities* in C, but if it is also able to *revise* falsely acquired lexical knowledge. So, one of the central elements in this work is the formulation of a couple of criteria for intelligent lexical acquisition systems subsumed under one paradigm: the Learn-α design rule.

The thesis describes the design and quality of a prototype for such a system, whose acquisition components have been developed from scratch and built on top of one of the state-of-the-art Head-driven Phrase Structure Grammar (HPSG) processing systems. The quality of this prototype is investigated in a series of experiments, in which the system is fed with extracts of a large English corpus.

While the idea of using machine-readable language input to automatically acquire lexical knowledge is not new, we are not aware of a system that fulfills Learn-α *and* is able to deal with large corpora. To instance four major challenges of constructing such a system, it should be mentioned that a) the high number of possible structural descriptions caused by highly underspecified lexical entries demands for a parser with a very effective ambiguity management system, b) the automatic construction of concise lexical entries out of a bulk of observed lexical facts requires a special technique of data alignment, c) the reliability of these entries depends on the system's decision on whether

it has seen 'enough' input and d) general properties of language might render some lexical features indeterminable if the system tries to acquire them with a too high precision.
The cornerstone of this dissertation is the motivation and development of a general theory of automatic lexical acquisition that is applicable to every language and independent of any particular theory of grammar or lexicon.
This work is divided into five chapters. The introductory chapter first contrasts three different and mutually incompatible approaches to (artificial) lexical acquisition: cue-based queries, head-lexicalized probabilistic context free grammars and learning by unification. Then the postulation of the Learn-α design rule is presented. The second chapter outlines the theory that underlies Learn-α and exposes all the related notions and concepts required for a proper understanding of artificial lexical acquisition. Chapter 3 develops the prototyped acquisition method, called ANALYZE-LEARN-REDUCE, a framework which implements Learn-α. The fourth chapter presents the design and results of a bootstrapping experiment conducted on this prototype: lexeme detection, learning of verbal valency, categorization into nominal count/mass classes, selection of prepositions and sentential complements, among others. The thesis concludes with a review of the conclusions and motivation for further improvements as well as proposals for future research on the *automatic induction of lexical features*.

Acknowledgements

I wish to thank everyone who contributed to this thesis.

First of all, I would like to thank my supervisor, Prof. Dr. Walter Bisang, for his continuous support, advice, encouragement and the numerous inspiring discussions.

My gratitude extends to my second reviewer, Prof. Dr. Ulrich Heid, for his invaluable suggestions and the opportunity to present my project at the Institut für Maschinelle Sprachverarbeitung in Stuttgart.

I also thank the other members of my thesis committee for their constructive feedback.

Many thanks go out to my IT business colleagues for showing respect and consideration for this project.

A big thank you goes to the open source community as their products have contributed to the prototype presented within the thesis.

On a personal note I heartily thank my family for their patience, understanding, and support. Without the tireless backup, encouragement and advice of my beloved wife this thesis would never have been accomplished.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The first chapter of this thesis consists of five parts. The first section deals with the problem of manually constructing knowledge resources for natural language processing (NLP) systems and with principle solutions to overcome these problems. Different lines of research in this area will be characterized and the contribution of this thesis will be pointed out in section 1.2. Due to the broadness of the field, this survey can only be superficial and by no means exhaustive. However, it might be useful to work out a couple of basic questions and dimensions which also have to be dealt with by the narrower problem of automatic acquisition of *lexical* knowledge. Even the different solutions provided in this research topic cannot be surveyed in full depth. Rather, section 1.3 contrasts three different strategies of computer-aided lexicon acquisition: cue-based corpus extraction, head-lexicalized probabilistic context free grammars and the unification-based approach. Arguments for the shortcomings of the former two will be presented and unresolved issues of the latter strategy will be put forward, to a great extent forming the motivation of this research. Section 1.4 argues for a radical position by formulating the design rule which is underlying this work: the Learn-α Design Rule.
At the end of this chapter a series of examples, retrieved from the British National Corpus (BNC) (Leech 1992), illustrate the acquisition of lexical knowledge by means of feature unification, the starting point for the second chapter which presents the *theory* of Learn-α.

1.1 The Language Engineering Bottleneck

It has been pointed out by many researchers of the NLP community that the manual construction of language resources (such as grammar and lexicon) is nearly impracticable at least if a broad-range NLP system is in scope that

has to deal with real, unrestricted language. This finding has evoked much effort in the area of automatic (or at least computer-aided) provision of linguistic knowledge resources. Going a little deeper into that matter, it can be stated that hand-crafted resources of this kind tend to be expensive, defective, incomplete, inconsistent and relative. This is not only because of the mere complexity and diversity of knowledge that has to be provided (phonological, morphological, syntactic and semantic rules and the entries for ten or more thousands of lexical items), but is also due to the inherent openness of at least the lexical component, which means that every robust NLP system has to be enabled to react properly when it faces unknown words in the input. While the fact that highly trained people have to formulate appropriate grammatical rules or lexical entries makes it obvious that the resources are *expensive* to build, some more details for the other problematic properties of manually constructed NLP system resources are in order. First, the resource is *defective* in the sense that there is no guarantee that all the relevant properties can be found by introspection of the lexicographer(s) or grammar writer(s). The experience of building the COMLEX dictionary reported in Grishman et al. (1994) is a good example although the authors provide a method to minimize the rate of error. A closer look at one of the current state-of-the-art broad-range computational grammars, the English Resource Grammar (ERG) (Flickinger 2000) reveals that many rules (e.g. those responsible for appositives) are far from being stable in the sense that they either strongly under- or overgenerate.
The *incompleteness* of hand-crafted resources has been pointed out in many publications, e.g. Heid and Kermes (2002)[1]. A resource like a lexicon might be incomplete with respect to missing entries or with regard to a lexical property which has not been in focus during development but might be recognized as an important lack when the resource is re-used for another NLP task or when the grammatical component is revised. One well-known example is the lack of frequency information in dictionaries. Without this information a probabilistic parser has to treat each lexical entry as equiprobable - with a serious negative impact on the capability of finding an appropriate ranking of alternative parses. Highly connected with incompleteness is the problem of *inconsistency*: In practice it is sometimes difficult to stick to the initially developed criteria for the assignment of words to certain classes, especially

[1]

> 'For the more frequent syntactic constructions, data are easily accessible; idiomatic constructions often readily spring to the lexicographer's mind when the headword is being analyzed. But there are facts that are less in sight and yet should be noted in a large dictionary.'

in cases where little agreement among linguists with respect to certain properties can be observed. Argumenthood of prepositional phrases is a typical example of the difficulties of creating a valency or subcategorization frame dictionary.
Finally, it is worth emphasizing the *relative* nature of an NLP resource which often decreases its reusability: The knowledge resource is relative with regard to

1. the other components of the NLP system, especially the content and the structure of grammar
2. the linguistic background theory of the researcher / the engineer
3. the application of the system
4. the language(s) in scope
5. the acquisition procedure.

This is why the *language engineer* (henceforth LE) who is in charge of developing an NLP system or of adapting an existing one to a new domain faces a hurdle that tends to constrain the whole development process: the *language engineering bottleneck*, which can be seen as an instance of the so-called knowledge engineering bottleneck, one of the main problems for the realization of intelligent systems[2].

If the focus is on building or adapting a lexical resource for an NLP system (as in this thesis), we refer to this problem by the notion *lexical acquisition bottleneck*, which in turn is an instance of the language engineering bottleneck. The relativity of the lexicon is an instance of the relative nature of an NLP resource and it applies to content and form[3].
That this work is dedicated to the development and implementation of the *lexical* component does not mean that the complexity and costs of the implementation and maintenance of the *grammar* component should be underestimated. In fact, both the lexical and grammatical component have been

[2]The bottleneck of a project always determines its overall duration. Of course bottlenecks can be resolved by employing many human resources working in parallel, but this is not what the knowledge engineering bottleneck is about. Here we may think of one engineer doing the whole process: design, development and testing of a) the framework and b) the knowledge resources. Usually - especially if central parts of the framework are already available (like the programming language, for instance) - it is b) that determines the duration of the whole project. This may certainly depend on the domain that has to be modeled and the quality that is to be achieved, but if the domain is broad and the ambition is high then b) is very likely to be the bottleneck.

[3]The relativity is underlined by the observation that '[t]here is virtually no consensus on the representation of the lexicon.' (Matsumoto and Utsuro 2000: 563)

the target of research on computer-aided acquisition of linguistic knowledge, which has become an important, though quite heterogeneous field in computational linguistics. Some researchers from this field witness a 'dramatic shift' (Cardie and Mooney 1999) in the last but one decade from manually constructing language resources to 'partial or totally automating this process', a shift which may be traced back to the early success of statistical approaches to *speech recognition*, which fertilized approaches on the acquisition of other aspects of language. This shift goes along with 'a resurgence in research on empirical methods in natural language processing' (Brill and Mooney 1997) and culminates in the call for systems with self-organizing dictionaries, recently formulated by Briscoe (2001) in the context of valency acquisition: 'A better model of a valency lexicon [...] is of an adaptive self-organizing knowledge base which continually monitors data to update associations and the strengths of associations between predicates and valency frames.' In this thesis it will be argued that this is not only to be demanded for valency, but for the whole lexical component, in line with what could be called *in vivo* general-purpose deep lexical acquisition coined in Baldwin (2005c) and Baldwin (2007)[4].

The field has grown too large to be surveyed in an introductory chapter, but at least some important dimensions can be worked out, along which the different subfields may be characterized. Very broadly, Wermter et al. (1996) distinguish three basic learning methods: connectionist, statistical and symbolic approaches (and hybrids that combine methods across these classes). Connectionist approaches make use of neural networks, statistical approaches try to acquire linguistic knowledge by the analysis of large corpora, and symbolic approaches apply techniques from machine learning research. In the context of *supervised learning*, for example, in which the learning system has to learn from labeled examples, one can find all kinds of techniques being applied to the automatic or semi-automatic acquisition of NLP system resources: case-based learning (Cardie 1993), memory-based learning (Daelemans et al. 1999), inductive logic programming / rule induction (Brill 1995, Zelle and Mooney 1997), decision tree learning (Hermjakob 1997), explanation-based learning (Samuelsson and Rayner 1991), genetic programming (Siegel and McKeown 2000) and active learning (Thompson and Mooney 1998). In *unsupervised learning* the system has to learn from unlabeled input, most notably from raw text or speech. Various techniques have been carried over from machine learning research such as minimum de-

[4] *In vivo* lexical acquisition uses the target system itself for learning lexical items, as opposed to *in vitro* accounts in which linguistic properties are learned outside a particular target system using special-customized acquisition procedures tailored to the linguistic property in scope.

scription length (MDL) (De Marcken 1996, Goldsmith 2001), maximum entropy modeling (Berger et al. 1996, Zhang and Kordoni 2005) or expectation-maximation (Rooth et al. 1998).
Going hand in hand with the basic strategy, the underlying *representation* or model of language (neural networks versus linguistically shallow models versus linguistically deep accounts based on formal grammar research) may certainly constrain the availability of methods for language acquisition. The present work will not cover connectionist frameworks at all, so the focus is on the remaining two basic strategies, namely statistical and symbolic approaches. Furthermore, the two aspects of basic strategy and representation might be conflated into one important dimension: *linguistic precision*, a notion which covers the question of how much linguistic knowledge has been employed in advance before the learning system starts acquiring further linguistic knowledge. As an example, systems that extract lexical information from texts by means of regular expressions are linguistically less precise than systems that make use of large, broad-coverage unification-based grammars. Typically, less precise methods are supplemented by statistical devices that are aimed to compensate for errors introduced by the analyzing system components.
Further important dimensions are

underlying motivation: such as foundations of cognitive modeling or practical tasks such as speech recognition, natural language understanding, information retrieval, or learning lexical items for a precision grammar (*deep lexical acquisition*, Baldwin 2005a, Baldwin 2005c, Blunsom and Baldwin 2006, Baldwin 2007)

linguistic phenomenon: the target of the acquisition procedure across all linguistic description levels (phonology, morphology, syntax, semantics, pragmatics)

NLP system component: the component which is in focus, mainly lexicon and grammar

investigated language(s): the language type may have an impact on the acquisition procedure and performance

investigated sub language: different domains can reveal different usages of words In a sense, unsupervised learning can be thought of as finding patterns in the data above and beyond what would be considered pure unstructured noise.

language source: the input for the acquisition system: raw texts, annotated text corpora, speech corpora, mono- versus bilingual corpora, labeled vs. unlabeled training data (supervised vs. unsupervised training)

etc.

degree of autonomy: the question of how much manual intervention is required

degree of embeddedness: an important aspect of the system architecture: whether the acquisition procedure is isolated from the NLP system(s) which are subject to improvement or whether the acquisition device is an embedded feature of an NLP system, see Cardie (1996) for the call for embedded learning components and Baldwin (2007) for the dichotomy of *in vivo* versus *in vitro* acquisition methods

initial knowledge: the question of how much linguistic knowledge is accessible to the system before it starts learning

statistical model: the model's underlying statistical components

method of evaluation: the way of determining the acquisition procedure's quality along with the question of how different systems can be compared.

From the language engineering perspective, computer-aided language acquisition deals with the question of exploiting machine-readable *language sources* of a given *(sub)language* to automatically acquire missing information about certain *linguistic aspects* in *NLP system components* in order to fulfill a certain *application task*.

1.2 Aim and Structure of this Thesis

The list of dimensions presented in the last section is designed to help formulate the contribution of this thesis.
First of all, the work presented here deals with the machine-driven improvement of the lexical component while the grammar is kept constant. Further, as machine-readable text is the only language source in the current implementation, phonological aspects are not covered at all. Additionally, the model of language adopted here can neither cover pragmatic phenomena nor word sense. So the work is restricted to morphological, syntactic and semantic aspects of written texts, where semantic aspects are limited to phenomena of argument linking (valency and control). Although the underlying principles outlined in section 1.4 and more precisely in section 2.10 are thought to cover all lexical features of all languages, there are a couple of practical limitations in the implementation and evaluation of these principles: first, the investigation is limited to the acquisition of *English* lexical features only; second, the selected set of lexical features has a bias towards

the syntax/semantics interface: part-of-speech of unknown words, nominal count/non-count distinction, verb particle combinations and verbal, adjectival and nominal subcategorization frames. Because the grammar is kept constant, learning of lexical features means the acquisition of lexical constraints that restrict the availability of potential structural analyses resulting in improved *disambiguation performance*. Transferring the basic question of computer-aided language acquisition to the scope of this thesis, this leads to the question of how one can exploit *machine-readable unannotated texts* of *English* to *automatically* acquire missing information about *morphological, syntactic and semantic* aspects of the *lexical component* of an NLP system in order to improve its *disambiguation performance*. The approach is hoped to extend to other languages (and probably to other linguistic phenomena) in a similar vein, although it seems reasonable to claim that the limitations explored here vary with the type of input language.
Now the scope is settled, which we believe to be sufficient to investigate the benefits and limits of the idea which underlies the logic of artificial lexical acquisition exposed in this work. This idea is quite simple: If an NLP system is required to perform deep analysis of language input in order to fulfill its task, why not exploit the results for the goal of automatic improvement? **Can the lexicon be a natural fall-out and logical consequence of the system's a-priori knowledge (its grammar and initial lexicon) taken together with its experience (the system's input)?** To outline this idea, consider that the analysis can be characterized as a mapping of utterances in a given input C to a set S of linguistically motivated structures with the help of linguistic information encoded in a grammar G and a lexicon L:

$$G \ + \ L \ + \ C \ \rightarrow \ S \tag{1.1}$$

The idea that underlies intelligent lexical acquisition systems is to modify this simplistic and schematic formula in such a way that the system is able to exploit the information encoded in S to create a new, *improved* version of the lexicon:

$$G \ + \ L \ + \ S \ \rightarrow \ L' \tag{1.2}$$

While this idea is not new (cf. section 1.3.6) this thesis tries to contribute to a deeper understanding of its logic and limits. In our theory we thoroughly describe the perspective of both the learning machine and the language engineer (LE) who designs the machine such that it automatically improves by experience to show better accuracy with future input. The learning procedure is independent of the feature system and grammar that the LE constructs. In a word, learning means filling the underspecified features of

lexemes with values that are consistent with the system's experience. More than that, feature values are learned if their existence is a necessary condition for the analysability of utterances of the experience set, which means that the system automatically detects utterances that unambiguously prove the existence of a feature value for a certain lexeme. The thesis will show that this paradigm of unambiguous 'reference' utterances is limited because of the structure of language which renders a lot of feature values indeterminable.
The thesis presents a prototype of a system that is able to improve automatically on the basis of the above formulas - including the capability to revise 'incorrect' lexical entries. A statistical component is included to facilitate induction and to cope with noise. Further, the system is able to detect indeterminable feature values and to automatically switch from a high precision mode into a fuzzier mode under which indeterminable feature values get visible. To the best of our knowledge, no such system has been developed yet.
This work is organized as follows. The remaining part of this introductory chapter presents three different and incompatible approaches to artificial lexical acquisition and motivates the design rule for intelligent NLP systems, called Learn$-\alpha$, which is set forth in a preliminary version. The second chapter concerns the theory of Learn$-\alpha$ with the aim to make the underpinning logic as explicit as possible. It concludes with the final version of Learn$-\alpha$. In the third chapter, an implementation of Learn$-\alpha$ is presented, called ANALYZE-LEARN-REDUCE. This prototype allows for the conduction of various kinds of experiments the designs and results of which are the topic of the fourth chapter. The final chapter reviews the conclusions and suggests further improvements along with proposals for future research.

A final remark with regard to the *initial state* of the prototype's lexical component is in order. Although it would be very attractive from a theoretical point of view to start the learning cycles with a completely 'empty' lexicon, this would be computationally intractable. Instead, the starting point for the experiments will be the ERG grammar and the lexicon that comes along with it. The first experiment investigates how the system learns words that are not in the lexicon. For all the remaining tests, all lexemes that are in the scope of the acquisition experiment are removed from the lexicon before learning starts.
Having narrowed down the scope of the present work, the next section will provide a closer look on selected extant work on computer-aided lexical acquisition.

1.3 Previous Research on Computer-aided Lexical Acquisition

As different accounts for computer-aided language acquisition in general vary across all the dimensions developed in the last but one section, the same is true for the research on computer-aided *lexical* acquisition. For instance, one line of research aims at the support of lexicographers (Heid and Kermes 2002) whereas others emphasize the simulation of the adult's learning of lexical information (Kilbury et al. 1994, Russel 1993) or try to acquire information from corpora that can supplement NLP systems[5] (Brent 1993[6]). Different motivation has an impact on the result of the various proposals. Tools that are designed to support lexicographers are not expected to be fully automatic. Instead they provide lists of unambiguous observations in the language source which eases the burden of finding all the possible lexical properties of a given word by introspection. This is a very useful feature, typically missing in mere statistical approaches.
It should also be mentioned that the investigated languages only make up a small subset of the world's languages. Research is done mainly on European languages and some East-Asian languages (mainly Chinese and Japanese) which might be a consequence of the availability of machine-readable language sources. Although it might be very interesting, the current stage appears to be still too early to pursue the impact of typological features of a language on the performance of different acquisition strategies. It would not come as a surprise if languages like Chinese and Thai which show relatively spare morpho-syntactic cues at the surface (if at all) would decrease the performance of approaches which heavily rely on such cues.

1.3.1 Lexical Acquisition Tasks

Many different aspects of the lexicon have been the target of machine-driven acquisition and very broadly, they can be classified into two major *lexical acquisition tasks*: *lexeme detection* and *induction of lexical constraints*. While the former deals with finding lexical units that are not covered by the com-

[5]See Carroll and Fang (2004) for a study of the impact of an automatically acquired subcategorization lexicon on the performance of an HPSG parser

[6]'The scenario outlined above [facing a language where no dictionary is given, just a grammar textbook and a very large text corpus] is adopted in this paper as a framework for basic research in computational language acquisition. However, it is also an abstraction of the situation faced by engineers building NLP systems and it is widely agreed that current lexical resources are inadequate' (page 244)

putational lexicon, the latter is about learning constraints that are imposed by lexemes that are already specified (in the lexicon or explicitly in a list of words that are to be learned). If not stated otherwise we use the notion *lexeme* both for single-element as well as multi-element lexemes. Lexeme detection has a considerable overlap with *unknown word handling* of NLP systems, but is not necessarily the same. It can be argued that not all of the 'unknown words' shall be listed in the lexicon. So, for example, named entity recognition (NER) can be seen as a part of the unknown word handling subsystem, but if named entities are kept out of the (general) lexicon, NER does not fall under the notion of lexeme detection. The same might hold for *nonce words* and *neologisms*. The experiments related to lexeme detection, reported in chapter 4, reveal that there is a good portion of missing lexical entries in the ERG system - words that are neither named entities nor highly infrequent words, complementing the work of Zhang and Kordoni (2006). Surprisingly this is not only the case for lexicalized multiword expressions (MWEs) but also for simple words.

Lexeme detection has various aspects. First, some research aims at the semantic classification of unknown words (Ciaramita 2002), highly related to the work on *word sense disambiguation*.

Second, different word forms have to be associated with the same lexical entry (lemmatization), especially challenging in the context of irregular forms (Yarowsky and Wicentowski 2000), extremely rich morphology (Erjavec and Džeroski 2004) or lexical rules (Viegas et al. 1996). In the context of HPSG (Pollard and Sag 1994) research has been conducted on the maximal exploitation of the context of unknown words (Erbach 1990, Kilbury et al. 1992, Kilbury et al. 1994, Barg and Walther 1998) and inflectional type induction (Barg and Kilbury 2000).

Third, the classification of MWEs as being compositional or lexicalized is required as a basis for the decision whether to store a multiword expression in the lexicon or not. Research on MWEs (and collocations) has a long tradition starting with Firth (1957) and has gained much attention of the lexical acquisition research community[7]. A subclass of MWEs are those in which an open-class item[8] forms a non-compositional multiword combination with a closed-class item: verb particle combinations (VPCs) (Blaheta and Johnson 2001, McCarthy et al. 2003, Baldwin and Villavicencio 2002, Sag et al. 2002, Villavicencio and Copestake 2002, Baldwin et al. 2003a, Villavicencio 2003a, Baldwin 2005b, Kim and Baldwin 2006, Kim and Baldwin 2007), *de-*

[7]See, for example, the results of the Lingo multi-word expression project (http://mwe.stanford.edu/)

[8]We use the distinction of open- versus closed-class items in the sense of Quirk et al. (1994).

terminerless PPs (Baldwin et al. 2003b), or non-heads selecting for heads they combine with (Soehn and Sailer 2003).
A minimal lexical entry can be considered as the combination of word-forms (either explicitly listed or in form of lemma + inflectional type) and the associated word sense. Further constraints have to be added to specify which morphological, lexical or syntactic constructions it can enter or with which other words it is compatible. The acquisition of morpho-syntactic constraints like gender (Barg and Walther 1998, Cucerzan and Yarowsky 2003, Nicholson et al. 2006) has been investigated, as well as lexical properties related to lexical rules such as diathesis alternations (Lapata 1999, McCarthy 2001). An additional area of interest was semantic classification, such as count/mass noun distinction (Bond and Vatikiotis-Bateson 2002, Baldwin and Bond 2003b), noun specificity (Caraballo and Charniak 1999), qualia structure in the framework of Pustejovsky (1995) (Claveau et al. 2003, Yamada and Baldwin 2004, Cimiano and Wenderoth 2005, Yamada et al. 2007), meronymy (Girju et al. 2003), verb aspect (Siegel and McKeown 2000), lexical-semantic verb classification (Stevenson and Merlo 1999, Schulte im Walde 2003, Joanis and Stevenson 2003, Korhonen and Briscoe 2004) or general lexical semantic classes (inferred using derivational affixes as cues, cf. Light 1996). A large body of research was spent on the acquisition of verbal subcategorization frames[9] (Brent 1993, Ushioda et al. 1993, Manning 1993, Ersan and Charniak 1996, Briscoe and Carroll 1997, Carroll and Rooth 1998, Kuhn et al. 1998, Wauschkuhn 1999, Sarkar and Zeman 2000, Schulte im Walde et al. 2001, Korhonen 2002, Sarkar and Tripasai 2002, Stevenson and Merlo 1999, Briscoe 2001, Carroll and Fang 2004), verbal lexical conceptual structures (Dorr and Jones 1996, Dorr 1992), adjectival subcategorization frames (Yallop et al. 2005), and PP attachment preferences (Fabre and Bourigault 2001, Hindle and Rooth 1993, Baldwin 2005d). Further work has been published on the acquisition of selectional preferences/restrictions (Resnik 1993, Ribas 1995, Poznański and Sanfilippo 1996, McCarthy et al. 2001).

1.3.2 Evaluation of Lexical Acquisition Methods

From the viewpoints of quality assurance and methodology the *evaluation* of the proposed methods is of the highest importance. Evaluation addresses (at least) two basic questions: firstly, how well did the acquisition procedure meet the expectation of the developers? One typical approach consists of comparing the results with a resource that stores the lexical properties as they are intended by the developer: the so-called *gold standard*. This re-

[9] For a review see also Matsumoto and Utsuro (2000) and Schulte im Walde (2009).

source can be either a compilation of features hand-judged in advance by the developer, for instance all subcategorization frames of the verbs (or at least a certain subset of verbs - if not all verbs should be evaluated) as they are detected by introspection of the developer or by going through parts of the corpus and judging sample sentences by hand. Alternatively, it can be an extract of already available 'authoritative' resources such as the Oxford Advanced Learner's Dictionary (OALD), the Alvey Natural Language Tools (ANLT) dictionary or the COMLEX dictionary, for example. It should be noted that this kind of evaluation is not always straightforward, because the gold standard might also suffer from incompleteness and inconsistency, especially if it was designed for human users in the first place. Schulte im Walde (2002) compares the subcategorization frames acquired by a German statistical grammar with the 'DUDEN - Stilwörterbuch' that acted as the gold standard in her study. The study revealed inconsistencies with regard to the free dative construction ('the mother baked *him* a cake') and incompleteness with regard to the subcategorization of non-finite and finite clauses and a couple of other shortcomings in the gold standard.
Secondly, how much did the NLP system which is supplied with the results of the acquisition procedure improve its performance? To achieve this, one first needs a set of criteria for the performance of the system as such, for example the accuracy in ranking different parsing alternatives. Then it can be measured how much the accuracy increases after the modification of the system, which is either performed by supplementing it with acquired information or more directly by training it with sample data (*training set*) if the system itself is able to learn from input.
Another common method of evaluation is the comparison of the system's performance with a *baseline*, which is according to Manning and Schütze (1999: 234) 'the simplest possible algorithm' to perform the task. For example, in the context of detecting PPs that are verbal arguments, a baseline could be to classify all PPs as adjuncts and then evaluate how much the system improved when supplemented with the acquired information compared to the baseline.
Evaluation is typically quantified using metrics like *precision* and *recall* that originate from the field of information retrieval. Although there are additional metrics in the literature, the focus here is on precision and recall, because these are the metrics used for the evaluation of the implementation proposed in this thesis. They quantify the degree of correctness and completeness of the acquisition procedure. Type precision deals with the question, how good the system is in identifying *just* the correct features of a lexeme, while type recall gives a measure of how good the system is in identifying *all* the correct features of a lexeme. A prerequisite for their definitions is the

classification of the particular acquisition results into true positives (TPs), false positives (FPs) and false negatives (FNs). An item *in* the result set is a TP if it is predicted by the gold standard (expected by the investigator), otherwise it is a FP. Items *missing* in the result set although predicted by the gold standard (expected by the investigator) are FNs. The definitions of the metrics follow straightforwardly:

$$\text{Type Precision} = \frac{\text{Number of overall TPs}}{\text{Number of overall TPs} + \text{Number of overall FPs}} \quad (1.3)$$

$$\text{Type Recall} = \frac{\text{Number of overall TPs}}{\text{Number of overall TPs} + \text{Number of overall FNs}} \quad (1.4)$$

In addition to type precision/recall, an additional metrics is often used to characterize the quality of the procedure: token recall, which is:

$$\text{Token Recall} = \frac{\text{Number of TP tokens}}{\text{Number of overall tokens in the test set}} \quad (1.5)$$

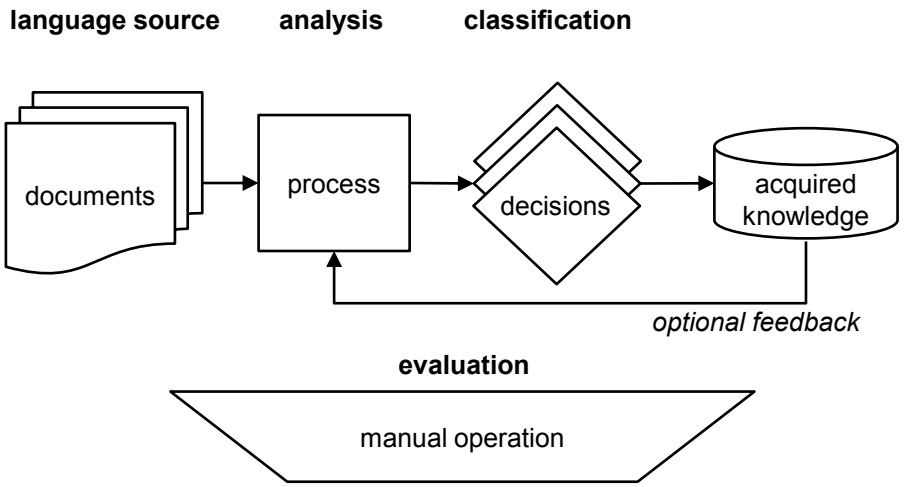
In the context of subcategorization frame (SCF) acquisition, for example, this amounts to the number of particular SCF tokens correctly classified by the system divided by all SCF test tokens in the test data. Token recall is derived based on manual analysis of the test data. A further important standard metrics is the *F-measure* which is the harmonic mean of type precision and recall. In general, it is difficult to use F-measure to predict parser performance as has been shown in Zhang et al. (2007).

1.3.3 Different Strategies for Lexical Acquisition

With the help of figure 1.1 this section summarizes important aspects of machine-driven lexical acquisition while adding some additional notes. The figure shows a simplified process flow typical for many approaches as well as for the present work. The task of the process is to automatically acquire lexical knowledge from some given language source, which can be a raw text or a collection of texts compiled according to certain criteria - either an annotated corpus or a raw corpus. If it is annotated, it is important to notice whether the annotation has been achieved manually or by means of some (often stochastic) NLP tools which are already in place. This is important, because one has to realize the quality of information associated with each

component of the process flow - starting with the input. While expensive to build, hand-tagged corpora usually show a higher accuracy than corpora that are annotated by tools.

Figure 1.1: Typical process in lexical acquisition



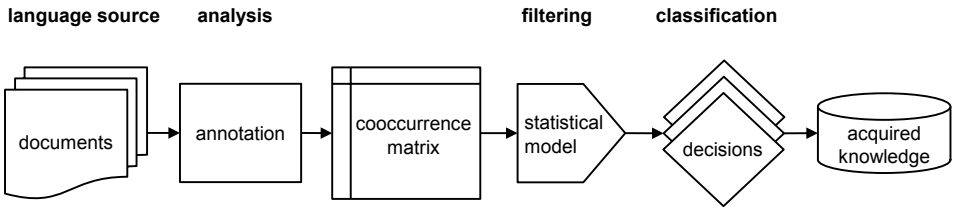
If the corpus is raw or it does not have the annotation which is required by the acquisition process, a first analysis phase (depicted by a simple process box) has to provide that kind of information which is needed to make some decisions about the class membership of a given lexeme. Note that the notions 'having a lexical feature' and 'belonging to a lexical class' are interchangeable throughout this work. If a lexeme 'has' the feature F, it can be said that it is a member of the class of lexemes which have the feature F. All that the language acquisition procedure has to do is to decide whether a certain lexeme belongs to a lexical class or not (for example: is 'car' a count noun or not). This is the *classification* step. The acquired knowledge can be seen as the assignment of all investigated lexemes to lexical classes. If the analyzing component is a system which can exploit this knowledge for further processing, this knowledge might be fed back to it. Finally, the output of the process (and with it the whole process) is evaluated, typically against a gold standard. The evaluation is a manual operation although it can be supported by tools.
Here it becomes obvious that the whole procedure suffers from a problem that has not been touched so far: the *sparse data* problem. This affects lex-

emes which occur quite infrequently as well as lexical phenomena that are infrequently realized during language production, hence there is little (if at all) evidence in the corpus. It is hoped that this effect can be compensated by the use of very large corpora but it is far from trivial to estimate how much data is needed to cover all lexemes and all phenomena that the language engineer aims at. Recent research activities deal with the exploitation of the world-wide web to make even larger corpora available, as in Villavicencio (2003b) or the web-as-corpus website of the Association for Computational Linguistics (ACL)[10].

The following describes five different learning strategies in a nutshell where many details are omitted for the sake of a concise presentation. Three of these strategies are described in more depth in the subsequent sections.

The first strategy deals with statistically significant cooccurrences of words. Figure 1.2 shows a process flow which can be used to detect - to a certain extent - prepositional arguments of verbs, for example. The underlying assumption is that if a verb subcategorizes for a certain prepositional phrase, then it shows cooccurrence with the corresponding preposition more often than chance. For the identification of this kind of word association, the procedure must have at least part-of-speech (PoS) information available. This is achieved by the first annotation step (which is unnecessary if the corpus is already PoS-tagged). During analysis, a cooccurrence table is computed, which counts how often a verb cooccurs with a certain preposition in the same sentence or in a previously defined collocational window. After that, all insignificant cooccurrences are filtered out by means of a statistical model (t-test for example). Classification in this case would be trivial: if a verb-preposition pair passes the filter, the verb is classified as a verb which subcategorizes for the corresponding preposition.

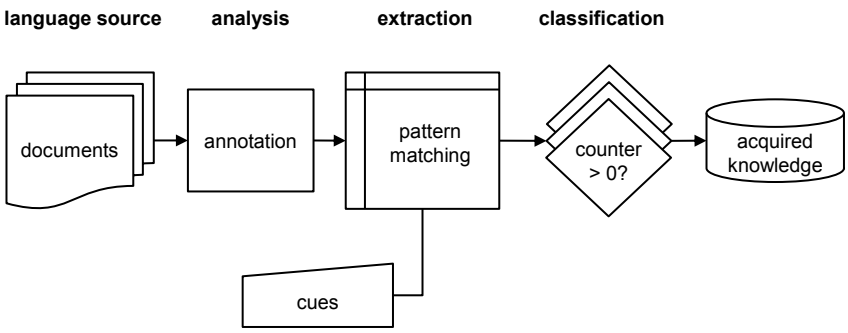
Figure 1.2: Find statistically significant cooccurrences



[10] http://www.sigwac.org.uk/

Another strategy tries to identify lexical features by identifying examples in the input, that unambiguously prove the class membership of a given lexeme. These examples are extracted from the corpus by regular patterns that are given to the extraction component in advance. The extraction step also includes the counting of the corresponding matches. The patterns are designed in a way that they are reliable *cues* for the identification of lexical properties. If a lexeme cooccurs with a pattern that is a cue for the lexeme's property, then the classifier will assign the lexeme to the corresponding class (figure 1.3). This cue-based extraction method is explained in more detail in section 1.3.4.

Figure 1.3: Find cues in the input



Another idea exploits the learning capabilities of probabilistic parsers, for which many approaches and algorithms are already in place. This type of parser can be modified in such a way that the acquisition of probabilities also extents to lexical items (whereas the 'classic' variant of that method focuses on the distribution of *syntactic* rule applications). In section 1.3.5 we present this approach in more detail.

A further strategy developed recently and set forth in a series of papers is called *supertagging* (Baldwin 2005c, Blunsom and Baldwin 2006, Baldwin 2007). This approach is designed for the deep lexical acquisition (DLA) of lexical items for precision grammars. Supertagging is an *in vivo* acquisition method, 'where the target resource that we are hoping to perform DLA relative to is used directly to perform DLA' (Blunsom and Baldwin 2006: 165). It 'can be defined as the process of applying a sequential tagger to

the task of predicting the lexical categorie(s) associated with each word in an input string, relative to a given DLR [deep language resource]' (Baldwin 2007: 5). The tagger is trained by means of token-level annotations for a particular DLR (the one that has to be improved) similar to the training of a PoS tagger using PoS tagged data. Thus the procedure can be seen as a *bootstrapping* method that starts with a given grammar and initial (seed) lexicon and that increases the coverage of the lexicon by classifying unknown lexemes by (super)tagging of the training data.

The last strategy to be outlined here emphasizes linguistic precision over mathematical precision. In this case the input is *deeply analyzed* using a constraint- and unification-based parser. In such a framework, lexical features restrict the application of grammatical rules by unification. To give a short example, consider the following fragment of a unification-based grammar:

$$\begin{bmatrix} \textbf{cat} & s \\ & \end{bmatrix} \rightarrow \begin{bmatrix} \textbf{cat} & n \\ \textbf{agr} & \alpha \\ \textbf{case} & nom \end{bmatrix} \begin{bmatrix} \textbf{cat} & v \\ \textbf{agr} & \alpha \\ & \end{bmatrix} \tag{1.6}$$

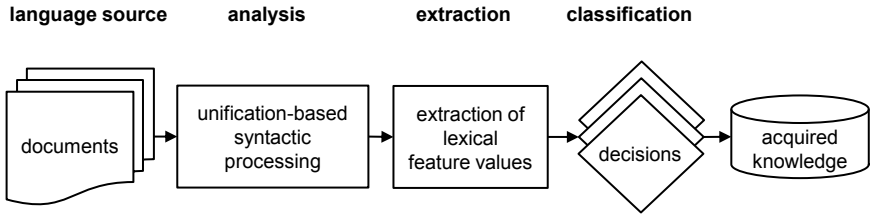
$$\begin{bmatrix} \textbf{graph} & `he' \\ \textbf{cat} & n \\ \textbf{agr} & +3rdsg \\ \textbf{case} & nom \end{bmatrix} \quad \begin{bmatrix} \textbf{graph} & `come' \\ \textbf{cat} & v \\ \textbf{agr} & -3rdsg \end{bmatrix} \quad \begin{bmatrix} \textbf{graph} & `comes' \\ \textbf{cat} & v \\ \textbf{agr} & +3rdsg \end{bmatrix} \tag{1.7}$$

One property of feature structures in such a grammar is one of the things that make this framework so powerful: feature structures can be *underspecified*, perfectly accounting for partiality of information or unknownness. For instance, the specification of case for content nouns can be omitted as opposed to the pronominal entries in (1.7). Thus, with regard to case, feature matrices of content nouns would be underspecified, allowing these matrices to unify with a nominative subject (as in rule 1.6) or with an accusative object in some other rule. Here is exactly the point where lexical acquisition comes into play. If some lexical features in a lexicon entry are missing, this results in *underspecified* feature structures with which the lexical look-up procedure comes up. While the missing features cannot *restrict* parsing they can be *assigned values during parsing*. Consider the highly underspecified feature matrix for the unknown word 'gnarf', for instance, as given in (1.8).

$$\begin{bmatrix} \textbf{graph} & `gnarf' \\ & \end{bmatrix} \tag{1.8}$$

This structure does not specify any category. The system does not *know* whether it is a verb or a noun or any other category. All it knows is the graphical representation (graph), but when this entry is unified with the verbal part of the rule (1.6) during parsing of the sentence 'I gnarf', it will be assigned the PoS value v for the category feature and the system has evidence to *hypothesize* that 'gnarf' is a verb. All underspecified lexical features have a chance to receive a value during parsing and after parsing is completed, the lexical features can be extracted from the resulting trees. Using this information, the classifier can assign the lexemes to the according lexical classes (figure 1.4).
Many questions might arise now, for instance how different hypotheses about the same token might converge to a new lexical entry or how the classifier might work in detail. The answers are to be worked out throughout this dissertation.

Figure 1.4: Extract lexical features from a unification-based parser



The next four sections contrast three of the above mentioned *strategies* of research on the acquisition of lexical features. The first strategy is exemplified by the unsupervised acquisition of verbal subcategorization frames via the *extraction* of unambiguous examples from a given input (section 1.3.4). It typically goes along with the elaborate application of statistical filters and therefore has a strong mathematical bias.
The second strategy bases on the work of Carroll and Rooth (1998) outlined in section 1.3.5. The third paradigm which bases on the idea that lexical features can be automatically extracted from the output of unification-based parsers is elaborated in section 1.3.6. The pros and cons of these methods are then discussed in section 1.3.7.

1.3.4 Cue-based Statistical Approaches: Brent 1991 and Subsequent Accounts

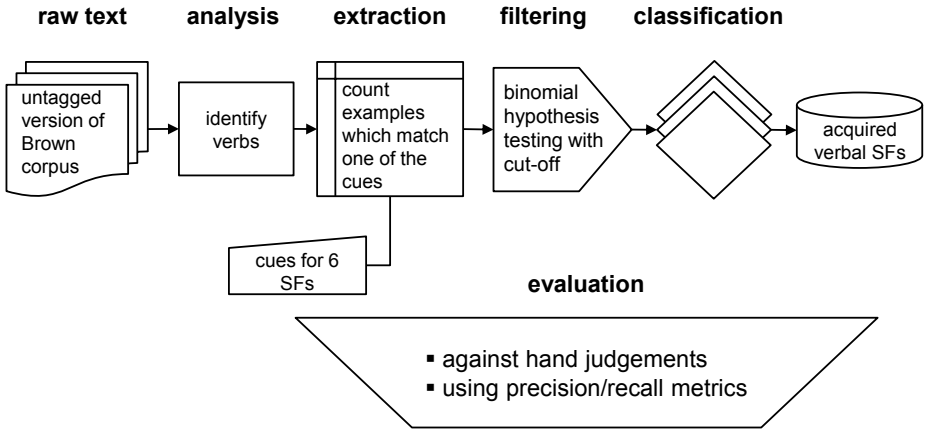
In a series of publications (Brent 1991, Brent 1993, Brent 1994) Michael Brent initiated a strategy for lexical acquisition research that is denoted by the notion 'cue-based extraction method' throughout this work. His idea was '[...] the first attempt to design a system that autonomously learns syntactic frames from naturally occurring text' (Brent 1993: 244). The discussion of his method bases on Brent (1993). The 6 SCFs studied in this paper are listed in table 1.1.

Table 1.1: The six subcategorization frames in Brent (1993)

SCF Description	Symbol	Good Example	Bad Example
NP only	NP	greet them	*arrive them
tensed clause	cl	hope he'll attend	*want he'll attend
infinitive	inf	hope to attend	*greet to attend
NP & clause	NPcl	tell him he's a fool	*yell him he's a fool
NP & infinitive	NPinf	want him to attend	*hope him to attend
NP & NP	NPNP	tell him the story	*shout him the story

The basic procedure is depicted in figure 1.5 using the terminology developed in the last section.

Figure 1.5: The process flow in Brent (1993)



Input of the procedure is raw text (Brent uses the untagged version of the Brown corpus for his experiments). The first step is finding verbs in the input. It consists of 2 phases:

1. Strings which occur with and without suffixed 'ing' are identified as potential verbs.
2. Potential verbs are only used in their stem-form or with suffix 'ing' attached [11]

Although the procedure developed by Brent is not an annotation in the classical sense, from an abstract point of view (and to make it more comparable to subsequent approaches), it might be treated as if those strings in the input which adhere to the above criteria are tagged as verbs. The next step - the extraction phase - identifies right-hand contexts of the previously located verbs, which serve as indicators of the usage of a SCF for the particular verbs. To ensure that these contexts are syntactically unambiguous (at least with high reliability), only those strings are considered that match one of the regular expressions listed in table 1.2 with the respective lexical categories defined in table 1.3. Brent calls them *local morpho-syntactic cues*. Here, 'cap' is any string starting with capital letter and cap+ is any sequence of cap-strings.

Table 1.2: Local cues in Brent (1993: 247)

Frame	Symbol	Cues
NP only	NP	(OBJ | SUBJ_OBJ | cap) (PUNC | CC)
Tensed Clause	cl	(that (DET | SUBJ | SUBJ_OBJ | cap+)) | SUBJ | (SUBJ_OBJ +TNS)
Infinitive VP	inf	to V
NP & clause	NPcl	(OBJ | SUBJ_OBJ | cap+) cl
NP & infinitive	NPinf	(OBJ | SUBJ_OBJ | cap+) inf
NP & NP (dat.)	NPNP	(OBJ | SUBJ_OBJ | cap+) NP

[11] 'There are several reasons for this. First, forms ending in -s are potentially ambiguous between third person singular present verbs and plural nouns. [...]
Second, past participles do not generally take direct objects: *knows me* and *knew me* are OK, but not **is known me*. Further, the past tense and past participle forms of some verbs are identical, while those of others are distinct. As a result, using the *-ed* forms would have complicated the statistical model substantially.' Brent (1993: 246).

Table 1.3: Lexical categories for cue definitions (Brent 1993: 247)

SUBJ:	I | he | she | we | they
OBJ:	me | him | us | them
SUBJ_OBJ:	you | it | yours | hers | ours | theirs
DET:	a | an | the | her | his | its | my
	| our | their | your | this | that | whose
+TNS:	has | have | had | am | is | are | was |were | do | does |
	did | can | could | may | might | must | will | would
CC:	when | before | after | as |while | if
PUNC:	. | ? | ! | , | ; | :

So if the query matches the string 1.9

$$\ldots \quad see_V \quad Peter_{cap} \quad before_{CC} \quad \ldots \tag{1.9}$$

by the expression [V cap CC], the counter for SF_{NPonly} of the verb 'see' is increased by one. Further examples are given in table 1.4. The counters are used to *classify* the verbs: if and only if a counter for a SCF of verb V is >0, V belongs to the class of verbs which can appear in this SCF.

Table 1.4: Further examples of matching cues

Verb	right-hand context	Cue	SCF (symbol)
know	that a	V that DET	cl
want	to give	V to V	inf
tell	him that he	V OBJ that SUBJ	NPcl
ask	him to go	V OBJ to V	NPinf
give	her yours!	V OBJ SUBJ-OBJ PUNC	NPNP

Applied to the untagged version of the Brown corpus, the cue-based queries select for instance 6 *NP*-contexts and 6 *cl*-contexts of the verb 'recognize' which occurred 71 times in total. This is encouraging because this verb is in fact transitive and also subcategorizes for tensed clauses.
However, the correspondence between the cues and the syntactic properties of the verbs is not perfect. Brent notes two shortcomings of his approach: 'The cues are fairly rare, so verbs [...] that occur fewer than 15 times tend not to occur with these cues at all. Further, these cues occur fairly often in

structures other than those they are designed to detect.' (page 249)

One example for misleading contexts is the verb 'record' with two selected *inf*-contexts (the verb does not subcategorize for infinitives in fact), based on the following observations (cf. page 249), repeated here as (2). In these sentences 'record' functions as a noun, rather than a verb.

(2)

a. But I shall campaign on the Meyner *record* to meet the needs of the years ahead.

b. Sposato needed a front, some labor stiff with a clean *record* to act as business agent of the Redhook local.

In the first sentence of (3), the token 'recovering' is a verb but the following infinitive VP is not an argument of the verb. The second sentence shows the token 'referring' (a verb) followed by a PP (starting with 'to change') which the learning algorithm misclassified as an infinitive VP.

(3)

a. The last season the Birds tumbled as low as 11-19 on May 19 before *recovering* to make a race of it and total 86 victories.

b. But I suspect that the old Roman was *referring* to change made under military occupation - the sort of change which Tacitus was talking about when ...

In other words, if the investigator compares the results of the procedure with his own judgments about the 'real' behavior of the particular verbs, he[12] finds that the algorithm does not only deliver

True Positives (TPs): detected subcategorization frames which he *also* would assign to the particular verb (SF_{NP} for 'recognize') and

True Negatives (TNs): undetected subcategorization frames which he *also* would *not* assign to the particular verb (SF_{NPNP} for 'recognize')

but also

False Positives (FPs): detected subcategorization frames which he would *not* assign to the particular verb (SF_{inf} for 'record') and

[12]We use the masculine pronominal forms for the LE or the investigator in order to avoid inconvenient expressions like 'he/she', 'his/her' and the like.

False Negatives (FNs): undetected subcategorization frames which he *would* assign to the particular verb (consider for instance all the verbs which did not appear in *any* cue)

In order to cope with this kind of *noise* produced by the extraction component of the learning procedure, Brent uses a statistical model which improves the reliability of his approach. Reliability means: the resulting set of classifications should contain as few FPs and FNs as possible. The model can be seen as a *filter* applied to the results of the extraction step. Recall that occurrences in sequences of the input that match the cues are counted during the extraction step. Those counter values that are statistically not reliable are filtered out (or reset to zero).
A requirement for the detailed explanation of this filter is the categorization of true and false positives and negatives. It is important to distinguish TPs detected by the whole procedure (including the filter) from true positive *examples* in the input of the procedure. While the example 'want to give' (cf. table 1.4) is a true positive example (TPE) for the assignment of SF_{inf} to the verb 'want', the final classification of 'want' as an SF_{inf}-verb would be called a TP. In analogy, the example '[...] Meyner record to meet [...]' (2a) is a false positive example (FPE) for the assignment of SF_{inf} to 'record'. Note that typically the input does not contain any negative evidence for the classification, so one does not encounter any true (or false) *negative* examples.

The statistical model which Brent applies is called binomial hypothesis testing (BHT). This model is used to decide - with a certain level of significance - which of two complementary hypotheses is true, based on the (assumed) binomial distribution of the investigated data.

Suppose it is known how unlikely it is to encounter an FPE for the subcategorization frame S of a verb V (a verb which does in fact not permit S). The probability to encounter such an FPE is called *error rate*, denoted here by the symbol $\pi_{V,S}$. If one analyzes n occurrences of V in the input, the probability that the classification step will come up with exactly m FPEs is (based on binomial distribution, see also section A.2 in the appendix):

$$P(m, n, \pi_{V,S}) = \frac{n!}{m! \quad (n-m)!} \quad {\pi_{V,S}}^m (1 - \pi_{V,S})^{n-m} \tag{1.10}$$

And the probability of showing m or more FPEs is:

$$P(m+, n, \pi_{V,S}) = \sum_{i=m}^{n} P(i, n, \pi_{V,S}) \tag{1.11}$$

If this value is small, for instance P ≤ 0.02 - this is the value Brent uses as a threshold in his experiments -, it can be assumed with a certainty of 98% that the verb does not permit the frame S. The assumption that all verbs display approximately uniform error rates allows to measure the quality of the method over all verbs using an overall error rate π_S.
In two of the experiments reported in his paper, Brent measures the performance of BHT on the data collected by the cues as a function of $\pi_{SF_{cl}}$ and $\pi_{SF_{inf}}$, respectively (note that one does not know these values a priori). The performance is measured with precision/recall values and the percentage of misclassified verbs. Based on the tables 6 and 7 in Brent (1993: 252-253), figures 1.6 and 1.7 show how precision and recall depend on the error rate which ranges from 2^{-5} to 2^{-13}. A high error rate ($2^{-5} = 1$ error in 32 occurrences) results in a precision of 100% - no false positives are reported by the procedure -, but recall is bad.
An additional experiment tries to estimate the error rate for each of the six subcategorization frames with regard to the Brown corpus. Omitting further details about Brent's estimation technique, table 1.5 shows estimated error rates for the different subcategorization frames together with the resulting precision/recall values. The proper estimation of the error rate, however, does not solve the problem that a high precision value goes in line with a low recall value and vice versa (figure 1.6).

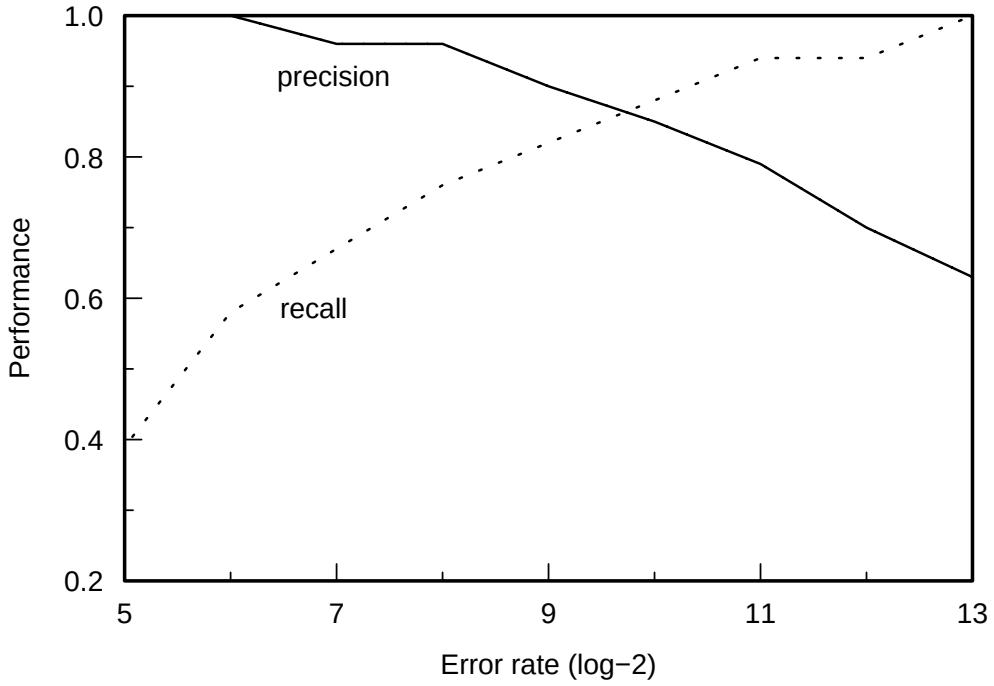
Although Brent's pioneering work shows promising results at first sight - especially if the simplicity of the procedure is taken into account - it has many important limitations with respect to all components of the process.

1) Location of potential examples Brent's algorithm is very restrictive in its determination of potential examples. This can be traced back to a miss-

Table 1.5: Estimated error rates (based on Brent 1993: 255)

SCF	Error Rate	Precision (%)	Recall (%)
cl	0.0037	96	76
inf	0.0048	96	81
NPcl	0.0002	60	100
NPinf	0.0005	100	71
NPNP	0.0004	100	50
NP	0.0132	98	47

Figure 1.6: Performance of BHT for SCF=*cl* as a function of the error rate



ing parser component, which is able to decide much more reliably whether a string in the input is functioning as a verb or not. Furthermore, according to Manning (1993), Brent's heuristic is error-prone and introduces additional noise to the process. Figure 1.8 shows how this deficient component is replaced in Manning's (1993) account. He uses a stochastic PoS tagger for annotation and a finite state parser to detect potential examples. The parser scans through the annotated text until an auxiliary or verb is found (noting whether the verb is active or passive) and then parses the right hand complement of the verb until it determines the end of the subcategorized elements (such as a period or conjunction). The cues for identification of SCFs are built into the parser. The detected frame contexts are entered into a histogram, which is input for the filtering step, similarly to Brent's method. Unfortunately, due to the noise introduced by the first two components, he is forced to restrict his stochastic filter by higher error rates, so that his proposal does not lead to a considerably higher performance. At least, it *enables* him to explore a *higher number* of subcategorization frames, which leads to the next point:

Figure 1.7: Performance of BHT for SCF=*inf* as a function of the error rate

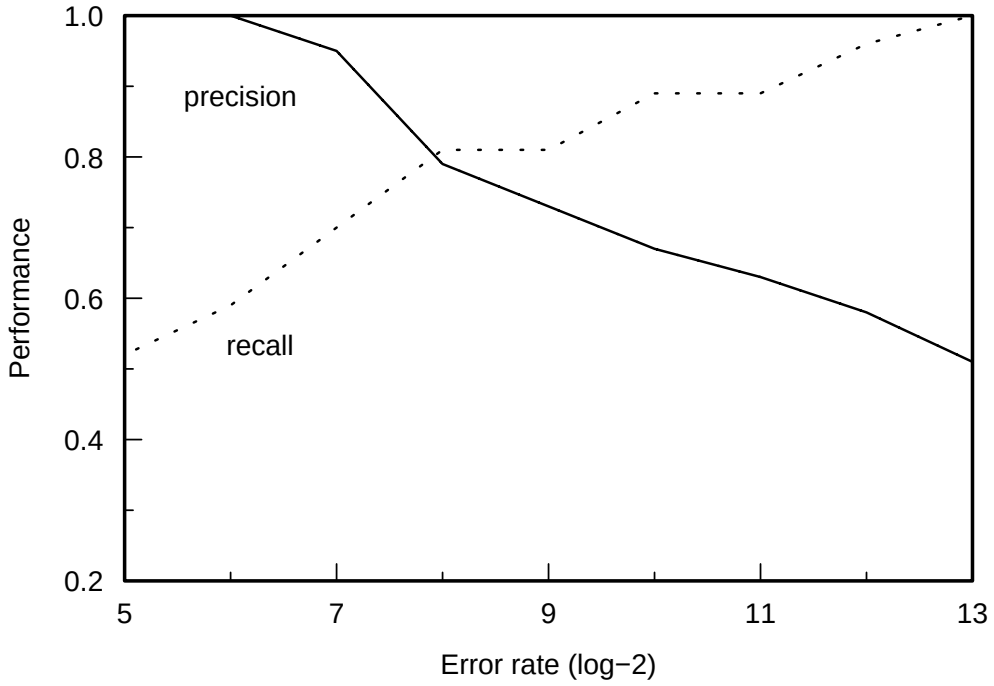
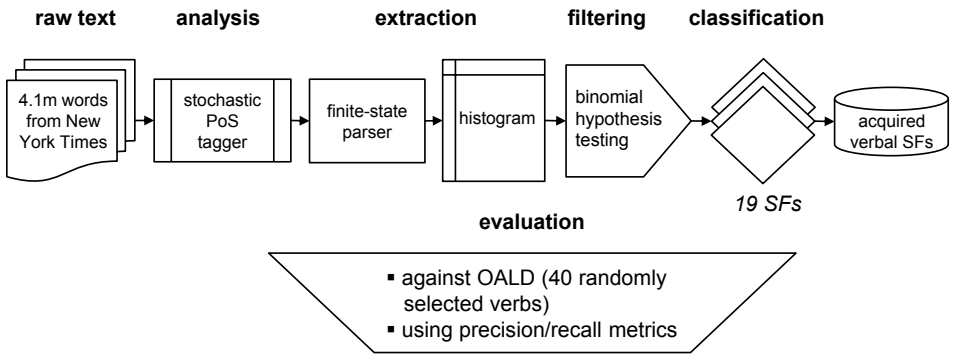


Figure 1.8: Process flow in Manning (1993)



2) Cue-based extraction Manning (1993) argues that the six SCFs in the scope of Brent's paper allow for the definition of cues that are 'very accurate predictors' for the SCFs at hand, but that other SCFs - especially those which

include prepositions - might be more difficult to detect reliably[13]. Apart from the problem that it is not trivial for an NLP component to decide whether a PP attaches to the verb phrase or to other material in the input, it is widely accepted that the boundary between being an *adjunct* or *argument* of a verb is not always easy to draw, especially if prepositions are involved which are *not* subcategorized for by the majority of verbs and are usually projected to sentence level adjuncts (e.g. location phrases).
Moreover, Brent's cues are highly accurate at the expense of their restrictiveness (heavily relying on closed class items like pronouns) and consequently might need a considerable amount of input before they match a string, e.g. for SF_{NPNP}, both NP slots must be filled by pronouns or strings which start with capital letters. And trivalent verbs are in any case rare. This limitation might contribute to the low recall for this SCF which is only 50% (cf. table 1.5).

3) The statistical filter Whereas the shortcomings of the extraction component can be compensated by the employment of a sophisticated annotation phase, the enhancement of the filter seems to pose an even more puzzling challenge and much work has been dedicated to its improvement. The design of the filter bases on a couple of assumptions, which have been questioned by several researchers. Brent himself addresses two critical issues: First, his approach does not take into account the variation in the percentage of verbs that appear in each frame. He observes that because of the high frequency of transitive verbs and the low frequency of verbs in a SF_{NPcl} frame, '[t]his results in too few verbs being classified as +NP and too many being classified as +NPcl [...]' (Brent 1993: 257). Table 1.5 shows that the estimated error rate of SF_{NP} is the highest of all, leading to a low recall, whereas the error rate of SF_{NPcl} is low and a recall of 100% is reached - at the expense of a bad precision.
Second, the assumption that the error rate for *one SCF* is uniform across all verbs does not hold either. Verbs that are homographic to nouns (or adjectives) should be assigned a higher error rate than verbs like 'operate' which have no homographs in other PoS classes.
This appears to be particularly true if no parser-based annotation step is part of the design, because a parser is far more reliable in resolving PoS ambiguities than the heuristic used by Brent.
Nevertheless, even a sophisticated annotation phase as developed in Briscoe and Carroll (1997) cannot solve the problem. Their approach (see figure 1.9)

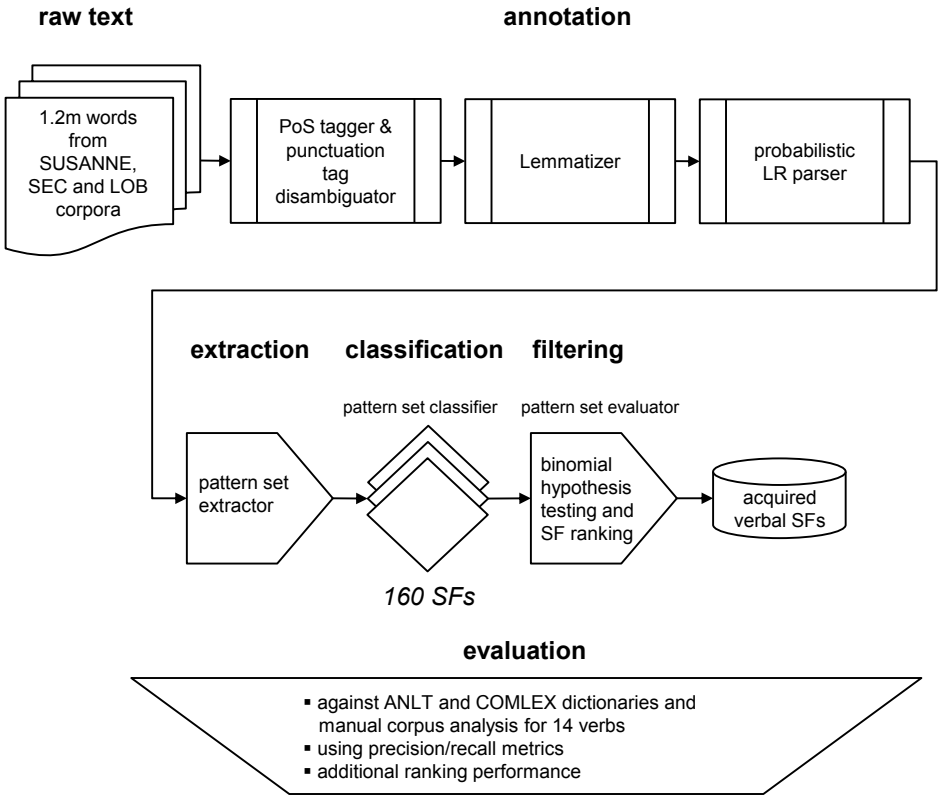
[13]Manning's paper deals with 19 SCFs. In addition to Brent he integrates intransitive verbs and verbs taking -ing complements and prepositions.

includes a PoS tagger, a lemmatizer and a probabilistic LR parser, but - as they note - the weak link in the whole process is the statistical filter and they are not able to obtain a better performance than the previous accounts, despite their additional effort to better estimate different error rates on the ground of relative SCF class size as observed in the automatic noise level estimation (ANLE) dictionary (number of dictionary verbs in class S / number of dictionary verbs) and the distribution of cues for a particular SCF (number of cues for class S / number of all cues).
Thus, Korhonen (2002) claims that '[a]lthough statistical filters have been widely recognized as problematic, the reasons for their poor performance have not been investigated.' (page 21). Guided by the idea, that the incorporation of linguistic a-priori knowledge into the acquisition process might lead to significant improvements (see for instance Manning and Schütze 1999: 276)[14], she demonstrates how the knowledge about the verb's membership to *semantic* classes can improve the filter's performance. It is also not justified, whether binomial hypothesis testing is an appropriate statistical inference procedure at all. Korhonen's (2002) experiments on the comparison of three different filtering types - BHT, likelihood ratio test (LRT) and the relatively simple maximum likelihood estimate (MLE), which just bases on the ratio of count for verb+SCF over the verb's count - show that BHT and LRT are outperformed by MLE.[15]
Coming back to the need for a better 'back-off'[16] estimate of the verb's error rates, the discussion of the statistical filter concludes with Korhonen's idea of integrating semantic knowledge into the design of the filter. In general, the quality of all the statistical filters suffers from the sparse-data problem. They fail to account for highly infrequent events. Observed frequencies are therefore often 'smoothed' to assign non-zero frequencies to unobserved events in some meaningful way. In Korhonen's approach smoothing is achieved for infrequent SCFs by switching to a prior frequency distribution based on semantic classes. The reasoning behind this is that verbs of the same semantic class are likely to admit the same SCFs. Her approach shows a significant increase in performance compared with the results obtained by Briscoe and Carroll (1997) and her own preliminary experiments on MLE based filtering. Table 1.6 summarizes the results - again focusing on precision/recall metrics

[14]Korhonen (2002: 18): 'Given that the current conception of a computational lexicon has a firm foundation in linguistic theory, one of the challenges and currently underused approaches in this area is to constrain the acquisition process using linguistic insights [...]'
[15]Additionally, cf. Sarkar and Zeman (2000) for experiments on t-score performance
[16]Korhonen (2002: 22): 'By back-off estimates, we refer to SCF "prior" probability estimates used for guiding SCF acquisition is [in] some way.'

only.[17]

Figure 1.9: Process flow in Briscoe and Carroll (1997)



4) The classifier Every cue-based learning system includes a component, which assigns word types to classes, even if this component is not always mentioned explicitly: the *classifier*. In case of learning subcategorization frames, the task is to assign a verb to one or more classes of verbs that take a certain SCF. The verb 'serve', for instance, is assigned to the following classes by Brent's algorithm (cf. table 10 in Brent 1993: 261): verbs that take SF_{NP} and verbs that take SF_{inf}. However, the classifier does not report which class is more likely - important information which is often said to be also

[17] We also omit the result of another preliminary test reported in her thesis.

part of the knowledge of the human natural language user and which can help probabilistic parser to obtain a better ranking of parsing alternatives. Furthermore, the likelihood of SCF usages can vary by domain, as shown in the work of Roland and Jurafsky (1998). The limitation of Brent's approach with regard to missing frequency information has been addressed by Ushioda et al. (1993). Consequently, their approach is able to deliver verbal SCF frequencies, which are statistically straightened out.

Linguistic Precision

Reviewing the development of the cue-based statistical approaches, it can be recognized that related components have been improved over time along the dimension called 'linguistic precision' in section 1.1: Brent started with the extraction of examples on *morphological* grounds, Manning takes into account the *syntactic* context of a verb by a shallow finite state parser as followed by Briscoe and Carroll (1997) who implement a linguistically 'deeper' annotation phase and Korhonen incorporates *semantic* knowledge into the filter. Much of this effort is made in order to cope with the noise introduced by the analyzing components. And of course - as one deals with *real* texts - there is also some noise in the corpus itself: misspellings and utterances in the texts which are considered linguistically 'incorrect' from the viewpoint of the engineer who wants his system to learn linguistically warranted features of the investigated language. This kind of noise - unless we have infallible parsers and linguistically 'ideal' texts available - is a fact which every learning system has to face. So, it is reasonable to claim that any practical language acquisition system will need some sort of statistical filter.

Table 1.6: Average results for 45 semantically classified verbs in Korhonen (2002: 135)

Method	Precision	Recall
BHT version of Briscoe and Carroll (1997)	55.3	49.4
MLE thresholding	84.5	47.2
Semantically driven 'back-off' estimates	87.1	71.2

On the Expense of Reliable Cues

The main disadvantage of the cue-based account has not yet been addressed: all the cues (or patterns) have to be worked out by hand. This means that a cue-based system might be unsupervised once the cues have been defined, but it is not autonomous at all in finding lexical properties if only the language and the corresponding grammar is given. All in all, the cues are designed for the identification of unambiguous examples for a certain lexical feature (as the verbal subcategorization frame), but the property of not being ambiguous can be traced back to the structure of the grammar combined with the lexical features it uses. If one wants to lexically enhance an NLP system which already has a grammar component at its disposal, then there is no need for hand-made cues as they are *inherent* in the structure of the grammar component. Moreover, if one is able to exploit the grammar component of the NLP system itself, one might come up with cues that are even more reliable in finding unambiguous examples.
To demonstrate how much manual effort on cues has to be spent to guarantee that the matching examples are (virtually) always unambiguous, Eckle-Kohler (1998) is a perfect example. She outlines a process of semi-automated acquisition of German verb SCFs under the perspective of quality assurance. The carefully designed cues (called 'automated linguistic tests' in her paper) which extract examples from raw texts for subsequent manual assessment should minimize the lexicographer's effort during the revision step. An example is given in table 1.7. The pattern described there is used to detect reliable examples for verbs taking a Subject-NP and a 'daß'[18]-complement. The resulting precision of this cue is due to the *constraints*, which are given in square brackets. Constraints [1] and [2] ensure that no material in the first noun chunk after the subordinating conjunction can serve as a base for the 'daß'-complement. Note that this constraint can only work if the system already knows which nouns subcategorize for and which adverb/adjective can be the correlative of a 'daß'-complement. A second noun chunk is only allowed in the pattern, if it is unambiguously genitive (constraint [3]), so that it is impossible to function as a verbal object. Similarly, an optional preposition chunk is allowed only if the preposition cannot be subcategorized by the verb (verbs taking prepositional arguments are not subject of this cue, hence constraint [4]). The following optional adverbial chunk is also not permitted to contain any material that could serve as a correlative of the 'daß'-complement (constraints [5] and [6]). Finally, the verb chunk itself is not allowed to appear in a past tense form built with the auxiliary 'sein', because the pattern could not distinguish active examples as 'Sie sind

[18] Or 'dass', according to the German orthographic reform.

übereingekommen, daß...' ('they have agreed that ...') from quasi-passives like 'Sie sind benachrichtigt, daß...' ('they are notified that ...'). Cf. Eckle-Kohler (1998) and Kuhn et al. (1998) for further examples.
This example shows how much linguistic knowledge has to be taken into account to design a single reliable cue. The claim made in this thesis, however, is that a full-blown NLP system with a grammar component of comparable quality can take care for these potential ambiguity problems itself in a *fully autonomous manner*.

Table 1.7: Pattern for detection of German verbs with subject NP and 'daß'-complement in Eckle-Kohler (1998)

Pattern	Description	Example
SubordConj	subordinating conjunction	*weil*
NC[1,2]	noun chunk: determiner, as well as modifying adjectives and adverbs are optional [1]: no modifying adverb or adjective which can function as a correlative of an adverbial daß-clause [2]: noun does not subcategorize for a daß-clause	*der Besitzer*
(NCgen[3])	optional genitive noun chunk [3]: case of noun chunk is unambiguously genitive	*des Hauses*
(PC[4])	optional prepositional chunk: preposition followed by a noun chunk [4]: preposition is not subcategorized for by verbs	*trotz schwerer Bedenken*
(AdvC[5,6])	optional adverb chunk [5]: no adverb which can function as a correlative of an adverbial daß-clause [6]: no pronominal adverb which can function as a correlative of a daß-clause indicating its function as a prepositional object	*jetzt endlich*
VCactive[7]	verb chunk in the active voice, containing one lexical verb, including all possible constructions with auxiliaries and modals [7]: forms where the past tense is formed with the auxiliary *sein* must not occur	*zugegeben hat*
daß		*daß*

1.3.5 Lexical Acquisition with Head-lexicalized Probabilistic CFGs: Carrol/Rooth 1998

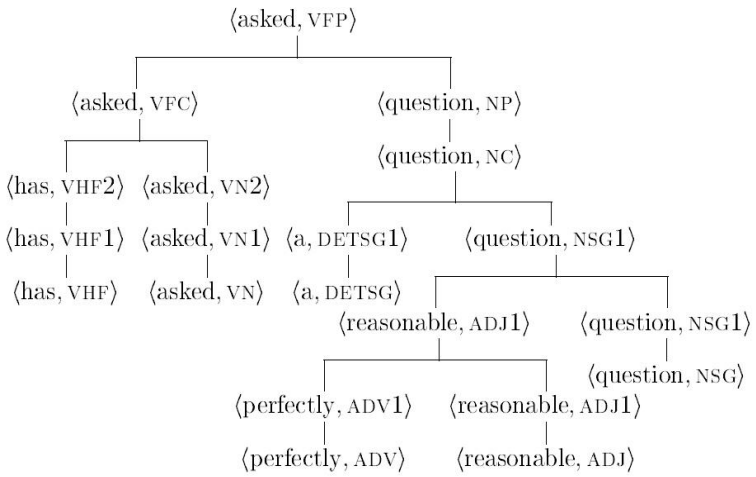
Carroll and Rooth (1998) describe a method that extracts lexical knowledge from *head-lexicalized* probabilistic context free grammars trained on large corpora. Probabilistic context free grammars (PCFGs) are grammars in which every rule is assigned a certain probability which expresses how likely it is for a non-terminal of the rule's left-hand side to expand to the structure given with the rule's right-hand side. PCFGs can be trained on corpora so that the probabilities decoded in the grammar (which is given in advance) reflect the distribution of syntactic structures in the seen input. A 'head-lexicalized' version of a probabilistic context free grammar (PCFG) additionally marks the head of each node in the tree. The authors employ a shallow X'-grammar based on Abney's (1991) concept of chunking. 'Shallow' means that the sentences in the input are not fully parsed. Rather, the analysis of a sentence is a sequence of noun chunks, adjective chunks, finite verbal chunks, prepositional chunks etc. strung together to clauses using an *n-gram*-based finite state model.
For example the phrase 'really should have fully recovered' is a finite verbal chunk (VFC) with 'recover' functioning as its head. The authors enrich the standard context free grammar (CFG) notation with a prime (') that marks the head (or phrase containing the head) on the right hand side of a rule. Consider the following simplified verbal complementation rules for finite verb phrases (VFPs):

$$
\begin{array}{lcl}
\text{VFP} & \rightarrow & \text{VFC'} \\
\text{VFP} & \rightarrow & \text{VFC' NP} \\
\text{VFP} & \rightarrow & \text{VFC' AP} \\
\text{VFP} & \rightarrow & \text{VFC' PP} \\
\ldots & &
\end{array}
$$

Here, for example, the phrasal category VFP can expand to a VFC and an adjective phrase AP. Whereas ordinary PCFGs aim at describing how likely this expansion is (compared to the alternative VFP rules), Carroll and Rooth (1998) want to detect how likely this expansion is for a certain *verb* - or more general, how likely it is for a certain *head*. In other words, they are able to observe how often a certain head enters a certain complement rule during parsing of a training corpus. Moreover, the observation of the head's lexical choice is not restricted to the frame, it is also possible to determine the selectional preferences of single heads. To achieve this, the PCFG is modified so that each phrase that is a projection of a lexical head 'knows' from which head it has been projected. Figure 1.10 serves as illustration. Here

we see a VFP expanded to the verbal chunk and the complementary noun phrase (by rule VFP → VFC NP). Note that the head 'asked' is assigned to the corresponding VFC. Similarly, the NP is labeled with its head 'question', ADJ1 is labeled with 'reasonable' and ADV1 with 'perfectly'.

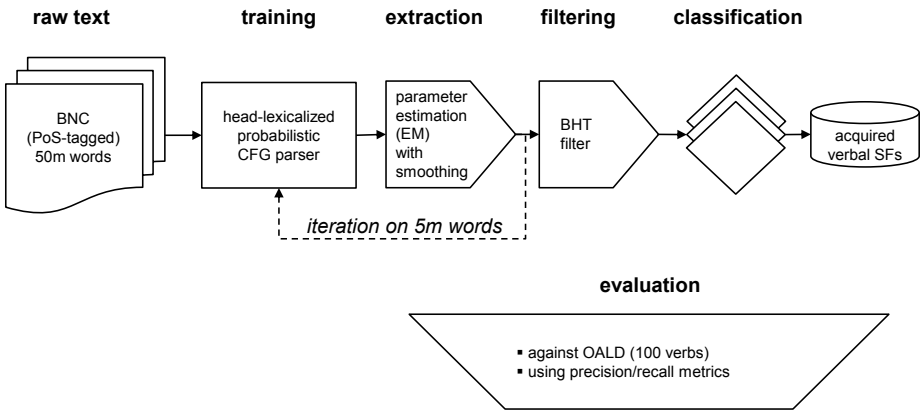
Figure 1.10: Lexicalization of categories in Carroll and Rooth (1998)



The training of the grammar (more technically the *parameter estimation*) is carried out by applying the expectation-maximization (EM) algorithm, which is a familiar procedure for estimating PCFGs. In the next step, the probabilities of the heads occurring in complementation rules are collected and smoothed. Smoothing consists of a) smoothing of rule distribution, where lexicalized rules are smoothed against the (also existing) non-lexicalized grammar rules and b) smoothing of word distributions by using an absolute discounting scheme so that zero probabilities for unseen events are avoided (an event is a head occurring in a rule).
The authors report an experiment on 50 million words from the PoS-tagged BNC, sketched in figure 1.11. Training is performed in iterations on 10 consecutive 5m word segments from the BNC. The distributions are mapped to lexical classes using the filtering method proposed by Brent with the exception that the cut-off values are heuristically determined so that an optimal trade-off between precision and recall is achieved for each frame.
From 200 randomly selected verbs (each occurring more than 500 times in the training set) one half is used to compute the optimal trade-off parameters, the other half is evaluated against the OALD. For verbal subcategorization

an overall precision of 78,88% and an overall recall of 75,06% is achieved.

Figure 1.11: Process flow for the experiment in Carroll and Rooth (1998)



The authors mention 3 major issues with the test: first, the performance with regard to intransitive frames is bad because the parser resolves unparsable constructs as intransitives. Second, the model does not distinguish PP complements from PP adjuncts, hence, adjunct PPs are responsible for a lower precision of frames that include PPs (or verbs that subcategorize for PPs). Third, the detection of frames that cover verb+particle suffers from disagreement of particle tagging in the BNC and the corresponding markup in the OALD.

The authors point out that it is difficult to compare the efficiency of their approach with the results of former approaches. One reason - which has a positive influence on recall in their experiment - is the use of a corpus like BNC that varies much more in genre than a homogeneous corpus like the Wall Street Journal Corpus (WSJ), for example.

Apart from these methodological problems, the strategy adopted here has important advantages compared to the cue-based extraction method. First of all, CFGs have a higher descriptional power than regular expressions. Second, there is no need to supply the acquisition procedure with hand-crafted patterns because the 'cues' are already entailed in the grammar. From this perspective, lexical acquisition comes as a by-product of parsing.

On the other hand, however, all the disadvantages that apply to CFGs also apply to this approach. It is well-known that CFGs simply do not reach the level of linguistic descriptional power that unification-based frameworks

provide. It is also not clear how the problem of unknown words is tackled. From the perspective of the LE there is yet another problem having to do with the acquisition of `discrete` lexical feature values: clearly, the results of the acquisition process can be fed back to the PCFG parser in order to improve its parsing performance (note that classification is not needed). All that is required is to feed back the extracted (and smoothed) probabilities to the parser. In this respect, lexical acquisition is fully autonomous and *in vivo* in terms of Baldwin (2007), but the classification step - which would be required for porting the results to other non-PCFG systems, as well as for the comparison with a gold standard - is not a trivial procedure. This is because aspects of discrete lexical features are scattered over grammatical rules. The system simply *does not know* which rules (and which probabilities) are related to which lexical class. For example, one has to tell the classifier which rules are relevant for the subcategorization properties of verbs (which verbal complement rule is responsible for which SCF class). Anyone who aims at detecting count-nouns, for instance, would have to tell the classifier which rules are responsible for mapping probabilities to the lexical classes of count-nouns and mass-nouns, respectively.
The present research aims to establish a mechanism that automatically fills the classes of a given lexicon without the need of explicitly telling the system about the relation between rules and lexical classes. This can be achieved by exploiting the power of unification-based grammars as shown in the next section.

1.3.6 Learning on the Job: Lexical Acquisition as a By-product

After having reviewed some of the cue-based approaches on the acquisition of lexical features, as well as the head-lexicalized PCFG account, this section will come to a completely different line of research, which might be embraced by the notion of *'learning-on-the-job'*. Here, the key idea is to exploit the linguistic knowledge which is inherent in the grammatical and lexical components of an NLP system. These components are either already there or have to be developed anyway, so using this kind of a-priori knowledge in order to perform unsupervised lexicon learning will save the costs for the preparation and design of appropriate extraction patterns.

Erbach 1990

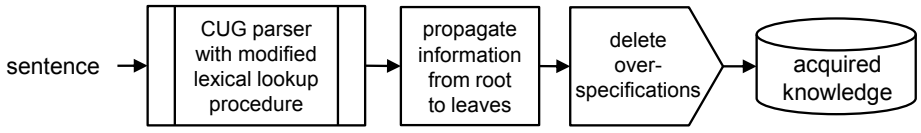
One of the first papers to introduce this idea was by Erbach (1990). From the perspective adopted there, syntactic processing is based on the relation be-

tween a) lexicon, b) grammar, c) strings of words and d) their corresponding syntactic and semantic structures. If one of these entities is partly or fully *unknown*, then syntactic processing is required. In other words, the concepts of *parsing* (structure is unknown), *language generation* (word string is unknown), *grammar acquisition* (grammar is unknown) and *lexical acquisition* (lexicon is unknown) are subsumed under one common account, in which the syntactic processor mediates between known and unknown information within an NLP system.
Erbach argues that unification-based grammars are especially qualified for the lexicon learning task, because they are able to deal with partial structures. His approach deals with unknown words only, but the idea can easily be extended to unknown features of lexical entries which are already specified in the lexical component. Erbach poses two constraints on the input:

1. The input must be a sentence.
2. An unknown word in the input belongs to one of the word classes of the input language or - as a stronger constraint - if the closed word classes are already fully specified in the lexical component, the unknown word is assumed to be a *member of one of the open word classes.*

The second constraint seems to be typical for the 'learning-on-the-job' approach and is also adopted in the implementation presented in chapter 3. Additionally, the system does not (or cannot) distinguish between *new words* which should find their place in the lexicon and *misspelled words* that the language engineer wants to prevent from entering the lexical component. Erbach therefore proposes to combine the processing of unknown words with spelling correction. While the first constraint will be rejected in subsection 1.4.1, the second will be rephrased as an important assumption in chapter 2, subsection 2.1.5. As of now, consider how Erbach's account is working. According to him, nothing more is necessary than a modification of the lexical look-up procedure, which - instead of simply failing when it finds an unknown word in the input - provides a 'highly underspecified' disjunction of all the open word classes in the language. Depending on the context of the unknown word, the parser in step 1 will select the correct feature structure during the syntactic processing step (Erbach uses Categorial Unification Grammar of Uszkoreit 1986). Step 2 consists of the propagation of information from the root node of the parse tree down to the leave node which represents the unknown word. Some of the features collected during this step might be irrelevant for the lexical entry (e.g. gender of the direct object of a verb - given that the verb does not pose any constraint on this), so, in step 3 the 'overspecifications' are deleted from the resulting feature structure. Figure 1.12 depicts the process flow accordingly.

Figure 1.12: Process flow in Erbach (1990)



Russel 1993

While in principle Erbach's idea can be implemented in any unification-based grammar, subsequent accounts (to which we had access) - including our implementation - all work within the framework of HPSG (Pollard and Sag 1994). Such an account is presented in Russel (1993). Russel's system is an extension to a unification-based parser, called UNICORN, which was developed by Gerdemann and Hinrichs (1988). Although Russel's grammar is implemented in HPSG, throughout this discussion, no technical details are provided that would make a thorough review of this grammar formalism necessary.

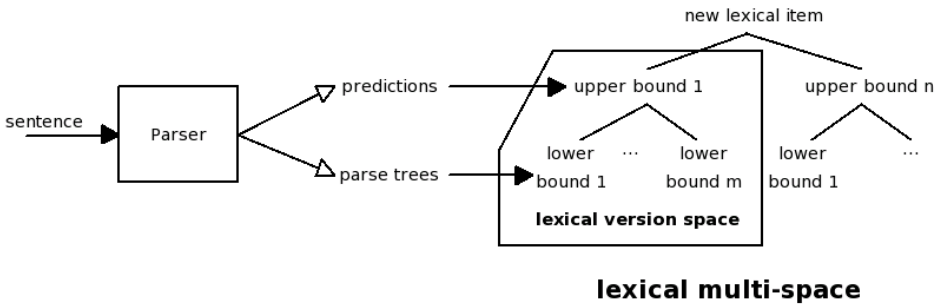
The algorithm of the grammar-independent UNICORN system has its roots in Earley (1970) and was subsequently confined to make it compatible to unification-based grammars. During processing of the input, which runs from left to right, *predictions* are passed to the scanner, which tries to identify constraints on the feature structure of the next input token, so that unnecessary lexical look-up is avoided. When scanning for the next token, the system already has an expectation towards the possible categories to which the token might belong, based on the grammar and on the token's context.
This is where Russel's extension comes into play. If the lexical look-up procedure cannot find the word in any of the predicted categories, processing is handed over to a component that is responsible for the construction of *hypotheses* about the token based on the parser's predictions. Note that this concept of unknown word differs from those accounts for which an unknown word is simply a word that is not stored in the lexicon.
The inability to look-up a token in one of the predicted categories now becomes a learning opportunity for the system. The predictions about the token's feature structure at the point of failure are exploited to derive a set of hypotheses about the token. Russel calls this set the 'upper bound' for the

feature structure of the unknown word. The predictions of the parser result from a function called 'restrictor'[19] that has to be provided by the grammar writer. In addition to the restrictions, some further feature structure is added to each hypothesis, depending on the predicted lexical category. While, for example, the PHON feature (which decodes the representation of the input strings) shows the same structure for every category, the features encoding semantic properties vary across lexical categories. Similar to the restrictor, the feature structure of a lexical category has also to be stipulated by the grammar writer.

After construction of the upper bound, the hypothesized lexical entries for an unknown word now temporarily augment the system's grammar and parsing is resumed. If a hypothesis does not lead to a successful parse, then it will be discarded. If none of the asserted hypotheses 'survive' parsing, then the input sentence is considered ungrammatical. However, if parsing succeeds, then one of the constructed hypotheses is a correct, though not necessarily complete, analysis of the unknown word. In order to fully determine the feature structure of the new word, further information from the resulting parses has to be taken into account. So, for each unknown word, the feature values that have been assigned during parsing by means of unification are stripped out from the resulting parse trees. At this point of time, the system does not know which of the feature values are essential for the target definition of the word. This is why the extracted features at first provide a 'lower bound' of the definition. The correct structure of the new lexical entry will be 'somewhere between' the upper bound and one of its lower bounds (if there is more than one). So, the result of the procedure is a set of new lexical items, each item having a set of upper bounds, each upper bound being assigned a set of lower bounds, as show in figure 1.13. In Russel's work, the data structure consisting of one upper bound and a set of corresponding lower bounds is called 'lexical version space'. The disjunction of all lexical version spaces of a lexical item makes up the 'lexical multi-space' assigned to this item.

[19] This concept goes back to Shieber (1985).

Figure 1.13: Lexical multi-space in Russel (1993)



As an example, Russel (page 107-108) provides a small grammar fragment, expressed in phrase structure rules (rather than HPSG path equations)[20]:

1)	S	→	NP VP
2)	VP	→	V VP
3)	VP	→	V ADV-OBJ
4)	VP	→	MOD-ADV VP
5)	VP	→	INTRANS
6)	NP	→	NP CONJ NP
7)	NP	→	*John*
8)	INTRANS[PAST]	→	*left*
9)	INTRANS[PPAST]	→	*left*
10)	ADV-OBJ	→	*left*

Now, the following sentence (4) includes the unknown word 'glarf' and we want to see how the lexical multi-space for this word is constructed:

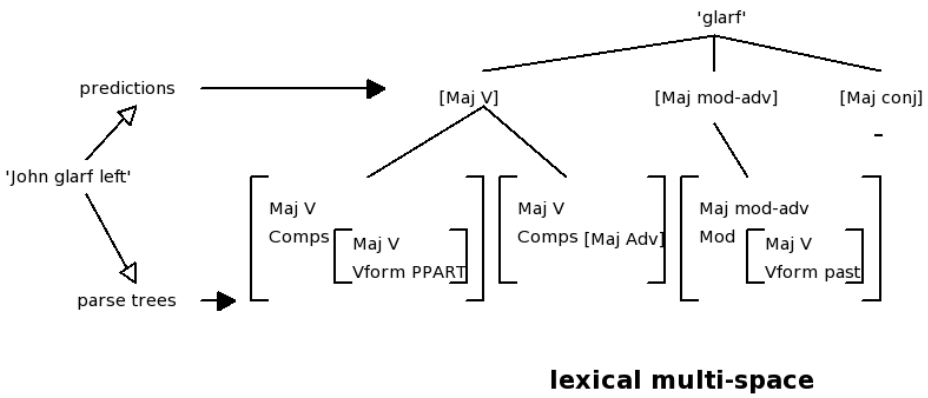
(4) John glarf left

First, when the system tries to parse the sentence, lexical look-up fails at the point where 'glarf' is encountered. The predictions of the parser for the

[20]Note that rule 2) is used for a combination of auxiliary verb (as part of category 'V') and verb phrase.

token's category are a) V because of rules 2 and 3, respectively, b) MOD-ADV ('modifying adverb') because of rule 4 and c) CONJ because of rule 6. Whereas the last hypothesis does not lead to a successful parse, the former two result in parses for which lower bounds of 'glarf' can be constructed: the word can be a V, taking either a past participle verb (equivalent to 'John has left') or an adverbial object (as in 'John runs left') or it can be an adverbial that modifies a verb (as in 'John quickly left'). Figure 1.14 shows the resulting lexical multi-space of 'glarf' with upper and lower bound using Russel's notation ('Maj' is the lexical category, 'Mod' denotes the modified category, 'Comps' denotes the verbal complement).

Figure 1.14: Lexical multi-space for 'glarf'



From this single encounter of the unknown word 'glarf', the system cannot yet determine the exact feature definition of the this word. So, all lexical multi-spaces of the new word derived from different sentences are combined, usually reducing the number of upper bounds and generalizing or eliminating lower bounds (cf. page 123). The goal 'is a single definition of the word 'glarf' which satisfies all the constraints of all contexts in which it has been seen.' (page 126). Omitting further details on the implementation of the combination of lexical multi-spaces, the focus is on the key ideas of Russel's account:

First of all, his system does not seek for unambiguous examples that prove a certain property of a new word. Rather, it derives possible hypotheses about a new word from each sentence in which the word occurs. The

combination of these sets of hypotheses is achieved in a way that ensures consistency of the word's target definition with the seen input. If two sets of upper bounds cannot be combined (because they are incompatible with each other), the system assumes two corresponding target definitions of the word. The disadvantage of this procedure is addressed on page 127: Suppose, a new word that in fact belongs to two categories, let's say V and N, is encountered in two sentences, (accidentally) leading to an upper bound set {V, Adj} for the first and {N,Adj} in the second sentence. The combination of these sets (which is usually similar to intersection) would yield {Adj}, because this category would be consistent with both sentences. Russel claims that those incorrect conclusions 'while not advancing the knowledge of the system, will not be a serious hindrance to it, either' (page 128). Here, we disagree, because in a large-coverage grammar often many possible parses are derivable and a false conclusion about a lexical item will result in an incorrect selection of a parse.
Second, similar to Erbach (1990), the instantiations of a lexical item in a parse tree typically contains feature settings that do not belong to a proper definition of the lexical item. The 'overspecifications' are filtered out by the comparison of the available lower bounds of the lexical version spaces. According to Russel (page 128), '[f]or lower bounds, this means finding the most informative possible feature structure all of whose constraints are consistent with all of the instantiated parses licensed by that feature structure.'. While this procedure ensures consistency, it can be argued that it might be too restrictive, because words typically belong to more than one subclass. A verb, for example, subcategorizes for different constituents and each subcategorization frame corresponds to a certain verbal subclass. What will happen, if a verb is encountered in, say, 3 different sentences each time appearing in a different frame? There is one upper bound that determines that the word is a verb and three different lower bounds, one for each frame. The assignment to the three verbal subclasses cannot take place, as the different complementation features are filtered out by the learning procedure. There is a further question that arises at this point: given that the lexical structures have to be provided by the grammar writer anyway, why shouldn't he simply mark, which are *essential features* for a given lexical category? This would free the system from the burden of detecting these essential features automatically. One of the major problems of lexical acquisition in a unification-based framework is the question, when do we think the system has seen enough input to converge all acquired pieces of information to 'stable' lexical entries. In Russel's system, '[c]omplete convergence on the correct definition of a word occurs when only a unique lower bound remains at some point during a learning session, and it is equivalent to the unique upper bound.'. How-

ever, there is no guarantee that complete convergence will ever take place for a given lexical item and Russel notes that, '[i]t becomes a matter of extra-grammatical heuristics to decide when a lexical item has been completely learned.' No such heuristic has been implemented in his system, but Russel proposes that after a 'reasonable number of encounters with a variety of input data', the system should assume a lower bound to be the correct lexical definition although it has not converged to the upper bound.

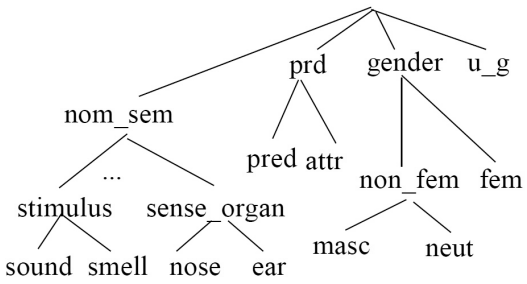
Incremental Lexical Acquisition: the Work of Barg et al.

While the previous accounts mainly dealt with the acquisition of unknown words, the work of Barg and colleagues (Kilbury et al. 1992, Barg 1996b, Barg and Walther 1998, Barg and Kilbury 2000) makes further progression to a more general account[21]. For them, the openness of the lexicon is an inherent aspect - and not just an unwelcome shortcoming - of language, which has to be properly reflected by any NLP system, '[...] so that the model itself encompasses incompleteness. Thus, when a sentence is parsed the problem may arise that no adequate entries for certain words can be found in the given lexicon, but the system must, nevertheless, be able to parse such a sentence. Better yet, it should utilize the sentence as a source of new information in order to extend the lexicon.' (Kilbury et al. 1992). This means that an NLP system should not only *deal with* incompleteness, but should be able to *improve* its capabilities from the linguistic information entailed in the system's input. Consequently, in order to optimize learning from input, the design of a lexical acquisition system should enable it to 'maximally exploit the context' (Barg and Walther 1998) of unknown words, which makes full grammatical processing necessary. Additionally, as lexical incompleteness does not only refer to (completely) unknown words (that is no entry in the system's lexicon is matched at all by a token of the input string), the authors rather view incompleteness as '[...] a gradual, information-based concept of 'unknownness' providing a uniform treatment for the range of completely known to maximally unknown lexical entries' (Kilbury et al. 1992).
According to them, unknown information is *revisable* information that can be divided into two classes: generalizable information and specializable information. This distinction can be illustrated along a simplified type hierarchy in figure 1.15[22].

[21] For an implementation of lexical acquisition within the DATR framework Evans and Gazdar (1989), see Barg (1992), Kilbury et al. (1992), Barg (1996a) and Barg (1996b)

[22] We leave the class *u_g* unexplained here as this is a pseudo-class needed for the technical implementation.

Figure 1.15: Excerpt from type hierarchy (Barg and Walther 1998)



The account of a gradual concept of 'unknownness' extends to all kinds of lexical information, morpho-syntactic (e.g. gender), semantico-syntactic (predicative versus attributive usage of adjectives), as well as semantic (ontological) classification. Hence, the type hierarchy depicted in figure 1.15 is a mixture of hierarchies from different domains. Learning is an incremental process of revision of already learned class assignments. Suppose, the system has already learned from some input sentence, that the gender of noun N is non-feminine. A new sentence now can be evidence for a more *specific* class assignment and the system might infer that N belongs to the class of masculine nouns. In other words, specialization of information means the revision from more general class assignments to less specific class assignments. This is achieved by *type unification*: $non\_fem \wedge masc = masc$.
Generalization is the other way round: if the system has learned from input that the adjective A can be used predicatively (resulting in assignment to the pred-class) and it has additional evidence for the attributive usage (attr-class), it will assign A to the more generic prd-class. Formally, this is an instance of *type union*: $pred \vee attr = prd$.
Note that the language engineer must tell the system about revisable features and whether they are generalizable or specializable. In their paper, Barg et al. mention the following lexical properties as examples for generalizable or specializable information, respectively:

Specializable: semantic type of noun, gender, inflectional class of noun (for some worked-out examples, see Barg and Kilbury 2000)

Generalizable: selectional restriction of verbs and adjectives, predicative/attributive usage of adjectives, valence class of verbs, case and form of PP arguments

Both semantic type of nouns and the selectional restrictions show that a certain type (sub-)hierarchy (in this case noun ontology) is not dedicated

to one of the information categories. Whereas the former is specializable information, the latter is generalizable information.
Keeping all the technical details apart, we want to illustrate the incremental learning process using the following German examples from Barg and Walther (1998), repeated here as (5).

(5)

i. Die Nase ist für Gerüche **sensibel**. (the nose is **sensitive** to smells)

ii. Die **sensible** Nase reagiert auf Gerüche. (the **sensitive** nose reacts to smells)

iii. Die **Nase** ist ein Sinnesorgan. (the **nose** is a sense organ)

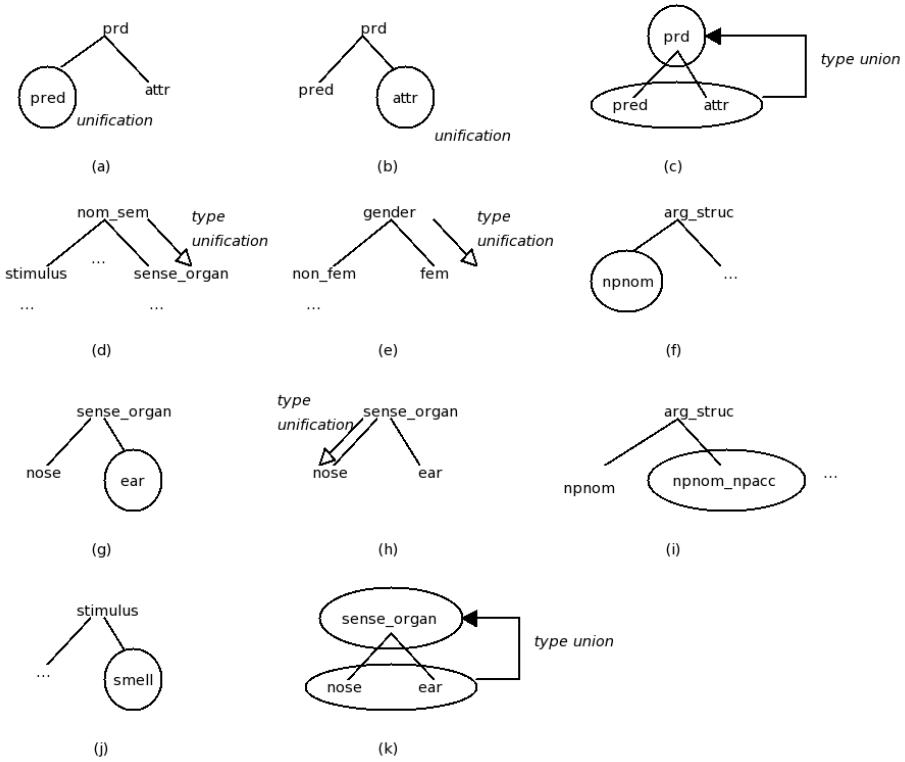
iv. Das Ohr **perzipiert**. (the ear perceives)

v. Eine verschnupfte **Nase perzipiert** den Gestank. (a bunged up **nose perceives** the stench)

In the examples, words that are subject to revision are printed in bold font. The revision of information is additionally depicted in figure 1.16. Example i) provides evidence that the adjective 'sensibel' can be used predicatively (as a complement of German copula 'sein', which has the third person singular form 'ist'). This learning step (a) can be achieved by normal unification (if there is a rule of 'sein' + adjective phrase, which puts the constraint on the adjective of being of class pred). Example ii) gives evidence that the adjective can also be used attributively (here as a modifier of the noun 'Nase'), see (b) in the figure. Having learned these facts, type union of *pred* and *attr* yields the more general class assignment of *prd* (c). Example iii) would enhance the system's knowledge by two type unifications: First, assuming that there is a rule for the structure [N1 copula N2] which allows the semantic type of N2 to percolate to N1, the unification of *nom_sem* (that is, the system does not know any specific semantic class of 'Nase') with *sense_organ* (let's assume that the system knows that 'Sinnesorgan' is a sense_organ), results in *sense_organ*: $nom\_sem \wedge sense\_organ = sense\_organ$ (d). Second, the information about gender of 'Nase' is specialized to *fem* (e) because the noun phrase must be singular (in agreement with the copula) and the determiner 'die' can only be used for feminine nouns when used in singular form. From example iv) the system can learn that the verb 'perzipiert' can be used in a subcategorization frame which is called *npnom* here: The only argument is a nominative NP (f). Additionally, it learns about selectional restrictions (g) on this argument: it is 'Ohr' ('ear'). This information will be revised by example v). The verb also accepts 'Nase'

('nose') for the first argument. As 'ear' and 'nose' are the only subtypes of *sense_ organ* in this simplified ontology, the system will generalize: this class now constrains the selectional properties of the first argument (k). It also learns an alternative subcategorization frame (i) - encountering a second (accusative) NP -, and it learns about the selectional restrictions for the second argument (j). In addition, the semantic class of 'Nase' is again specialized (h). Assuming that the system already knows about the selectional restrictions of the adjective 'verschnupft' (*'das verschnupfte Ohr'), it can reason that the modified noun must refer to a nose.

Figure 1.16: Revision of lexical information



It is important to note that the incremental acquisition of linguistic knowledge does not immediately restrict the parsing process. Although the system learns from input that 'sensibel' is a *pred*-adjective, this does not mean that a sentence in which this adjective occurs attributively would be impossible to be processed afterwards. However, the question arises *when* a lexeme in the system will be updated so that it really restricts parsing. The authors leave this topic for further research. There is another important limitation of the account with regard to the requirement that every subtype of a type T has to be learned before T can be generalized, as noted by the authors themselves: 'In contrast to humans, who seem to leap to conclusions based on incomplete evidence, our approach employs a conservative form of generalization, taking the disjunction of actually observed values only. While this has the advantage of not leading to overgeneralization, the requirement of having to encounter all subtypes in order to infer their common supertype is not realistic (sparse-data problem).' Despite these problems the strategy outlined here seems quite promising because it makes maximum usage of linguistic insights - encoded in a sophisticated grammar framework - and that learning should come as a natural by-product of what an NLP system is designed to do: processing of natural language.
Deep processing of the input is, however, much more expensive - compared to more 'shallow' accounts that work with CFGs or regular patterns. The situation gets even worse when the lexicon is highly underspecified, making room for a vast amount of possible analyses for complex sentences, which calls for a high-performing *ambiguity management*. This might be the reason that - to our knowledge - so far no massive experiments on unification-based lexical acquisition with large corpora have been reported (cf. Zhang et al. 2007: 153)[23].

Horiguchi et al. 1995

Horiguchi et al. (1995) sketch an HPSG system which was designed to learn lexical entries of Japanese content words. The motivation of this paper is the development of an observation module (which collects unambiguous examples), which is far more efficient than the one described in Brent (1991,1993). Section 1.3.4 has already shown that this account suffers from a low yield of possible examples and from the noise introduced by the analyzing components. Horiguchi et al. (1995) start their discussion by addressing these two principal problems. Moreover, they are concerned with the fact that *underlying* lexical features like verbal subcategorization may result in different

[23]Fouvry (2003) makes some suggestions how to cope with the ambiguity problem of lexical acquisition within the HPSG framework.

surface realizations caused by morpho-syntactic processes: 'The defect of his [Brent's] approach becomes more obvious when one applies this [cue-based extraction] method to languages like Japanese, in which the subcategorization frame of a verb goes through changes caused by, for example, suffixes attached to it.' (Horiguchi et al. 1995: 320).
The employment of a unification-based parser provides a straightforward solution to this. For a demonstration let us consider the following two Japanese sentences (examples 7 and 8 in their paper)[24]:

(6) hanako ga susi wo tukut-ta.
Hanako NOM sushi ACC make-PAST
Hanako made sushi.

(7) taroo ga hanako ni susi wo tukur-ase-ta.
Taro NOM Hanako DAT sushi ACC make-CAUS-PAST
Taro made Hanako make sushi.

The first sentence shows a canonical use of the transitive verb 'tukur-' ('to make'): the two arguments are realized on the sentential surface as nominative NP ('hanako ga') and accusative NP ('susi wo'), respectively. In the second sentence the verbal suffix '-(s)ase' introduces an additional argument - the causer - which appears as nominative NP. The agent of 'tukur-' is marked by dative case now.
In order to enable the acquisition system to learn about the possible SCFs of 'tukur-' efficiently, the grammar component should incorporate some knowledge about the causative suffix. The same applies to other argument structure changing suffixes of that language. The more the system knows in advance about these important closed class items[25], the more effective is the extraction of potential examples from corpora. One of the lexical entries for '-(s)ase' proposed by Horiguchi et al. (1995) is shown in figure 1.17. The SUBCAT feature of this suffix specifies the nominative case of the additional argument [6], which is semantically linked to the CAUSER role in the semantic (SEM) specification of the entry. The causee ('hanako' in the above example) is assigned dative case and is also semantically linked to the (canonically nominative) NP in the SUBCAT specification of the verb itself [4], while remaining arguments [3] ('susi' in the example) are passed through from the verb's SUBCAT to the SUBCAT list of the suffix.
During the syntactic processing of the above sentences, the parser would fill

[24]Note that the interlinear translation for 'tukur' in example (8) in their paper is mistakenly 'eat'.

[25]We want to use this notion as a cover term for function words and affixes.

in the missing information in the feature structure (here the SUBCAT list) of the unknown word.

The proposed system (see figure 1.18) first annotates the input sentence with the help of a PoS tagger. Function words are augmented with hand-made lexical entries and content words are given one of the (underspecified) lexical templates that are tailored for the corresponding part of speech. This annotation phase is now followed by a parsing step, which maps the input to the corresponding syntactic structure, as a by-product filling (by unification) the missing information in the tree nodes. The paper, however, is not very specific about what exactly happens with the output of the parser, but at least it is a good example how linguistically precise a-priori definitions of closed class items may increase the efficiency of lexical acquisition.

Figure 1.17: Proposed lexical entry for '-(s)ase' in Horiguchi et al. (1995)

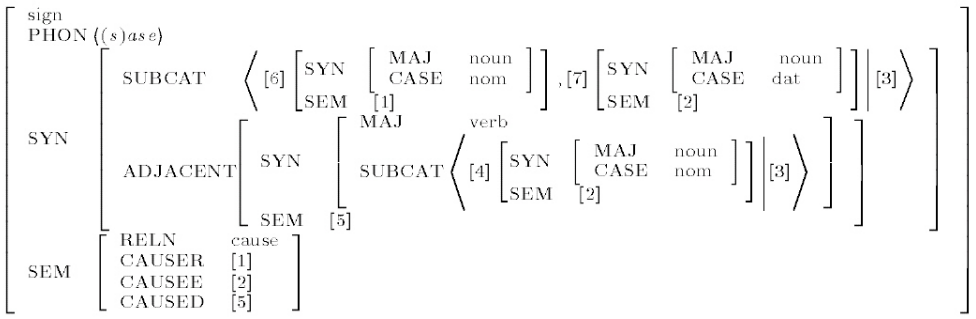
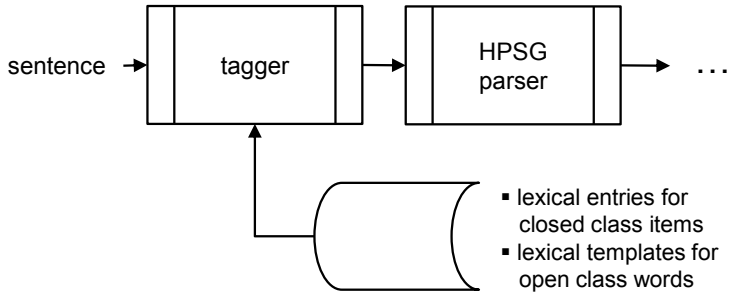


Figure 1.18: Process flow in Horiguchi et al. (1995)



1.3.7 Discussion

The last sections discussed different language acquisition strategies. For the sake of having at least some common aspects approaches have been chosen that mainly deal with the same language (English) and with the same linguistic phenomenon (verbal subcategorization)[26]. Other dimensions of lexical acquisition research - as discussed in the beginning of this chapter - varied, three of which should be addressed here.

1) Degree of linguistic precision This property is crucial for the comparison of the different strategies. The least precise approach is certainly the cue-based extraction method because it relies on regular expressions or finite state machines. PCFGs are able to reflect the recursive nature of language and supersede regular-pattern approaches with regard to linguistic precision, but they do not have the descriptional power of unification-based frameworks (cf. section 1.3.3). Deficiency in linguistic precision may be compensated by the use of very large corpora and sophisticated stochastic filters, but it is far from clear how large the input must be in order to satisfactorily make up for the loss of precision.

2) Degree of autonomy and embeddedness From the perspective of a language engineer who is targeting an NLP system that automatically learns knowledge missing in the lexical component, the cue-based approach is not

[26]For illustrations reference is sometimes made to accounts on other languages as well as other phenomena.

autonomous because all the cues have to be designed by hand. Further, the learning facility is fully isolated from any NLP system that is hoped to benefit from the acquired knowledge - called *in vitro* in Baldwin (2007) and related papers. Each result from a learning session is to be carried over to the target NLP system and from now on to be regarded as static knowledge until it gets replaced by the results of a new learning session. This fundamental problem has been emphasized in Briscoe's review of this type of research (Briscoe 2001). It is the problem of sparse data predicted by Zipf's (Zipf 1949) famous law: 'Zipf (1949) demonstrated that several distributions derived from natural language approximate to power laws in that the probability mass is distributed non-linearly between types with a few of the most frequent types taking the bulk of the probability mass and a very long tail of rare types. Both the unconditional distribution of valency frames and the conditional distributions of frames given specific predicates are approximately Zipfian [...] Two conclusions that can be drawn from this are: 1) that, because the power law is scaling invariant, any finite sample will not be representative in the statistical sense, and 2) that power law distributions are very often a clue that we are not sampling from stationary source but rather from a dynamical system [...] From this perspective it is not surprising that classical statistical models of learning, which rely on representative samples from stationary sources, do not perform optimally. A better model of a valency lexicon, given these observations, is of an adaptive self-organizing knowledge base which continually monitors data to update associations and the strengths of associations between predicates and valency frames.' (Briscoe 2001: 86-87). The limitations addressed here will certainly carry over to all kinds of lexical knowledge.

The situation with head-lexicalized PCFGs is much better because the acquired lexical properties - decoded in form of probabilities of rule applications - are inherent in the grammar and feedback of the learning results into the system appears to be straightforward. However, the incorporation of linguistically precise knowledge such as complex ontological hierarchies and other semantically motivated devices might prove difficult. Further, the development of a function for mapping of probabilities to lexical classes (if such an 'export' would be required for any other system or for evaluation purposes) is a manual operation similar to the design of cues.
The unification-based approach fully accounts for the principle openness of the lexicon, hence, a) learning of lexical features is integrated into the NLP system, b) the detection of learning opportunities is a natural by-product of the system's job and c) the 'cues' for unambiguous evidence are inherent in the grammars.

3) Evaluation of learning performance While the learning-on-the-job approach appears to be favorable with regard to the previously mentioned aspects, it is an experimental matter how it scales up (cf. Baldwin 2007: 153). The stochastically biased approaches have been tested on large corpora and the quality of the acquired knowledge measured in terms of precision/recall metrics, among others. Thus, the quantification of the unification-based learning-on-the-job approach is a missing piece in the map of lexical acquisition strategies (ibid.) and is emphasized in the present work.
But it must be noticed that the method of evaluation is not uncontroversial and Baldwin (2007: 152) claims that '[o]ne stumbling block in DLA research has been the lack of standardisation in evaluation [...]'.
The performance values of the various accounts are difficult to compare for a couple of reasons. Apart from different motivations, the published proposals also vary with regard to language type (e.g. fixed word order languages versus free word order languages), language source (heterogeneous versus homogeneous corpus, different corpus sizes), different linguistic phenomena (inflection, subcategorization[27], selectional restrictions, semantic classes etc.) coupled with different linguistic background theories and last but not least different and partly questionable gold standards. It even sometimes turns out that partial results of the acquisition procedure supersede the quality of the gold standard, from the perspective of the developer. All this does not mean that evaluation, as it has been carried out, is useless, but quite the contrary. Careful analysis of the resulting pieces of knowledge compared to what the developer has initially intended reveals the insight about shortcomings of the learning algorithm or parameters of the learning system. Rather, the point made here is to stress the problem of comparability. Hence, we will conclude this section with some final remarks on evaluation in general and the way we plan to measure the performance of our implementation.
Although the classic gold-standard-based way of evaluation will be pursued in this thesis, we want to outline the benefits of an alternative evaluation method that 1) accounts for the nature of the acquired knowledge being relative to the learning system and 2) minimizes the influence that different linguistic theories might have on the evaluation procedure. This can be illustrated with the help of figures 1.19 and 1.20. The first figure shows one possible method of evaluation that incorporates a gold standard. On the basis of linguistic theorizing, six sets of criteria are involved. First, there are some criteria to choose the language source. Second, there is a set of criteria

[27] Even within a certain phenomenon such as verbal subcategorization, there is a variation in quantity: Brent (1993) studies 6 SCFs, Manning (1993) 19 SCFs, Briscoe and Carroll (1997) 160 SCFs (!) etc.

for the parametrization of the acquisition system (e.g. which cues, which grammar, values of thresholds etc.). Third, in order to compare the acquired knowledge with a gold standard, this knowledge must somehow be extracted from the system on the basis of criteria set 3 (which is straightforward for cue-based extraction methods, but not necessarily for other approaches). The comparison step itself also depends on some criteria, for instance, for the determination of the sample set of words. Further, the gold standard has to be provided according to criteria such as: Which are the sources for the gold standard? How is information provided by these sources mapped to the gold standard so that a meaningful comparison with the acquired knowledge can be achieved? Finally, the information decoded in the sources has typically been accumulated by other people on the basis of some criteria in set 6.

Figure 1.20 outlines the basis of an alternative evaluation method. Clearly, nothing releases the LE from making decisions about the language source (criteria set 1), which is provided to the learning system which also has to be customized according to a second criteria set. Instead of using a gold standard, however, it might be advisable to measure the quality of the acquired knowledge more indirectly by comparing the behavior of the initial system with the performance of the system after learning. The evaluation of the system's performance is due to criteria set 3. Relying on 3 sets of criteria as opposed to 6 sets would minimize the influence of linguistic theorizing on the evaluation method dispensing with the direct judgment of pieces of acquired knowledge, instead one would measure its quality by quantifying how much the system's capabilities benefit from learning.
However, this kind of indirect performance measurement has a major drawback: it is extremely time consuming, especially in the bootstrapping phase where the lexicon is poor and the arising ambiguities are vast. For this reason, we will use the 'classic' gold standard approach acknowledging its deficiencies. The only exception will be the test reported in section 4.13 in which we compare the overall disambiguation performance of an updated model with the corresponding initial one.

Figure 1.19: Evaluation of acquired knowledge using a gold standard

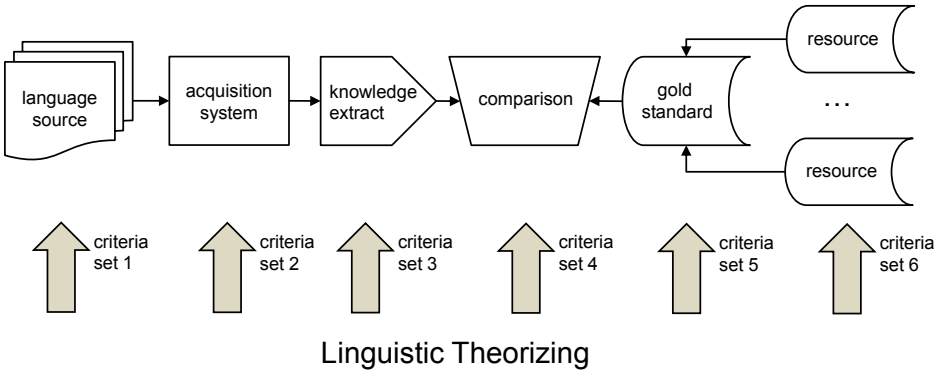
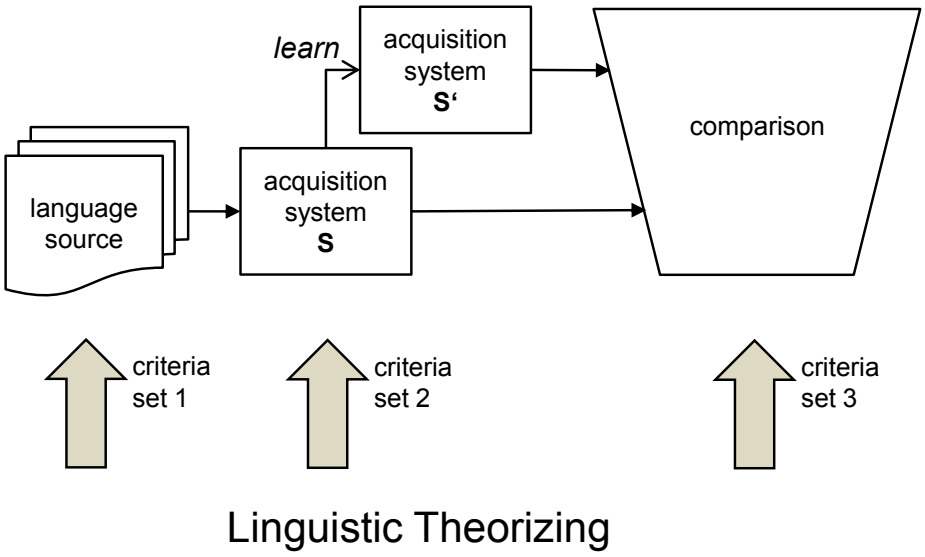


Figure 1.20: Indirect evaluation of acquired knowledge



1.4 The Learn−α Design Rule

This section aims to motivate the guiding rules underpinning our implementation of an NLP system that automatically induces lexical features from unrestricted text. This chapter started with the discussion of the language engineering bottleneck faced by the LE. His target is a system which is able to map English sentences to linguistically motivated structures that properly encode the relations between words and phrases of the input sentence. In his approach, he uses a symbolic representation of rules which operate on lexical features that enable the system to distinguish lexical classes having different properties. Emphasizing the symbolic character of his system serves to illustrate that the account here has a clear symbolic bias. So, for the sake of the argument, suppose that the LE does not have any probabilistic nor connectionist frameworks at his disposal[28]. Because of the inherent openness of the lexicon the NLP system has to be equipped with a component that deals with input tokens that can not be mapped to items in the current lexicon. For example, the system that is able to run the ERG (cf. section 3.3.2) already comes with an unknown word handling device which has been refined over the recent years (Adolphs et al. 2008). It will not matter for the argument which of the 'unknown words' will be candidates for new lexical items (recall the discussion of named entities in this context); what matters is that if they enter the lexicon then the system should be able to derive the corresponding lexical constraints *on the job*. If this is the case, why not use the very same device for the ongoing refinement of already existing lexical entries, ending up with a fully dynamic and autonomously learning lexicon? If this is possible, it would then be just a logical step to use the very same component in order to bootstrap from an initial seed lexicon. This is exactly what is at stake here.
How much the system has to be 'educated' before it can work productively is another question which is not in the scope of this section, rather the point here is the question what is *the logic behind such a learning system.*

1.4.1 Experience and Challenge

To make the discussion a bit more precise, a few notions are to be introduced that will be used throughout this work. First of all, because the notion 'sentence' is too narrow, it will be replaced by the term 'utterance'. Nothing should prevent the learning system from acquiring knowledge from word se-

[28] In chapter 2, we will motivate a statistical component.

quences in the input that make up linguistic entities other than just sentences. It is up to the LE how he wants his system to split up the input text into smaller parts, so from now on the input is considered as a sequence of utterances $< u_1, \ldots, u_n >$. On the time line, this sequence of utterances falls into two categories, as depicted by figure 1.21: the utterances that are already processed - the *experience* of the system - and future utterances, dubbed here as the *challenge*. Because the order of utterances in the input does not play any role in this work we will henceforth use the notion *experience set* when we refer to the set of utterances that the system experienced.

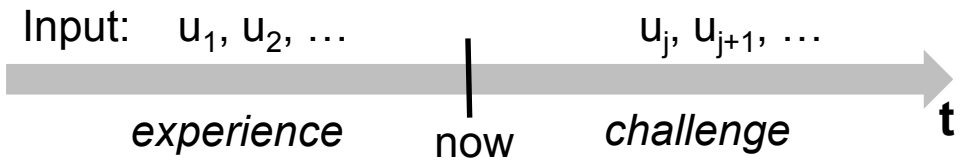


Figure 1.21: The sequence of utterances on the time line

The LE wants the system to benefit from its experience in order to better pass the challenge. But why should this be feasible at all? This is because of an essential aspect of language that *links* the utterances of the experience to those of the challenge. They all are made up of the *same* finite set of principles, rules and constraints that are usually ascribed to the grammar of the language.
Surely, grammar can change over time, but this is a fairly slow process which seems to be negligible compared to other factors such as noise. We therefore exclude diachronic change from further discussion fully acknowledging that it remains an unresolved issue. Also, the *openness* of the lexicon does not contradict the assumption that language extensionally can produce an infinite set of utterances by finite means. Lexical entries simply do not *belong* to this finite set, just the rules that structure the lexicon. In other words, unknown words in the challenge do not question the principle ability of an NLP system to benefit from its experience if it is assumed that the new words satisfy those constraints which the system already knows. As an example, one can expect that any new word belongs to (at least) one of the lexical types which the system knows in advance (cf. section 1.3.6).

1.4.2 A-priori Knowledge of the Learning System

The last point is crucial and it is important to mention that this thesis does not deal with systems that induce grammars or even grammar *and* lexicon (which would make extralinguistic evidence necessary). The grammar of the system is considered constant and as a part of this, the corresponding lexical structure is given a-priori, too, but the factual 'link' between experience and challenge does not help the system if it does not properly reflect the underlying grammar of the given language. So, if the LE encounters that the system does not learn efficiently or - in the worst case - that it does not learn at all, he will be forced to question the adequacy of its grammatical and/or lexical component. It also follows from this discussion, that a system can only learn about lexical phenomena to which it is sensitive. For instance, if a system does not exploit semantic properties of lexemes, because those properties are not encoded in its grammar, it will never learn them. On the other hand, the system has no *need* to learn semantic distinctions because they are not required for his task. This is what we dubbed as the 'relative' character of the lexicon at the outset of this chapter. The question remains whether a system that properly fulfills a certain task such as syntactic disambiguation does have a grammar at its disposal - which *enables* the system to perform the task - which is strong enough to serve as a proper link between experience and challenge.
In other words: is the 'enabling' knowledge sufficient for automatic enhancement of the knowledge? This thesis cannot provide a general answer to this question, but can provide some experience with a system prototype that reflects a broad range of grammatical facts of English and that tries to improve automatically. For the moment, it is a working hyothesis that in principle it is indeed possible to build NLP systems that have enough a-priori knowledge about a given language so that automatic improvement can take place.

1.4.3 The Logic of Artificial Lexical Acquisition

How does learning from experience take place in such a way that the acquired knowledge leads to further improvement? To seek for an answer, consider the requirements. First, the LE wants the system to increase its performance, at least with respect to certain utterances which the system would deal with in a 'better' way after learning has taken place, for example in reducing the number of possible parsing alternatives. This would not happen, if the system would either ignore the learning opportunities available during the experience phase, or gain some 'knowledge' that simply does not contribute to further improvement. Consider these criteria in more detail.

First, the enhancement of the system performance with regard to at least one utterance should not be at the price of the system's overall performance. This would be the case if the system would acquire knowledge that prevents it to successfully process an utterance that has been successfully processed before learning happened. Thus, it is required that the acquired knowledge must be *consistent* with the whole set of experienced utterances.
Consistency alone, however, might not be a sufficient condition. In addition a second criterion of the learning system must be fulfilled: the acquired knowledge should show some effect on analyzing the challenge. In other words, if the system would process the experience set once more after acquisition has taken place, there should be some effect. If not, how could we expect that the acquired knowledge shows some positive effect on the challenge? This claim could be subsumed under the headline *'effective learning'*, but the demand for effectiveness would still not be sufficient. That learning should be effective does not say anything about the precision of the system. In order to arrive at high-quality lexical knowledge that leads to a good performance on the challenge, a *maximum of precision* has to be demanded. A setting of a feature α can be consistent with a number of utterances but nevertheless not be true for a given lexeme (cf. section 1.3.6). Hence it is required that the system detects at least one utterance for which the setting of α is a *necessary condition* to be able to be processed successfully. Being a necessary condition, there is no room for an erroneous feature setting, at least as long as it is assumed that the utterances and the grammar are linguistically perfect. If no such utterance can be found because of systematic ambiguity, the system should be able to detect that circumstance and use statistic means to arrive at the best decision feasible.
A final requirement is that the system is able to decide whether it has seen sufficient input for the induction of a particular feature α of a particular lexeme. Ideally the system should arrive at an optimum in balancing out acquisition speed and recall[29].

In summary, the learning system should draw useful conclusions out of the experience set:

a. The system should acquire knowledge that is not indifferent to the input. Otherwise it could be said that it does not make intelligent use out of the experience. (This will be the weak version of Learn-α.)
b. The acquired knowledge must be consistent with the experience. Otherwise it could be stated that the system would have ignored the depen-

[29]This goes in line with Baldwin (2007) who argues for the importance of high recall.

dency between certain feature values and certain utterances. (Forming the strong version of Learn-α.)

In practice these requirements might result in conflicts and the system should somehow balance them out (cf. section 2.10). A system that widely refuses to fulfill these criteria could be called an 'ignorant' one (because it ignores the information provided by the experience).
But even if the system is not an ignorant one, this is no guarantee for not decreasing the performance when facing the challenge.

It may be that the experience was *not sufficient* to ensure consistency with future utterances. In this case the system must be able to *revise* its acquired knowledge, otherwise we will call it a 'stubborn' system. If a system is neither ignorant nor stubborn, there is reason to call it an *intelligent* system.
Now everything is in place to formulate a design rule for intelligent NLP systems - intelligent with regard to the capability to automatically improve by acquiring lexical knowledge on the job: the Learn-α Design Rule, given in (1.12).

(1.12)

The Learn-α Design Rule (preliminary)

The design of an NLP system shall enable it

i. to induce any lexical feature α from the experience set, if
   (a) the setting of α is **consistent** with the experience set
   (b) the determination of α's values is **not neutral** to the experience set (weak version)
   (c) the determination of a value of α is a **necessary condition** for the successful analysis of at least one utterance in the experience set (strong version)
   (d) it has seen a **sufficient number of utterances** to warrant high recall when processing the challenge set
ii. to **revise** the determination of a feature α if consistency with further input is not maintainable.

From the strong version of this principle it follows why the detection of unambiguous evidence in the input is crucial for the learning process.
Before considering this implication, an important assumption needs to be mentioned, namely the assumption that all utterances in the experience are grammatically correct. Although in section 1.3.4 it was already noticed that this is not always the case - first, real systems have to deal with some noise

in the input and second, there will always be some noise introduced by the analyzing component - let us assume for this discussion that we live in an ideal world, ensuring that the resulting analyses of utterances only comprise linguistically warranted structures (later on the theory will be enabled to deal with noise as well).

Assumption 1

a) The experience set of the learning system is a set of grammatical utterances.

b) The grammar of the system is sound.

It must be noted that the first item is problematic, not only because of its being unrealistic, but also because it somehow restricts the classes of NLP systems that can be equipped with the kind of lexical acquisition device developed here. An NLP system the aim of which is to distinguish grammatical from ungrammatical input simply cannot stick to this assumption, at least not if it is designed to learn the lexicon on the job. However, it might be 'educated' with a grammatical experience set (which is then to be called a training set) before it starts fulfilling its service.

Assume there is an utterance u mapped to a set S of structures resulting from processing u. If there is an utterance u in which a lexeme L is realized that is underspecified in a sense that the system does not know the admissible values of a feature α of L (the *setting* of α), how can this set of values be determined?
For simplicity consider a feature α that can take one of the two values a and b. The system can try to process the utterance u assuming a setting of $\alpha \in \{a, b\}$ for a given lexeme L, that means L admits α=a or α=b. The result of this tiny experiment obviously does not tell anything about whether L *really* admits a, b or both. Instead, the system should try to process the utterance u assuming a setting of $\alpha = a$ and find out whether the resulting structure set is empty or not. Afterwards it should try to process u under the assumption of $\alpha = b$. Let the resulting sets be labeled S_a and S_b, respectively. Note that at least one of the sets must be non-empty, otherwise u would be ungrammatical, independent of α, contradicting assumption 1. If S_a is empty (and S_b non-empty), this means that $\alpha = a$ leads to ungrammaticality and $\alpha = b$ does not. Hence, excluding $\alpha = b$ from the lexicon entry of L would contradict assumption 1 and would lead to inconsistency with the input, violating the strong version of Learn-α. If both sets S_a and S_b are non-empty, the utterance does not provide any ground for hypothesizing one of the feature values. In other words, the setting of α is indifferent to u. In this case, including $\alpha = a$ or $\alpha = b$ into the lexicon *based on u* would violate

the weak version of Learn-α.
If the utterance u provides evidence to infer the setting of α for a lexeme L, we call u a *learning opportunity* for $\langle L, \alpha \rangle$. If the system really exploits this learning opportunity to hypothesize a value v for $\langle L, \alpha \rangle$, we call u a *reference* for $\langle L, \alpha = v \rangle$.

1.4.4 Relative Degree of Ambiguity

The last subsection concluded that a feature setting that is indifferent to a given utterance u must not enter the lexicon on the grounds of u. However, it might be worthwhile to think about the information that is provided by an utterance with regard to a $\langle L, \alpha \rangle$, if, say, only a small portion of values would be admissible, small compared to the set of values that are defined for α. As an example, take a feature for which ten values $v_1, \ldots, v_{10}$ are defined. Ten experiments like the ones sketched out above might reveal that for u only two sets, say, S_{v_3} and S_{v_7} are non-empty. That means that one of only two values of ten is a necessity for the grammaticality of u. Furthermore, if experiments on a very large experience set reveal that, for instance, S_{v_2} and S_{v_9} are empty for every utterance, this would be good reason to exclude $\alpha = v_2$ and $\alpha = v_9$ from the lexicon.
Consequently, it is extremely valuable to assess 'how ambiguous' an utterance is with regard to $\langle L, \alpha \rangle$. This motivates a last notion that shall be introduced in this section: as the number of possible (non-empty) sets yields a measure for the degree of ambiguity of the utterance u relative to a feature α of a lexeme L, we refer to this value by the notion *Relative Degree of Ambiguity* (henceforth RDoA). This notion emphasizes the independence of two different aspects of ambiguity. The overall ambiguity of u simply does not matter. The system can perhaps produce a set of 100 possible structures for u. If all 100 structures reveal the same value for a certain feature of a lexeme (RDoA=1), then there is an unambiguous learning opportunity. So, what matters is the utterance's ambiguity with regard to a piece of missing knowledge of the system. A formal definition of RDoA will be provided in section 2.5.5.

Again, this can be a useful heuristic parameter of the learning system, due to the following reason: If there is a feature with n possible values and RDoA of some $\langle u, L, \alpha \rangle$ is r, then this proves that $n-r$ values are not true for $\langle L, \alpha \rangle$ in u. If $n - r$ is large, the information gained by making a disjunctive hypothesis like $\alpha = v_i \vee \ldots \vee \alpha = v_j$ (with r disjuncts in total) might be reasonably high. The system could benefit from this hypothesis because it excludes many values for α that are not true for L - at least in that particular

utterance.
In the current implementation, however, the principal demand is that RDoA=1 for all u, α, L, which implements the strong version of the design rule. This restriction will only be relaxed for features that are indeterminable otherwise (cf. section 2.5).

1.4.5 Why a Unification-based Grammar is a Good Framework for Learn−α

Finally it is worthwhile to point out why a unification-based grammar is a good platform for the implementation of Learn−α.
First, as outlined in section 1.4 and more thoroughly in section 1.3.6, missing lexical information can be straightforwardly represented by underspecified lexical entries. Further, if a free variable that represents missing knowledge of a feature α is entering a rule that binds this value to a constant c, this yields the same effect as if the system would have set $\alpha = c$ immediately after lexical look-up in order to try whether processing is possible under this assumption.
If all structures resulting from an underspecified feature setting lead to the same value c for α, this is the same as if the system would have tried $\alpha = c$ and subsequent processing would have 'survived' this decision.
The unification strategy frees the system from the computationally inefficient burden of trying out all possible values of all underspecified features in the utterance. To check whether a feature setting is a necessary condition (strong version of the principle), the system just has to check whether all resulting structures yield the same value for (α,w).
Additionally, condition i.b) of Learn−α in the weak version is also maintained easily: no binding of any feature will occur that is fully indifferent to the grammar.
At the beginning of this section it was said that the unification based framework only *partially* meets the conditions of learn-α. This is the case because consistency with the input is just maintained in accordance with *one* particular utterance. A word - or more precisely a word token - in the input can unambiguously show up with a certain value for a feature α but it may also unambiguously come up with another value for α after processing of another utterance. Thus conditions i.a) and ii) - consistency with the whole set of utterances - are not trivial to meet. This topic is, however, left for the second chapter.

1.5 Learning by Unification: some Illustrations

This last section of the introductory chapter will present a couple of examples that illustrate *learning by unification*. Although the observation of a *single* particular feature value realization is not sufficient for induction - as it will be outlined in the second chapter - it is the starting point of lexical acquisition as implemented in chapter three.
Consider the examples given in (8):

(8) a. She gave him a book.
b. She was given a book.
c. I looked up the entry
d. noble house wines

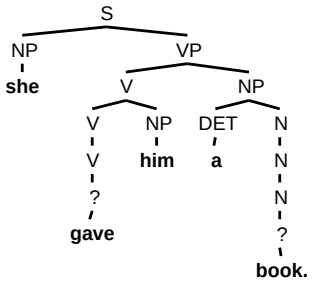
In the first two examples we are interested in the realization of valency and voice of the verb `give`. In the third example, the VPC entered by `look` is investigated and in the last example two feature value realizations are of interest: attributive usage of `noble` and part of speech of `house wine`. The last lexeme is completely unknown to the system employed here, so this is even a matter of *lexeme detection*. The first three lexemes are highly underspecified. Only the part of speech is known. The examples are parsed with the Linguistic Knowledge Building (LKB) system (Copestake et al. 1999b, Copestake and Flickinger 2000) using the ERG modified for our prototype. Many technical details are left unexplained here. They will become clear at the end of chapter three. The main point, nevertheless, can be demonstrated ignoring them for the time being. All figures shown here are screen shots of LKB windows.

1.5.1 Example 1 and 2: Verbal Valency and Voice

Example (8a) shows the rare property of having only one reading (figure 1.22) - despite the massive underspecification.

Figure 1.23 shows the partial chart related to this single reading. The tokens are given on the far left side along with their token positions. The numbers of related edges of the chart are displayed in square brackets. The strings in capital letters are the names of the lexical types, lexical rules or syntactic rules. Edge 322, for example, spans the whole sentence after application of the ERG rule 'SUBJH'.
The open-class lexical types are called LSEED entries. The concept of lexical seeds (LSEEDs) will be detailed in section 3.2. Lexical seeds combine all lexical features in a single structure. Consider the LSEED structure of `give`,

Figure 1.22: Learning by unification: Example 1, Parse Tree



shown in figure 1.24. All the system knows at that point is that the lexeme is a verb (type **lseedbasicverb**) with [LSSTEM < **give** >] for the orthographic representation and [LSPRED **_give_v_pred**] for the meaning of the lexeme. The value of feature LSPOS is predicted by the type and can be ignored. All other features in LSEED are completely underspecified, the values are just the most general values appropriate: LSVAL (valency), LSPP (subcategorized prepositions), LSSC (sentential complements), LSVOICE (voice), LSVPRT (verb particle) and LSCPSUBJ (+ if the verb allows sentential subjects). In this configuration the lexeme `give` accepts all verbal subcategorization patterns.

In the course of the derivation these feature values are unified with values that are specified in the corresponding lexical or syntactic rules. For technical reasons, the LSEED structure is stored under a feature called MAPPEDLSEED. Figure 1.25 shows the structure found in the feature structure related to the top edge number 322.
By looking at this structure the following lexical feature value realizations can be observed:

- ditransitive valency: [LSVAL **ls_ditrans**]
- no prepositional phrase: [LSPP **ls_pp_noprep**]
- no sentential complement: [LSSC **ls_sc_nosc**]
- no verbal particle: [LSVPRT **ls_no_vprt**]
- active voice: [LSVOICE **ls_act**]

Note that a further feature LSDATSHFT was added. This was introduced by the type **lseedditrverb**. Its value is + if the lexeme enters a *dative shift* construction, which is not the case in the example.

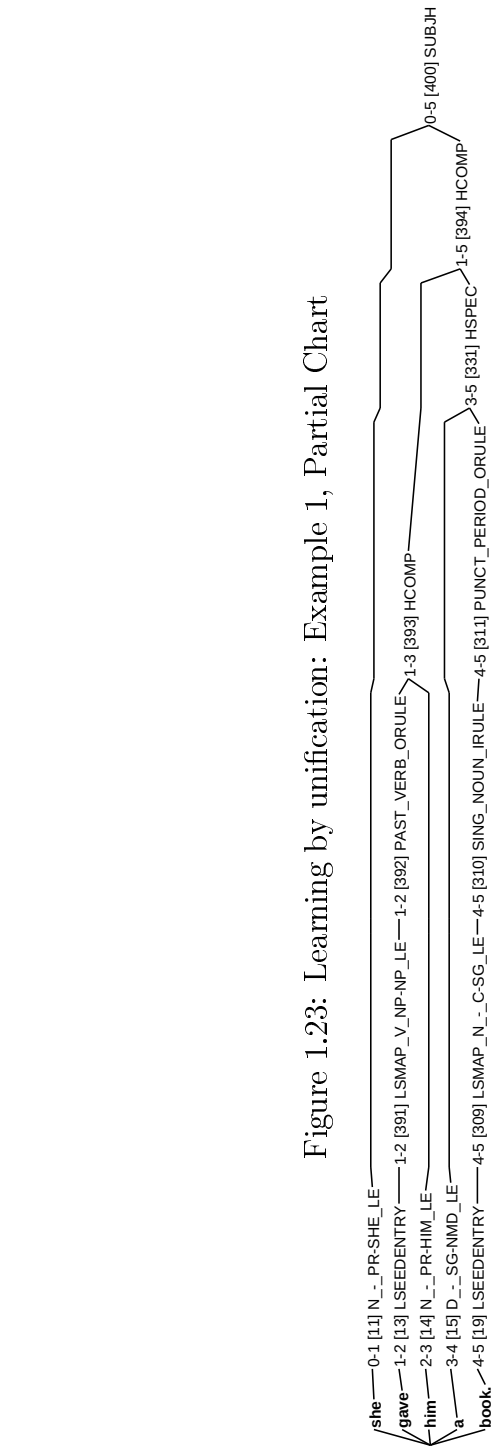


Figure 1.23: Learning by unification: Example 1, Partial Chart

Figure 1.24: Learning by unification: Example 1, LSEED Entry of `give`

```
Edge 13 P - FS

[lseedentry
 SYNSEM: synsem_min0
 --LSEED: +
 STEM: <0> = [*cons*
              FIRST: give
              REST: *null*]
 TOKEN: tok_min
 --LSGOV: ls_gov
 KEY-ARG: bool
 LSEED: [lseedbasicverb
         LSSTEM: <0>
         LSPRED: _give_v_pred
         LSPOS: ls_pos_v
         LSVAL: ls_val
         LSPP: ls_pp
         LSSC: ls_sc
         --LSMWE: bool
         LSVPRT: --ls_vpc
         LSVOICE: ls_voice]]
```

Figure 1.25: Learning by unification: Example 1, MAPPEDLSEED of `give`

```
FIRST: [arg123_relation
        PRED: <16> = _give_v_pred
        LBL: <7>
        LNK: *list*
        CFROM: 4
        CTO: 8
        MAPPEDLSEED: [lseedditrverb
                      LSSTEM: [*cons*
                               FIRST: give
                               REST: *null*]
                      LSPRED: <16>
                      LSPOS: ls_pos_v
                      LSVAL: ls_ditrans
                      LSPP: ls_pp_noprep
                      LSSC: ls_sc_nosc
                      --LSMWE: -
                      LSVPRT: --ls_no_vprt
                      LSVOICE: ls_act
                      LSSSUBJ: bool
                      LSDATSHFT: bool]
```

In the second example (8b) the verb realizes passive voice. There is again one reading only (cf. the tree in figure 1.26). While the LSEED entry is the same as in the previous example, the MAPPEDLSEED feature structure reflects the passive voice (figure 1.27). It is the *lexical rule* connected to edge 643 (cf. the chart in figure 1.28) that checks the feature value [LSVOICE **ls_pas**]. If the lexical entry were to specify [LSVOICE **ls_act**] (which is the case for verbs that do not passivize) then this lexical rule would not unify and thus not be applicable. The sentence would be ungrammatical for the system.

Figure 1.26: Learning by unification: Example 2, Parse Tree

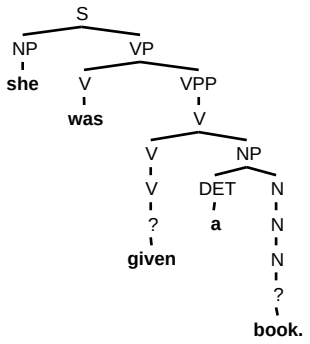


Figure 1.27: Learning by unification: Example 2, MAPPEDLSEED of `give`

```
FIRST: [arg123_relation
        PRED: <16> = _give_v_pred
        LBL: <7>
        LNK: *list*
        CFROM: 8
        CTO: 13
        MAPPEDLSEED: [lseedditrverb
                      LSSTEM: [*cons*
                               FIRST: give
                               REST: *null*]
                      LSPRED: <16>
                      LSPOS: ls_pos_v
                      LSVAL: ls_ditrans
                      LSPP: ls_pp_noprep
                      LSSC: ls_sc_nosc
                      --LSMWE: -
                      LSVPRT: --ls_no_vprt
                      LSVOICE: ls_pas
                      LSSSUBJ: bool
                      LSDATSHFT: bool]
```

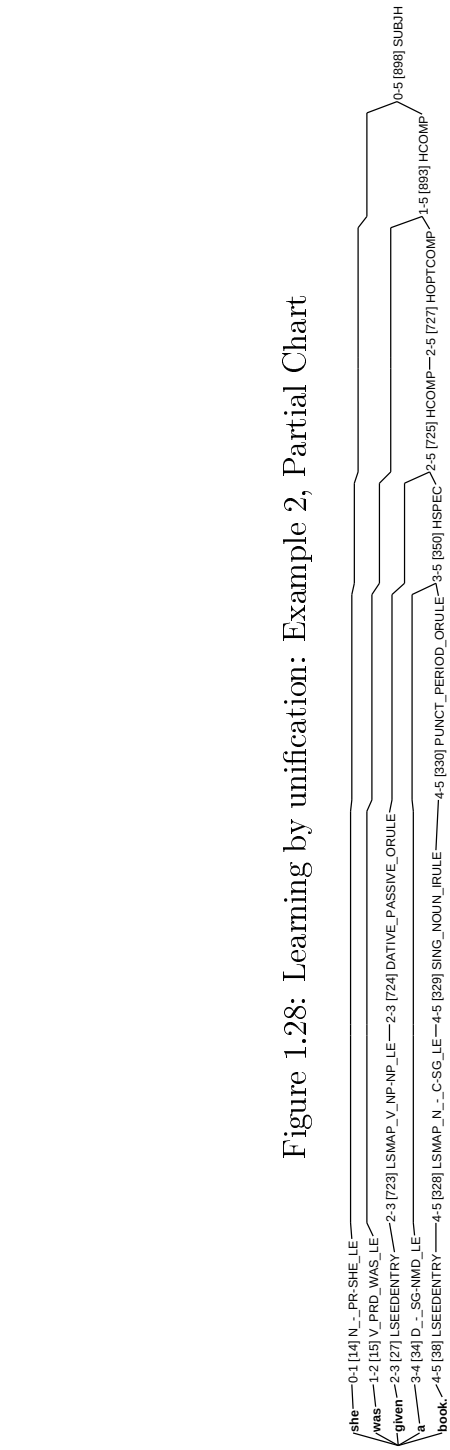
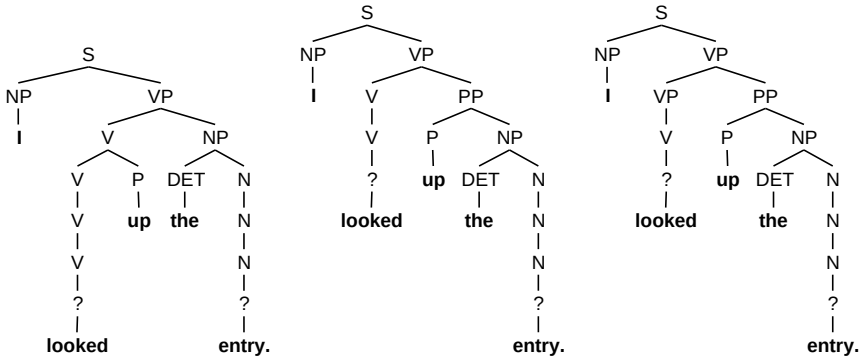


Figure 1.28: Learning by unification: Example 2, Partial Chart

1.5.2 Example 3: Verb Particle Construction

Example (8c) is ambiguous between at least three possible readings: either *up* is a particle with which `look` enters a VPC or it is a preposition heading a PP that either attaches to the verb, which means that it is subcategorized, or that attaches to the VP (figure 1.29)[30].

Figure 1.29: Learning by unification: Example 3, Parse Trees



The LSEED entry for `look` is similar to the one of `give` (figure 1.24), except for the features LSSTEM and LSPRED. Because the verb is realized in a different valency frame, the MAPPEDLSEED feature differs, as can be seen in figure 1.30: in the first reading the type of the MAPPEDLSEED feature structure becomes **lseedmtrverb_prt** indicating that the lexeme is a mono-transitive phrasal (VPC) verb. The value of feature LSVPRT is bound to the PRED feature value of the particle. The type introduces a new lexical feature LSPRTBEFN which is bound to + if the particle precedes the direct object[31].

The type of MAPPEDLSEED is different in the second reading. It is **lseeditrverb** which is used for non-phrasal intransitive verbs[32]. This time the LSPP feature value is bound to a structure specifying that the lexeme takes one PP with preposition *up*. The third reading shows yet another

[30] In the implementation there are more readings, for example the interpretation of the lexeme as a directional verb.

[31] LSPRTBEFN is the code for *particle before noun*, sometimes called *joined configuration*, e.g. in Baldwin (2005b: 4). This lexical feature controls the applicability of the ERG lexical rule *NP_particle_lr*, cf. Villavicencio and Copestake (2002: 359).

[32] The type introduces yet another lexical feature LSLOCINV for *locative inversion*.

MAPPEDLSEED structure. The type is the same as in the previous reading, but the LSPP feature is bound to **ls_pp_nopp** because the PP does not attach to the verb.
The question of whether the lexeme `look` is part of a VPC[33] cannot be unambiguously answered by virtue of this *particular* example, which shows how underlying lexical features may be unrecoverable from the surface forms of utterances, at least if one tries to recover them unambiguously. In section 2.5 we will motivate the notion *lexical neutralization* for this phenomenon. The ambiguity is covered by the different MAPPEDLSEED structures belonging to a particular utterance. Note that there might be features whose values can be unambiguously determined, as part of speech or voice in the example. If a learning system is expected to satisfy the Learn$-\alpha$ design rule, then it should exploit the resulting structures as much as possible. In section 2.8.1 we will propose a mechanism which will take care of that.

[33]The question can be rephrased to: is there a lexeme `look up`?, depending on how it is encoded in the grammar.

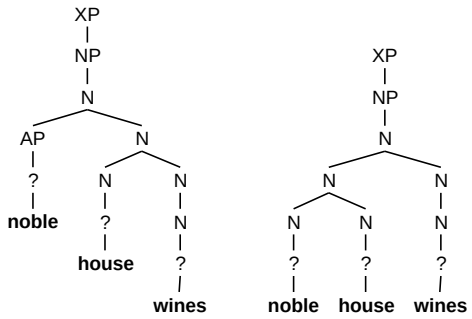
Figure 1.30: Learning by unification: Example 3, Reading 1-3, MAPPEDLSEEDs of `look`

reading 1 reading 2 reading 3

1.5.3 Example 4: Adjectives and Noun Compounds

The last example (8d) produces a lot of readings, from which two are selected for discussion (figure 1.31). In the first reading the two nouns `house` and `wine` combine to a new noun `house wine`. This noun is not available during *lexical lookup* (cf. the chart in figure 1.32) so it is an unknown multiword noun and becomes a lexeme candidate (whether it is a lexeme has to be decided during induction).
The MAPPEDLSEED structure of this compound is created by an ERG rule connected to edge 646: 'NOUN_N_CMPND'; it is of type **lseedbasicnoun** (figure 1.33).

Figure 1.31: Learning by unification: Example 4, Parse Trees



reading 1 reading 2

Figure 1.32: Learning by unification: Example 4, Reading 1, Partial Chart

Figure 1.33: Learning by unification: Example 4, Reading 1, MAPPEDLSEED of `house wine`

```
FIRST: [prep_notense_relation
        PRED: compound_rel
        LBL: <15>
        LNK: *list*
        CFROM: 6
        CTO: 17
        MAPPEDLSEED: [lseedbasicnoun
                      LSSTEM: [*cons*
                               FIRST: <16> = [*cons*
                                              FIRST: house
                                              REST: *null*]
                               REST: <17> = [*cons*
                                             FIRST: wine
                                             REST: *null*]]
                      LSPRED: unknown_noun_compound
                      LSPOS: ls_pos_n
                      LSVAL: ls_no_val
                      LSPP: ls_pp
                      LSSC: ls_sc]
```

In the second reading `noble` is realized as a noun which builds a compound with `house`. This compound in turn forms a larger compound with `wine`. Figure 1.34 shows the MAPPEDLSEED of this hypothesized noun.

Figure 1.34: Learning by unification: Example 4, Reading 2, MAPPEDLSEED of `noble house wine`

```
FIRST: [prep_notense_relation
        PRED: compound_rel
        LBL: <14> = [handle
                     INSTLOC: <13>
                     SORT: *sort*]
        LNK: *list*
        CFROM: 0
        CTO: 17
        MAPPEDLSEED: [lseedbasicnoun
                      LSSTEM: [*cons*
                               FIRST: <15> = [*cons*
                                              FIRST: <16> = [*cons*
                                                             FIRST: noble
                                                             REST: *null*]
                                              REST: <17> = [*cons*
                                                            FIRST: house
                                                            REST: *null*]]
                               REST: <18> = [*cons*
                                             FIRST: wine
                                             REST: *null*]]
                      LSPRED: unknown_noun_compound
                      LSPOS: ls_pos_n
                      LSVAL: ls_no_val
                      LSPP: ls_pp
                      LSSC: ls_sc]
```

The two alternative readings show how neutralization can already begin at the point of lexeme determination, for which we will motivate a separate notion in the theoretical part (paragraph 2.5.3 on page 144).
Apart from this ambiguity, the example is a potential learning opportunity

for attributive usage of the adjective `noble` whose LSEED is presented in figure 1.35 The lexical feature LSATTR specifies whether the adjective can be used attributively or predicatively, respectively. In the LSEED the value of this feature is the most general, namely attributively *and* predicatively.

Figure 1.35: Learning by unification: Example 4, LSEED Entry of `noble`

```
Edge 9 P - FS

[lseedentry
 SYNSEM: synsem_min0
 --LSEED: +
 STEM: <0> = [*cons*
              FIRST: noble
              REST: *null*]
 TOKEN: tok_min
 --LSGOV: ls_gov
 KEY-ARG: bool
 LSEED: [lseedbasicadj
         LSSTEM: <0>
         LSPRED: _noble_a_pred
         LSPOS: ls_pos_a
         LSVAL: ls_val
         LSPP: ls_pp
         LSSC: ls_sc
         --LSMWE: bool
         LSDEGREE: ls_degree
         LSATTR: ls_attrprd]]
```

The MAPPEDLSEED structure of `noble` accounts for the fact that the adjective is used attributively here (figure 1.36).

In summary, this section has shown how various linguistic phenomena (valence, VPC, voice, attributive usage of adjectives and noun compounds) are covered by one and the same account in the implementation, which will be presented more thoroughly in the third chapter. In this context the MAPPEDLSEED structure simplifies the harvesting of lexical feature values bound by unification. From *single* observations, however, no feature values can be reliably induced. So what is needed is a general theory of lexical feature value induction which is the topic of the next chapter.

Figure 1.36: Learning by unification: Example 4, Reading 1, MAPPEDLSEED of `noble`

```
FIRST: [adj_relation
        PRED: <14> = _noble_a_pred
        LBL: <15> = [handle
                     INSTLOC: <13>
                     SORT: *sort*]
        LNK: *list*
        CFROM: 0
        CTO: 5
        MAPPEDLSEED: [lseednormaladj
                      LSSTEM: [*cons*
                               FIRST: noble
                               REST: *null*]
                      LSPRED: <14>
                      LSPOS: ls_pos_a
                      LSVAL: ls_xarg
                      LSPP: ls_pp_noprep
                      LSSC: ls_sc_nosc
                      LSDEGREE: ls_degree
                      LSATTR: ls_attr]
```

Chapter 2

Theory of Learn$-\alpha$

This chapter develops a theory of artificial lexicon acquisition which will support the requirements of a learning system outlined in section 1.4. The roadmap is as follows: section 2.1 will define the structure of lexical knowledge with a level of abstraction that is appropriate for the proposed theory. The perspective of the language engineer, an indispensable building block of the theory, will be introduced in section 2.2, which will also lay out the model-theoretic part. The core of the theory is presented in section 2.3 which deals with observation, induction and the quantification of evidence. Sections 2.4 and 2.5 show how the major hurdles of lexicon acquisition, noise and systematic ambiguity, are accounted for. Further attention is turned to multiword expressions in terms of lexical associations in section 2.6 and lexeme detection in section 2.7. Section 2.8 explicates how the learning system arrives at concrete decisions based on statistical inference. In this context the third obstacle, namely data-sparseness will be briefly discussed. The revision of those decisions that cannot be maintained in the light of further experience is discussed in section 2.9. The chapter ends with the revised version of the Learn$-\alpha$ Design Rule in section 2.10 which then incorporates the findings discussed up to that point.

2.1 Lexical Knowledge

When a natural language processing (NLP) system produces or when it analyzes an utterance of a particular language, what does it know about the lexical items the utterance is made of?[1] Likewise, when an NLP system with

[1]The notion 'NLP system' may appear too broadly used here. In the context of Learn$-\alpha$ it mainly refers to a class of computer programs that include a parser and eventually a component that is able to generate sentences from logical form. In most of the text

an *in vivo* lexical acquisition component acquires a lexical item, what kind of knowledge does it gain? Both questions are related and both are crucial for a theory of lexical acquisition, of which the theory of Learn$-\alpha$, a theory of artificial lexicon acquisition, is an instance. These questions make up the topic of the current section, which constitutes the groundwork for the present chapter.
As it has become a standard in linguistics to use 'features' for the formal description of linguistic entities, it is obvious that, whatever lexical items are believed to be, it is reasonable to state their properties in terms of features. This is why the starting point here will be a discussion of the nature of features in subsection 2.1.1. Subsection 2.1.2 will then discuss the characterization of *lexical* features and their employment in lexical entities: lexical tokens and types. The fact that these entities stand in a hierarchical relationship, e.g. transitive verbs are verbs, verbs are open class lexemes etc., will be captured by the notion of lexical type hierarchy, as outlined in subsection 2.1.3.
In subsection 2.1.4 the notion of *symbolic language model* will be coined, which is intended to encode the linguistic knowledge of an NLP system and which is subject to change during the process of lexical acquisition. This facilitates a definition of what it means to know a lexical feature value (subsection 2.1.6).

2.1.1 Features and Feature Structures

This subsection is dedicated to aspects of the notion 'feature' that are worthwhile discussing here although this notion is in the 'flesh and blood' of every linguist. As the theory of Learn$-\alpha$ intends to be a theory of the *logic of lexical acquisition* it is fully embedded in first-order logic. Moreover it will be provided with a modal theoretic foundation later on in section 2.2.4. This makes it necessary to review how features are rooted in logic, thereby anticipating general properties of induction. Not giving a precise account of what 'induction of lexical features' is intended to mean can otherwise give rise to misunderstandings.

The problem of induction is (of course) not restricted to linguistic features, but to features in general such as those that allow to formulate that swans are white: Suppose a first-order logical statement has to be formulated expressing the (incorrect) hypothesis that all swans are white. The statement has to select predicates from an inventory of concepts such as a

such a program is simply referenced by 'the system' or sometimes 'the learning system'.

unary predicate Swan$x$ and a 2-ary predicate Colorxy, which relates, say, physical objects to colors. The hypothesis can then be stated as:

$$(\forall x)\ \text{Swan}x\ \supset\ \text{Color}x\mathbf{white}$$

Note that a particular member from the set of colors, typeset in bold font, was picked out. Given the axiom that no individual can have two different colors ($(\forall x)\ (\forall y)\ (\forall z)$ Colorxy & Color$xz\ \supset\ y = z$), the formulation of the hypothesis entails that swans cannot be red, for example. The axiom states that the relation Colorxy is a function. As this is the main criterion for features[2], the predicate Color is indeed a *feature*.

The inventory of concepts makes up the ontology in which the hypothesis is embedded. Applying Quine's famous slogan that 'to be is to be the value of a variable' (Quine 1953: 15) to the example one cannot believe in the hypothesis without believing in individual swans and individual objects that can have a particular color and in individual colors (like **white** and **red**). So with each introduction of a feature some ontological overhead is bought in. As a matter of ontological hygiene it is assumed that the set of entities that a given predicate is applicable to is always specified. So it is with the set of values a feature can take (which is normally called the range of a function). We will from now on use $V(F)$ to denote the set of appropriate values of a feature F.

A-priori Knowledge, Underspecification and Induction of Features

The above example can be used to illustrate that the theory of Learn$-\alpha$ is not intended to make a system learn, for example, which features are appropriate for which kind of object or which values are appropriate for which feature or which features have to be added to the universe of concepts. It is also not intended to learn statements like $(\forall x)\ (\exists y)$ Swan$x\ \supset$ Colourxy. All this is dedicated to the realm of a-priori knowledge.
But the learning system should acquire - by observation - reasonable constraints like

$$-(\exists x)\ \text{Swan}x\ \&\ \text{Color}x\mathbf{red}$$

- constraints which are consistent with the system's experience. Similarly it should learn *admissions* in the sense that swans 'admit' being white based on the observation:

[2]Indeed, the formula $(\forall x)(\forall y)(\forall z)(\text{F}xy\ \&\ \text{F}xz\ \supset\ y \doteq z)$ is one of the three axioms of a first-order feature theory that Backofen and Smolka (1993) demonstrate to be complete.

$$(\exists x)\ \text{Swan}x\ \&\ \text{Color}x\mathbf{white}$$

The same will hold for lexical items (cf. subsection 2.2.6).
Furthermore, it is crucial to see that the *absence* of the statement

$$(\exists x)\ \text{Swan}x\ \&\ \text{Color}x\mathbf{c}$$

does not mean that the system shall infer:

$$(\forall x)\ \text{Swan}x\ \supset\ -\text{Color}x\mathbf{c}.$$

If both clauses $-(\exists x)\ \text{Swan}x\ \&\ \text{Color}x\mathbf{c}$ and $(\exists x)\ \text{Swan}x\ \&\ \text{Color}x\mathbf{c}$ are missing and cannot be inferred, then this at most indicates that the color of swans is unknown. We say that the feature Color is *underspecified* for swans[3]. Negation $(-(\exists x)\ \ldots)$ and underspecification are different things and we are tempted to say that it is underspecification that allows the system to assume any color for swans if needed for the interpretation of a given experience. Note that the system of natural deduction always allows to add a statement like

$$(\exists x)\ \text{Swan}x\ \&\ \text{Color}x\mathbf{c} \vee -(\exists x)\ \text{Swan}x\ \&\ \text{Color}x\mathbf{c}$$

to the premises of the system's process of logical deduction. It might then determine which of the terms 'survives' reasoning when faced with a swan of color **c**.
Learning as it is conceptualized in this work is the supplementation of a knowledge database by adding clauses like $-(\exists x)\ \text{Swan}x\ \&\ \text{Color}x\mathbf{c}$ which are consistent with experience. How this knowledge is exploited when faced with further experience is another question. Here the position is adopted that based on induced knowledge a system does not recognize a red swan as long as there is a valid alternative for classifying the object in front of it.
For reasons of compactness consider that the knowledge database can be such that all the particular clauses about the (un)observed colors of swans (admissions and constraints) are put into one all-quantified statement like

$$(\forall x)\ \text{Swan}x\ \supset\ (\text{Color}x\mathbf{white} \vee \text{Color}x\mathbf{black})$$

[3]The concept of underspecification firstly arose in phonology and has been adopted by some semantic theories where underspecification in the lexicon allows for contextual instantiation of semantic features leading to contextual disambiguation. In morphology, underspecification is sometimes used to describe inflectional paradigms and syncretisms in a systematic and economic manner, e.g. Wunderlich and Fabri (1995). We are using the concept of underspecification to model unknownness. Here, it is the target of lexical acquisition to instantiate underspecified lexical features based on experience and to store these instantiations in the lexicon.

The disjunction represents the system's knowledge about swans *with respect to the feature Color*. It is the result of a process that the notion 'induction of features' refers to in this work. For completeness of the discussion some more background on the applications and important aspects of features is provided in the appendix, section B.1.

Feature Theory

This paragraph turns from the logical perspective of using features to encode knowledge to the more technical definitions that make up the building blocks of feature theory. These comprise the definitions of feature signature, feature system and feature structure, cited from Rounds (1997) and slightly adapted to the notation used throughout this thesis. We think that these are the most general definitions possible that capture the notion of feature as it is used in the context of the present work.

Def. 1 *Let L be a set of* **feature names**, *and A be a set of* **sort names**. *We call the pair $\langle L, A\rangle$ a* **feature signature**.

Def. 2 (Feature System) *A feature system of signature $\langle L, A\rangle$ is a tuple*[4]

$$\mathbf{FS} = \langle D, \{f_l\}_{l\in L}, \{D_a\}_{a\in A}\rangle,$$

where for each feature name l, f_l is a partial function on D, and for each sort name a, D_a is subset of D.

In the context of linguistic theorizing, a feature system contains the domain D, the set of entities the linguist is going to describe and is using for descriptions. The domain is partitioned into subsets according to concepts that are called sorts. In a word, D and all its subsets entails the ontological commitment[5] of the linguistic theory described in terms of a particular FS. Note that particular orderings of sorts are left to the particular theory. Nothing, however, prevents a set D_a to fully comprise another set D_b (of sort b), becoming an instance of subsumption. Each feature f_l is a partial function $f_l\colon D \to D$, partial because the feature is not necessarily defined for each entity in D.

[4] We will use **FS** to denote a feature system. Rounds (1997) uses the letter $\mathcal{A}$.

[5] The discussion of the ontological commitments of linguistic theories is not a mere theoretical artifact. It has recently found practical application in the field of ontology engineering within the Semantic Web. For an ontology of HPSG expressed in the web ontology language OWL, see Wilcock (2007) on top of GOLD (General Ontology for Linguistic Description - http://linguistics-ontology.org/) set out in Farrar and Lewis (2005).

Before we come to the final definition of feature structures, we have to remark that for function symbols f, *df* is a common notational variant of $f(d)$. The up- and down-arrows (*df* $\uparrow$ and *df* $\downarrow$) are used to express that f is undefined at d or defined at d, respectively.

Def. 3 *A subsystem of a feature system* **FS** *consists of (i) a subset E of* $D^{\mathbf{FS}}$ *such that if f is a feature,* $d \in E$*, and df* $\downarrow$*, then df* $\in E$*, and (ii) subsets* E_a *of* D_a *such that* $E_a \subseteq E$*.*
We also define the principal subsystem *P(d) generated by an element of* $D^{\mathbf{FS}}$*. The domain E(d) of this subsystem is the set* $\{d\pi \mid \pi \in L^* \;\&\; d\pi \downarrow\}$*, and* $E_a = D_a \cap E(d)$*. A feature system is* point generated *or* principal *if* $D = P(d_0)$ *for some* $d_0 \in D$*.*
Another official name for such a system is feature structure*.*

According to this definition, a feature structure picks out entities from the feature system's domain such that going out from a given entity d which 'generates' this structure, all entities to which a path from d is defined are part of the structure as well (and implicitly all the paths coming with them). In other words, a feature structure is such that all entities are connected to a common entity via feature paths. This entity is usually called the *root* of the feature structure.
This subsection concludes with a few remarks on the logical first-order description language we use henceforth. Unary predicate symbols are employed for sorts in the sense that Ae is true if and only if $e \in D_A$. Quantifiers can bind variables that replace individuals as in $(\exists x)\ \mathrm{A}x$. Features and relations in general are written as binary predicate symbols. Particular features (taken from a presumed signature) are written in small caps, constants (and types in typed feature logic) in bold font (FEATUREx**v**) - except phonological matrices. Outside logical formulas, we write features in lower letters. Logical operators are & (and), $\vee$ (or), $\supset$ (conditional), and - (negation) with the usual denotations and operator precedences. The logical description of figure 2.1, for example, would be:

$$\begin{array}{ll} (\exists x)\ (\exists y) & \mathrm{Word}x \;\&\; \textsc{sem}x, \textsc{car} \;\& \\ & \textsc{spell}xy \;\&\; \mathrm{Spell}y \;\&\; \textsc{phon}y, /ka\text{:}/ \;\&\; \textsc{graph}y\mathbf{car} \end{array} \tag{2.1}$$

Figure 2.1: Attribute Value Matrix (AVM) Representing a Feature Structure

$$word\begin{bmatrix}\text{SPELL} & spell\begin{bmatrix}\text{PHON} & /\text{kɑː}/\\ \text{GRAPH} & \textbf{car}\end{bmatrix}\\ \text{SEM} & \textbf{CAR}\end{bmatrix}$$

The symbols ϕ and ψ denote logical formulas and $\boldsymbol{\phi}^{\sigma}$ denotes a feature structure whose root is of type (or sort) σ. The superscript σ is omitted if it can be inferred from the context.

2.1.2 Lexical Features, Tokens and Types

An utterance u is empirically given by a chain of sounds or graphical symbols (or the physical sensation caused by it) and a *context* in which u is uttered. Focusing on written utterances, the context may be a particular text and a particular position[6] in the text. The feature POSITIONx**v** may be used to identify unequivocally the context of an utterance using some appropriate coordinates. From this it follows that POSITIONx**v** is thought to be an *extensional* feature[7]. For example, in our implementation, every utterance is assigned a unique ID, displayed throughout this work in the format ⌊TID:SID⌉, where TID is the unique identifier of the text and SID the identifier of the utterance unique within the text. All other features shall be interpreted intensionally. A feature structure that specifies POSITIONx**v** is pointing to a particular linguistic object which is then called a *token*, otherwise it denotes a set of tokens called a *type* (not to be confused with the type in feature structures).
Following Saussurean thinking, each utterance that is part of a natural language L conveys some meaning. If the meaning of an utterance u is compositional in the Fregean sense, it can be predicted from the meanings and structural relationships of smaller units of u by the application of *rules*. The set of rules that allow for the prediction can be called a linguistic theory Θ (of L). Ontologically, Θ uses the concepts provided by a given feature system, for instance the set of utterances D_u. Let us assume that from Θ we

[6]Position does not necessarily mean just a tuple (x,y) with x and y being character positions that encode beginning and end of a character stream. It is more complicated because tokens might be discontinuous strings.

[7]For the notion of extensional features cf. section B.1. We treat texts as types, so that copies of the *same* text do not double the number of lexical tokens.

can deduce a linguistic description for each grammatical utterance. Let us call this description a **u-model** μ of the utterance. μ is an entity of a set D_M (provided by the feature system used by Θ) and is intended to model one particular interpretation of u as it is licensed by L. Let M be the relation $M \subseteq D_u \times D_M$ such that $\mathrm{M}u\mu$ is true if and only if μ is an admissible interpretation of u. Consequently, it can be assumed that the u-model is deducible from Θ if u is grammatical: $\Theta \vdash \mathrm{M}u\mu$.
The theory can be used to single out the grammatical utterances $U^+ \subseteq D_u$ such that

$$U^+ = \{u \quad | \quad \Theta \vdash (\exists x)\ \mathrm{M}ux\} \tag{2.2}$$

The u-model can be described by a feature structure $\boldsymbol{\phi}^M$ that reflects some linguistic knowledge about the particular interpretation of u[8].
If Θ is used to split u into smallest parts as long as u's meaning can be predicted from these, one arrives at those entities that can be called *lexical tokens*. The removal of the POSITIONx**v** feature from these tokens then leads to the lexical types $x \in D_{lex}$ the utterance (or more precise: a particular interpretation of it) is made of. We will call these types *lexemes* and we will use the symbol $\mathcal{L}$ for the corresponding sort predicate. The properties of lexemes of a language L are described in terms of feature structures $\boldsymbol{\phi}^{lex}$, the set of which is usually called the *lexicon* of L. Because the analysis of utterances into smaller and even smaller parts is a recursive process, it is sometimes stated that lexemes ground the recursion inherent in the language's grammar[9]. We will call a feature specified in a lexeme's feature structure $\boldsymbol{\phi}^{lex}$ a **lexical feature**. In other words, lexical features are those features that are appropriate for the sort *lex* and all its subsorts.

Relativity of the Lexicon: the Black-Box Model

A theory of artificial lexical acquisition that takes the relativity of the lexicon seriously (cf. section B.3) shall not make any claims about the *structure* of lexical entries, a notion it should rather dispense with. Nor should it rely on details of particular grammar formalisms, which is why we have to figure out the least common denominator of the above mentioned theories

[8]Note that the information content in $\boldsymbol{\phi}^M$ is independent of the particular nodes (or entities) in $\boldsymbol{\phi}^M$. All that is needed to grip the content is the set of path values and the set of structurally shared paths, cf. the notion of *abstract feature structure* in Carpenter (1992).

[9]'Generally speaking, a grammar in the frameworks of GB, LFG, GPSG and early versions of HPSG includes a way to license constituent structure and a lexicon licensing the words grounding the recursion.' (Meurers 2001)

(as prominent representatives of linguistic theories) with respect to the lexicon. It will not matter, for instance, whether a theory labels a subset of its generalizations 'lexical rules', stating that these are 'in the lexicon'. We will treat all generalizations except the abstraction that relates lexical tokens to lexical types as part of the grammar.
So what is common to all theories? How can lexical knowledge be described without capturing the details and tenets of a particular theory of language? We think that the only way to establish a theory of Learn$-\alpha$ that is independent of particular grammatical theories is to treat the linguistic theory as a 'black box' leaving over the set of its logical consequences as the only path of access. Whatever the particular details are, one thing is certain: given a closed-world assumption, the omission (or change) of one of the lexical entries above logically alters the relation M. For instance, let u be the utterance *John walks* and μ one of the predicted interpretations of u deducible from the theory Θ ($\Theta \vdash \mathrm{M}u\mu$), then the omission of the lexical type for *walk* (or *walks*) resulting in a theory Θ' (all else being equal) that renders $\mathrm{M}u\mu$ undeducible and even more - because of the closed-world assumption - the set $\Theta' \cup \{\mathrm{M}u\mu\}$ is inconsistent. Let us say that ϕ_{walk} is the logical description of the lexical entry's feature structure. Then it makes a difference to M whether $\Theta \vdash (\exists x)\ \phi_{walk}x$ or not. Consequently, no more is required than a formulation whether a logical statement about a lexeme is entailed by a theory (or its lexicon) or not.

Flat Lexical Feature Structures

For ease of exposition of the theory nested feature structures in lexical descriptions are avoided. This means that all values of lexical features are atomic. Such a structure might be called *flat*. If intended, more complex lexical descriptions can be projected from these flat feature structures. As an example, the feature structure in figure 2.1 shall be recast to:

$$lex\begin{bmatrix} \text{PHON} & /\text{kɑ:}/ \\ \text{GRAPH} & \texttt{car} \\ \text{SEM} & \mathbf{CAR} \end{bmatrix} \tag{2.3}$$

The logical description is straightforward:

$$\Theta \vdash (\exists x)\ \mathcal{L}x\ \&\ \text{GRAPH}x, \texttt{car}\ \&\ \text{SEM}x, \texttt{CAR}\ \&\ \text{PHON}x, \texttt{/kɑ:/} \tag{2.4}$$

As a more complex example, let us express that this lexeme is a count noun (and only a count noun):

$$\begin{array}{lll} \Theta \vdash & (\forall x)\ \mathcal{L}x\ \&\ \text{GRAPH}x, \texttt{car}\ \supset\ \text{CAT}x, \texttt{noun} \\ \& & (\forall x)\ \mathcal{L}x\ \&\ \text{GRAPH}x, \texttt{car}\ \&\ \text{CAT}x, \texttt{noun}\ \supset\ \text{COUNT}x, \texttt{+} \end{array} \tag{2.5}$$

This does not impose any restriction on the implementation but simplifies the formulation of the theory. Note the omission of the PHONx**v** feature in this formula. being concerned with written input only, the feature will be henceforth ignored. This means that a lexeme is empirically given by its graphical representation. Let us further abbreviate GRAPHx**v** to Gx**v**. The sense-encoding feature will be schematically represented by Sx**s**.

2.1.3 Lexical Types and Lexical Type Hierarchy (LTH)

Suppose that the COUNT feature of the lexeme `car` would be underspecified. What are the conditions for the system to be able to induce it? At least, as part of its a-priori knowledge, it has to know whether COUNT is applicable to the lexeme or not. Note that since the theory Θ is a black box, it is not required that it comprises typing and appropriateness specifications. However, this has to be required at least for the lexical feature structures in order to ensure proper inference. Further, the system must know the possible values of the lexical features, as given in the set V(F) of every lexical feature f. Given a feature signature $\langle Feat, Type\rangle$, Carpenter (1992: 86) develops a definition of appropriateness that is adopted here:

Def. 4 (Appropriateness Specification) *An* appropriateness specification *over the inheritance hierarchy* $\langle$**Type**, $\sqsubseteq\rangle$ *and features* **Feat** *is a partial function Approp* : **Feat** $\times$ **Type** $\rightarrow$ **Type** *that meets the following conditions:*

- (Feature Introduction)
  for every feature $F \in$ **Feat**, *there is a most general type* $Intro(F) \in$ **Type** *such that* $Approp(F, Intro(F))$ *is defined*
- (Upward Closure / Right Monotonicity)
  if $Approp(F, \sigma)$ *is defined and* $\sigma \sqsubseteq \tau$, *then* $Approp(F, \tau)$ *is also defined and* $Approp(F, \sigma) \sqsubseteq Approp(F, \tau)$

It shall be assumed further that the lexical feature structures be *well-typed* in the sense of Carpenter (1992: 88). That means that they include only features that are appropriate for the respective type.

Assumption 2 *Lexical feature structures are of type* lex *and have to be well-typed.*

The type *lex* and all its subtypes will be called 'lexical types' and the inheritance hierarchy is called **lexical type hierarchy** (LTH) accordingly. It is important to distinguish the notion 'lexical type' from the notion 'type' in the context of lexical token/type distinctions (such as 'type precision/recall'). To avoid confusion the notion 'lexeme' is preferred in the latter context. It is assumed that the system properly infers the most general type of a lexical feature structure ϕ such that ϕ is well-typed. Take 2.6 as an example. Provided that the value of the category feature CATx**v** cannot be **noun** except for the type *noun*, type inference assigns *noun* to the feature structure:

$$\underset{???}{\begin{bmatrix} \text{PHON} & /\text{kɑː}/ \\ \text{GRAPH} & \texttt{car} \\ \text{CAT} & \textbf{noun} \end{bmatrix}} \rightarrow \underset{noun}{\begin{bmatrix} \text{PHON} & /\text{kɑː}/ \\ \text{GRAPH} & \texttt{car} \\ \text{CAT} & \textbf{noun} \end{bmatrix}} \tag{2.6}$$

Before coming to some illustration of the lexical type hierarchy, we want to motivate an important distinction between those features that influence type inference and those that do not. The idea to be captured is that some features are *essential* to lexical types while others encode *additional* information. The distinction allows to state that if a lexeme admits feature values $f_i(x) = v_i$ then the feature, say, f_j is defined, too. This will have an impact on the acquisition procedure: before a system can learn the verbal valency of *walk*, for example, it has to learn that it is a verb - otherwise verbal valency is not defined. We call those features that constrain appropriateness *type constraining features*:

Def. 5 (Type constraining feature) *Given is a lexical type hierarchy* $\langle lex, \sqsubseteq \rangle$ *and two subtypes of lex,* σ *and* τ *such that* $\sigma \sqsubseteq \tau$. *A* type constraining feature F *is a feature that meets the following condition:* $Approp(F, \sigma)$ *is defined and* $Approp(F, \sigma) \neq Approp(F, \tau)$

In a word, if the appropriateness specification of a feature F changes somewhere along the hierarchy, then F is a type constraining feature.
Consider the following example of a lexical type hierarchy. A prerequisite is the specification of some types that define admissible values for lexical features. In line with Carpenter, types are written in bold fonts:

$$\begin{array}{ll} i) & \textbf{boolean} \sqsubseteq \{+, -\} \\ ii) & \textbf{lexcat} \sqsubseteq \{\textbf{opencat}, \textbf{closedcat}\} \\ iii) & \textbf{opencat} \sqsubseteq \{\textbf{noun}, \textbf{verb}\} \\ iv) & \bot \sqsubseteq \{\textbf{nouninfl}, \textbf{verbinfl}\} \end{array} \tag{2.7}$$

In the type hierarchy, lexemes universally fall into two classes: the closed class (not extensible, for instance all function words) and open class to which at least nouns and verbs belong (if the language makes this categorical distinction, cf. Bisang 2011). Further, words differ with regard to its semantic content and are usually divided up to the two classes of content words and function words. The types **nouninfl** and **verbinfl** are intended to encode some inflectional property.
Let us further use the following features: *open* (+ if the lexeme is of the open class, - otherwise), *cont* (+ if the lexeme is a content word, - if it is a function word), *cat* (lexical category), *nflex* and *vflex* (encode inflectional properties of nouns and verbs, respectively) and *count* (+ if the noun is a count-noun, - otherwise). The type hierarchy is depicted in the format used by Carpenter (1992), the most general type at the bottom and appropriateness specifications written below the types (features in italics). The repetition of an appropriate specification is omitted if a type does not alter the specification of its supertype. Lexical types are underlined for better readability.

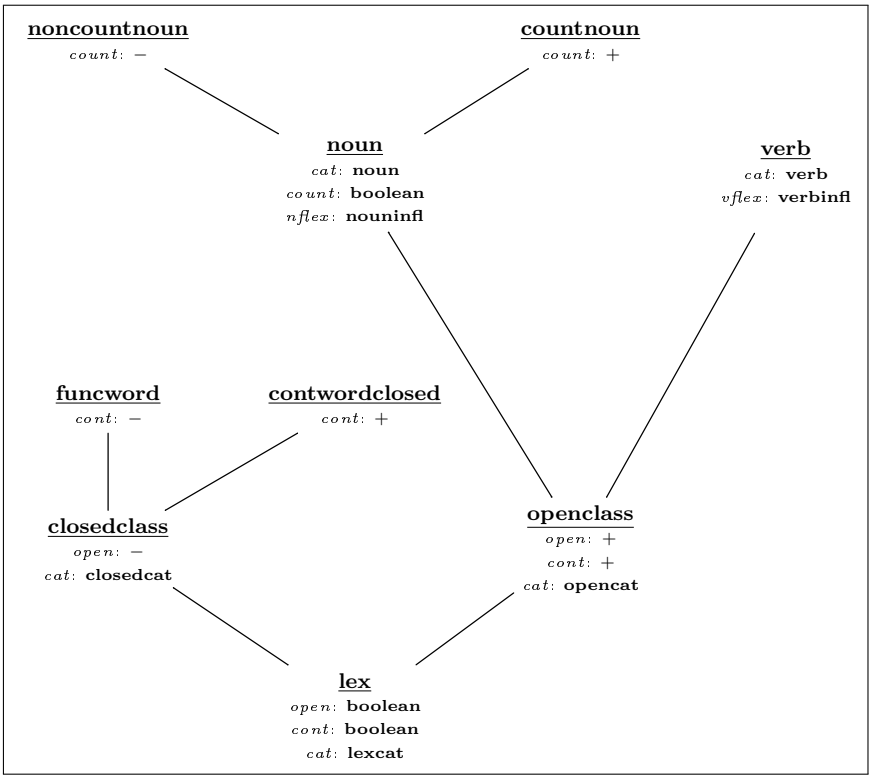
The application of definition 4 is illustrated by the following set of appropriateness specifications:

(2.8)

i. $\mathit{Intro}(\textsc{open}) = \textbf{lex}$
ii. $\mathit{Intro}(\textsc{cat}) = \textbf{lex}$
iii. $\mathit{Intro}(\textsc{nflex}) = \textbf{noun}$
iv. $\mathit{Intro}(\textsc{vflex}) = \textbf{verb}$
v. $\mathit{Intro}(\textsc{count}) = \textbf{noun}$
vi. $\mathit{Approp}(\textsc{open}, \textbf{lex}) = \textbf{boolean}$
vii. $\mathit{Approp}(\textsc{open}, \textbf{closedclass}) = -$
viii. $\mathit{Approp}(\textsc{open}, \textbf{openclass}) = +$

All features except NFLEX and VFLEX are type constraining features.

Figure 2.2: Example of a Lexical Type Hierarchy



2.1.4 Symbolic Language Models

Recall our concept of learning as inducing constraints like

$$-(\exists x)\ \text{Swan}x\ \&\ \text{Colour}x\mathbf{red}$$

based on observations of swans. In a similar vein, within the theory of Learn$-\alpha$, the learning system is supposed to induce constraints like

$$-(\exists x)\ \mathcal{L}x\ \&\ \text{G}x, \texttt{car}\ \&\ \text{CAT}x, \texttt{verb}$$

from the observation of lexical tokens of *car*. Hence the system must be enabled to *adapt* its theory Θ to a theory Θ' such that the constraint is deducible from Θ'. In order to capture the traditional division of linguistic knowledge into lexical and grammatical knowledge properly, we will from now on partition Θ into the sets $\Theta = \Gamma \cup \Lambda$ where Γ is thought to be the grammar and Λ is thought to be the lexicon of the system in the sense that all generalizations except the abstraction from lexical tokens to types are

covered by Γ and the only individuals about which Λ entails information are lexemes.
This allows one to state that Λ can be extended (and is inherently open) while Γ is fixed (as long as we are concerned with a theory that does not account for the acquisition of grammatical structures). Λ is open in the sense that there might be an extension Λ' of it that entails information about lexemes that are 'unknown' to Λ. Both sets pick out the required concepts and features from a given feature system **FS**. The feature system along with the grammar and the lexicon are intended to model aspects of language on symbolic, non-probabilistic grounds. This is why we call the triple a *symbolic language model* (or sometimes simply a model if there is no room for confusion):

Def. 6 ((consistent) symbolic language model) *Given is a feature system* **FS** *and two sets of logical statements* Γ *and* Λ *for a language* L *such that for each utterance* $u \in D_u$ *(the set of all utterances included in* **FS***) the set* $\Gamma \cup \Lambda$ *predicts zero or more linguistically motivated models of* u*:* $(\Gamma \cup \Lambda) \vdash Mu\mu$*. We also write:* $\mathbf{M} \vdash Mu\mu$*. Then we call the triple* $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *a symbolic language model. The model is said to be consistent if and only if* $(\Gamma \cup \Lambda)$ *is consistent.*

A symbolic language model allows the system to predict the u-models of an utterance by logical deduction. This logical entailment is called *analysis of an utterance*:

Def. 7 (analysis of an utterance) *Given is a consistent symbolic language model* $\mathbf{M}$ *and an utterance* $u \in D_u$*. The logical entailment of the set* $\mathcal{M}^u$ *of all u-models of* u *is called the analysis of u.* $\mathcal{M}^u = \{x \mid \mathbf{M} \vdash Mux\}$*. We write:*

$$\mathbf{M} : u \Rightarrow \mathcal{M}^u$$

Everything that can be known about the structure of an utterance is deducible from the language model. For example consider the utterance u that is graphically represented by `'my house'`. Then the language model might allow to deduce that the entity which is represented by `house` is the head of u:

$$\mathbf{M} \vdash (\exists x)\ (\exists y)\ Mux\ \&\ \mathrm{G}x,_{\texttt{my house}}\ \&\ \mathrm{G}y,_{\texttt{house}}\ \&\ \mathrm{Head}xy \tag{2.9}$$

The entity x in this formula is the u-model of u.

2.1.5 The Structure of Lexical Facts

In Learn$-\alpha$ the acquisition system is supposed to learn *lexical facts*. So it is necessary to give a precise account of what a lexical fact constitutes. Recall that the internal structure of the lexicon Λ is unknown. All that can be said is whether it entails a lexical feature description or not. Additionally these descriptions are assumed to be flat. Given a lexical type hierarchy and appropriateness specification, the system knows a-priori that a certain feature F$x\mathbf{v}$ is not defined for a lexeme as long as it is not positively specified for the respective constraining features that allow the inference of the most general type which introduces F. Let ψx be the conjunction representing the specification of the constraining features. We will call ψ the *realization context* of F$x\mathbf{v}$. This leads to the definition of *lexical feature*.

Def. 8 (lexical feature) *Given is a lexical type hierarchy $\langle lex, \sqsubseteq\rangle$ and a set A of appropriateness specifications. Let σ be a subtype of lex ($lex \sqsubseteq \sigma$) and ψ a logical description of a feature structure whose informational content just suffices to allow to infer σ. ψ is called the* realization context *of each feature F which is appropriate for σ. Then $\alpha = \langle \sigma, \psi, F\rangle$ is called a lexical feature. If $(Approp(F, \sigma) = \tau) \in A$ then the set of admissible values of α is denoted by $V(F)$ and is $V(F) = \{\tau' \mid \tau \sqsubseteq \tau'\}$.*

This leads to the following template of lexical facts:

$$\underbrace{\Lambda}_{\text{lexicon}} \quad \underbrace{\vdash}_{\text{entails}} \quad \underbrace{(\exists x)}_{\substack{\text{there}\\ \text{is one}}} \quad \underbrace{\mathcal{L}x}_{\text{lexeme}} \quad \& \quad \underbrace{\psi x}_{\substack{\text{realization}\\ \text{context}}} \quad \& \quad \underbrace{\mathrm{F}x\mathbf{v}}_{\text{feature:value}} \tag{2.10}$$

We will use the symbol $\alpha^{\mathcal{L}}_{\mathbf{v}}$ as an abbreviation of the entailed formula. It says: 'there is a lexeme that is positively specified for the features in the realization context ψ and the feature F$x\mathbf{v}$. We will call $\alpha^{\mathcal{L}}_{\mathbf{v}}$ the feature value of a lexeme's feature $\langle \mathcal{L}, \alpha\rangle$. And we will call $-\alpha^{\mathcal{L}}_{\mathbf{v}}$ a *lexical constraint*. We will further use $\alpha^{\mathcal{L}}_{\mathbf{v}\in V}$ to abbreviate the formula

$$(\exists x)\ \mathcal{L}x\ \&\ \psi x\ \&\ (\mathrm{F}x\mathbf{v_1} \vee \ldots \vee \mathrm{F}x\mathbf{v_n}) \tag{2.11}$$

where $\mathbf{v_i} \in V$ and $V \subseteq V(F)$.
A further comment with regard to the realization context is in order. First,

for the definition 8 to work it has to be made sure that there is always at least one realization context, otherwise ψ would not be properly defined. Second, it has to be ensured that a lexeme is somehow empirically grounded. This means that the features Gx**v** and/or PHONx**v** are required to be part of each realization context - being constraining features by definition. As the implementation of the theory deals with written input only, it is assumed that Gx**v** is always part of the realization context or in other words it is the *ultimate realization context*. Moreover, this feature does not count as a lexical feature in terms of definition 8.

Assumption 3 *The specification of the value of feature* Gx**v** *is assumed to be the ultimate realization context of all lexical features.*

The definitions of symbolic language model and of lexical feature facilitate a precise definition of lexical underspecification.

Def. 9 (underspecification of α) *Let* $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be a consistent symbolic language model and* $\alpha = \langle \sigma, \psi, F \rangle$ *a lexical feature in* $\mathbf{M}$*. Then* α *is said to be* underspecified *for a lexeme* $\mathcal{L}$ *if and only if*

$\Lambda \cup \{-\alpha^{\mathcal{L}}_{\mathbf{v}_i}\}$ *is consistent and*
$\Lambda \cup \{\ \alpha^{\mathcal{L}}_{\mathbf{v}_i}\}$ *is consistent*
for every $v_i \in V(F)$.

We will call the opposite of lexical underspecification a *full determination* of a lexical feature.

Def. 10 (full determination of α) *Let* $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be a consistent symbolic language model and* $\alpha = \langle \sigma, \psi, F \rangle$ *a lexical feature in* $\mathbf{M}$*. Then a lexical feature* $\alpha = \langle \sigma, \psi, F \rangle$ *is said to be* fully determined *by* Λ *if and only if* $\Lambda \vdash \alpha^{\mathcal{L}}_{\mathbf{v}_i}$ *or* $\Lambda \vdash -\alpha^{\mathcal{L}}_{\mathbf{v}_i}$ *for every* $v_i \in V(F)$.

The most extreme form of lexical underspecification is an unknown lexeme - which henceforth replaces the notion of 'unknown word'.

Def. 11 (unknown lexeme) *Let* $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be a consistent symbolic language model. Let* Sx**v** *and* Gx**v** *be the predicates encoding sense and graphical representation, respectively. A lexeme* $\mathcal{L}$ *is said to be* entailed *in* Λ *if and only if (at least) the features S and G are fully determined by* Λ[10]*. Otherwise it is said to be* unknown *in* Λ.

[10] In recent developments of ERG the option of *semantic underspecification* is built in to the system in order to capture the fact that 'surface words often have multiple senses which are not syntactically distinguished in one language but are in another' (Flickinger et al. 2005: 170). However this does not conflict with the definition of lexeme given here because it is not the case that there are lexical entries in ERG left without *any* sense related specification.

The definition of lexeme detection is straightforward. Again, we are using the predicate G to anchor the lexeme to the empirical world:

Def. 12 (lexeme detection) *Let* $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be a consistent symbolic language model. A lexeme* $\mathcal{L}$ *is said to be detected by the induction step* $\Lambda \sqsubset \Lambda'$ *if and only if* $\mathcal{L}$ *is unknown in* Λ *and* $\Lambda' \vdash (\exists x)\ \mathcal{L}x$ & Sx**s** & Gx**g**

Note that in this definition we already anticipate the notion of induction (cf. the definition in section 2.3.3).

If a lexeme is unknown, even its feature OPEN is underspecified, which leaves room to assign a negative value to it - contradicting the semantics of this feature. Therefore the following assumption seems appropriate:

Assumption 4 *In the course of the induction of the lexical features of an unknown lexeme, the feature* OPENx**v** *will be assigned the + value.*

2.1.6 What it Means to Know a Lexical Feature Value

The previous subsections provide all the necessary ingredients for a precise formulation of what is meant by saying that a system 'knows' a lexical feature value.
The theory of knowledge has a long tradition in Philosophy and the definition of knowledge has been worked out more and more precisely in order to capture all kinds of possible deception and fallacy. One such - very fine-grained - definition has been presented by Lehrer (1998: 21): 'S knows that p if and only if (i) it is true that p, (ii) S accepts that p, (iii) S is justified in accepting that p, and (iv) S is justified in some way that does not depend on any false statement.'[11] How does this definition apply to a system that is said to 'know' a particular lexical fact? First of all, according to condition (i) the lexical fact must be true. Hence it must be stated what makes a lexical fact true. This is simply not possible because of the relative nature of the lexicon - there is no such thing as an objectively true lexical fact. All that can be done is to stipulate which lexical facts are true on a meta-level, which is in Learn$-\alpha$ the perspective of the language engineer (LE). All the system 'knows' is relative to the beliefs of the LE who is the ultimate authority[12]. Thus this meta-level has to be worked out in more detail - which is the topic of the next

[11] We acknowledge that this definition may not be uncontroversial in the field of epistemology. For the points to be made here, however, we think the definition is sufficient.

[12] In practice this does not require that the LE does know the lexical fact upfront. It might be the case that review of the system's justification of the fact convinces the LE to accept the lexical fact as being true.

section. The term 'accept' in condition (ii) replaces the more traditional notion *believe*. The difference does not play any role in Learn$-\alpha$, so we will prefer the latter[13]. Condition (iii) translates in Learn$-\alpha$ to the requirement that all inductions of lexical features shall be empirically grounded (by the system's experience set) with statistical significance. Moreover, to enable the LE to review the 'knowledge' gained by the system it is of highest importance to represent the justification for it in a format that is easily accessible for the LE. The last condition (iv) means in the context of Learn$-\alpha$ that the system cannot be said to 'know' a lexical fact p - even if p is true (for the LE) and there are significantly many examples in the experience set that 'prove' p so that the system induced p - if all the examples are false positives (noise). In this case it was no more than mere coincidence that the system has induced a correct fact - but it does not 'know' it according to the definition of knowledge.

In favor of a logical foundation of artificial lexicon acquisition, we propose to model the system's epistemic and doxastic states within a modal-theoretic account defined over the symbolic language model. This will be worked out in section 2.2.4 - which in turn is the prerequisite for the account of a dynamic system that is able to adapt its doxastic state in the light of new experience (section 2.9). The model requires a formal understanding of the LE's perspective, which is the topic of the following section.

[13] According to Lehrer (1998: 13) 'acceptance' as opposed to 'believe' is 'defined in terms of some purpose', namely the 'purpose of attaining truth and avoiding error'.

2.2 Adequacy and Soundness: The Perspective of the Language Engineer

Section 1.1 discussed the LE's desire to arrive at a model of a language L that correctly analyzes all utterances of L. Technically, in terms of the definitions of the last sections, this means that all utterances that the LE considers grammatical are assigned one or more feature structures the type or sort of which corresponds to the domain of all utterances. This entails that all utterances that are judged ungrammatical shall not be assigned to any feature structure. Further, the demand for a *correct* analysis of an utterance u can be seen as a requirement that the model only assigns those feature structures to u that the LE would assign as well. Whether the model is able to do so is a property of its grammar and its lexicon. Consequently, the LE either constructs a model from scratch which has this property or he employs a model - henceforth called the *initial* model $\mathbf{M}_0$ - that does not have this property but which enables the system to arrive automatically at a modification $\mathbf{M}'$ that finally has it. In this case, it would be appropriate to say that the system automatically improves because of learning and it was argued that because of the language engineering bottleneck this would be the preferable choice. To characterize this kind of learning in terms of *improvement* a notion must be coined that covers the feature structures that the LE is after. We will call these the *intended* feature structures. This notion allows to state that the final model $\mathbf{M}'$ should be such that for every utterance u all and only the intended feature structures should be derived. A model that has this property may be called a sound model ($\mathbf{M}_{sound}$)[14].

2.2.1 The Optimal Model

Focusing on lexical acquisition it must be noted that the claim that the learning process in general must arrive at sound models is probably too strong. This is due to the grammar that is considered a constant and which plays a crucial role for the question whether all and only the intended feature structures are derived. It is easy to think of a grammar that accepts ungrammatical utterances regardless of what the lexicon entails. Consequently, all the lexicon can do is to add *further constraints* to the rules imposed by the grammar. In other words the lexicon can not do more than rule out

[14]Note that the lexicon Λ_{sound} of a sound model can be thought to be a finite set (if required for theoretical purposes) if we restrict the soundness of the model to a certain point of time t_0 in such a way that all and only those lexemes are covered that belong to the language at t_0.

utterances that are judged grammatical by the grammar. This is always the case when the feature structure derived by the grammar alone (without any lexicon) is inconsistent with some knowledge entailed by the lexicon. Coming back to lexical acquisition, the LE will target a model that includes a lexicon that has as many constraints as possible - ruling out as many utterances as possible - as long as all intended feature structures are derivable. Such a model should be called *optimal* ($\mathbf{M}_{opt}$).

2.2.2 The Adequate Model

The *intention* of the LE can be formally expressed by means of a function that maps utterances to a possibly empty set of all intended u-models: $I : D_u \mapsto \mathfrak{B}(D_M)$. This function fosters a couple of notions that describe properties of language models:

Def. 13 (sound language model) *Let* $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be a consistent symbolic language model. The model is called sound if and only if* $\mathbf{M} : u \Rightarrow I(u)$ *for all* $u \in D_u$ *(the domain of all utterances).*

Def. 14 (overgeneration) *Let* $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be a consistent symbolic language model. The model is said to overgenerate w.r.t. an utterance* $u \in D_u$ *if and only if* $\mathbf{M} : u \Rightarrow \mathcal{M}^u$ *and* $\mathcal{M}^u \supset I(u)$.

Def. 15 (undergeneration) *Let* $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be a consistent symbolic language model. The model is said to undergenerate w.r.t. an utterance* $u \in D_u$ *if and only if* $\mathbf{M} : u \Rightarrow \mathcal{M}^u$ *and* $\mathcal{M}^u \subset I(u)$.

Def. 16 (adequate language model) *Let* $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be a consistent symbolic language model. The model is said to be adequate if and only if* $\mathbf{M} : u \Rightarrow \mathcal{M}^u$ *and* $\mathcal{M}^u \supseteq I(u)$ *for all* $u \in D_u$.

A model that undergenerates requires revision (cf. section 2.9). Consequently, in order to avoid revisions, a learning step $\mathbf{M} \mapsto \mathbf{M}'$ should always lead to an adequate model. This claim leads to the following theorem:

Theorem 1 *No* $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *can be an adequate language model if the corresponding zero-lexicon-model* $\mathbf{M}_\emptyset = \langle \mathbf{FS}, \Gamma, \emptyset \rangle$ *is not adequate.*

Or in a slightly weaker version: no extension can be more adequate than the initial model. The reason is obvious: as further induced lexical facts can do nothing but constrain the derivation of feature structures, an intended feature structure that is ruled out by $\mathbf{M}_\emptyset$ can never be 'rehabilitated' by

later successors of $\mathbf{M}_\emptyset$.

Usually the LE will not start with a completely empty lexicon. He might provide his system with some initial lexical knowledge encoded in Λ_0 so that the corresponding $\mathbf{M}_0$ is the system's starting point - the state of the system before any induction takes place. To avoid revisions at the outset, it is a desire to ensure that the initial lexicon Λ_0 is adequate, a desire which may be hard to meet in practice.

2.2.3 Improvements: Targeting the Optimal Model

The definitions above help formalize the notion of learning for the type of systems Learn$-\alpha$ is dealing with. As pointed out, a lexicon - or a learning step - can only add further constraints. Consequently, if a learning step has some effect at all, then it is the exclusion of some feature structures that have been assigned to utterances before learning took place. Therefore a definition in terms of exclusion of feature structures appears to be reasonable:

Def. 17 (extend a language model) *Let* $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be a consistent symbolic language model. The model is said to be* extended *by a model* $\mathbf{M}' = \langle \mathbf{FS}, \Gamma, \Lambda' \rangle$ *(*$\mathbf{M} \sqsubset \mathbf{M}'$*) if and only if*
for every $u_i \in D_u$
$\mathbf{M} : u_i \Rightarrow \mathcal{M}^{u_i}$ *and* $\mathbf{M}' : u_i \Rightarrow \mathcal{M}^{u_i\prime}$ *and* $\mathcal{M}^{u_i\prime} \subseteq \mathcal{M}^{u_i}$
and there is at least one u_k *such that* $\mathcal{M}^{u_k\prime} \subset \mathcal{M}^{u_k}$.
The model $\mathbf{M}'$ *is said to* properly extend $\mathbf{M}$ *if and only if* $\mathbf{M}'$ *is adequate. Because it is the lexicon only that is subject to modification, we can also write* $\Lambda \sqsubset \Lambda'$.

From this it follows that an induction step $\Lambda \sqsubset \Lambda'$ is an *improvement* if and only if Λ' properly extends Λ. The optimal model introduced in subsection 2.2.1 can now be defined on top of definition 17.

Def. 18 (optimal extension) *The model* $\mathbf{M}_{opt}$ *is said to be an* optimal extension *of a model* $\mathbf{M}$ *if and only if* $\mathbf{M}_{opt}$ *properly extends* $\mathbf{M}$ *and there is no other model* $\mathbf{M}$*' that properly extends* $\mathbf{M}_{opt}$. $\mathbf{M}_{opt}$ *is called the* optimal model *and* Λ_{opt} *the optimal lexicon.*

This definition does not work if the number of lexemes is assumed to be infinite. However, there is no need for such an assumption. As the life cycle of the system is reasonably finite, so is the number of lexemes.

2.2.4 A Modal Logic for Learn$-\alpha$

A learning system that fulfills Learn$-\alpha$ can be modeled by a set of symbolic language models that share the same feature system and grammar. Learning takes place by extending a given model or more precisely by extending a given lexicon ($\Lambda \sqsubset \Lambda'$) until an optimum is reached.
In the optimal model all feature values for all lexemes are fully determined because then the model gains the highest precision possible. The optimal model is the best of all those sets of logical statements in which all feature values are fully determined. This set can be seen as a set of possibilities W and can be embedded in the semantics of possible worlds. Every underspecified lexical feature $\alpha^{\mathcal{L}}_{\mathbf{v}}$ in a theory Θ is then represented as a subset of W, namely those worlds consistent with Θ in which $\alpha^{\mathcal{L}}_{\mathbf{v}}$ holds plus those worlds consistent with Θ in which $-\alpha^{\mathcal{L}}_{\mathbf{v}}$ is the case. A given symbolic language model can be mapped to the set of worlds with which it is actually consistent. W must then be such that every world in it is consistent with the initial model. For example, there is no world in W in which a lexeme is a noun only which allows passive voice.
The possible world semantics allows the introduction of logical operators by means of which statements of one world can be made 'visible' to another. This bears the advantage that the statements which hold in predominant lexicons like Λ_{opt} can be syntactically integrated with statements in the current theory Θ of a learning system - in one logical statement.
Let $\Theta_0 = \Gamma \cup \Lambda_0$ be the initial theory of the learning system. Let $[T]$ denote the non-empty set of all theories that are compatible with a consistent theory T in the sense that $T' \in [T]$ if $T' = T \cup \phi$ - i.e. T expanded by a logical formula ϕ - is consistent, too[15]. The set $W = [T]$ can be interpreted as a set of possibilities ('possible worlds'). In Learn$-\alpha$ this amounts to $W = [\Theta_0]$. In terms of machine learning, it is W that describes the *search space*. Put differently, the search space is structured by the initial model given a-priori. In order to express the semantics of a logic based on W, a *valuation function* $\mathcal{V}$ is to be defined that maps a logical statement p to all worlds $w \in W$ in which p is the case. The set of these worlds, denoted by $[\![p]\!]_{\mathcal{V}}$ is called a proposition. The interesting propositions in Learn$-\alpha$ are $[\![\alpha^{\mathcal{L}}_{\mathbf{v}}]\!]_{\mathcal{V}}$ and $[\![-\alpha^{\mathcal{L}}_{\mathbf{v}}]\!]_{\mathcal{V}}$, respectively. The set of possible worlds taken together with the valuation function build a *model system* $\mathcal{M} = \langle W, \mathcal{V} \rangle$ that allows to recast boolean operations as set operations, e.g.:

[15] In the context of believe revision theories, it is often assumed that T (and every expansion T') is closed under deduction as well.

#1	for any formula ϕ:	$[\![-\phi]\!]_{\mathcal{V}} = W \setminus [\![\phi]\!]_{\mathcal{V}}$
#2	for any formulae ϕ_1 and ϕ_2:	$[\![(\phi_1 \ \& \ \phi_2)]\!]_{\mathcal{V}} = [\![\phi_1]\!]_{\mathcal{V}} \cap [\![\phi_2]\!]_{\mathcal{V}}$

Semantically, the truth conditions can be stated as follows using the notation $\mathcal{M}/w \vDash \phi$ expressing that the formula ϕ is true in world w of model system $\mathcal{M}$:

#3	$\mathcal{M}/w \vDash \phi$	iff $w \in [\![\phi]\!]_{\mathcal{V}}$
#4	$\mathcal{M}/w \vDash -\phi$	iff not $w \in [\![\phi]\!]_{\mathcal{V}}$
#4'	$\mathcal{M}/w \vDash -\phi$	iff not $\mathcal{M}/w \vDash \phi$
#5	$\mathcal{M}/w \vDash \phi_1 \ \& \ \phi_2$	iff $\mathcal{M}/w \vDash \phi_1$ and $\mathcal{M}/w \vDash \phi_2$

The model system is turned into a modal-logical model system by introducing the first modal operator, which we want to call the *optimality operator*. It can be used to express that ϕ holds in the optimal model represented by the world w_{opt}. Its truth condition is:

#6 $\mathcal{M}/w \vDash \Box\phi$ iff $\mathcal{M}/w_{opt} \vDash \phi$

This axiom allows for a *deontic* interpretation along the lines of Kanger (1970). In a deontic logic system, $\Box$ is interpreted as *ought, should* and the analog to Λ_{opt} would be a *welfare program* - a world in which everything what is wanted in 'our' world is really the case - in a learning system every feature setting that should be learned is the case in Λ_{opt}.

The semantics of a modal operator is captured by a binary *accessibility relation* R that holds between words of the model system. Properties of R characterize the axioms that are valid in the model system regarding the operator. The accessibility relation $R_{\Box}$ of the optimality operator is a serial one: For every world w the relation $wR_{\Box}w_{opt}$ holds (every world 'sees' the optimal one). This sustains the deontic interpretation of the operator.
The model system is extended to $\mathcal{M} = \langle W, R_1 ... R_n, \mathcal{V}\rangle$ if it contains n modal operators. The logical foundation of Learn$-\alpha$ is then given by the following definition:

Def. 19 (model system over symbolic language model)
Let $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda\rangle$ *be a consistent symbolic language model,* $\mathbf{M}_0 = \langle \mathbf{FS}, \Gamma, \Lambda_0\rangle$ *the corresponding initial model and* $\mathbf{M}_{opt} = \langle \mathbf{FS}, \Gamma, \Lambda_{opt}\rangle$ *the corresponding optimal model.*
The model system defined over $\mathbf{M}$ *is* $\mathcal{M} = \langle [\Gamma \cup \Lambda_0], R_{\Box} ..., \mathcal{V}\rangle$ *with a valuation function* $\mathcal{V}$ *and a serial accessibility relation* $R_{\Box}$ *for the optimality operator* $\Box$*. For every theory ('world')* $\Theta \in [\Gamma \cup \Lambda_0]$ *and formula* ϕ *the following axiom holds:*

$$\mathcal{M}/\Theta \vDash \Box\phi \quad \textbf{iff} \quad \mathcal{M}/\Theta_{opt} \vDash \phi \tag{2.12}$$

Three notes for definition 19 are important: First, it is assumed that axioms #1 - #5 above hold, too. Second, it is open to add further modal operators (the three dots in $R_{\Box}...$). Third, the model system as defined above does not account for the predicate calculus employed in the modeling of lexical facts. It can be extended to do so, but this complicates the matter unnecessarily. In such an extension, additional functions have to be introduced that control the intension/extension of terms and their relation to the objects existing in the possible worlds. In Learn$-\alpha$ the only objects of interest that transcend worlds are the lexemes. Let us therefore simply assume that a) all lexemes 'exist' in all worlds in $[\Theta_0]$, whether they are known or not and that b) lexemes of different worlds can be identified by virtue of their extensional feature Gx**v** which grounds them empirically.

For all worlds $w \in W$ and statements p (that can be expressed with the feature system underlying the model system) it holds that p is in w or it is not, *tertium non datur*. From this the following theorem can be deduced, which becomes important for the induction step later on:

Theorem 2 *Let* $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be a consistent symbolic language model and* $\langle [\Gamma \cup \Lambda_0], R_{\Box}..., \mathcal{V} \rangle$ *its model system. Then the following statement is valid in the system:*

$$- \Box \phi \quad \supset \quad \Box - \phi \tag{2.13}$$

Proof: Let $\mathcal{M} = \langle W, R_{\Box}..., \mathcal{V} \rangle$ be the model system over $\mathbf{M}$. $- \Box \phi$ is true in a world $w \in W$ iff $\mathcal{M}/w \vDash - \Box \phi$. Suppose that $\mathcal{M}/w_{opt} \vDash \phi$. By #6 this would entail $\mathcal{M}/w \vDash \Box\phi$ contradicting the assumption because all worlds in the system are considered consistent. Therefore $\mathcal{M}/w_{opt} \vDash \phi$ does not hold. From #4' it follows that $\mathcal{M}/w_{opt} \vDash -\phi$. #6 links this back to all worlds (including w of the outset): $\mathcal{M}/w \vDash \Box - \phi$ ■

To account for the *doxastic* state of the learning system the model system will be extended by a second operator B_s with the intended meaning that $B_s\phi$ is true if the system s believes that ϕ is the case[16]. All worlds $w \in W$ in which ϕ is the case are doxastic alternatives (Hintikka 1962: 34) with regard to ϕ (all other worlds are disbelieved). The intersection of the sets of doxastic alternatives related to all believed statements ϕ_i constitute the actual *belief set* of the system. Let R_{B_s} be the corresponding accessibility relation (called relation of alternativeness in Hintikka 1962) that connects the factual world to all doxastic alternatives. The belief set is a subset of W which means

[16]It is usual to talk about the beliefs of an *agent*. In our work, the notion 'agent' is consistently replaced by 'system'.

that the possible belief states are in the boundaries of the system's a-priori knowledge. Because some of the lexical constraints are merely *believed* (not known) by the system, the resulting outcome of an analysis of an utterance u is a *believed* analysis. As the model system is assumed to be a Kripke system the axiom K: $\Box(p \supset q) \supset (\Box p \supset \Box q)$ is valid for all modal operators. If all the premises of an analysis are known (i.e. all realized lexemes are a-priori) then the resulting analysis is known otherwise merely believed by the system.

Def. 20 (modal operator B_s) *Let $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ be a consistent symbolic language model and $\mathcal{M} = \langle [\Gamma \cup \Lambda_0], R_\Box, R_{B_s} ..., \mathcal{V} \rangle$ the model system over it. Then the learning system believes that ϕ is the case iff $B_s\phi$.*

Learning of a lexical fact can be seen as the resulting belief that the statement encoding the lexical fact is entailed by the optimal world: $B_s \Box \alpha^{\mathcal{L}}_{\mathbf{v}}$. In order to *apply* the learned fact the system has to assume that ϕ (in the premises of an analysis) if it believes that ϕ is in the optimal model - assuming that the system is designed for the optimal outcome:

$$B_s \Box \phi \supset B_s\phi \tag{2.14}$$

2.2.5 Grammaticality: What the System Knows About the Optimal Model

As a prerequisite for the learning steps consider what the system knows about $\mathbf{M}_{opt}$ and what it does not know. To start with the latter, it is obvious that the system does not know *which* of the feature structures that are entailed in an initial model for a certain utterance are really intended. If the system knew that, then no learning would be necessary at all. Let $\mu \in I(u)$ be an intended u-model. Then it follows that it must be derivable from the optimal model as well, otherwise this model would not be adequate. By use of the optimality operator introduced in the last subsection, it can be stated that $\mathbf{M} \vdash \Box \mathrm{M}u\mu$, but as said, the system does not have this knowledge, so it must be assumed that the initial model is indifferent with regard to the analysis of an intended u-model in the optimal model:

Assumption 5 *For all intended u-models $\mu \in I(u)$ of an utterance u the initial model is consistent with $\Box Mu\mu$ and it is consistent with $\Box - Mu\mu$.*

The only cue the system has about the analysis of u in the optimal model is that there must be at least *one* u-model, because of assumption (1a) (subsection 1.4.3) which states that all utterances that the system experiences are grammatical:

Assumption 6 *For the initial model* $\mathbf{M}_0$ *of the learning system (and all extensions), it holds that for every* $u \in E$

$$\mathbf{M}_0 \vdash \Box(\exists x)\ \mathit{Mux}.$$

2.2.6 What it Means to Admit a Feature Value

Additionally, we can give a formal explication of the notion 'a lexeme really admits a feature value'. The intuition behind this is that a lexeme $\mathcal{L}$ *realiter* admits a feature value $\alpha^{\mathcal{L}}_{\mathbf{v}}$ if it is entailed in the optimal model:

Def. 21 (admit a feature value) *A lexeme* $\mathcal{L}$ *is said to* admit *a feature value* $\alpha^{\mathcal{L}}_{\mathbf{v}}$ *if and only if* $\Box\alpha^{\mathcal{L}}_{\mathbf{v}}$.

2.3 Towards a Logical Account of Learn−α

This section concerns the logical foundation of the question raised in section 1.2: Can the lexicon be a natural fall-out and logical consequence of the system's a-priori knowledge taken together with its experience?
This is the basic idea behind the *automatic induction of lexical features*. It is the idea that if the system is provided with a sufficiently large experience set E of grammatical utterances, then it can induce knowledge about lexemes - encoded in the form of lexical features. The induction is based on the system's observation of the behavior of the lexemes that are realized in the utterances and on the assumption that this behavior *will be the same* when facing the challenge.
The single observations are somehow encoded in the feature structures $\mathcal{M}^u$ that result from applying a model $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ on the single utterances in E - they are entailed in the *analysis* of E. Considering the current knowledge in Λ, the system can induce an improved lexicon Λ' with enriched and/or revised knowledge. Both phases - observation and induction - are summarized in the following formula:

i. observation: $\langle \Gamma, \Lambda \rangle : E \mapsto \mathcal{M}^E$

ii. induction: $\langle \Lambda, \mathcal{M}^E \rangle \mapsto \Lambda'$

The observation step will be explicated in subsections 2.3.1 and 2.3.2. The induction step is dealt with in subsection 2.3.3.

2.3.1 Observations: Exploiting Learning Opportunities

One claim of Learn−α is that the acquired knowledge must be consistent with the experience set. We will now give a more precise explanation of what consistency means in this context. Consistency with the experience set E simply means that the determination of the feature α is consistent with every u ∈ E.

Def. 22 (Consistency of Lexicon and Utterance(s))
Let $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be a consistent symbolic language model and u an utterance. Then* Λ *is said to be* consistent *with u if and only if* $\mathbf{M} \vdash (\exists x)$ *Mux (cf. subsection 2.2.5).*
Further, Λ *is said to be* consistent *with a set of utterances U if and only if* Λ *is consistent with every* $u \in U$.

Obviously it follows from theorem (1) (the zero-lexicon must be adequate) that the zero-lexicon must be consistent with the set of grammatical

utterances to warrant that subsequent extensions are adequate, too. The induction step developed in subsection 2.3.3 has to ensure that for every lexicon consistent with E its extension is consistent with E, too. So it is crucial to find out by experience which feature values a lexeme admits in order to avoid that the extension entails a feature constraint that is not warranted and which will rule out some utterances in E, which would dissatisfy the restriction that the extension must be consistent with E. Consequently, if an utterance's parsability (or grammaticality in a model) depends on a feature value then this feature value must be deducible from the optimal lexicon. Or the other way round: if an utterance u is considered grammatical by the LE (hence it must have an u-model in the optimal language model) and the feature constraint $-\alpha^{\mathcal{L}}_{\mathbf{v}}$ *would* prevent any successful analysis of u then the feature value $\alpha^{\mathcal{L}}_{\mathbf{v}}$ is deducible from the optimal lexicon. We will express this implication in a formula that we like to call the **(Learn−α) observation schema**:

$$(\Box(\exists x)\ \mathrm{M}ux\ \&\ (-\alpha^{\mathcal{L}}_{\mathbf{v}} \supset -(\exists x)\ \mathrm{M}ux)) \supset \Box\alpha^{\mathcal{L}}_{\mathbf{v}} \tag{2.15}$$

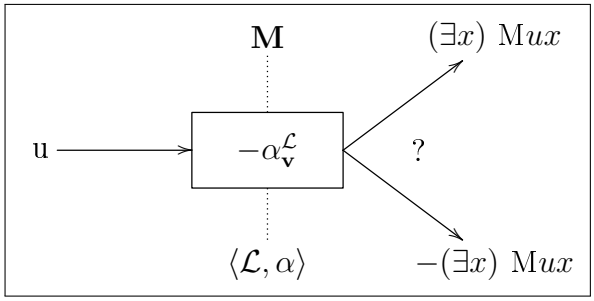
So the reasoning of the system goes as follows: 'I know that u is grammatical. I also have *observed* that if $-\alpha^{\mathcal{L}}_{\mathbf{v}}$ holds then I cannot find any solution (or u-model) for u. Consequently I will believe that $\alpha^{\mathcal{L}}_{\mathbf{v}}$ is optimal.' In this case, we call u a *reference* for the feature value and the transition from observation to a piece of knowledge is henceforth expressed in the notation given in the following definition:

Def. 23 (reference) *If an utterance u is analyzable if and only if $\alpha^{\mathcal{L}}_{\mathbf{v}}$ is true, then u is called a reference for $\alpha^{\mathcal{L}}_{\mathbf{v}}$. We notate:*

$$\mathbf{M} : u \mapsto \Box\alpha^{\mathcal{L}}_{\mathbf{v}}$$

The schema is depicted in figure 2.3:

Figure 2.3: Feature Value Observation Schema



This definition will be replaced by a more general one on page 148. The strong version of Learn−α demands that the determination of a $\alpha^{\mathcal{L}}_{\mathbf{v}}$ is a prerequisite for the successful analysis of at least one utterance in the experience set. This amounts to claiming that the observation schema has to prove a certain feature value unambiguously (RDoA=1). If this restriction would not hold we would arrive at the following schema:

$$(\Box(\exists x)\ \mathrm{M}ux\ \&\ (-(\alpha^{\mathcal{L}}_{\mathbf{v}_1}\vee\ldots\vee\alpha^{\mathcal{L}}_{\mathbf{v}_n}\supset -(\exists x)\ \mathrm{M}ux))\supset\Box(\alpha^{\mathcal{L}}_{\mathbf{v}_1}\vee\ldots\vee\alpha^{\mathcal{L}}_{\mathbf{v}_n}) \quad (2.16)$$

By implication, however, it does not follow which of the values the lexeme really admits. All we can say is that the disjunction $(\alpha^{\mathcal{L}}_{\mathbf{v}_1}\vee\ldots\vee\alpha^{\mathcal{L}}_{\mathbf{v}_n})$ holds in the optimal model. But using this schema the system does not *arrive at* the optimal model because if only one of the values - say v_k - is not admitted then there might be an utterance the analysis of which could be restricted if the system *would* entail that $-\alpha^{\mathcal{L}}_{\mathbf{v}_k}$ and in this case there would be a proper extension of the lexicon.

At a first glance, the observation schema is computationally extremely inefficient because it appears to force the system to analyze an utterance as many times as there are possible combinations of lexemes, lexical features and feature values in the utterance. However, as argued in subsection 1.4.5, in a unification based grammar the lexical features will either be bound to values during the analysis of an utterance (by means of path equations) or they will be left unbound. In the latter case the setting of the respective feature is completely indifferent to u's analysis (which means that $-\alpha^{\mathcal{L}}_{\mathbf{v}}\supset -(\exists x)\ \mathrm{M}ux$ does not hold) or the values can be extracted from the derived feature structures. The formulation of the logic that underlines Learn−α will, nevertheless, abstract from any possible implementation and hence from any aspect of computational efficiency. It is worthwhile noting that a formalism like Lexical Functional Grammar (LFG) that makes a distinction between *defining equations* (where feature values come into being in the course of a derivation) and *constraining equations* (which do not create feature settings but merely check the existence of feature values)[17] does not guarantee that all lexical features are bound if necessary. In this case the mere *extraction* of features from the feature structures (f-structure in LFG) would not suffice to fulfill Learn−α.

The same holds for the use of inequations (cf. section B.1), as the following example will illustrate. For that purpose, let ±TR be a verbal feature that indicates whether a verb can be used transitively or not and let us assume that rule 2.17 is the only rule with access to this feature. While the rule will

[17] Kaplan and Bresnan (1982). Cf. Manning (1991) for a critique of this distinction.

ensure that no verb which is marked [-TR] can be followed by an NP, the feature value will never be instantiated if underspecified.

$$\begin{bmatrix} \text{CAT} & \text{VP} \\ & \end{bmatrix} \rightarrow \begin{bmatrix} \text{CAT} & \text{V} \\ \text{TR} & \boxed{1} \\ \boxed{1} \neq - & \end{bmatrix} \begin{bmatrix} \text{CAT} & \text{NP} \\ & \end{bmatrix} \tag{2.17}$$

Of course, 2.18 is to be preferred and it seems naive to consider 2.17 at all, but a similar situation may indeed occur somewhere during the development of a complex grammar that uses inequations.

$$\begin{bmatrix} \text{CAT} & \text{VP} \\ & \end{bmatrix} \rightarrow \begin{bmatrix} \text{CAT} & \text{V} \\ \text{TR} & + \end{bmatrix} \begin{bmatrix} \text{CAT} & \text{NP} \\ & \end{bmatrix} \tag{2.18}$$

All this does not pose any problem to Learn−α, but it has an impact on computability. In other words, inequations might be counterproductive to a mechanism that just reads-off lexical feature values as they have been instantiated during analysis.

2.3.2 Lexical Acquisition Space

The derivation of a feature structure $\boldsymbol{\phi}$ from an observed utterance with the help of the observation schema can be seen as a search through a search space that we call *lexical acquisition space*, a search space which is structured by the language model. The nodes of this search space are lexical feature structures; the arcs are transitions from an already known feature structure $\boldsymbol{\phi}_i^{\sigma}$ (feature structure of sort or type σ) to an acquired feature structure $\boldsymbol{\phi}_k^{\sigma'}$. Because the transition is made with reference to an utterance u, u may be the arc's label. Diagram 2.4 shows an example within an untyped feature theory in which the lexical type hierarchy is assumed to be flat. The entry into the search space is a feature structure on the lexical sort *lex*. This feature structure $\boldsymbol{\phi}_0^{lex}$ just encodes the knowledge that a given lexeme $\mathcal{L}$ spelled **g** exists (let us assume, the system does not know anything else about it). The feature structures are annotated with the according logical description. Based on the evidence from the application of the observation schema on some utterance u_1, the system constructs a feature structure $\boldsymbol{\phi}_1^{lex}$ that specifies values for some of the features introduced by *lex*. Note that the acquired feature structure must a) be consistent with the known feature structure and b) contain more terms than the known feature structure. Otherwise the transition would contradict Learn−α.

Figure 2.4: Lexical Acquisition Space in Untyped Theory

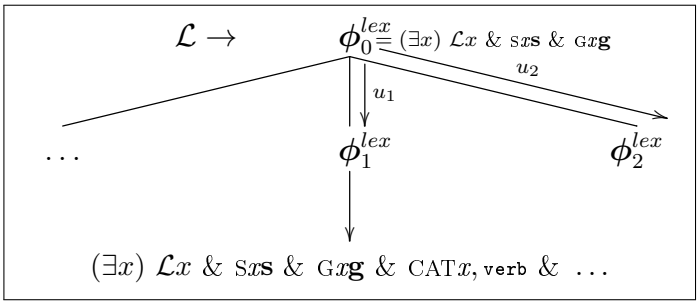
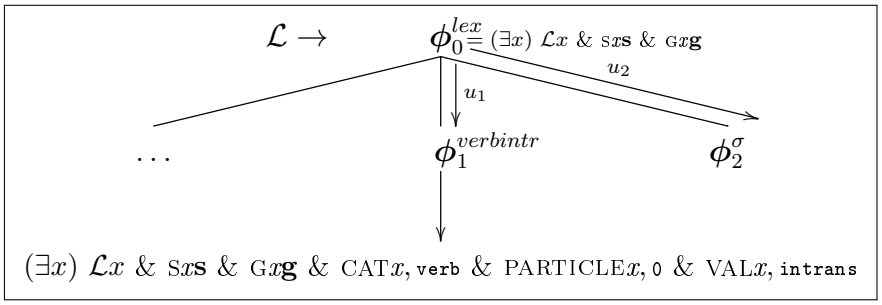


Figure 2.5 depicts an example where a typed feature theory is used. As with the untyped case, consistency between the feature structures must be maintained. This requires that the type of the acquired feature structure $\boldsymbol{\phi}_k^{\sigma'}$ is a subtype of the known feature structure $\boldsymbol{\phi}_k^{\sigma}$: $\sigma \sqsubseteq \sigma'$. In the example, there is one utterance (u_1) that evidences facts about a certain verb: the graphical representation, the verbal particle (none) and the valency (intrans). By virtue of type inference, the transition produces a feature structure that is of type *verbintr* (which shall denote the class of intransitive verbs). The point here is that from this moment on, additional features might appear at the horizon of learnability, namely those that are introduced by this particular type.

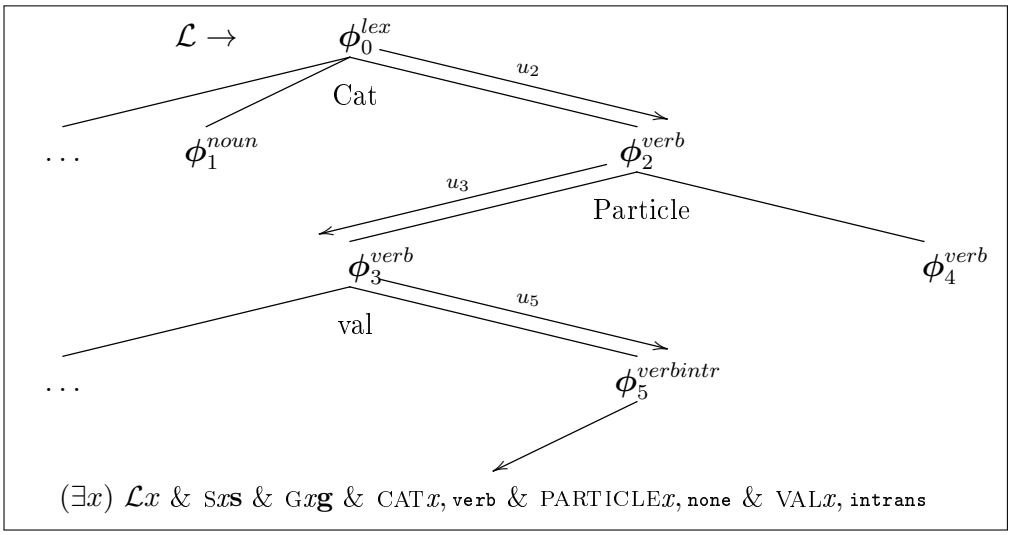
Figure 2.5: Lexical Acquisition Space in Typed Theory



The last example in figure 2.6 is an instance of *incremental acquisition*. Here, the observations are exploited to incrementally refine existing knowledge about a lexeme. The acquired knowledge proved by an utterance becomes the known feature structure for the analysis of a subsequent utterance, which is used to prove further lexical facts. In the example, it takes four of

them to arrive at the feature structure that was acquired in one step in the previous example. For better illustration, we note down the constraining features under the nodes. Because the search through the lexical acquisition space goes from the most general to the most specific type, this kind of search can be adumbrated with the slogan 'Learning Down the Hierarchy', coining a notion that will be formulated more precisely in section 2.8.1.

Figure 2.6: Incremental Lexical Acquisition



2.3.3 Induction

Inductive reasoning is the process of inferring a) the unknown from the known and/or b) the universal from the particular. For example, if all swans ever encountered (particular known objects or *tokens*) were white, it could be hypothesized that a) the next swan to be observed (the unknown) will also be white and b) *all* swans (the universal *type*) are white[18]. The cogency of the inference highly depends on the sample or the experience and it is valid until the first contradicting token is observed, which cannot be reconciled with the hypothesis - a black swan in the example. That means that an inductive conclusion is always afflicted with some uncertainty, as opposed to deductive arguments, where the premises *logically* entail the conclusion. Whether the principal fallibility of induction gives rise to radical skepticism or whether

[18]Put logically: $(\forall x)$ Swan$x \supset$ COLOURx**white**. A weaker form would claim: it is very likely that all swans are white.

one can live reasonably well with it in practice, is a philosophical question that will not be tackled here. But at least it highlights that *revisability* is a necessary feature of a system that learns by induction.
Applied to lexical acquisition, the inductive process generates a universal statement about a certain lexeme $\mathcal{L}$ based on the single facts observed from the behavior of the related tokens in the utterances of the investigated language. In some respects the particular token showing a certain feature based on the observation schema of the last subsection are the white swans - the 'feature' COLOR of this object set to **white** during the observation of each instance. Further, the fact that we never observed a green swan gives reason to assume that no swan on this planet is green. Likewise, the fact that a lexeme $\mathcal{L}$ never unambiguously shows a certain feature value should lead the system to the conclusion that the lexeme never will show this value in any utterance - or in other words: that it does not admit the feature value. This kind of reasoning should be called **(Learn$-\alpha$) induction schema** and can be formally expressed as follows:

$$-(\exists x)\ (\Box(\exists y)\ \mathrm{M}xy \supset \Box\alpha^{\mathcal{L}}_{\mathbf{v}}) \supset -\Box\,\alpha^{\mathcal{L}}_{\mathbf{v}} \tag{2.19}$$

Opposed to the observation schema that deals with the extraction of lexical facts out of one particular utterance, the induction schema deals with the universe of utterances (filling the first variable of the u-model relation) - this is what the variable x ranges over. With the help of theorem (2) of section 2.2.4 we arrive at the induction of the corresponding feature value constraint:

$$-(\exists x)\ (\Box(\exists y)\ \mathrm{M}xy \supset \Box\alpha^{\mathcal{L}}_{\mathbf{v}}) \supset \Box -\alpha^{\mathcal{L}}_{\mathbf{v}} \tag{2.20}$$

This second variant of the induction schema is important, because if the system does not arrive at constraints then it never finds an extension to a given model (cf. subsection 2.2). If induction takes place based on formula 2.20 then both a) consistency with the set of input utterances is met and b) the weak condition of Learn$-\alpha$ is maintained. This is because a) no utterance would be ruled out by believing in $-\alpha^{\mathcal{L}}_{\mathbf{v}}$ and b) because the system will add a constraint that will rule out some utterances which are considered grammatical if the assumption is not made - that means that the induced knowledge is an extension and shows some effect ('effective learning').
At the beginning we stated that induction produces all-quantified sentences, but it seems that the induction schema does not make use of all-quantifiers. However if we replace $\alpha^{\mathcal{L}}_{\mathbf{v}}$ by what it stands for and if we apply the rules of quantifier negation then we arrive at:

$$\Box -(\exists x)\ \mathcal{L}x\ \&\ \psi x\ \&\ \mathrm{F}x\mathbf{v} \quad \supset \quad \Box(\forall x)\ -(\mathcal{L}x\ \&\ \psi x\ \&\ \mathrm{F}x\mathbf{v}) \tag{2.21}$$

The only problem is the interpretation of the first quantifier $-(\exists x)$ in formula (2.20) (there is no utterance x such that ...). If it quantifies over the set of all grammatical utterances U^+ (there is no utterance x $\in U^+$ such that ...) then the formula is logically valid - and would be an instance of so-called *complete induction*. However, applied to a finite set of utterances E $\subset U^+$ the formula is no logical truth anymore. It becomes a mere assumption - probably an approximation of truth depending on how good E represents U^+. This is the nature of induction. Now we can define the induction step in terms of experience and lexical extension:

Def. 24 (induction of lexical knowledge) *Let* $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be a consistent symbolic language model and E the set of experienced utterances. Induction of lexical knowledge is an extension of the lexicon:* $\Lambda \sqsubset \Lambda'$ *under application of the induction schema to E (we write* $E \mapsto \Box(-)\alpha^{\mathcal{L}}_{\mathbf{v}}$*) such that*

$$\begin{array}{llll} \textbf{if } E \mapsto \Box \quad \alpha^{\mathcal{L}}_{\mathbf{v}} & \textbf{then} & \Lambda' \vdash \quad \alpha^{\mathcal{L}}_{\mathbf{v}} & \textbf{and} \\ \textbf{if } E \mapsto \Box - \alpha^{\mathcal{L}}_{\mathbf{v}} & \textbf{then} & \Lambda' \vdash -\alpha^{\mathcal{L}}_{\mathbf{v}} & \end{array} \tag{2.22}$$

When does learning end? Never, as a matter of principle, because the lexicon is open. More precisely, learning stops at the end of the system's life cycle.

2.3.4 Feature Value Realization

Before we can introduce a probabilistic account of Learn−α it has to be stated more precisely what it means that a given lexeme $\mathcal{L}$ *realizes* a feature value $\alpha^{\mathcal{L}}_{\mathbf{v}}$ in an utterance u.
To start with an example, we want to be able to state that the lexeme `want` in (9a) realizes its disposition to subcategorize for an infinitive complement, whereas it does not do so in (9b). The notion should additionally allow for the statement that the lexeme `marry` in (9c) also does not realize the corresponding feature value:

(9)
a. He never wanted to be a star.
b. He wanted a car.
c. He married to get rich.

While the statement about (9c) goes back to the linguistic fact that `marry` does not admit the feature value that encodes the subcategorization for an infinitive complement, the reason for the statement about (9b) is that the

according disposition of `want` simply does not matter for the interpretation or analysis of the utterance. So the question whether $\mathcal{L}$ realizes $\alpha_{\mathbf{v}}$ in u depends on two underlying aspects: a) a lexeme $\mathcal{L}$ that does not admit $\alpha_{\mathbf{v}}$ can never realize it, b) a lexeme $\mathcal{L}$ that realizes $\alpha_{\mathbf{v}}$ in u influences the analysis (parsability and the possible readings) of u *because* it admits $\alpha_{\mathbf{v}}$. Whether a lexeme really admits a feature value can not be scrutinized here; we simply leave it to the symbolic language model that the LE *has in mind* when he wants his system to learn. Section 2.2 introduced the model $\mathbf{M}_{opt}$ as a 'final' model that the LE aims at. Part of this model is the optimal lexicon Λ_{opt} which makes $\alpha_{\mathbf{v}}^{\mathcal{L}}$ deducible if $\mathcal{L}$ admits it (cf. subsection 2.2.6).
From a formal perspective it can be said that a lexeme realizes a feature value if and only if the analysis of an utterance depends on whether $\mathcal{L}$ admits $\alpha_{\mathbf{v}}$ or not. Imagine two possible models $\mathbf{M}_{opt}$ and $\mathbf{M'}_{opt}$ that are identical except of one difference: the first makes the feature value deducible, the second does not. If the analysis of u under $\mathbf{M}_{opt}$ differs from that under $\mathbf{M'}_{opt}$ then $\mathcal{L}$ realizes the feature value in u:

Def. 25 *Given are two possible optimal models* $\mathbf{M}_{opt} = \langle \mathbf{FS}, \Gamma, \Lambda_{opt} \rangle$ *and* $\mathbf{M'}_{opt} = \langle \mathbf{FS}, \Gamma, \Lambda'_{opt} \rangle$ *that only differ with regard to the following entailments:*

$$\begin{array}{ll} \Lambda_{opt} \vdash & \alpha_{\mathbf{v}}^{\mathcal{L}} \\ \Lambda'_{opt} \vdash & -\alpha_{\mathbf{v}}^{\mathcal{L}} \end{array}$$

A lexeme $\mathcal{L}$ *is said to* <u>realize</u> *the corresponding feature value* $\alpha_{\mathbf{v}}$ *in an utterance u under* $\mathbf{M}_{opt}$ *if and only if*

$$\begin{array}{lll} \mathbf{M}_{opt}: & u \Rightarrow \mathcal{M}^{u} & and \\ \mathbf{M}'_{opt}: & u \Rightarrow \mathcal{M}^{u'} & and \\ \mathcal{M}^{u} \neq \mathcal{M}^{u'} & & \end{array}$$

2.3.5 Decisions, Quantification of Evidence and Testing Power

Experience-based induction is always afflicted with a certain amount of uncertainty, however persuasive the evidence might be. Because of that, *statistical estimation theory* is a central building block of a theory of Learn−α. It helps the system to quantify *how much* the induction steps are supported by experience. Moreover, it will be the basis for dealing with two major hurdles that come along with lexical acquisition: noise (section 2.4) and systematic ambiguity (section 2.5). This subsection develops the probabilistic,

frequentist-based formulae required for the induction step in an *ideal world*. They will be finalized in section 2.8. A set of probabilistic notions is required. They will be introduced here without further justification. In case the reader is less familiar with statistics all notions are motivated in section A in the appendix.
Let $E_\alpha^{\mathcal{L}}$ denote the sample, which is the set of utterances in which $\mathcal{L}$ realizes one of the values of the feature α. Probabilistic trials are seen as picking out one $u \in E_\alpha^{\mathcal{L}}$ at random and observing which feature value $v_i \in V(F)$ for $\mathcal{L}$ and α has been realized. The realization happens with an unknown probability p - a parameter of the distribution to be estimated. Such parameters will be denoted by θ. In an ideal, noise-free world, the parameter could be approximated by MLE. We denote MLE-estimated parameters with a hat such as $\hat{p}(v_i)$. Each feature value v_i is assigned a random variable X_i that models the outcome of an experiment with $n = |E_\alpha^{\mathcal{L}}|$ trials. The observed number of realizations in general is denoted by x. $X \sim D$ means that the random variable is D-distributed. Let $f(x|\theta)$ denote a known distribution f with an unknown parameter θ stemming from a parameter space Θ (not to be confused with a theory).

It will now be discussed how the learning system can make use of these instruments in order to arrive at a good decision based on its experience. Formally, we want to express the *decision* related to a particular induction step by a function $\delta : E_\alpha^{\mathcal{L}} \mapsto \{\Box\alpha_{\mathbf{v}}^{\mathcal{L}}, \Box - \alpha_{\mathbf{v}}^{\mathcal{L}}\}$. In other words, the result of the function δ fed with the experience set is the belief of the system whether the constraint will hold in the optimal model or not.
The simplest tests for Learn−α are the *frequency test* given in 2.23 and *MLE thresholding* (2.24), respectively.

$$\delta(E_\alpha^{\mathcal{L}}) \quad = \quad \begin{cases} \Box\alpha_{\mathbf{v}_i}^{\mathcal{L}} & \textbf{if} \quad x(v_i) \geq freq_{min}(\alpha_{\mathbf{v}_i}^{\mathcal{L}}) \\ \Box - \alpha_{\mathbf{v}_i}^{\mathcal{L}} & \textbf{else} \end{cases} \tag{2.23}$$

$$\delta(E_\alpha^{\mathcal{L}}) \quad = \quad \begin{cases} \Box\alpha_{\mathbf{v}_i}^{\mathcal{L}} & \textbf{if} \quad \hat{p}_i \geq MLE_{min}(\alpha_{\mathbf{v}_i}^{\mathcal{L}}) \\ \Box - \alpha_{\mathbf{v}_i}^{\mathcal{L}} & \textbf{else} \end{cases} \tag{2.24}$$

Here, both $freq_{min}$ and MLE_{min} are thresholds that might be stipulated by the language engineer or determined via system training. Both tests often show better results than one might expect. For example, Korhonen (2002)

demonstrates how the MLE-based test outperforms tests based on BHT and LRT when applied to cue-based verbal subcategorization frame acquisition. The disadvantage of these methods is the training period required to setup optimal thresholds.
We now turn to decision procedures based on *statistical inference* developed in the last century. The underlying theories, philosophies and methodologies are often mutually incompatible and there is no general consensus among statisticians which is the 'best' way to follow. Because of that and in order to distinguish incommensurable notions of statistical inference theory properly (motivated by Berger 2003 and Hubbard and Bayarri 2003), we aim at strictly disclosing from which school of thought a certain concept stems. We will start with the Fisherian school (Fisher 1925) and the notion of p-value, followed by the discussion of classical hypothesis tests rooting in the Neyman-Pearson school (starting with Neyman and Pearson 1933). Learn−α embedded in the Bayesian tradition would take Jeffreys (1961) as a starting point, but this is out of the scope of this thesis. We try to cast the crucial formulae using a consistent set of symbols, but do not want to belie the incompatibilities mentioned above.

Significance Testing and P-Value

The approach of *significance testing* (Fisher 1925) aims at providing an 'objective' criterion of statistical induction. To achieve this one has to setup a *null hypothesis* H_0 that a sample stems from a hypothetical infinite population with a known distribution, say $X \sim f(x|\theta)$. This null hypothesis is to be rejected if the observed sample (the outcome of an experiment) shows *significant evidence* against it. The measurement of evidence expresses how much the observation contradicts H_0 or how likely it is to come across the sample or any sample that is more unlikely in a world in which H_0 holds. This probability is called the *p-value* π (we use π instead of the usual p to avoid confusion), which is calculated by means of a *test statistics* $T = t(x)$ that has to be chosen upfront: $\pi(x) = P_0(t(X) \geq t(x))$ (cf. Berger 2003: 2). The smaller π the more evidence against H_0. To come to a decision whether H_0 should be rejected or not one has to setup an arbitrary threshold, called the *significance level* with typical values 0.5, 0.1 or 0.01 established in practice. This is usually denoted with α. As outlined below there is a subtle ambiguity between this threshold and the one that is used in N-P style hypothesis testing. We therefore use α_s for the former. The decision rule is then:

$$\delta(x) \quad = \quad \begin{cases} \text{reject } H_0 & \textbf{if} \quad \pi(x) < \alpha_s \\ \\ \text{do not reject } H_0 & \textbf{else} \end{cases} \tag{2.25}$$

This rule makes one important point explicit: it does not indicate whether to *accept* H_0 or not. There is also no alternative hypothesis to be accepted if H_0 is rejected. If $\pi < \alpha_s$ this just means that either a rare event has occurred or H_0 is false (or the sample is not random). It is up to the user of this method to make further decisions accordingly. In addition, the interpretation of the absolute value of π is not straightforward. For further critics against the p-value driven procedure see Berger (2003). However it is recommended to report the p-value along with the results of a study so that others are enabled to '[...] reach a verdict based on the significance level of their choice.' (Lehmann and Romano 2005: 64).
In the Learn−α framework we cannot simply rely on the p-value, but we propose to utilize it as an important meta data of feature value constraints / admissions. This meta data can help the system to question induced feature constraints / admissions with 'weak' evidence during the revision process. It can be seen as the strength of the system's doxastic state concerning the feature value. How is H_0 to be formulated in this case? Because the null hypothesis often claims the *absence* of any significantly observable effect, it is reasonable to propose that the null hypothesis states that a lexeme $\mathcal{L}$ does **not** admit a feature value - regarding the actual realization of a feature value as the 'effect' under investigation. In this case, a small p-value indicates that there is reason to believe in $\Box\alpha_{\mathbf{v}}^{\mathcal{L}}$ (which then means the induction of the feature value). From this point of view the underlying sampling distribution is binomial with a hypothesized probability of feature value realization (if $\mathcal{L}$ allows for the realization then it does so with a certain probability). Unfortunately the probability is not known because it varies from lexeme to lexeme and feature value to feature value. We therefore propose to stipulate a *minimum feature value realization probability* that specifies the acquisition system's maximal sensitivity to feature value realizations: p_{min}^{fv}. The null hypothesis then states that the feature value realizations belong to a population with a maximum realization probability of p_{min}^{fv} leading to the *left-sided test*: $H_0 : p \leq p_{min}^{fv}$ where $X \sim \mathbf{B}(p, N)$ (this denotes binomial distribution with probability p and sample size N). Without the provisioning of this kind of threshold no hypothesis can be formulated in the ideal world setting. The same holds for the application of the Neyman-Pearson theory introduced below and for the application of statistical estimation theory to Learn−α in

general. Using p^{fv}_{min} the binary question of whether a lexeme admits a feature value or not is recast into a continuous view: there is *always* a probability that a lexeme is *used* (or realized) with the feature value. If this probability is below p^{fv}_{min} then this usage is supposed to be excluded from the lexicon. It is important to note that p^{fv}_{min} is *not* a threshold against which relative frequencies are *compared* during the hypothesis test. It is a parameter that allows one to *formulate* a hypothesis.
H_0 may also be formulated the other way round ($H_0 : p > p^{fv}_{min}$), which leads to a *right-sided test.*

The p-value of a binomial left-sided hypothesis can be computed using the cumulative distribution function (cdf) of either the binomial or the Beta distribution (cf. Weerahandi 1995: 49–55). Here are the formulae using the binomial cdf:

$$\pi(x) = 1 - F_{\mathbf{B}(p^{fv}_{min},N)}(x-1) \tag{2.26}$$

where x is the observed number of feature value realizations in $E^{\mathcal{L}}_{\alpha}$ and $N = |E^{\mathcal{L}}_{\alpha}|$ the sample size.
For right-sided tests ($H_0 : p \geq p^{fv}_{min}$) the p-value is:

$$\pi(x) = F_{\mathbf{B}(p^{fv}_{min},N)}(x) \tag{2.27}$$

Note that a binomial distribution can only be assumed if the single observations are mutually independent, which might be a simplification. We do not pursue the impact on the theory if this assumption does not hold.

Assumption 7 *The single observations (references as per definition 23) are mutually independent.*

In addition the probability distributions of the values of a given feature are not independent of each other. But hypothesis testing in the multivariate setting is much more complicated and it is questionable whether it would be superior to the univariate model. Multinomial hypothesis testing (based on *multinomial goodness-of-fit*) would be required if we were interested in the exact likelihood that a distribution vector $\langle p_1, ..., p_n \rangle$ (representing the probabilities of the individual feature values) fits an observed sample. The focus here, however, is on the decision whether individual feature values are admitted or not. So for the time being we restrict the theory to the following assumption:

Assumption 8 *The fact that* $\sum_i p(v_i) = 1$ *for the values* $v_i \in V(F)$ *can be neglected in practice.*

For an illustration of the equations above assume a $p_{min}^{fv} = 0.001$ and a sample size of $N = 1000$. According to the gold standard of A. Korhonen[19], the verb `communicate` shows a relative realization frequency of 0.008 for the SCF 104 (intransitive + sentential complement with `that`). If it had realized this feature 8 times in the sample, the p-value would be:
$1 - F_{\mathbf{B}(0.001,1000)}(7) = 10^{-5}$, significant evidence for rejecting the null hypothesis that the verb does not allow this SCF. If it had occurred with that SCF only 2 times the p-value would be $\pi = 0.26$, no significant result against H_0. If the null hypothesis would claim that the verb *does* allow this SCF and if 0 references would have been observed, the p-value would be: $F_{\mathbf{B}(0.001,1000)}(0) = 0.37$. This is again not significant, but a more significant result cannot be achieved given the small sample size (see 2.3.5 for required sample sizes in the ideal world model). A higher sample size of $N = 5000$, for example, would yield a p-value of 0.0067.

Error Probabilities, Hypothesis Testing and Optimal Decisions

From the point of view of the theory set forth in Neyman and Pearson (1933) testing of a null hypothesis without an explicit alternative hypothesis is meaningless. While Fisher emphasized the objective measurement of the significance of a particular outcome of an experiment, Neyman and Pearson aimed at the *reduction of the potential error* of the decisions based on an experiment. That means that the committed decision errors during the (hypothetical) ongoing repetition of the same experiment should be minimal on the long run, justified by the *frequentist principle*[20]. The framework of two competing hypotheses, the hypothesis H and its alternative K, allows for a straightforward account for the probabilities of *type I errors* (rejecting H in favor of K while H is true) and *type II errors* (accepting H while it is false)[21]. To see why this is the case, consider that the two hypotheses allow to split up the parameter space into two mutually exclusive subsets $\Theta_H \cup \Theta_K = \Theta$.[22]

[19] http://www.cl.cam.ac.uk/~alk23/subcat/subcat.html, last checked on 03.06.2009.

[20] In Berger (2003: 3) expressed as: 'In repeated practical use of a statistical procedure, the long-run average actual error should not be greater than (and ideally should equal) the long-run average reported error.'

[21] Because the hypothesis H usually stands for the null hypothesis (the absence of a measurable effect), a type I error is often called *false positive* and a type II error a *false negative* in the literature.

[22] Under this view a decision problem can be cast as a class of distributions $\mathcal{P} = \{P_\theta, \Theta\}$ where $\theta \in \Theta$. For hypothesis testing in which a decision is to be made among two competing hypotheses, the class $\mathcal{P}$ can be divided into two mutually exclusive subclasses H and K ($H \cup K = \mathcal{P}$) so that H contains the distributions for which the hypothesis is true and K, the class of alternatives, contains those for which it is false. See Lehmann

Because the decisions are to be made on ground of a sample $X = x$, so that $\delta(x) = d_H$ or $\delta(x) = d_K$, respectively, the sample space can be split up into two complementary regions A and R so that H is accepted if x falls into A and rejected otherwise. A is then called the *region of acceptance* and R the *region of rejection* or the *critical region*. The probability of a type I error is $P_\theta\{\delta(x) = d_K\} = P_\theta\{x \in R\}$ for all $\theta \in \Theta_H$ and the probability of a type II error is $P_\theta\{\delta(x) = d_H\} = P_\theta\{x \in A\}$ for all $\theta \in \Theta_K$. To arrive at an *optimal decision* based on x there are two strategies: either one assigns costs (or losses) to the various errors[23] or one assumes uniform costs. In the latter case one usually assigns an upper bound on the probability of rejecting H while it is true (committing a *type I error*) using an (again arbitrary) *level of significance* α_e (we use α_e in order to differentiate it from α_s, the significance level in Fisherian testing):

$$P_\theta\{x \in R\} \leq \alpha_e \quad \text{for all} \quad \theta \in \Theta_H. \tag{2.28}$$

This condition states that the probability of observing x in the critical region should be lower than or equal to the level of significance for all probability distributions that belong to the null hypothesis.
Along with minimizing the probability of a type I error it is desired to minimize the probability of a *type II error* (accepting H when it is false) by maximizing:

$$P_\theta\{\delta(x) = d_K\} = P_\theta\{x \in R\} \quad \text{for all} \quad \theta \in \Theta_K. \tag{2.29}$$

The probability of rejecting hypothesis H for a given $\theta \in \Theta_K$ is called the *power of the test* against the alternative θ. Seen as a function of θ, the formula in 2.29 is called *power function* $\beta(\theta)$. This is an important feature of the theory because it helps to assess the quality of a test. The sample size, for example, is a crucial factor for the power.
According to Fisher and Jeffreys, however, (see Berger 2003: 3) the procedure falls short of accounting for the variation in evidence. In Learn−α, we therefore propose to compensate that disadvantage by retaining the p-values connected to particular decisions as meta data which can be reassessed whenever necessary.

---

and Romano (2005).

[23] In this case $L(\theta, d)$ is the loss if parameter θ were true (or the variable X were truly underpinned by distribution P_θ) and decision d were made. The *risk* of the decision is the average loss on the long run and is expressed by the *risk function* $R(\theta, \delta) = E[L(\theta, \delta(X))]$ (Lehmann and Romano 2005). In the version of Learn−α developed in this thesis, we will assume uniform risk functions. It would be interesting to explore how a system could assign different losses to different decisions, for example to incorporate the number of counterexamples that are ignored in the experience set.

To find a theoretical optimum according to the above equations one has to *randomize* the test. A randomized test is completely characterized by the *critical function* $\phi(x)$ (with $0 \leq \phi(x) \leq 1$). The probability of rejection is then $E_\theta\phi(X) = \int \phi(x)dP_\theta(x)$ Lehmann and Romano (2005: 58). The problem is to find a ϕ that maximizes the power of the test (2.30) while satisfying the condition in 2.31:

$$\beta_\phi(\theta) = E_\theta\phi(X) \quad \text{for all} \quad \theta \in \Theta_K. \tag{2.30}$$

$$E_\theta\phi(X) \leq \alpha_e \quad \text{for all} \quad \theta \in \Theta_H. \tag{2.31}$$

In the case of simple hypotheses (that is only one distribution per hypothesis) the problem of hypothesis testing is completely specified by 2.30 and 2.31 and the determination of a uniformly most powerful test (UMP) can straightforwardly determined. In many situations, however, the hypotheses are composite as in the binomial case, in which, say, $K : \theta \leq \theta_0$ is to be tested. It can be shown (Lehmann and Romano 2005: 65) that for distributions with monotone likelihood ratio in their statistics $T(x)$ (which is the case for all one-parameter exponential distributions such as the binomial one) there exists a UMP test for testing $H : \theta \leq \theta_0$ against $K : \theta > \theta_0$ given by:

$$\phi(x) = \begin{cases} 1 & \textbf{if} \quad T(x) > C \\ \gamma & \textbf{if} \quad T(x) = C \\ 0 & \textbf{if} \quad T(x) < C \end{cases} \tag{2.32}$$

The values of C and γ can be determined by solving the equation:

$$E_{\theta_0}\phi(X) = \alpha_e \tag{2.33}$$

The solution for testing $H : \theta \geq \theta_0$ against $K : \theta < \theta_0$ is given by interchanging the inequalities in 2.32.
In the randomized setting, α_e will be exactly the probability of committing type I errors. In non-randomized, discrete cases, the solution of 2.33 has to be an approximation that guarantees that the real expected probability of committing type I errors $\hat{\alpha}$ does not exceed the upper bound α_e. $\hat{\alpha}$ is called the *size of the test*[24].

[24]The size of the test is usually denoted by α, however, to avoid confusion, we use the notion $\hat{\alpha}$.

Hypothesis Testing in Learn−α: The Ideal World

How does the Neyman-Pearson framework apply to the ideal, noise-free world of Learn−α?

Let H be the (null) hypothesis that $-\alpha^{\mathcal{L}}_{\mathbf{v}}$ is the case in the optimal model[25] and K the hypothesis that it is not. In the ideal world one single reference would already be sufficient to reject H and to decide in favor of K. This means that the rejection region is $[1, \infty]$, the acceptance region is $\{0\}$ and the critical value is $C = 0$. In the first case it does not matter how much data is supplied. In the second case it does, because if the decision is made on ground of zero references there is always a certain amount of uncertainty. This amount is a function of $N = |E^{\mathcal{L}}_{\alpha}|$ and the actual realization probability $p(\alpha^{\mathcal{L}}_{\mathbf{v}})$ (which is < 1). If H is accepted in a K-world the consequence is a false negative (FN) in the lexicon. Let β_{FN} be the maximum error rate that this will happen on the long run. This means that N must be such that $P_K[x = 0] \leq \beta_{FN}$ (let P_K be the probability in the K-world). In the binomial setting this amounts to:

$$P_K[x = 0] = \binom{N}{0} \quad p(\alpha^{\mathcal{L}}_{\mathbf{v}})^0 \quad (1 - p(\alpha^{\mathcal{L}}_{\mathbf{v}}))^N = (1 - p(\alpha^{\mathcal{L}}_{\mathbf{v}}))^N \leq \beta_{FN} \quad (2.34)$$

The inequation can be solved for N. Because $p(\alpha^{\mathcal{L}}_{\mathbf{v}})$ is unknown, however, one has to make a worst-case assumption and replace it by p^{fv}_{min}:

$$N \geq \frac{log(\beta_{FN})}{log(1 - p^{fv}_{min})} \quad (2.35)$$

Based on this inequation the decision function can be formulated as follows:

$$\delta(x) = \begin{cases} \Box\alpha^{\mathcal{L}}_{\mathbf{v}} & \textbf{if} \quad x > 0 \\ \Box - \alpha^{\mathcal{L}}_{\mathbf{v}} & \textbf{if} \quad x = 0 \quad \textbf{and} \quad N \text{ satisfies ineq. 2.35} \\ \text{undecided} & \textbf{else} \end{cases} \quad (2.36)$$

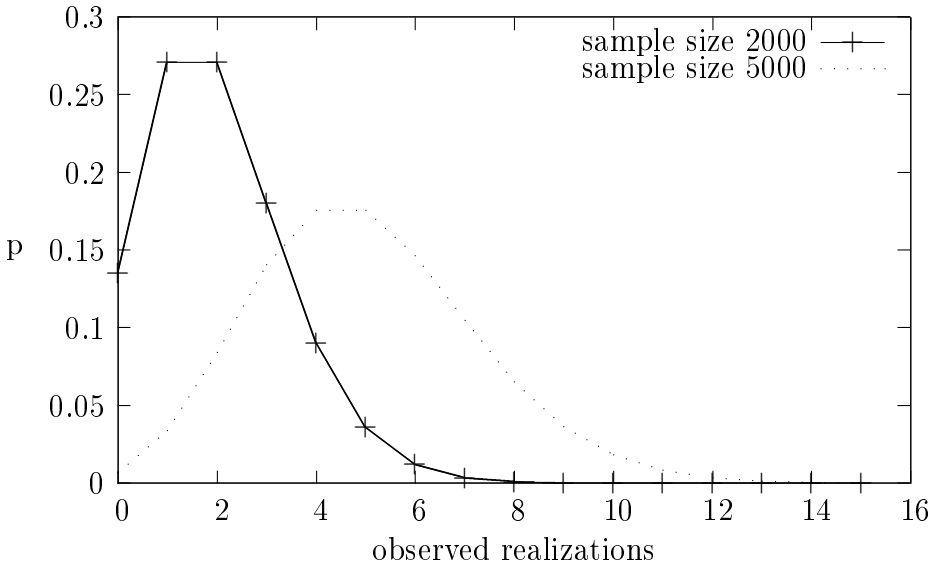
To get a feel for the required sample size depending on p^{fv}_{min} and β_{FN}, we compiled the following table:

[25] In line with section 2.3.5 we treat the feature value constraint as the null hypothesis, this means the feature value realization is the *effect* under investigation.

β_{FN} / p^{fv}_{min}	0.1	0.05	0.01	0.005	0.001
0.001	2301	2994	4603	5296	6904
0.002	1150	1496	2300	2647	3450
0.003	766	997	1533	1763	2299
0.004	574	747	1149	1322	1723
0.005	459	598	919	1057	1378
0.006	383	498	765	880	1148
0.007	328	426	656	754	983
0.008	287	373	573	660	860
0.009	255	331	509	586	764
0.01	229	298	458	527	687

The table shows how sensitive the choice of the minimal sample size is to the setting of the parameters. Figure 2.7 plots two 'worst-case' distributions with $p^{fv}_{min} = 0.001$ and different sample sizes. The point where the graph crosses the y-axis (x=0) is the actually expected error rate β_{FN}.

Figure 2.7: Distribution of Feature Value Realizations



Likelihood Ratio Tests

The hypothesis testing procedures described so far are *exact tests* because they do not approximate the model by distributions such as the normal distribution or the χ^2 distribution. In this section we briefly discuss an asymptotic test method that is widely used in statistical NLP: the Likelihood Ratio Test (LRT). The test was introduced by Neyman and Pearson (1928) and is a popular test due to a couple of features (Lehmann 2006): a) it has intuitive appeal because it is based on the likelihood $L(\theta|x)$ of parameters θ, b) it agrees with tests obtained from other principles such as being UMP unbiased and c) it shows good asymptotic properties under certain conditions (cf. Dunning 1993, who claims that this test shows good performance even for small samples)[26]. The LRT produces decisions based on the *likelihood ratio*:

$$\lambda(x) = \frac{sup_{\theta \in \Theta_H} \quad L(\theta|x)}{sup_{\theta \in \Theta_K} \quad L(\theta|x)} \tag{2.37}$$

This ratio compares the best explanation of the data given the hypothesis H with the best explanation of the data given the the alternative hypothesis K. Under certain restrictions the test can be asymptotically approximated with the χ^2 distribution. One restriction is that the number of free parameters D_1 in the numerator of the fraction must be less than the number of free parameters D_2 in the denominator. Then the quantity $-2log(\lambda(x))$ asymptotically approximates $\chi^2_{D_2-D_1}$.
The test can be applied to Learn−α, again with the null hypothesis $H = \Box - \alpha^{\mathcal{L}}_{\mathbf{v}}$. Let κ be the $1-\alpha_e$ percentile of $\chi^2_{D_2-D_1}$, then the decision rule is:

$$\delta(x) = \quad \begin{cases} \Box\alpha^{\mathcal{L}}_{\mathbf{v}} & \textbf{if} \quad -2log(\lambda(x)) \geq \kappa \\ \Box - \alpha^{\mathcal{L}}_{\mathbf{v}} & \textbf{else} \end{cases} \tag{2.38}$$

For subcategorization acquisition, this procedure has been applied in Sarkar and Zeman (2000) and Korhonen (2002).

Pointwise Mutual Information

This subsection presents a hypothesis testing approach based on an information-theoretical account of word associations. It can be used to detect multiword

[26]Lehmann, however, does not find any of these features convincing. His article discusses a large class of situations in which LRT performs poorly. But for the class of problems we are interested in, namely the testing of a simple hypothesis against a simple alternative, 'the Neyman-Pearson Lemma completely vindicates the LR-test, which always provides the most powerful test.' (Lehmann 2006: 2).

expressions or, more general, lexicalized and/or institutionalized expressions that consist of more than one distinct free graphical entity, free in the sense that each graphical entity refers to some different lexeme in isolation. Let us suppose that the grammar comprises some rules which have the effect that a graphical entity $\mathbf{g}_0$ may be constructed out of two other graphical entities $\mathbf{g}_1$ and $\mathbf{g}_2$ so that $\mathbf{g}_0$ is the graphical representation of a lexeme $\mathcal{L}$. A typical grammatical rule that could serve as an example is the English noun compounding rule: $N \rightarrow NN$. If $\mathcal{L}$ is realized in the experience set then the two distinct graphical entities jointly occur in it more often than chance, in other words they are supposed to show some measurable association. One way to measure this kind of association is pointwise mutual information (MI), cf. Church and Hanks (1990). Let $P(\mathbf{g}_i)$ be the probability that the graphical entity $\mathbf{g}_i$ occurs in an utterance[27].

Then the pointwise MI value is calculated as follows:

$$MI = log \frac{P(\mathbf{g}_0)}{P(\mathbf{g}_1)P(\mathbf{g}_2)} \tag{2.39}$$

On the ground of a sufficiently large experience set E the MI can be approximated using the counts $c(\mathbf{g}_i)$:

$$\hat{MI} = log_2 \frac{c(\mathbf{g}_0)|E|}{c(\mathbf{g}_1)c(\mathbf{g}_2)} \tag{2.40}$$

The condition of using a 'sufficiently large' experience set can be satisfied by deferring the calculation until the point where E contains a minimum number of references for $\mathcal{L}$.

If the system is provided with a minimal MI value MI_{min} that has to be reached in order to believe into the hypothesis K that $\mathbf{g}_0$ is indeed the graphical representation of a lexeme in the optimal model (and to reject the null hypothesis H), then the decision function (this time ranging over E) is defined as:

$$\delta(E) = \begin{cases} \Box(\exists x)\ \mathcal{L}x\ \&\ \textsc{g}x\mathbf{g_0} & \textbf{if} \quad \hat{MI} \geq MI_{min} \\ \Box - (\exists x)\ \mathcal{L}x\ \&\ \textsc{g}x\mathbf{g_0} & \textbf{else} \end{cases} \tag{2.41}$$

[27]Note that in our theory a) the number of occurrences is related to the size of the experience set (the number of utterance) and not to the size of a corpus (the number of words) and b) there is no window w of joint occurrence as in Church and Hanks (1990: 77) because the use of construction rules such as the English compounding rule makes the specification of a window obsolete.

2.4 Noise

The account of artificial lexical acquisition provided so far is based on the assumption (1) in subsection 1.4.3 that a) the grammar of the learning system is sound and b) the system's experience always consists of grammatical utterances. Deviations from this assumption (unintended grammatical consequences or ungrammatical utterances) are usually subsumed under the notion 'noise'. Further, the initial and induced lexicons can also contain noise (unintended lexical entailments) and it is important to note that this kind of noise can mislead the system as much as the noise in the experience or the grammar can do.
Neglecting noise in the previous sections allowed us to give a straightforward logical account of Learn$-\alpha$, but in practice the LE can neither avoid nor ignore it. It was already mentioned that the input texts and the grammar of a real learning system are afflicted with this kind of noise - and this section is dedicated to a formal account of what this means.

2.4.1 The Noise Model in Learn$-\alpha$

In information theory the formal treatment of 'noise' has its root in Shannon (1948). This theory deals with the question of how the information content of a message can be reproduced if the message is transmitted over a channel distorted by noise (*noisy channel model*). The notion 'noise' in Learn$-\alpha$, however, has a slightly different meaning: it deals with the interplay of the major components of a learning system: the experience set, the grammar, the lexicon, and the acquisition component. In general, 'noise' refers to the *unintended output of such a system component.* In such a view, the component is seen as a device that generates items - and from time to time a generated item is undesired by the LE. So, as with other parts of this theory, noise is relative to the intentions of the LE. Because of the complexity and/or unknown parameters of the component the generated error is random, in other words the generation of the component is distorted by errors that can only be grasped by a statistical model with some unknown parameters - the error probabilities. Noise is most of the time a mixture of errors stemming from various sources and also having different effects on the analysis of utterances as well on lexical acquisition.

Different types of sources of noise can be distinguished: some sources are a-priori, most notably the grammar (which is not changed during learning) and the initial lexicon. Other sources are a-posteriori: the experience set

and induced lexical feature constraints. Because the grammar is a 'black box' in Learn$-\alpha$, 'noise' cannot directly refer to rules (or 'errors' in rules); it acquires meaning only in the context of the actual analysis of an utterance in terms of *u-models*. These can be related to the intention to the LE by the intention function I(u) introduced in section 2.2.1. Because the u-models are the basic pieces of information for the observation schema, we suggest that the respective errors are called *observation errors*. Two kinds of observation errors can be distinguished, perfectly mirroring the notions of 'type I' and 'type II' errors (cf. section 2.3.5, page 116):

type-I-observation-error : $\Theta \vdash \mathrm{M}u\mu$ **although not** $\mu \in I(u)$

type-II-observation-error : **not** $\Theta \vdash \mathrm{M}u\mu$ **although** $\mu \in I(u)$

To keep it simple, we assume that the occurrence of these errors is binomial distributed with a certain error probability ϵ that is a characteristic parameter of the component C. We call ϵ the *noise level* of C.

2.4.2 Noise in the Experience Set

Noise in the experience set E (the input of the system) is defined over utterances $u \in E$ that are not linguistically well-formed in view of the LE. In other words, the LE wants the NLP system, at least in its optimal state, to map these utterances to empty sets of u-models. (10) is an example for such an utterance in the BNC (these are relatively rare) that is considered ungrammatical from the viewpoint of the author (in the role of a LE):

(10) ⌊B38:2641⌉: plus a snakebite We didny now how to

A very prominent type of noise in the (written) experience set is a *spelling mistake*. Although this is a problematic one because it can hamper *lexeme detection* it should not be further treated in this thesis.
Assuming a binomial model underpinning the experience's noise, the corresponding noise level ϵ_E can straightforwardly be estimated via MLE. This leads to the following definition:

Def. 26 (noise in the experience set) *Let* $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be a consistent symbolic language model with the corresponding optimal model* $\mathbf{M}_{opt}$ *and* $E \subset D_u$ *a non-empty finite set of utterances that are fed into* $\mathbf{M}$. *Let* $E = E^+ \cup E^-$ *where* $E^- = \{u \mid \mathbf{M}_{opt} : u \Rightarrow \emptyset\}$. *Then* ϵ_E *is called the noise level of* E*:*

$$\epsilon_E = \frac{|E^-|}{|E|} \tag{2.42}$$

Noise in the experience set can be reduced by selecting well-maintained corpora or texts that are very 'trustworthy' with regard to their linguistic quality. But in general, it imposes a big challenge to the theory of Learn$-\alpha$, because the system cannot properly estimate ϵ_E by itself. All it can do is to observe the overall parser success rate and compare it with the rate yielded after the processing of some 'authoritative' text. If deviation is too high this might indicate that the noise level is too high to warrant proper lexical acquisition. We leave that issue for further research.

2.4.3 Noise in the Grammar

Intuitively, if the set of generated u-models for an error-free utterance u does not match $I(u)$ no matter what is entailed in the current lexicon then it must be an error due to the grammatical component of the system[28]. Normally these errors are due to *undergenerating* rules (evoking type-II-observation-errors) or *overgenerating* rules (evoking type-I-observation-errors). From this perspective the grammar 'generates' errors - it adds noise to the whole system. The 'no matter what is entailed in the lexicon' portion of the premises is perfectly modeled by building Λ_{opt} into the definition. We cannot, however, use the current definition of the optimal model any longer because if the grammar is such that the zero-lexicon-model $\mathbf{M}_\emptyset$ is not adequate (which is now possible) then the optimal model cannot be adequate either - consequently we have to drop the condition of proper extension (cf. definition (17) and (18)). It will be replaced by the condition that the optimal model has to optimally meet the LE's intention - or the other way round it has to show the minimal overall error rate of the system, to be defined as follows:

Def. 27 (overall error rate) *Let* $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be a consistent symbolic language model. Let I(u) be the function that maps utterances to a set of u-models intended by the LE and* $\mathcal{M}^u$ *be the set of u-models that* $\mathbf{M}$ *assigns to u. Let the error rate of the grammar w.r.t. u be* $\epsilon^u_{\mathbf{M}}$ *according to a performance measure defined by the LE as a function of the given sets* $\mathcal{M}^u$ *and I(u). Then* $\epsilon_{\mathbf{M}}$*, the overall error rate of the model, is the average of* $\epsilon^u_{\mathbf{M}}$ *over all* $u \in D_u$*.*

We leave it to the LE to define the performance measure on particular utterances. As an example, he might use F-measure, which combines precision (P) and recall (R) into one single measure of overall performance (Manning and Schütze 1999: 269). We will use the subscript u to indicate that the measure refers to the system's performance on a particular utterance u:

[28] Abstracting away from other possible failures such as tokenization, parser timeouts and the like.

$$F_u = \frac{1}{\gamma \frac{1}{P_u} + (1-\gamma)\frac{1}{R_u}} \tag{2.43}$$

Here γ determines the weighting of precision and recall, respectively. Precision and Recall can be defined in terms of intention and generated set of u-models:

$$\begin{aligned} P_u &= \frac{|\mathcal{M}^u \cap I(u)|}{|\mathcal{M}^u|} \quad \textbf{if } \mathcal{M}^u \neq \emptyset \textbf{ else } 0 \\ R_u &= \frac{|\mathcal{M}^u \cap I(u)|}{|I(u)|} \quad \textbf{if } I(u) \neq \emptyset \textbf{ else } 0 \end{aligned} \tag{2.44}$$

Note that if $\mathcal{M}^u = \emptyset$ or $I(u) = \emptyset$ then P and R are zero (there are no true positives) and F is zero as well. This means that F_u is always zero for utterances that the LE judges ungrammatical. This might be undesired because the performance of a system that rules out ungrammatical utterances should be considered higher than the performance of a system that does not. We leave it to the LE to adapt the performance measure of a model accordingly. To complete this excursus we want to note that the error rate of the grammar w.r.t. u would be:

$$\epsilon^u_{\mathbf{M}} = 1 - F_u \tag{2.45}$$

Given the notion of overall error rate, the definition of the optimal model follows straightforwardly:

Def. 28 (optimal extension revised) *The model* $\mathbf{M}_{opt}$ *is said to be an* optimal extension *of a model* $\mathbf{M}$ *if and only if* $\mathbf{M}_{opt}$ *extends* $\mathbf{M}$ *and there is no other model* $\mathbf{M}'$ *that extends* $\mathbf{M}_{opt}$ *such that* $\epsilon_{\mathbf{M}_{opt}} > \epsilon_{\mathbf{M}'}$. $\mathbf{M}_{opt}$ *is called the* optimal model *and* Λ_{opt} *the optimal lexicon.*

We can now return to the definition of 'noise in grammar'. The idea underlying this definition is that the 'misbehavior' of an optimal model cannot be due to incorrect or missing lexical constraints. It must be the grammar that is responsible for the system's overall error rate.

Def. 29 (noise in grammar) *Let* $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be a consistent symbolic language model and* $\mathbf{M}_{opt}$ *its optimal model.*
Then the noise level of the grammar is the overall error rate $\epsilon_{\mathbf{M}_{opt}}$.

Noise in the grammar is one of the biggest obstacles of lexical acquisition. So it is extremely important to reduce the noise level as much as possible.

Interestingly, an implementation of Learn$-\alpha$ can help with that. By investigation of the alleged references for wrongly induced feature constraints the information entailed in it can be systematically exploited to detect deficiencies in the grammar[29].

2.4.4 Noise in the Lexicon

Per definition, missing or erroneous belief of alleged lexical facts leads to a non-optimal model. Because all the errors of an optimal model $\mathbf{M}_{opt}$ are due to the noise in the grammar, it is logical to conclude that all errors of a non-optimal model $\mathbf{M}$ that vanish in $\mathbf{M}_{opt}$ must be due to noise in the lexicon of $\mathbf{M}$. This leads to the following definition:

Def. 30 (noise in lexicon) *Let* $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be a consistent symbolic language model and* $\mathbf{M}_{opt}$ *its optimal model.*
Then the noise level of the lexicon is $\epsilon_\Lambda = \epsilon_{\mathbf{M}} - \epsilon_{\mathbf{M}_{opt}}$.

ϵ_Λ measures the distance between a lexicon Λ and its optimum Λ_{opt} in terms of noise. Note that - per definition - the noise level of a lexicon Λ whose corresponding grammar has a noise level of $\epsilon_\Gamma = 1$ is always 0, regardless of what information is encoded in Λ. Once again this shows the relativity of the lexicon.
The noise level of the lexicon is a target of reduction similar to the one of the grammar. Here two different sources of noise must be distinguished: a) the a-priori source coming with the initial lexicon, which is quite similar to the noise in the grammar and b) the a-posteriori generated source of noise by inducing unintended feature constraints. The latter is especially important because the system might be endangered by operating under a vicious circle: the (improperly) induced lexicon Λ' generates noise that negatively impacts induction - yielding an even 'worse' lexicon Λ'' and so forth.
Moreover we need to differentiate the noise level ϵ^u_Λ due to *underspecification* invoking type-I-observation-errors from the noise level ϵ^c_Λ due to *unwarranted lexical constraints* responsible for type-II-observation-errors. We come back to this point in subsection 2.4.6.

2.4.5 Effective Noise

Crucial questions for a theory of artificial lexical acquisition are a) how much can the learning system approximate the theoretical optimum?, b) how effi-

[29] This fact was heavily made use of during the implementation of our acquisition system. It somehow provokes the slogan: 'show me what you learn and I show you what's wrong with your grammar'.

ciently does learning take place? and c) how accurate are the induction steps? Let us subsume accuracy and efficiency of learning under a cover term: *learning performance*. Noise has certainly an impact on learning performance. But in order to understand how it is influenced, we have to reconsider how learning takes place, we have to recall the logic of lexical acquisition.
How is the logic of Learn$-\alpha$ affected by noise? Do we have to modify, for example, the schemata for observation (2.15) or induction (2.20), respectively? No, we think that they still hold. It is the *premises* of an observation step that may be wrong, not the implication itself. Recall the observation schema, which we repeat here in a simplified version (p stands for: utterance u is grammatical, α stands for $\alpha^{\mathcal{L}}_{\mathbf{v}}$):

$$(\Box p \;\&\; (-\alpha \supset -p)) \supset \Box\alpha \tag{2.46}$$

So, if there is an ungrammatical utterance in E ($\Box - p$) - because of ϵ_E - and the system assumes $\Box p$ then the premise is wrong. Likewise, if the grammar/lexicon rules out an utterance on grounds of $-\alpha$, although this implication is not intended by the LE (an effect of ϵ_Γ and ϵ_Λ, respectively), then one of the premises, namely $(-\alpha \supset -p)$ is again wrong. Wrong premises can lead to wrong conclusions and thus to unintended references. The examples in (11) illustrate a case in point:

(11)
a. the very fact
b. the very poor man
c. the very poor
d. the very gnarf

Utterance (11a) and (11b) show that `very` can be used as adjective and adverb, respectively. Let us now suppose the initial lexicon would entail the latter but not the former information, that means the system assumes that `very` can not be used adjectively - as part of ϵ_{Λ_0}. Consequently, the u-model of (11d) in which the unknown lexeme `gnarf` is a noun is ruled out. The only derived u-model is the one in which `gnarf` is an adjective, modifying a $\emptyset$-noun, in analogy to (11c). Based on the premise

$$(-\text{ `}\texttt{gnarf}\text{ is adjective'} \supset -\text{ `(11c) is parsable'})$$

the system will conclude that `gnarf` is an adjective although it might be a noun in reality.
If `gnarf` *does not admit* adjectival usage, then (11d) is an unindented reference, to be defined as follows[30]:

[30]This definition replaces the notion *false positive example* introduced in chapter 1.

Def. 31 ((un)intended reference) *Let* $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be a consistent symbolic language model and* $u \in D_u$ *a reference for a feature value* $\alpha^{\mathcal{L}}_{\mathbf{v}}$*:* $u \mapsto \Box\alpha^{\mathcal{L}}_{\mathbf{v}}$*. Then* u *is said to be an* unintended reference *if and only if* $\Box - \alpha^{\mathcal{L}}_{\mathbf{v}}$*, otherwise it is said to be* intended*.*

It is important to estimate how likely an unintended reference of a feature value is for a lexeme that does *in fact not admit* the feature value. Because this parameter can then be used in binomial hypothesis testing (cf. next section). This motivates the definition of *effective noise level*:

Def. 32 (effective noise level) *Let* $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be a consistent symbolic language model and* $\mathcal{L}$ *a lexeme for which* $\Box - \alpha^{\mathcal{L}}_{\mathbf{v}}$ *holds. The probability with which unintended references for* $\alpha^{\mathcal{L}}_{\mathbf{v}}$ *are generated by the acquisition system is called the* effective noise level of $\alpha^{\mathcal{L}}_{\mathbf{v}}$ *and denoted by* $\epsilon_{eff}(\alpha^{\mathcal{L}}_{\mathbf{v}})$*.*

If the effective noise level of a feature value $\alpha_{\mathbf{v}}$ would be equal among the class of all lexemes that do not admit $\alpha_{\mathbf{v}}$ then the observation of just one lexeme which is definitely known to belong to that class would suffice to determine $\epsilon_{eff}(\alpha^{\mathcal{L}}_{\mathbf{v}})$ for all lexemes of that class. However, because different lexemes have different properties and different impact on the generation of u-models, this cannot be assumed. But if the model is such that the deviation of the real effective noise levels from an average noise level is not too high then it is reasonable to develop a function that serves as a good approximation of the (otherwise unknown) noise levels.

Def. 33 (approximated effective noise level and effective noise level confidence interval)
Let $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be a consistent symbolic language model and* $C^{\mathcal{L}}$ *a class of lexemes for which* $\Box - \alpha^{\mathcal{L}}_{\mathbf{v}}$ *holds and* Υ *be the set of all observed effective noise levels attached to the lexemes in* C*. Let* $f(\Upsilon)$ *be an approximation function that estimates the effective noise level of* $\alpha_{\mathbf{v}}$ *based on the observations.*
Then $\hat{\epsilon}_{eff}(\alpha_{\mathbf{v}}) = f(\Upsilon)$ *is called the* approximated effective noise level of $\alpha_{\mathbf{v}}$*. It is the mean of the confidence interval* $[\underline{\epsilon}_{eff}(\alpha_{\mathbf{v}}), \overline{\epsilon}_{eff}(\alpha_{\mathbf{v}})]$*.*

2.4.6 The Noise Barrier

In section 2.3.5 it was argued that statistical estimation theory can only be applied to Learn$-\alpha$ if an absolute minimum feature value realization probability (p^{fv}_{min}) is provided. This threshold is thought to determine the maximal sensitivity of the learning system with the consequence that a certain feature value $\alpha^{\mathcal{L}}_{\mathbf{v}}$ can not be acquired if its real realization probability is below that

threshold. But given that the system's observations of feature value realizations are distorted by noise the sensitivity with regard to a given feature value becomes a function of both p^{fv}_{min} *and* the effective noise level $\epsilon_{eff}(\alpha^{\mathcal{L}}_{\mathbf{v}})$. If the latter is greater than the former and the real realization probability of $\alpha^{\mathcal{L}}_{\mathbf{v}}$ is in the interval $[p^{fv}_{min}, \epsilon_{eff}(\alpha^{\mathcal{L}}_{\mathbf{v}})]$ then the feature value is invisible to the system *because of noise* as if it was hiding behind a 'wall', which in this case we would like to call the *noise barrier* of $\alpha^{\mathcal{L}}_{\mathbf{v}}$. Intended references cannot be statistically distinguished from unintended references any more and it is the noise level that determines the sensitivity of the system. Therefore we want to introduce a second parameter of the system: the *maximally assumed effective noise level* ϵ_{max}. It is the counterpart of p^{fv}_{min} in the sense that it specifies the maximum 'insensitivity' of the system: it is the assumed 'high water mark' of the lexical feature's noise barriers. In the next subsection ϵ_{max} will be calculated on the basis of other important parameters.

The closer the real feature value realization probability $p(\alpha^{\mathcal{L}}_{\mathbf{v}})$ comes to ϵ_{eff} the more data will be required to differentiate both. In order to parametrize the system for the degree of precision we thus introduce a further system parameter: SNR, the signal-to-noise ratio. It is the ratio of $p(\alpha^{\mathcal{L}}_{\mathbf{v}})$ to ϵ_{eff} and must be greater than 1.

As we said in the introduction of this section, noise cannot be avoided in practice, which means that we cannot retain assumption (1). On the other hand - given that too much noise might prevent (considerable) improvement of a system and there is no magic bullet against it - we do not want to drop it completely. So we have revised the assumption to a softened, more heuristic, version. Note that we do not impose any restriction on the noise level $\epsilon^{u}_{\Lambda_0}$ because an initial lexicon may be considered extremely noisy due to the lexical underspecification and it is the acquisition of lexical facts that shall reduce this type of noise. The noise that is introduced by unwarranted lexical constraints in the initial lexicon ($\epsilon^{c}_{\Lambda_0}$), however, needs to be kept to a minimum because these constraints may well invoke unintended references.

Assumption 1 revised
a. $\epsilon_E \ll 1$ *(The majority of utterances in the experience set is grammatically correct.)*
b. $\epsilon_\Gamma \ll 1$ *(The grammar correctly covers the majority of phenomena that the LE aims at.)*
c. $\epsilon^{c}_{\Lambda_0} \ll 1$ *(The initial lexicon is nearly free of erroneous lexical constraints)*
d. $\epsilon_{eff} \leq \epsilon_{max}$
e. $\epsilon_{max} \ll 1$

If this assumption is violated then the system cannot be expected to learn

successfully.

2.4.7 Noise Scenarios

For the application of statistical inference the system should know the effective noise levels in order to determine whether lexemes do admit certain feature values or not. But for the computation of the noise levels the system should know which lexemes admit certain feature values or not. So the acquisition system appears to be trapped in a vicious circle. Does the system have a chance to escape it? We think yes, at least partially, but the answer can only be a *heuristic* one. For a heuristic solution in which the actual noise levels are estimated it is valuable to differentiate possible cases, which we call *noise scenarios*. These scenarios are based on three factors:

i. how many lexemes admit a given feature value $\alpha^{\mathcal{L}}_{\mathbf{v}}$?

ii. how large is their real feature value realization probability?

iii. how large is the overall (observed) feature value realization probability (i.e. the average of all observed relative frequencies)?

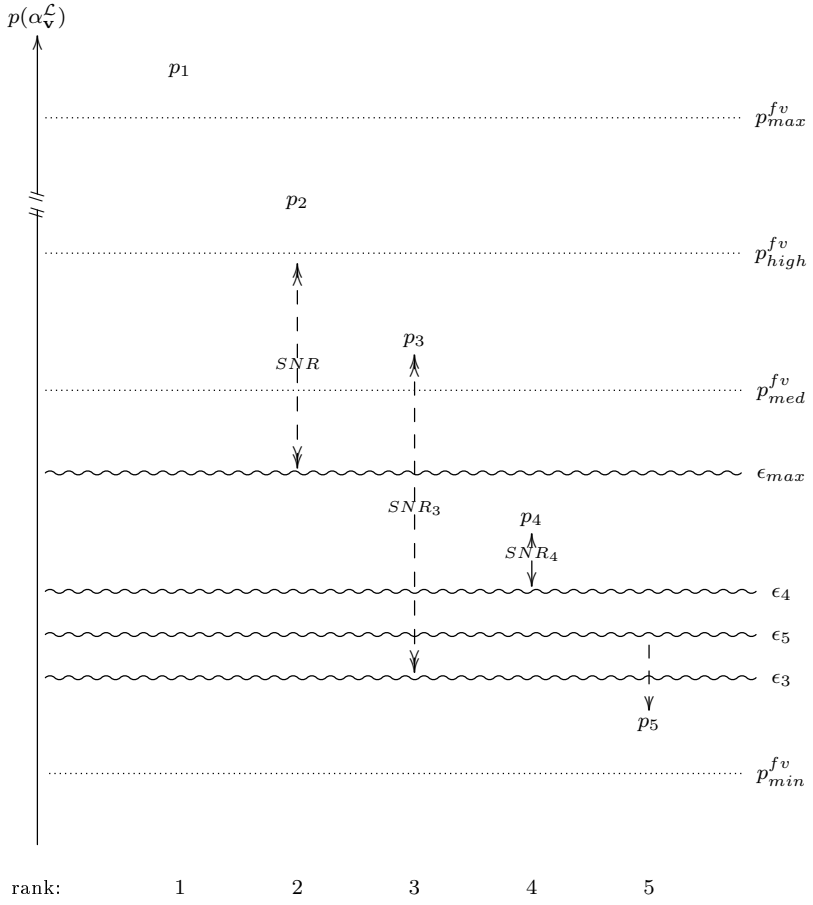
The first question is important because if only a few lexemes admit a feature value then many lexemes do not and the observed references for these can be used to estimate the noise level. Regarding the second question, let us partition the various real feature value realization probabilities. In section 2.3.5 it was argued that for the sake of hypothesis testing a minimal 'worst-case' feature value realization probability p^{fv}_{min} has to be assumed. On the other hand, it makes sense to stipulate an upper bound, p^{fv}_{max} in the sense that any observed feature value realization above this threshold should not undergo any hypothesis testing and the feature value admission should be taken for granted instead. These two thresholds form the range of interesting potential values for the real value $p(\alpha^{\mathcal{L}}_{\mathbf{v}})$. We propose to partition this range into two areas $[p^{fv}_{min}, p^{fv}_{med}]$ for the 'low' to 'medium' values and $[p^{fv}_{med}, p^{fv}_{max}]$ for the medium to very high values. It makes sense to choose the geometric mean of p^{fv}_{min} and p^{fv}_{max} for the p^{fv}_{med} in order to account for the usually Zipfian distribution of realization probabilities[31]
We further partition the upper range into two areas: $[p^{fv}_{med}, p^{fv}_{high}]$ for the medium-scale values and $[p^{fv}_{high}, p^{fv}_{max}]$ for the high values, p^{fv}_{high} being the geometric mean of the $[p^{fv}_{med}, p^{fv}_{max}]$ interval, and propose to set ϵ_{max} to p^{fv}_{high}/SNR. The intervals are depicted in figure 2.8 along with a hypothetical distribution

[31] Very often there is a 'canonical' value and some medium-range values and multiple low-frequent values. p^{fv}_{med} will be used as a threshold in another context later on.

of realization probabilities and hypothetical noise levels. Suppose there is a lexeme/feature combination and there are 5 appropriate values. The related realization probabilities are ranked in descending order. p_1 exceeds p^{fv}_{max} so that the related feature value never undergoes hypothesis testing. p_2 is the probability of a high-frequency feature value. Its assumed signal-to-noise ratio exceeds the maximally assumed signal-to-noise ratio (SNR) and can be induced with high precision (provided that ϵ_{max} is not exceeded). Induction of the feature value related to p_3, which is closer to ϵ_{max}, depends on the actual noise level which holds for this feature value. The actual SNR appears to be sufficiently high in this example. Unlike p_3, the probability p_4 seems to be too close to the related noise level in order for the related feature value to be meaningfully acquired on the long run. The acquisition of the feature value related to p_5 is impossible because it is swallowed by the noise barrier. Exceeding p^{fv}_{min} this feature value would be acquired in the ideal world.

This figure can be used to illustrate the different noise scenarios. Consider first the case in which only a few lexemes admit a given feature value. In the first scenario the realization probability is high (like p_3). For example English ditransitive verbs (`give`, `bring`) realize the ditransitive SCF very often, but the minority of verbs are ditransitive. If the observed overall realization probability is high (scenario 1A) this means a high noise level. In this case it depends on the actual SNRs whether acquisition works properly. If the observed overall realization probability is low (scenario 1B), however, this is the ideal situation because the few lexemes can well be singled out.
If the real realization probability is low (scenario 2) the situation is much more difficult. English verbs that allow for locative inversion might be a case in point. A high observed overall realization probability (scenario 2A) suggests that the feature value is swallowed by the noise barrier (similar to p_5). If it is low (scenario 2B) it depends on the actual SNR again (as with p_4).
Suppose there are many lexemes admitting the given feature value. In the third scenario the realization probability is high, for which the class of English transitive verbs may be an instance. If the observed overall realization probability is relatively high (scenario 3A) a certain amount of noise may be the cause and it depends once more on the actual SNR whether the feature value can be effectively acquired. A relatively low overall realization probability (scenario 3B) is again a nearly noise-free ideal situation.
The fourth scenario in which the realization probability is low is the worst case and requires special attention. Because we are not aware of a good example, this may be a hypothetical scenario. A high overall realization probability (scenario 4A) means a high noise level and it is likely that the

Figure 2.8: Hypothetical Distribution of $p(\alpha_{\mathbf{v}}^{\mathcal{L}})$



feature value is again hidden below the noise barrier. In case it is low, the situation is still critical because the average of feature value realizations cannot be distinguished from the noise level - it *is* the noise level. Only those (few) lexemes that show a relatively high feature realization will then be singled out as the 'real' cases.
The following table summarizes the discussion:

Scenario	Lexemes admitting FV	Real FV Real. Prob.	Overall FV Real. Prob.	Consequence for Learn$-\alpha$
1A:	few	high	high (noise)	SNR!
1B:	few	high	low	good recall/precision
2A:	few	low	high (noise)	noise barrier
2B:	few	low	low	difficult (SNR)
3A:	many	high	high (noise?)	SNR!
3B:	many	high	low	good recall/precision
4A:	many	low	high (noise)	noise barrier
4B:	many	low	low	critical

In section 2.8.5 we will propose a simple algorithm for the *automatic noise level estimation* of a given feature value $\alpha_{\mathbf{v}}$ based on its characteristics.

2.5 Lexical Neutralization

This section concerns the second obstacle to lexical acquisition: the problem that lexical features become indeterminable because of systematic ambiguity. It discusses various kinds of ambiguities that prevent the acquisition of a lexeme's feature value. These ambiguities *hide* the underlying properties of the lexemes, comparable to phonological processes which annihilate contrastive phonological features - a process that is called 'neutralization'. Here, we extend this notion to the annihilation of lexical features, covered by the term *lexical neutralization*. Before we outline the details, it is important to distinguish two different kinds of lexical neutralization:

a. Contexts in which a given lexeme is visible, but (at least) one of its lexical features is invisible. We will call this effect: *(lexical) feature value neutralization* or shorter for convenience: *feature neutralization* (cf. subsection 2.5.1).

b. Contexts in which a given lexeme is realized but invisible as a whole. We call this effect the *neutralization of lexeme determination* (cf. subsection 2.5.3).

2.5.1 Feature Value Neutralization

Regarding examples in this section, let VAL ('valence') be a feature with the appropriate values $\{\mathbf{intrans}, \mathbf{mtrans}, \mathbf{ditrans}\}$ and SC ('sentential complement') a feature with the appropriate values $\{\mathbf{none}, \mathbf{to}, \mathbf{that}\}$. The meaning of the features and feature values will be clear from the examples to come. Resuming example (9) of section 2.3.4, repeated below as (12) for convenience, it can be stated that the lexeme `want` in utterance (a) realizes the feature values $\langle[\text{VAL } \mathbf{intrans}], [\text{SC } \mathbf{to}]\rangle$. In (b) and (c) the verbs `love` and `marry` realize the feature values $\langle[\text{VAL } \mathbf{mtrans}], [\text{SC } \mathbf{none}]\rangle$ and $\langle[\text{VAL } \mathbf{intrans}], [\text{SC } \mathbf{none}]\rangle$, respectively.

(12)
a. I wanted to come.
b. He loved her.
c. He married to get rich.

These statements are possible because we *know* which feature values these verbs admit[32]. Turning to the perspective of the learning system that does not know the correct values of these features, the question arises whether the examples can be used as unambiguous references for SC. Obviously,

[32] If we have some sort of Λ_{opt} in our mind.

while (b) proves that a sentential complement is not obligatory, (a) and (c) don't prove any SC value, because of two concurrent readings, that make [SC **none**] as likely as [SC **to**] - with the effect of assigning the category of an adverbial phrase to the infinitival clause. That means that although `want` *in fact realizes* [SC **to**], the system cannot see it. The feature value is invisible, it is *neutralized* by an alternative value, namely **none**. This exemplifies what we call *feature value neutralization* (or shorter for convenience: *feature neutralization*), to be defined as follows:

Def. 34 *Let* $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be a consistent language model in which* α *is underspecified for a given lexeme* $\mathcal{L}$*. Further, there is a related optimal model* $\mathbf{M}_{opt}$*. Then feature value* $\alpha^{\mathcal{L}}_{\mathbf{v}}$ *is said to be neutralized in an utterance u if and only if* $\mathcal{L}$ *is realized in u under* $\mathbf{M}_{opt}$ *and both* $(\Lambda \cup \{\alpha^{\mathcal{L}}_{\mathbf{v}}\})$ *and* $(\Lambda \cup \{-\alpha^{\mathcal{L}}_{\mathbf{v}}\})$ *are consistent with u.*

2.5.2 Systematic and Total Neutralization

Feature neutralization hampers lexical acquisition and in extreme cases it can fully prevent a feature value from being acquired. This happens if it is neutralized in *every* utterance where it is realized, an effect that we call *total neutralization*:

Def. 35 *A feature value* $\alpha^{\mathcal{L}}_{\mathbf{v}}$ *of a lexeme* $\mathcal{L}$ *is said to be totally neutralized if and only if it is neutralized in every grammatical utterance in which it is realized.*

This effect certainly depends on the grammar employed. For instance, a grammar can be such that [SC **to**] is totally neutralized because its realization (like example (9a)) can not be distinguished from a corresponding adverbial phrase (example (9c)). If the grammar, however, properly models wh-movement such that wh-extraction out of infinitival complements is allowed whereas wh-extraction out of wh-islands (like adverbial phrases) is blocked, then utterances can be constructed that do not neutralize the feature value:

(13)
a. What did he want to have?
b. He married to have a big house.
c. *What did he marry to have?

Suppose that the system knows the possible VAL values of `have` as part of its a-priori knowledge (as it is the case in our implementation). Based on

the transitive nature of `have`, example (13a) is a reference for ⟨`want`, [SC **to**]⟩, because first, the interrogative pronoun must be the object of `have` - hence can not be the object of `want` - and second, wh-extraction out of the adverbial phrase is not possible (cf. (13b) and its ungrammatical counterpart (13c)). But even if the grammar is that sophisticated, the system might have to wait long until it comes across examples such as (9a), missing out on an overwhelming majority of learning opportunities that are systematically neutralized by virtue of the grammar's structure. As a softened version of total neutralization we call this effect, namely extremely low learning efficiency because of systematic ambiguities: *systematic neutralization*.

2.5.3 Types of Lexical Neutralization

Feature neutralization arises from one of the key characteristics of language: the conflation of linguistic distinctions during generation of surface structures. Linguistic entities like sentences are complex multidimensional structures that are mapped to a linear sound (or graph) chain during spell-out. Although these chains might show a complexity of themselves, it is nevertheless a fact that information about the underlying structures gets lost during this transition. It is the challenge for the hearer/reader to recover it from the received material and this is the very challenge with which we confront the learning system. If a feature value cannot be recovered (although 'it is there'), then neutralization takes place. Feature neutralization affects all linguistic dimensions:

Neutralization of Lexical Semantic and Pragmatic Features

A lexeme $\mathcal{L}$ can have more then one value for the lexical feature that encodes meaning (like semantic class membership). For instance, the lexeme `come` is assigned to 23 synsets in WordNet. Most of the time, the intended use of a lexeme can be recovered by human addressees - from linguistic context and/or from world knowledge. To enable machines to share this competence may be seen as one of the holy grails of computer linguistics, but our implementation lacks this facility completely. A system that is able to recover lexical semantic (and pragmatic) features must include a rich and complex semantic/pragmatic component and it is far from clear how much and what kind of a-priori knowledge such a system must have at its disposal - especially when it has to overcome the bootstrapping phase.

Neutralization of Lexical Syntactic Features

Lexical syntactic features like part-of-speech category (CAT), subcategorization or inherent number (NUMBER) are also candidates for feature neutralization, as exemplified in this paragraph. CAT feature values, to begin with, can be neutralized for syntactic and morphological reasons and they are particularly susceptible to neutralization in languages with impoverished morphology like English or Chinese.
To give an example, the CAT feature of the artificial lexeme `gnarf` in (14) is neutralized. The lexeme might realize [CAT **verb**] (with 'to' as an infinitive marker) or [CAT **noun**] in a determinerless PP (like 'to office').

(14) He came to gnarf.

The magnitude of category crossing homographs in English is relatively high due to the heavy use of *conversion by zero-derivation*[33]. In order to get an idea of how many of them are stored in WordNet (version 1), we first categorized the words on the basis of the feature values they admit (n: noun, v: verb, a: adjective, adv: adverb). Examples for these categories and the number of words belonging to the respective category are given in table 2.1 and some of the glosses and related utterances for the corresponding synsets are listed in table 2.2[34].

Table 2.1: Categories of Homographs in WordNet

Example	n	v	a	adv	Count
act	x	x			3297
cold	x		x		2121
just			x	x	369
go	x	x	x		281
fun	x			x	213
dead	x		x	x	170
correct		x	x		141
home	x	x	x	x	88
air	x	x		x	86
even		x	x	x	17
utter		x		x	12
					$\sum$ 6426

[33]It should also be mentioned that conversion does not only involve a shift from one open-class category to another open-class category. Moreover - and this challenges the idea of incorporating closed-class lexemes into the initial lexicon - it can also lead to a shift ('upgrade') from a closed class category to an open class category (Brinton and Traugott 2005: 38).

[34]It must be noted that many adverbial and adjectival usages stem from complex nouns like 'air war' in which 'air' is analyzed as an adverb in WordNet.

Table 2.2: PoS-crossing Homographs: Examples from WordNet V.1

Word	PoS	Example gloss	Example utterance
home	(n)	'where you live'	'home is where the heart is'
	(v)	'return home accurately from a long distance, as of some birds'	'homing pigeons'
	(a)	'relating to or being where one lives or where one's roots are'	'my home town'
	(adv)	'on or to the point aimed at'	'the arrow struck home'
go	(n)	'a time for working (after which you will be relieved by someone else)'	'it's my go'
	(v)	'have a particular form'	'as the saying goes...'
	(a)	'functioning correctly and ready for action'	'all systems are go'
air	(n)	'nowhere to be found in a giant void'	'it vanished into thin air'
	(v)	'make public'	'She aired her opinions on welfare'
	(adv)	'relating to or characteristic of or occurring in the air '	'all systems are go'
act	(n)	'the performance of some composite cognitive activity'	'the act of remembering'
	(v)	'perform an action'	'think before you act'
dead	(n)	'people who are no longer living'	'they buried the dead'
	(a)	'out of use or operation because of a fault or breakdown'	'the motor is dead'
	(adv)	'total'	'dead silence'
fun	(n)	'activities that are enjoyable or amusing'	'he is fun to have around'
	(adv)	'providing enjoyment; pleasantly entertaining'	'a fun thing to do'
cold	(n)	'the sensation produced by low temperatures'	'the cold helped clear his head'
	(a)	'no longer new; uninteresting'	'cold (or stale) news'
even	(v)	'become even or more even'	'even out the surface'
	(a)	'symmetrically arranged'	'even (or regular) features'
	(adv)	'to the full extent'	'loyal even unto death'
correct	(v)	'make right or correct'	'Correct the mistakes'
	(a)	'socially right or correct'	'correct behavior'
utter	(v)	'articulate; either verbally or with a cry, shout, or noise'	'He uttered a curse'
	(adv)	'total'	'utter seriousness'
just	(a)	'used especially of what is legally or ethically right or proper or fitting'	'a just and lasting peace'
	(adv)	'(intensifier) absolutely'	'I just can't take it anymore'

Another important case in point is the neutralization of features that encode subcategorization. (15) lists some examples, the verbs of interest being underlined:

(15)
a. It rains.
b. It followed that S won.
c. She bought a book for him.
d. She bought it for him.
e. Mrs. Lyons flushed with anger again.
f. She stopped smoking.
g. I said this laughing.
h. It will come round.
i. Fix it this way.
j. The day you marry.
k. We know that God is everywhere;
l. He told the audience that he was leaving.
m. I advised Mary to go.
n. John promised Mary to resign.
o. He helped to save the child.

In (15a) and (15b) the expletive 'it' is not bound to any semantic role, rather in a) it is the subject of an expletive verb and in b) the placeholder for an extraposed *that*-clause - facts that could be reflected by boolean features EXPLETIVE and EXTRAPOSED, respectively. Both features are subject to total neutralization because the system (at least in our implementation) can never infer from those utterances whether 'it' is an anaphora or an expletive[35].
The example (15c) is an instance of the classic PP-attachment problem: The PP can attach to the verb or alternatively to the nominal phrase headed by `book`. Because PPs usually do not attach to pronouns, no PP-attachment problem should be encountered in (15d). It is still unclear (for the learning system), however, whether the PP belongs to the argument structure of `buy` or whether it is an adjunct. This is another instance of total neutralization: there is no clue for the system to decide about the argumenthood of a PP that attaches to a verb. Apart from this problem, example (15e) demonstrates how the property of being intransitive can be neutralized: here the verb `flush` can either be the matrix verb (intransitive reading) or the head of a small relative clause with 'Mrs. Lyons' as (phonologically empty) direct

[35]Note that these non-anaphoric pronouns are not restricted to the subject position. In all of the following examples, the word 'it' does not have any anaphoric reference:

a. I take *it* then *that you are leaving.*
(cleft sentence, cf. Quirk et al. 1994: 349)

b. They really lived it up in Boston last night (Fraser 1974: 34)

c. I had a good time of it. (Fraser 1974: 34)

object, resulting in a transitive reading.
The ambiguities in the examples (15f) and (15g) do not arise from attachment issues. Rather, it is the status of the *-ing*-phrases (both attach to the verb) that is the source of the feature neutralization. In (15f) the verb realizes its disposition to subcategorize for an *-ing*-complement, but the complement cannot be recognized as such because it is superficially isomorph with an adverbial phrase like in (15g).
(15h) is an instance of systematic neutralization that will be treated in more detail in subsection 2.5.6: the status of 'round' is ambiguous: it could be the particle of a phrasal verb or an adjectival complement. Thus the system cannot decide which of the two different lexemes `come` and `come round` are actually realized. We propose to account for VPC-verbs by the employment of a verbal feature PARTICLE whose value is **none** for non-VPC verbs or a type representing the particle ({**round**, **up**, ...}) in case the lexeme is a VPC verb. Note that while the GRAPH feature is identical in both readings in the example (each time **come**), the sense feature and the feature PARTICLE will differ[36]. The neutralization of the sense feature notwithstanding, it is the two values of the PARTICLE feature (**none** and **round**) that neutralize each other. Although this is a neutralization of lexeme determination in the first place, it is mentioned here because this phenomenon has a considerable interplay with verbal subcategorization.
Example (15i) and (15j) show the property of certain nouns to be convertible to adverbs: 'this way' is surely no direct object of `fix`, 'the day' no direct object of `marry`.
(15k) and (15l) show neutralizations of the feature value [SC **that**]. In the first case this is due to the ambiguous status of `that`, which can be used as a complementizer or as a demonstrative pronoun (among others) - so that (15k) would become an instance of a complement with covert complementizer if `that` is analyzed as a demonstrative. In the second case, again an attachment problem is the originator of the neutralization: the complement headed by `leave` either attaches to the matrix verb or it attaches to the nominal phrase headed by `audience` provided that the SCF of `tell` is unknown. This example even shows multiple neutralizations: Because `that` can additionally be used as a relative pronoun, the phrase headed by `leave` could also be a relative clause with `that` functioning as the direct object of `leave`. It follows that not only the subcategorization feature of `tell`, but also the one of `audience` is neutralized.

[36] A similar approach is followed in ERG in the sense that both `come` and `come round` are verbs with the same STEM feature value (translates to our GRAPH), cf. Villavicencio and Copestake (2002).

(15m), (15n) and (15o) are examples of neutralization of control. Suppose the feature CONTROL would be used to encode control properties of verbs taking the values of $\{\mathbf{arb}, \mathbf{seq}, \mathbf{oeq}\}$ (for arbitrary control, subject-equi and object-equi, respectively). First of all, for reasons explained with example (9) it is not proven by the example that the *to*-clause is a complement of the verb and hence it is not guaranteed whether the feature CONTROL is applicable at all. Even if the status of the *to*-clause would be clear to the system the feature values of CONTROL all neutralize each other. They can be **arb** as in fact realized in (15o), **oeq** as in (15m) or **seq** as in (15n).
This paragraph concludes with examples that demonstrate the neutralization of countability and inherent number. The latter is the feature that determines with which number the lexeme is projected into morpho-syntax. The former encodes the distinction between count and mass nouns. Note that this feature is seen here from a mere structural point of view[37].

Technically, [SG −] may denote plural nouns like `trousers` and [SG +] may stand for singular nouns accordingly. The SG is appropriate for count nouns only, so the feature value, say, [COUNT +] is the realization context for it. Because plural nouns as well as mass nouns can directly be projected to a DP without the need of an (overt) determiner, as exemplified in example (16), the values of COUNT can neutralize each other - with the effect that the realization context for SG is not unambiguously realized:

(16)
a. I prefer water. ⟨[COUNT −]⟩
b. I like people. ⟨[COUNT +], [SG −]⟩

The example (17) shows, however, that these feature values are not totally neutralized as there are unambiguous references for both COUNT and SG:

(17)
a. A dog barks. ⟨[COUNT +], [SG +]⟩
b. Cattle are grazing. ⟨[COUNT +], [SG −]⟩
c. Dignity is desirable. ⟨[COUNT −]⟩

Neutralization of Lexical Morphological Features

This subsection deals with lexical features that control the applicability of derivational and inflectional morphological rules. These features come along with two different types of neutralization. First of all, it is not always obvious

[37] We acknowledge the facts that a) the feature is to be treated differently in different languages (in Chinese, for example, all nouns are mass nouns and must be rendered countable by means of numeral classifiers, see Chierchia 1998) and b) the setting of this feature is very much related to world knowledge. In this thesis, however, we focus on English and the mere distributional properties of the count/mass dichotomy.

whether an inflected or derived form belongs to a certain underlying lexeme or not[38]. This is an instance of 'neutralization of lexeme determination', discussed further below. Secondly, because affixes can serve multiple functions, ambiguities with regard to these functions may arise.
To begin with, such overlaps in function can lead to neutralization within paradigms of *inflection*. As an example, consider the paradigm that is shown in table 2.3. Here, the morphological forms of some German nouns[39] (and the correlated definite articles) vary with gender, number and case. We have chosen nouns from classes that take the plural suffix *-en*. This suffix can be found in paradigms of all *genera* (masculinum, femininum, neutrum). The table shows how the gender feature gets neutralized when the nouns occur in plural - across all cases. This is because the plural suffix *-en* is neutral with regard to gender (and case as well[40]). Note that the definite article is also neutral with regard to gender if the noun is plural.

Table 2.3: Neutralization of Lexical Morphological Features

	masc.		fem.		neutr.	
	sg.	pl.	sg.	pl.	sg.	pl.
Nom.	der Schmerz	die Schmerzen	die Frau	die Frauen	das Herz	die Herzen
Gen.	des Schmerzes	der Schmerzen	der Frau	der Frauen	des Herzens	der Herzen
Dat.	dem Schmerz	den Schmerzen	der Frau	den Frauen	dem Herzen	den Herzen
Acc.	den Schmerz	die Schmerzen	die Frau	die Frauen	das Herz	die Herzen

We want to emphasize that neutralization in *one* particular utterance should not be confused with total neutralization which make *all* utterances hide the underlying lexical feature in question. Here, only nouns of a certain paradigm are affected if they occur in plural, which is certainly no instance of total neutralization, at least unless a given noun is no plural-noun. There will be enough utterances realizing instances of the nouns in singular. Note that underspecification of morphological features used in inflectional paradigms concerns closed-class material like inflectional affixes. Here it becomes obvious how underspecification of closed class items is related to underspecification in the sense of unknownness: underspecification of the affix (in the example: the underspecification of gender) renders the lexical feature GENDER of the open class noun stem invisible and prevents acquisition if the

[38]To give an instance of oddity: the word 'mod' is stored as an adjective in WordNet (and as a noun, too). The learning system might - when it encounters the word 'modest' - be tempted to analyze this word as a comparative of 'mod'.

[39]'Schmerz' = 'pain', 'Frau' = 'woman', 'Herz' = 'heart'

[40]See Evert (2004) for quantitative and statistical aspects of this feature.

noun has the affix attached.

For the neutralization of features of derivational morphology we want to instance the English *-ly* suffix. More precisely, there are two suffixes, namely $-ly_v$ and $-ly_a$. Both suffixes turn an adjective into an adverb. But whereas $-ly_v$-adverbs are verbal adjuncts (like 'quickly'), $-ly_a$-adverbs adjoin to adjectives (like 'extremely'). If the context does not tell whether the adverb is attached to a verb or an adjective (like in (18)) then neutralization takes place[41]:

(18) It was greatly shaken.

Neutralization of Lexical Phonological/Graphical Features

Graphical and phonological features can be neutralized in contexts where the graphical/phonological rules render the underlying forms invisible. The German Auslautverhärtung, for instance, leads to this kind of ambiguity: because German obstruents in the coda of the syllable have to be voiceless - disregarding voice of the underlying phonemes - the underlying phonological form of the lexeme is neutralized.
A similar effect can be observed in the area of orthography: in English (and many other languages), sentence-initial words begin with capital letter - disregarding the underlying graphical form.

Neutralization of Lexeme Determination

The most extreme form of lexical neutralization is the (potential) uncertainty about which lexeme is used at all in a given utterance. This effect can be observed in systems like ours that use the graphical form as an anchor to a lexeme (in other words, the GRAPH-feature is the *ultimate realization context* of each lexeme): in (19), a possible title of an article (hence the capital letters), no unambiguous access to the (artificial) lexeme `gnarf` is given. The lexeme might be stored under `gnarf` or under `Gnarf`.

(19) Forced to Gnarf.

Lexeme determination is also neutralized when lexemes show *irregular forms*, because they can not be traced back to the graph feature of any lexeme (at least not on the basis of 'normal' morphological rules). As we do not have a general solution for this problem, irregular forms are part of the

[41] 'shaken' is a participle of `shake` and it is additionally stored as an adjective in WordNet.

a-priori knowledge in our implementation.

Neutralization of lexeme determination can also arise from the difficulty to detect the correct word boundaries of a lexeme. Languages like Chinese that do not have a graphical correlative to the space character used in standard European languages are especially susceptible to this problem (the so-called segmentation problem).
But even in English, word boundaries of multiple word items might be invisible, as the examples in (20) show:

(20) a. She also sent the railway company letters about her dissatisfaction.
b. Can you provide information about the flat line reinsurance?
c. New York is fun.

In (20a), it is the (possibly) unknown valency of `send` that is responsible for the alternative readings [$_{N^0}$ railway company] versus [$_{N^0}$ railway company letters]. Even in the ditransitive analysis there is an ambiguity between [$_{N^0}$ railway company] and [$_{N^0}$ company letters]. (20b) might lead the system to come up with the two readings [$_{N^0}$ flat line reinsurance] and [$_{NP}$ [$_{A^0}$ flat] [$_{N^0}$ line reinsurance]]. Utterance (20c) is another example of graphical neutralization mentioned in the last paragraph. Here, the alternation of 'new' and 'New' leads to the two readings [$_{NP}$ [$_{A^0}$ new] [$_{N^0}$ York]] and [$_{N^0}$ New York].

2.5.4 Consequences for Learn$-\alpha$

The last subsection has demonstrated that lexical acquisition under the restricted version of the Learn$-\alpha$ implementation would hardly be feasible or at least it would be quite patchy due to total neutralization. This enforced the relaxation of the restriction that an utterance prove a feature value unambiguously:

Theorem 3 *There are lexical features that are invisible when the system is restricted to* $RDoA_{max}$*=1.*

This conclusion may not come as a surprise. However, we consider it worthwhile to make it as explicit as that. Note that first, total neutralization depends on the quality of the grammar or more generally on the system which is able to reject impossible (or highly unlikely) u-models. Second, not all features are affected by total neutralization (again depending on the grammar). These *can* be learned with highest precision. It is just a matter of recognizing this kind of meta-property so that it is guaranteed that the system learns with the highest precision possible. Third, one strength of

Learn$-\alpha$ is its capability to increase the quality of lexical acquisition when the quality of the grammar gets enhanced. Especially when the system becomes sensitive to semantic and pragmatic distinctions more disambiguation would be possible and the degree of neutralization would decrease automatically. Again, the lexicon is completely relative to the grammar.

Assumption of Statistical Significance

While the conclusion says that our system cannot (constantly) work under $\text{RDoA}_{max}=1$, it does not predict the consequences of working with $\text{RDoA}_{max}>1$. The statement enforces the introduction of uncertainty (exactly the one that we wanted to avoid at the outset) - uncertainty about the adequacy of acquired feature constraints and, as a result, uncertainty about the overall quality of the acquisition method. If the system cannot trust a reference any more, the question occurs of *how many* references are needed to consider a given $\alpha_{\mathbf{v}}^{\mathcal{L}}$ proven.
This problem is similar to the issue of noise which furnishes (alleged) references with the uncertainty whether they are intended or not. There is a crucial difference, nevertheless, between noise and neutralization: noise is because of error (and might be reduced by various methods), lexical neutralization lies in the nature of language. Type-I-observation errors (cf. section 2.4.1) due to overgeneration can be caused by both noise and neutralization, but in the latter case it is not because of an inadequate language model but because of missing lexical knowledge - exactly the one that is the target of induction, which highlights another vicious circle.
It was argued that because of the nature of induction a statistical component must lie at the heart of a theory of lexical acquisition. And this component has to be adapted to the factor noise as well. But giving up $\text{RDoA}_{max}=1$ introduces even more uncertainty and the concern arises whether lexical features are always realized with statistical significance that enables the system to single them out. In other words, does, for instance, a prepositional verb that subcategorizes for a preposition P occur with P more often then chance? We cannot give a general answer, we cannot but *assume* it - adopting the tenet that underlies much of the research within statistical NLP[42]:

Assumption 9 *Lexical feature values are realized with statistical significance.*

[42]See for example Church and Hanks (1990) about mutual information, a measure for word association: 'If there is a genuine association between x and y [words in this case], then the joint probability P(x,y) will be much larger than chance P(x)P(y) [...]'. But see also Church et al. (1991) for a warning of blindly applying statistics to language.

At the end it will always be a matter of empirical investigation.

2.5.5 Generalized Observation Schema and RDoA$_{max}$

As a prerequisite for the final version of Learn$-\alpha$, the formalization of the notion 'Relative Degree of Ambiguity' (RDoA) coined in subsection 1.4.4 is in order. This notion covers the possibility that an utterance 'proves' a feature value with some ambiguity related to it. In a certain sense, RDoA 'counts' the number of disjunctions in the premise

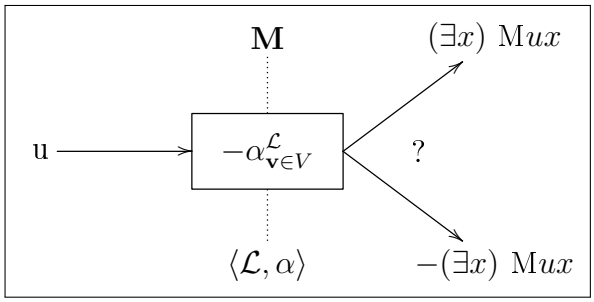
$$-(\alpha^{\mathcal{L}}_{\mathbf{v}_1} \vee \ldots \vee \alpha^{\mathcal{L}}_{\mathbf{v}_n}) \supset -(\exists x)\ \mathrm{M}ux$$

of the implication of the Learn$-\alpha$ observation schema (subsection 2.3.1). Let us abbreviate the conjunction by the term $\alpha^{\mathcal{L}}_{\mathbf{v}\in V}$, then we can state that $RDoA = |V|$.
Again simplifying the schema so that p stands for 'u is considered ungrammatical' leads to the following **generalized observation schema**:

$$(\Box p\ \&\ (-\alpha^{\mathcal{L}}_{\mathbf{v}\in V} \supset -p)) \supset \Box\, \alpha^{\mathcal{L}}_{\mathbf{v}\in V}) \tag{2.47}$$

We modify the graphical representation of this schema accordingly:

Figure 2.9: Generalized Observation Schema



The transition from an utterance to a feature value that is considered proven entails that RDoA=1. But as shown, this restriction might render feature values indeterminable (cf. section 2.5). This is why we now allow for the generalized transition from one single utterance to a proven (or referenced) feature value under a specified RDoA$_{max}$=n. This means that u is a reference although there are up to $n-1$ alternative feature values in the premise of the observation step. Whether the induction step will consider this reference or not, is a separate question of statistical significance. Here is the definition.

Def. 36 (reference under RDoA$_{max}$) *Let* $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be a consistent symbolic language model. An utterance u is said to be a reference for a feature value* $\alpha^{\mathcal{L}}_{\mathbf{v}}$ *under a maximum RDoA of n, if and only if it is consistent with* Λ *and V is the set of feature values such that* $\Lambda \cup \{-\alpha^{\mathcal{L}}_{\mathbf{v}\in V}\}$ *is inconsistent with u and* $|V| \leq n$*. We will use the following notation:*

$$u \mapsto^{\rho=n} \Box\alpha^{\mathcal{L}}_{\mathbf{v}} \tag{2.48}$$

2.5.6 Bifurcations in Lexical Acquisition Space

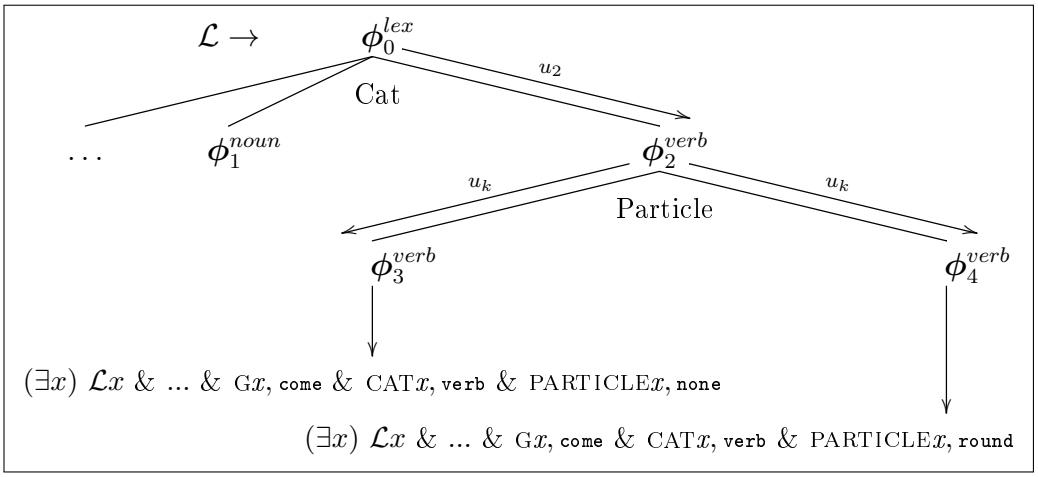
Constantly parameterizing the learning system to RDoA$_{max}>1$ would unnecessarily decrease the system's precision and thus contradict the Learn$-\alpha$ design rule, because those lexical features that are not affected by total neutralization *can* be acquired with a maximum of precision. Further, because a maximum of precision is demanded by Learn$-\alpha$, we are forced to find out whether the indeterminable features require a RDoA$_{max}=\infty$ or whether the degree of allowed ambiguity can be restricted.
To answer this question, we have to reconsider why total neutralization prevents the transition from one node in lexical acquisition space to the next node. This is because RDoA$_{max}=1$ requires that there must be only one feature value (of a given lexeme and feature) which is consistent with the utterance. A second, alternative value would block the transition. On the other hand, if there are *only* two alternatives, then the utterance proves after all that either the one or the other value is the correct one. There is no room for a third one, etc. This is what we want to call a *bifurcation in lexical acquisition space*. Figure 2.10 shows an example for the acquisition of PARTICLE in example (15h) ('It will come round'). Here the bifurcation happens for every utterance u_k that realizes the feature value [PARTICLE **round**].

Example (15h) suggests that all particles are subject to total neutralization. But at least they can all be tried to be acquired under RDoA$_{max}=2$. This is not necessarily the case for all features, for example CONTROL (cf. examples (15m-o)) will be indeterminable under RDoA$_{max}=2$. But a closer look at lexical features reveals that many (if not most) of them are subject to bifurcations only. Example (9c), repeated here as (21) is another instance:

(21) He married to get rich.

Assuming that the SC feature of `marry` is underspecified because of unknownness, it is neutralized in this utterance because the grammatical relation of the matrix verb to the infinitival VP is ambiguous. The VP can be

Figure 2.10: Bifurcation Example



(and in fact is) an adjunct, leading to the instantiation of [SC **none**] or the VP can be the verbal complement resulting in [SC **to**]. This example shows that there are only two competing values. But only if we find an instance of total neutralization by more than one competing value, the system would be forced to increase RDoA_{max} beyond 2. This leads to the claim:

Theorem 4 *Many feature values can maximally be totally neutralized by* **one** *alternative value. So* $RDoA_{max}$ *=2 is a prominent point in the interval* $[1, \infty]$.

This theorem has important implications for the design of our implementation because it structures the levels of precision: $1 > 2 > \infty$ (see the concept of *cascading focus* in section 2.8.8). It may be a consequence of the structure provided by the lexical type hierarchy. The flatter the hierarchy the less applicable is the theorem.

2.6 Lexical Associations and Compositionality

All examples discussed so far dealt with morpho-syntactic constraints. Surely, there *is* an interaction with semantics (argument structure, count/mass dichotomy), but the point is that a) the morpho-syntactic behavior of the lexeme is not fully determined by its semantics and b) there is a strong interplay with the grammar. For instance, the feature value [COUNT −] of the lexeme `flash` cannot be fully determined by its sense provided the observation that the German counterpart `Blitz` is definitely [COUNT +] (Baldwin

and Bond 2003a) and the grammatical consequence of the feature value - both in English and German - is strong: nouns with [COUNT −] can directly be projected into a DP, otherwise they can't.
The examples in (22) given below are completely different and this section addresses the question *if and how* the phenomena underpinning them are dealt with in Learn−α.

(22)
a. Can you do / #make me a favor?
b. Can you take / make the decision?
c. I like strong / #powerful tea.
d. I like dry wine / dry biscuits.
e. I performed / #executed the analysis.
f. I'll eat my cake.
g. I'll eat my hat.
h. I'll eat my gnarf.
i. #I'll eat my grasshopper / my sofa /
some electrons / an orthogonal triangle / the sixties.
j. #Colorless green ideas sleep furiously.
k. #My buxom neighbor is the father of two.
l. the bus / car / ship / space shuttle / door / drinking accident
m. my best man

None of the example utterances that are marked with '#' violate any morpho-syntactic constraint. Consider the perspective of the LE again. Why should the examples matter? Because lexical knowledge helps a) filter out undesired u-models generated from utterances and b) generate naturally sounding utterances from a give semantic structure. Example (22a) and (22b) show realizations of the English support verb construction (SVC). We adopt the definition of Erbach and Krenn (1993: 3), in which SVCs 'typically consist of a relational noun, i.e. a noun with an argument structure (predicative noun), which contributes the semantic relation of the expression, and a semantically deprived support verb, which contributes, at least, tense and causativity'. The meaning of the combination ⟨`do`,`favor`⟩ and ⟨`take/make`,`decision`⟩ is a function of the meaning of `favor` and `decision`, respectively, and the added meaning of the construction - provided the grammar Γ contains corresponding rules[43]. The difference between (22a) and (22b) is that in (22a) ⟨`make`,`favor`⟩ cannot be interpreted as a combination entering the construction. Hence the only possible interpretation is a fully compositional one. The interpretation of (22b) (or the preferences towards it), however, is a matter of dialectal variation. According to McKeown and Radev (2000)

[43]For sake of simplicity the additional benefactive argument position is ignored here.

⟨make,decision⟩ is used in American English, ⟨take,decision⟩ in British English.
Example (22c) is an often cited example of a *collocation*. It shows that the noun tea imposes constraints on the selection of its intensifying modifier - under the assumption that strong and powerful are synonyms. Similar to (22a), a violation of this constraint leads to a compositional non-intensifying interpretation (whatever a powerful tea might denote). In (22c) the combination ⟨dry,wine⟩ selects a specific meaning of the adjective: non-sweet, which is restricted to all kinds of wines (and not to the very lexeme wine), but not to other beverages or food. So the combination does permit the canonical interpretation of 'dry' only - although the interpretation of a sweet vs. non-sweet biscuit was readily available.
Example (22e) shows a verb-object collocation: ⟨perform,analysis⟩, which seems to be preferred to the alternative ⟨execute,analysis⟩, which is synonymous to the former[44].
Examples (22f) - (22k) point up a phenomenon of a very different kind. They concern *selectional restrictions* that are treated under the headwords *category mistake*, *selectional violation*, *type crossing* or *semantic anomaly* (Resnik 1993: 34). Example (22f)'s meaning is a fully compositional function of the usual meanings of eat[45] and cake. The isomorphic example (22g) conveys an additional meaning, according to OALD 'used to say that you think sth [something] is very unlikely to happen' (Hornby 2005: 484). This meaning is neither a function of the previously mentioned meaning of eat nor of the meaning of hat. It is fully idiomatic and the combination ⟨eat, hat⟩ has to be represented by a different and independent lexeme. As this phenomenon is to be treated along with *lexeme detection* in Learn$-\alpha$, it does not belong to 'lexical association'. It at most touches selectional restrictions in the effect that its meaning usually blocks the compositional reading because of the violation of a selectional restriction connected to eat[46].
In this context, example (22h) deserves special interest. Assuming that gnarf is an unknown lexeme, a reading is preferred in which this lexeme denotes something edible, hence some sort of food (Manning and Schütze 1999: 288)

[44] In WordNet Version 3 both verbs are part of the synset 201712704 = {perform, execute, do} which is glossed as 'carry out or perform an action'. A quick check of the number of retrieved pages from Google for the phrases 'execute an analysis', 'perform an analysis' and 'do an analysis' (15.06.2009) show a drastic quantitative difference: there were 7850 retrieved pages for 'execute an analysis', 376.000 pages for 'perform an analysis' and 308.000 pages for 'do an analysis'.

[45] Represented by the WordNet version 3 synset 201166351 with the gloss 'eat a meal; take a meal;'

[46] That somebody can indeed *eat* his hat is impressively demonstrated by Stan Laurel in the unforgettable movie 'Way Out West'.

- *by virtue of* `eat`'s selectional restrictions. The issue of how Learn$-\alpha$ treats this phenomenon crucially depends on the question whether selectional restrictions of this kind are lexical properties at all. We will argue for the claim that they are not. Examples (22i) to (22k) show why. The compositional interpretation of (i) is awkward due to different reasons. In the case that the personal pronoun refers to a speaker who lives in a culture in which grasshoppers definitely do not belong to the usual menu, the utterance comes as a surprise, but the compositional reading is absolutely fine. 'Eating a sofa' can only be interpreted metaphorically because of physical restrictions concerning nutrition. The awkwardness increases dramatically from left to right in (i) especially if the speaker is assumed to be part of our real world that obeys to the laws of a certain ontology which simply makes the nutritional consumption of physical particles, mathematical objects or decades meaningless. The compositional interpretation is rehabilitated in the context of fairy tales, for instance: 'Look! There is the monster who has eaten all my triangles, said the poor mathematician'. In these contexts the real-world ontological constraints are violated, not any lexical restrictions. Chomsky's famous example (22j) (Chomsky 1957) includes a series of such violations[47] plus a contradiction that spans the predicate-argument relationship of three words: the *logical* restriction that a thing cannot be green (have color) and colorless (not have color) at the same time. Example (22k), taken from Resnik (1993), shows a selectional restriction that is imposed by a (compositional) *phrase* 'buxom neighbor' and *not* by a single lexeme. The point is that both the logical contradiction in (22j) and the semantic anomaly in (22k) cannot be properly handled by the lexicon. All these anomalies are at most indirectly connected to lexemes by virtue of their sense predication $\mathrm{S}x\mathbf{s}$ that might be *linked* to an ontological component[48]. In a system that is able to detect such violations there must be a component that has a representation of at least logical necessities, ontological constraints, cultural and factual knowledge - a component that is outside of the symbolic language model in Learn$-\alpha$. Coming back to the perspective of the LE, his intention function $I(u)$ contains models of utterances that are passed to this component and evaluated further (filtered,

[47] Again there are contexts in which green ideas, for example, can be meaningful. In a brainstorming session, for instance, people might be asked to write down their ideas on red and green cardboards according to some distinguishing criterion. In this situation it sounds perfect to say something like: 'could you please fix the green ideas onto the right board?'

[48] According to Soehn (2005), who advocates the integration of selectional preferences into HPSG, these preferences help an NLP system with 'parsing, word-sense disambiguation and the resolving of anaphora'. Our point however is that disambiguation is not in the exclusive right of grammar.

ranked etc.). $I(u)$ only filters u-models by virtue of $\mathbf{M}$, nothing more. The question of how the additional component is functioning, is completely out of his scope. At least we want to express strong doubts that it uses similar devices as the ones constituting $\mathbf{M}$. Thus, the theory of Learn$-\alpha$ is in sharp contrast with the work of Barg et al. (cf. section 1.3.6) in which the concept of unknownness comprises morpho-syntactic *and* ontological dimensions. In our proposal sense candidates for `gnarf` in (22h) are to be inferred in the outside component, not by virtue of Learn$-\alpha$.
Finally, the examples (22l) and (22m) display another case of the compositional / non-compositional dichotomy. While (22l) is fully compositional within the boundaries of what constitutes our factual world, there is little compositionality in (22m) whose meaning is defined in WordNet as 'the principal groomsman at a wedding'. This is again an instance of *lexeme detection* to be deferred until section 2.7.

The discussion was intended to demarcate the boundaries of Learn$-\alpha$ with regards to lexical associations. The fully compositional expressions (or more precisely the compositional interpretations thereof) are not lexical by definition and the fully idiomatic ones are reserved for the topic of lexeme detection. The remaining - the *semi-productive* - ones are divided into the class of expressions that have to be treated outside of $\mathbf{M}$ and the class of expressions that are inside $\mathbf{M}$: the collocations. Although it is questionable whether the demarcation line can always be clearly and consistently drawn, we think that the reason why speakers do not use 'powerful tea' (in the sense of intensity) cannot be traced back to ontological or cultural or factual peculiarities. Otherwise there have to be stipulated very subtle and difficult-to-maintain semantic differences between `strong` and `powerful`. It is also clear that because of different degrees of grammaticalization different languages will show diverging feature systems due to the fact that semantic distinctions that are outside of $\mathbf{M}$ in one language might be reflected by lexical features in another one[49]. But even within one and the same language it will be difficult if not impossible for the system always to distinguish a statistical cooccurrence effect that is due to a collocation from one that is due to selectional preference.

[49]In Thai, for example, there is a degree of lexical fixedness how nouns select their numeral classifiers (Haas 1942), which justifies a nominal lexical feature to account for the phenomenon.

2.6.1 Collocations

The notion *collocation* is not homogeneously defined in the linguistic literature[50]. There are at least four contemporary types of definitions:
(i) collocation as a set of lexical items that occur together more often than chance (a probabilistic definition). This definition is used, for instance, in McKeown and Radev (2000)[51] or Copestake (2001).
(ii) collocation as a word combination that deserves its own lexical entry, regardless of its probabilistic properties, e.g. in Evert (2005: 17): 'A collocation is a word combination whose semantic and/or syntactic properties cannot be fully predicted from those of its components, and which therefore has to be listed in a lexicon' (collocations as idiosyncratic entities).
(iii) collocation as a 'word combination where one element cannot be easily substituted' (Zinsmeister and Heid 2004). This distributional definition covers support verb constructions, habitual combinations as well as idioms.
(iv) collocation as a *phraseme* which is not a lexeme in itself and in which the selection of one item A is a function of another item B but not vice versa (a simplified version of collocation in Meaning-Text-Theory (MTT), hence a phraseological definition, see Burger et al. 2007: 120).
Learn$-\alpha$ adopts the last definition because frequency (i) is not a building block of the symbolic language model, instead more a kind of metadata given a-posteriori[52]. Because definitions (ii) and (iii) comprise idioms they have a too strong overlap with lexeme detection that is treated separately.
So we use collocations as combinations of two lexemes in which the selection of one lexeme $\mathcal{L}_1$ is a function of the other $\mathcal{L}_2$. Following the terminology of Phraseology, we call $\mathcal{L}_1$ the *collocate* and $\mathcal{L}_2$ the base of the collocation. Collocations pose four major challenges to Learn$-\alpha$. Firstly, how can *selection as a function of a lexeme* be modeled by a lexical feature? Secondly, how can the selection process be modeled in the grammar so that for example 'strong tea' is preferred during generation of an utterance and the expression 'pow-

[50] This often stated claim is supported by the observation that many (if not most) authors who write about collocations start with a definition thereof. The present subsection follows this tradition.

[51] 'Unlike free word combinations, a collocation is a group of words that occur together more often than by chance.'

[52] We acknowledge that retaining frequency information may help the grammatical component ease the burden of a pragmatic component as proposed in Copestake and Lascarides (1997), however frequency is not a lexical feature in the scope of definition 8. So we are in line with Copestake (2001), who states that 'The probabilistic approach to semiproductivity can be seen as a matter of performance rather than competence and has to be formalized separately from the symbolic grammar.' See also Sag et al. (2002: section 3.4) for discussion.

erful tea' is not ruled out during analysis. Thirdly, how does the observation schema apply? And finally, what are the conditions for induction?
In search for an answer to issue one, we propose to store the collocate as the value of a *collocational feature* that is part of the base's lexical definition. As only sorts or types defined in the feature system of a symbolic language model can become the value of a feature, *lexemes by themselves* - usually identified by their *lemmas* - have to become sorts or types[53]. This means that lexemes should be directly represented in the type system of the symbolic language model. This move is not without controversy, however. It has been noted that selection might be more related to semantics than to lexemes (Zinsmeister and Heid 2004, Kuhn 1997: footnote 4). Until this issue is resolved we want to change the conception of $\mathcal{L}$ as an unary sort symbol to a conception in which $\mathcal{L}$ becomes a feature. For example, let **strong** then be the lemma of the lexeme `strong` in a type system, so that the statement 'there is a lexeme `strong`' can be logically expressed as $(\exists x)$ $\mathcal{L}x$**strong** and the statement '`strong` is a collocate of `tea`' as $(\exists x)$ $\mathcal{L}x$**tea** & COLLOCATEx**strong**.
The problem of the last formula, however, is that collocations are not the relation between two lexemes, but a restriction imposed on a certain relation between two lexemes. This relationship is modifier-modified in the 'strong tea' example and verb-undergoer in the 'perform analysis' example.

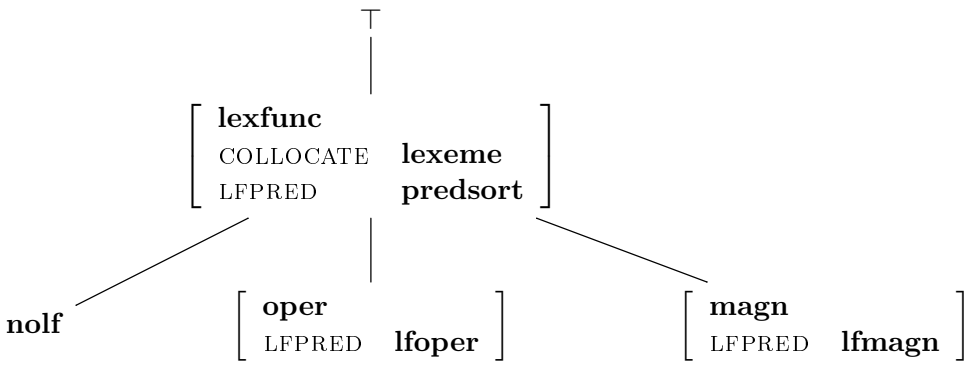
A further prerequisite to the application of the lexical feature account on collocations is the exposition of information about the collocate to the base. From the viewpoint of the base one wants to express that if the base should be modified by an adjective (in the *strong tea* example) and if the adjective belongs to the synonym set S (the one that denotes the intensifying meaning) then the lexeme `strong` is to be preferred. This kind of access to relations is not necessarily available (by the usual grammatical rules that implement subcategorization, for example). It is not unusual, however, in frameworks such as HPSG[54].
We would like to sketch out two possible implementations of collocations in HPSG and review the logic behind it afterwards. The first proposal is based on lexical functions established in MTT[55]. The input of a lexical function is a lexeme and the output, for the sake of simple exposition, is a lexeme, too. The lexical function determines the meaning of the word combination. In MTT there are some 60 lexical functions. The examples employ two of them:

[53]For a similar reason Erbach (1992) introduced the feature LEXEME within the HPSG framework. See Soehn and Sailer (2003) for a modification of this approach.

[54]In Soehn (2006) a mechanism is enabled that exposes the structure of even the whole utterance to the word or idiomatic phrase. However, he remarks that this may be a too powerful apparatus that deserves some restriction (Soehn 2006: 96).

[55]For a different account of lexical functions in HPSG, cf. Heylen et al. (1994).

Figure 2.11: Type Hierarchy for Lexical Functions



OPER (*perform* an analysis) and MAGN (*strong* tea). To be able to include lexical functions in feature descriptions, we propose a type **lexfunc** which carries the collocate (the result of the function) in form of a COLLOCATE feature. Because the semantics requires a PRED feature to be bound to some value encoding the sense of a predication, **lexfunc** shall specify this as well. Because the PRED feature is already defined on another type, we use LFPRED. Subtypes of **lexfunc** can specify the required values. Not all lexemes will have a lexical function. So it should be possible to specify **nolf** (no lexical function) as well. The fragment of such a type hierarchy is shown in figure 2.11.

The AVMs in 2.49 show two lexical entries for the transitive verb `execute`. The left one cannot be a lexeme selected by a lexical function, the right one can be selected by the base lexeme of a collocation. In addition let us assume two entries for `perform` in analogy.

$$\textit{verbtrans}\begin{bmatrix}\text{LEMMA} & \textbf{execute}\\ \text{LEXFUNC} & \textbf{none}\\ \text{PRED} & \textbf{execute'}\end{bmatrix}\quad \textit{verboper}\begin{bmatrix}\text{LEMMA} & \textbf{execute}\\ \text{LEXFUNC} & \textbf{oper}\end{bmatrix} \tag{2.49}$$

Let us also assume that there is a feature LEX in which the lexical features of a lexeme are collected. One of these lexical features be COLL that encodes collocations over lexical functions in the base. The next structure given in 2.50 is then the crucial one. It is thought to be in the specification of all lexeme types that opt for **oper**, hence a specification for **verboper**, too.

$$lexwithlf\begin{bmatrix}\text{LEMMA} & \boxed{1} \\ \text{LEXFUNC} & \boxed{2}\begin{bmatrix}\text{COLLOCATE} & \boxed{1} \\ \text{LFPRED} & \boxed{3}\end{bmatrix} \\ & oper \\ \text{PRED} & \boxed{3} \\ \text{COMPL} & \left\langle \text{INDEX} \mid \text{COLL} \quad \boxed{2} \right\rangle\end{bmatrix} \tag{2.50}$$

What happens if a sign with this specification enters the *head-complementizer* rule is that the LEXFUNC value (e.g. of the verb `perform`) will be unified with the value of the feature COLL of the complement (e.g. `analysis`). Because the collocate feature value unifies with the lemma of the head, this information is in a sense 'imprinted' on the base of the collocation. The structure projected from the base lexeme after parsing will look like this:

$$\begin{bmatrix}\text{LEMMA} & \textbf{analysis} \\ \text{INDEX} \mid \text{COLL} & \begin{bmatrix}\text{COLLOCATE} & \textbf{perform} \\ \text{LFPRED} & \textbf{lfoper}\end{bmatrix} \\ & oper\end{bmatrix} \tag{2.51}$$

The architecture in which the COLL feature is part of the INDEX feature accounts for the observation that collocations transfer to anaphora as in (23a), even cross utterances as in (23b)[56].

(23) a. The analysis$_i$ which$_i$ we performed / #executed was wrong.
b. Have you seen this analysis$_i$?
I would like to know who performed / #executed it$_i$.

The second approach dispenses with lexical functions. Instead, the collocate is seen as one of the possible selections out of a set of synonyms, such as {`perform`,`execute`,`do`}. While `perform` and `do` are used with `analysis`, `execute` seems to be less natural, it forms an *anti-collocation* with the base (cf. Pearce 2001). We propose to store collocations only for synsets that include more than one lexeme. Because lexical functions also inherently specify the relation between the lexemes (OPER: verb-undergoer, for example), this information is now missing. We propose to replace it by a feature COLLREL which takes values from a suitable type system that specifies relationships.

[56]The account is due to Erbach and Krenn (1993).

It is part of a type that specifies collocations, that has the additional features COLLSYNSET (to specify the synonym set) and COLLOCATE with the same function as in the former approach. In the type system every synset is a different type. The particular synset to which a lexeme belongs is specified in the lexical entry[57]:

$$\textit{verbtrans}\begin{bmatrix}\text{LEMMA} & \textbf{execute}\\ \text{SYNSET} & \textbf{s4711}\\ \text{PRED} & \textbf{execute'}\end{bmatrix}\ \textit{verbtrans}\begin{bmatrix}\text{LEMMA} & \textbf{perform}\\ \text{SYNSET} & \textbf{s4711}\\ \text{PRED} & \textbf{perform'}\end{bmatrix} \tag{2.52}$$

When two lexemes are combined, e.g. in the *head-complementizer* rule, the synset value of the lexeme will be unified with the value of COLLSYNSET, the relationship feature COLLREL will be set and the head becomes the collocate of the base:

$$\textit{verbtrans}\begin{bmatrix}\text{LEMMA} & \boxed{1}\\ \text{SYNSET} & \boxed{2}\\ \text{COMPL} & \left\langle \text{INDEX}\,|\,\text{COLL} \begin{bmatrix}\text{COLLREL} & \textbf{undergoer}\\ \text{COLLSYNSET} & \boxed{2}\\ \text{COLLOCATE} & \boxed{1}\end{bmatrix} \right\rangle\end{bmatrix} \tag{2.53}$$

Both accounts require special treatment in the grammar, because of two reasons. First, if all collocates of a lexeme are stored in the lexicon such that they get projected into syntax this will have a negative impact on the performance of the analysis. We leave this to the implementation of the particular system[58]. Second, if a collocation is specified for a lexeme (the base) this might strictly rule out violations of the collocational restriction, simply because the head does not unify with the collocate and/or the synset encoded in the lexeme's collocation feature - an undesired effect. Instead a kind of default/overwrite mechanism is appropriate. If the lexeme is underspecified

[57] for English, this information can easily be gathered from WordNet as part of the a-priori knowledge

[58] If collocations are required for *analysis* at all (and not just for generation) then it might be reasonable to check the restrictions after the parse trees have been created.

for a particular collocate, unification will take place (with the effect that the value - the collocate - can be harvested from the resulting structure). However if there *is* already a collocate specified for a certain synset then it gets overwritten by the actual collocate.

2.6.2 Collocations and the Observation Schema

The two approaches can be mapped back from typed feature structures into the logic of lexical knowledge, but because it may become a bit clumsy, we use some abbreviations. If a base lexeme specifies a collocate then the lexical function (approach 1) or the relation and the synset (approach 2) are a prerequisite, hence they are part of the *realization context* ψ (cf. section 2.1.5) with the slight difference that the realization context here relates *two* variables - one for the lexeme and one for the collocation:

(24) $\Lambda \vdash (\exists x)\ \mathcal{L}x\mathbf{base}\ \&\ \psi xy\ \&\ \text{COLLOCATE}y\mathbf{collocate}$

This formula reflects the information encoded in 2.51. This account leads to a straightforward application of the observation schema. In an ideal world situation only the naturally sounding utterances are encountered by the learning system. This means that if the lexeme `tea` is used with an intensifier (function MAGN), then `powerful` is never encountered in this context. From the observation schema it follows that if `strong` *is* encountered with `tea` in the context of MAGN then one cannot say that `strong` is not a collocate for this function and base lexeme. Of course, if the system cannot decide between the MAGN reading and the compositional reading, then there is total neutralization. In the real world situation there will be usages of anti-collocates, but these are treated as noise in line with all the other lexical features (cf. section 2.4). This means that there is no 'special' treatment of collocations in Learn$-\alpha$ apart from the application of collocational features in a default/overwrite model of language.

2.7 Lexeme Detection

As argued in the introduction, one of the two broad categories of lexical acquisition tasks is the detection of lexemes (cf. section 1.3.1). The present section concerns the logical perspective that the theory of Learn$-\alpha$ adopts towards this task. According to definition 11 in section 2.1.5 a lexeme is entailed in a lexicon Λ if at least its graphical representation and sense is

encoded in it ($\Lambda \vdash (\exists x)\ \mathcal{L}x$ & Sx**s** & Gx**g**). Lexemes can be given a-priori in Λ_0 or a-posteriori in any extension of Λ_0 by a process that is called *lexeme detection* in Learn$-\alpha$. It also follows from the discussion of knowledge of propositional facts in section 2.1.6 that a lexeme can only be said to be *known* in a lexicon Λ if it is entailed by it as well as its optimal extension Λ_{opt}.

Prerequisite for lexeme detection is a proper segmentation of the experienced utterances. This might be a more or less difficult NLP task depending on the orthographic properties of the given language. In this work proper segmentation is taken for granted for simplicity's sake. So what is at stake is the question of whether the induction of lexical features by virtue of the generalized observation schema and the induction schema includes Sx**s** as well or whether this very feature has to be excluded on general grounds. This question is difficult to answer without making any further assumptions on the internal structure of the feature's value. Note that, for example, the feature value does not necessarily have to be related to any semantic net in order to fulfill the definition mentioned above. It is fully up to Γ how much sense it differentiates. In the worst case of a semantically 'flat' language model all lexemes could be homonyms of each other[59].

The least that can be said about lexeme detection in Learn$-\alpha$ is that every utterance that is grammatical is made up of atomic expressions that have certain meanings or functions, whatever their specific meanings or functions may be. They are atomic in the sense that they cannot be further taken apart without loosing the assigned meaning or function. An atomic expression can be a lexeme, a nonce word, a neologism or a named entity (NE). From that perspective, lexeme detection is subject to the observation schema, because from the analysability in the optimal model it follows that every word in the utterance is either a single word lexeme, an NE, a nonce word/neologism or it is part of an expression that comprises multiple words and which is either a single word lexeme, an NE or a nonce word/neologism. Otherwise the utterance cannot be mapped to any structural description. In general because of the ambivalent status of an atomic expression its status of being a lexeme is neutralized and thus to be decided with $\mathrm{RDoA}_{max}>1$. This means that the lexical status of expressions can be acquired to that very extent as lex-

[59] In ERG, for example, the value of feature PRED that relates to the meaning of signs is of type *predsort* which subsumes an ontology for the closed class items only. For open class items the type is equivalent to an arbitrary string so that different lexical entries can encode different senses. However these senses do not relate to any ontology, except those for which special noun types are defined, such as temporal noun types ('Tuesday', 'September') with the aim to capture conversion of NPs to PP-like phrases (cf. Copestake 2001). See also the footnote on definition 11.

emes, NEs, and nonce words/neologisms show different statistical properties. What the system can learn about the content of the sense feature value is very much dependent on the grammar employed, but it is obvious that the challenges discussed in section 2.6 have to be faced likewise. However, this is not to say that lexeme detection by virtue of Learn$-\alpha$ is useless, but quite the contrary. It can be used to trigger system components that deal with word sense acquisition and disambiguation, for example based on algorithms such as explored in Yarowsky (1995).

The following discussion is divided into the two cases of a) single word lexemes in subsection 2.7.1 and b) MWEs in subsection 2.7.2. Subsection 2.7.3 classifies MWEs and discusses their handling in Learn$-\alpha$. The reader who is familiar with MWEs may therefore want to skip subsection 2.7.2.

2.7.1 Detection of Single Word Lexemes

Lexeme detection splits into a) the identification of *single word lexemes* and b) the detection of multiword expressions. Much research has been dedicated to the latter although the former is not trivial either. A principal issue with the former is the circumstance that the lexeme candidate cannot be semantically related to any components it is made of because its components do not convey meaning. So the challenge remains to separate them from alleged lexemes - like named entities or neologisms (if those should be excluded from the lexicon), misspelled words, irregular forms, foreign words or words that are not independent of *fossilized MWEs* (such as 'kith and kin', 'go/be haywire', 'to and fro' or German 'klipp und klar' (non-literally: 'clear as daylight'))[60]. It may be that the real lexemes can be reliably retrieved from the set of candidates by looking at their distribution. This account would link Learn$-\alpha$'s feature $sx\mathbf{s}$ to theories that see word sense *as* distribution, like those presented in Widdows (2004)[61]. For the time being, however, this link is rather speculative and therefore out of the scope of the present version of Learn$-\alpha$. Note also that mere frequency is not sufficient given the power-distribution of the lexemes according to Zipf's law, because the relative frequency of those lexemes that are in the tail of the distribution gets so low that they may hide behind the noise barrier. The same reason might render the application of distributional theories of meaning difficult as well. The consequence of the

[60] See Moon (1998: 78–79) for more examples. Interestingly, 'kith' and 'haywire' are listed in WordNet (version 3) as single word lexemes as opposed to 'fro' that is only part of a multiword expression.

[61] For a related word sense disambiguation algorithm based on context clusters see Schütze (1998).

discussion is that our implementation will indeed exploit its unknown word handling to find single word lexeme candidates, but it will not be able to properly filter out the noise along with that. It deserves an empirical investigation how much noise is introduced into the lexicon due to that (cf. chapter 4).

2.7.2 Detection of Multiword Expressions

The situation with MWEs is different because the circumstance that they are made up of multiple words can be exploited for their identification. The behavior of an MWE can be meaningfully compared with the expected behavior of the single words occurring in it.
The proper treatment of MWEs in the context of linguistically precise NLP appears to be one of the prominent research topics (Sag et al. 2002) and their automatic acquisition is still a great challenge, which is why we dedicate some space for it in this subsection.
We adopt the working definition used by the Stanford MWE project[62]: an MWE is 'any phrase that is not *entirely* predictable on the basis of standard grammar rules and lexical entries'. At least five types of idiosyncratic behavior can be distinguished: lexical, morpho-syntactic, semantic, pragmatic and probabilistic idiosyncrasies. Not every MWE shows each type of markedness, but at least one of it. Consequentially, not every MWE is a lexeme on its own - only those MWEs that have idiosyncratic meanings will be represented by dedicated lexemes. The strong interplay, however, between these subclasses of MWEs justifies treating them in one section (which is dedicated to lexeme detection in the first place). According to the MWE project the only sufficient conditions for MWE-hood are semantic/pragmatic non-compositionality, syntactic irregularity[63], and non-identifiability (meaning cannot be predicted from its surface form). This sets them clearly apart from collocations as defined in section 2.6.

Lexical Idiosyncrasies

MWEs are lexically marked if (at least) one word included is not homophone or homograph to any single word lexeme. Such a word does not occur outside

[62] http://mwe.stanford.edu/reading-group.html

[63] Not including lexicogrammatical fixedness, defined as 'formal rigidity, preferred lexical realisation, restrictions on aspect, mood, voice, etc.', which is neither necessary nor sufficient (http://mwe.stanford.edu/reading-group.html)

MWEs and does not have a meaning of its own. We adopt the term *cranberry expression* from Trawiński et al. (2008)[64], but our definition is more restrictive: it excludes examples such as *jn. aufs Abstellgleis schieben* which has the metaphorical meaning *to put so. on inactive reserve or to deprive so. of his/her influence*, discussed in Trawiński et al. (2008). While this MWE is in their list of cranberry expressions - with *Abstellgleis* (*holding track*) as a cranberry word - it is a not lexically idiosyncratic in our account because the word has a meaning (albeit rather technical) outside the MWE.
Cranberry words are sometimes called *fossil words* because they have lost their status as a free lexeme that they might have had in the past. Some examples are given in subsection 2.7.1. A list of English and German cranberry words is freely available on the web site of the Collection of Distributionally Idiosyncratic Items (CoDII)[65]. Because the meaning of a cranberry expression cannot be determined by the meaning of its parts - as there is no meaning for at least one of its parts - such a phrase automatically counts as an MWE. The cranberry word is supposed to show an extremely high association score with other parts of the MWE, because the MWE *is* its distribution. So the detection of cranberry expressions should be fairly straightforward.

Morpho-Syntactic Idiosyncrasies / MWE Paradigms

The most extreme form of morpho-syntactic idiosyncrasy is the violation of grammatical rules by MWEs such as `at all`, `by and large` or `every which way`[66]. They are called *ill-formed collocations* in Moon's classification of MWEs (Moon 1998)[67] and *fixed expressions* in Sag et al. (2002). Because of their syntactic invariability they are subject to a *words with spaces* treatment (ibid.). In Learn$-\alpha$ they are treated as closed-class lexical entries, hence they

[64] A similar effect can be observed on sub-word level, where some morpheme-like parts do not occur outside of the particular word, as in 'cranberry', which is why those MWEs are sometimes called *cranberry collocations*.

[65] http://www.sfb441.uni-tuebingen.de/a5/codii/, discussion cf. Trawiński et al. (2008).

[66] The existence of these word combinations of course does not justify grammatical rules that could account for them.

[67] 'Ill-formed collocations break the conventional grammatical rules of English' (ibid., page 21). Note that in Moon's work ill-formed collocations form one macrocategory of fixed expressions and idioms that relate to 'problems of lexicogrammar' (ibid., page 19): *anomalous collocations* along with cranberry collocations, defective collocations (that have an idiosyncratic meaning component) and phraseological collocations, which are paradigmatic but semi-productive. She uses the probabilistic definition of the term *collocation*, cf. section 2.6.1.

are given a-priori[68].
Not all fixed expressions need to be syntactically ill-formed. Consider the MWE `in and out` in the idiomatic sense 'going regularly to a place' (Hornby 2005: 782) in example (25a). The modification in (25b) seems to distort the idiomatic meaning:

(25) a. He was in and out of jail in most of his life.
(example by Hornby 2005: 782)
b. ??He was in and often out of jail in most of his life.
[in the idiomatic interpretation]

Another form of morpho-syntactic idiosyncrasy are constraints that an MWE imposes on its realization in syntax, constraints which do not hold in the fully compositional interpretation of it. This behavior is often described under the headword *syntactic variability* (e.g. Sag et al. 2002 who categorize these MWEs as *semi-fixed expressions*). An often cited MWE is (26a) which has the additional interpretation 'John died'. The MWE `kick the bucket` is not a fixed expression, because it allows inflection on the verb. However, neither passive voice in (26b) nor modification in (26c) allows for the idiomatic reading (the last example is adopted from Sag et al. 2002):

(26) a. John kicked the bucket.
b. *The bucket was kicked by John
[in the idiomatic interpretation]
c. *John kicked the great bucket in the sky
[in the idiomatic interpretation]

According to Sag et al. (2002) the only types of lexical variation are inflection (*kick*, *kicks*, *kicked*) and variation in reflexive form (*wet oneself*, *wet yourself*, ...). In addition proper names show a high degree of syntactic idiosyncrasy. However, within lexicon acquisition, we propose to treat them separately (under the headword *named entity recognition*).

The other pole of the continuum of morpho-syntactic marked MWEs are *semi-productive constructions*, called *phraseological collocations* in Moon's typology. These are of special interest for the theory of Learn$-\alpha$, because of their vast magnitude that makes it difficult to provide them a-priori indicating that they are part of the *language engineering bottleneck* (cf. section 1.1). These MWEs are best described in a paradigmatic manner, so that the

[68]This is in line with the way followed in ERG, see also Sag et al. (2002: section 3.1) for discussion.

multiword lexeme just specifies whether it opts for the paradigm or not[69]. Henceforth, we will call them *paradigmatic MWEs*. They can be divided into two subclasses: a) MWEs that belong to a paradigm which is made up of at least one closed class item and b) those that are not. Here we discuss two for a), namely *determinerless PPs* and VPCs, and one for b), that is transitive verbs taking bare-nouns as direct object.

Classified in terms of their syntactic idiosyncrasy, determinerless PPs fall into two categories, the unmarked and the marked ones (Baldwin et al. 2003b). A determinerless PP is unmarked - as in example (27a) below - if the head noun of the governed NP can indeed project into syntax without (overt) determiner evidenced by examples such in (27b). Both a) and b) are due to an uncountable sense of the noun. A construction as in (27c) is considered anomalous given the fact that the noun `car` is countable and hence requires an overt determiner in other syntactic contexts such as (27d)[70]. This anomaly must be accounted for by the lexicon. Because of the interaction with the count/mass dichotomy c) and d) are subject to (total) neutralization and form another example of bifurcations in lexical acquisition space (cf. section 2.5.6).

(27)
a. in school
b. school is over
c. by car
d. *(the) car is damaged

The question remains *where* the idiosyncrasy has to be stored. Baldwin et al. (2003b) argue that example (27c) is due to the capability of the preposition `by` which subcategorizes for determinerless nouns that belong to a class that denotes MEANS/INSTRUMENTs (examples (28a) and (28b)). Even in the context of a flying carpet this shows productive behavior. Moreover the preposition even rules out noun complements with determiner as in (28c). It can be added that there is also no modifier allowed ((28d) versus (28e)).

(28)
a. I arrived yesterday by train / by carpet
b. by hammer / by computer
c. *by a/the car
d. *by fast car
e. as former president

[69] As a paradigm becomes part of the grammar, it is questionable whether the related expressions should be categorized as syntactic idiosyncrasies.

[70] Baldwin et al. (2003b) note, however, that the noun can be used without determiner in sentences such as 'car costs less than train for trips to the city'.

Thus this PP headed by `by` is paradoxically a fixed, but highly productive MWE and it makes sense to encode the paradigm via a lexical entry of `by`. Consequently, if the semantic classes (±MEANS/INSTRUMENT), as well as the lexical entry for `by` are given a-priori (and if there is a means for coercion etc.) then there is no issue of lexeme detection at all.
However, not all determinerless PPs are like that. Baldwin et al. (2003b) discuss determinerless PPs with the nouns `sea` and `hand`. These are fully compositional but the set of available prepositions seems idiosyncratic: *at sea, to sea, from sea to ..., %by sea, *in sea, *over sea.* As long as there is no semantic analysis explaining the differences, the system will have to store these idiosyncrasies along with the respective noun. This means that the non-head (the noun) also selects the head (the preposition). The mechanism developed in Soehn and Sailer (2003) for HPSG can be employed to account for this type of determinerless PPs.

Verb particle combinations (VPCs) are MWEs with a rather complex paradigm, which we will now discuss using the terminology developed in Quirk et al. (1994) and summarized in table 2.4 (ibid. page 1161). The table shows free (non-idiomatic) instances of VPCs (A) and the lexicalized counterparts (B). The VPCs in (1) and (2) consist of a head verb or verb-object phrase followed by a spatial adverb. If the VPC is lexicalized, then the respective verb is called a *phrasal verb.* In (3) and (4), the verb (or verb-object phrase) is followed by a PP instead of an adverb, hence the verbs - if they subcategorize for the PP - are called *prepositional verbs.* In (5) and (6) we find a combination, consisting of verb (or verb-object phrase) + adverb + PP. The verbs that subcategorize for this construction are called *phrasal-prepositional verbs.* Within these three different headings, Quirk et al. (1994) distinguish two types: Type I for the intransitive, Type II for the transitive variant.

It must be noted that most of the transitive phrasal verbs (2) are not restricted to realize their objects between head and particle (as the table might suggest), the objects - if they are non-pronominals - can also follow the particle. There is even a minor class of Type II phrasal verbs that do not allow the object to intercede between head and particle - we call them Type IIb phrasal verbs[71]. We shall call the class of phrasal verbs that restrict their objects to precede the particle *Type IIc phrasal verbs*[72].
Apart from semantic idiosyncrasies of the lexicalized instances, there are a couple of important *grammatical* restrictions that accompany these construc-

[71] See the example ?*carry the threat out* in Baldwin and Villavicencio (2002)
[72] For instance **I was crying out my eyes*, cf. Quirk et al. (1994: 1155).

Table 2.4: Multiword verbs in Quirk et al. (1994)

		Lexical Verb	Direct Object	Particle Adverb	Pre-position	Pre-positional Object
1 (free combination)	(A)	come	-	in	-	-
Type I Phrasal Verb	(B)	crop	-	up	-	-
2 (free combination)	(A)	send	someone	away	-	-
Type II Phrasal Verb	(B)	turn	someone	down	-	-
3 (free combination)	(A)	come	-	-	with	+ me
Type I Prepositional Verb	(B)	come	-	-	across	+ a problem
4 (free combination)	(A)	receive	something	-	from	+ me
Type II Prepositional Verb	(B)	take	someone	-	for	+ a fool
5 (free combination)	(A)	run	-	away	with	+ it
Type I Phrasal-Prepositional Verb	(B)	come	-	up	with	+ an answer
6 (free combination)	(A)	send	someone	out	into	+ the world
Type II Phrasal-Prepositional Verb	(B)	put	someone	up	for	+ election

tions. Because particles like 'up', for instance, can be realized as an adverb *or* a preposition, it is especially interesting to find out whether a given phrase is a VPC or a Type I prepositional verb. According to Fraser (1974) it is 'the clearest syntactic difference between these pairs' that the first may realize its particle after the object[73], whereas the second cannot, as exemplified in the sentences in (29) ibid. page 1-2:

(29) a. He sped { the process / *the pole } up / up { the process ('hurried') / the pole ('climbed quickly') }

b. Harry will look { the client / *the fence } over / over { the client ('examine') / the fence ('peer above the top of') }

c. The man reeled { the line / *the street } in / in { the line ('wound up') / the street ('staggered') }

d. She ran { the pamphlets / *the stage } off / off { the pamphlets ('print') / the stage ('fled') }

Further grammatical differences are exemplified by (30). First, phrasal verbs of Type II do not allow adverbs to precede the particle (30a), whereas this restriction does not apply to free combinations; second, whereas freely adjoined PPs may often occur sentence-initially, this is not possible with

[73]It should be added: if it is not a Type IIb phrasal verb.

phrasal verbs (30b); third, the particle cannot be a residue in gapping constructions (30c). Further, the particle cannot be pied-piped in wh-constructions like relative clauses (30d) or questions (30e)[74].

(30)
a. *Harry looked furtively over the client
b. *In the line, the man reeled as if drunk
c. *He sped up the process, and she, up the distribution
d. *the man up whom they called
e. *Up which man did they call?

There is a further phonological difference: the particle of a phrasal verb usually receives stress, whereas the preposition is weakly stressed. This is why we generally prefer the notion *homography* instead of *homonymy* throughout this work, as we are dealing with written input only.
Type I phrasal verbs show similar grammatical restrictions that appear to support the cohesion between the verbal head and the particle, as demonstrated in (31)[75]:

(31)
a. *The mine caved quickly in
b. *The car slowed all the way up
c. *The varnish coat wore down, and the undercoat away
d. *The hippies want to turn ON, not OFF
e. *Up blew the tank.

Like Type II phrasal verbs, it is not grammatical for adverbs to precede the particle ((31a) and (31b)), whereas this restriction does not apply to free combinations ('Walk straight in', also cf. Quirk et al. (1994: 1152)). Again, the particle cannot be the residue of verbal gapping ((31c) and (31d)). Further, the particle normally cannot be fronted (31e)[76].
The structural isomorphisms between the compositional and the non-compositional utterances lead to total neutralization and pose a major challenge for the detection of VPCs. The problem is that if an utterance is observed which realizes a structure that is not available for a VPC (such as (31b)) then this does not prove more than that the verb *in this particular utterance* is not part of a VPC. Likewise, an utterance that realizes the [V NP P] structure shows that V can enter a VPC, but it does not prove that [V P] is a phrasal

[74]The examples a) to c) are from Fraser (1974: 2), examples d) and e) are from Quirk et al. (1994: 1167).
[75]The examples a) to d) are from Fraser (1974: 4), example e) is from Quirk et al. (1994: 1153).
[76]but cf. Quirk et al. (1994: 1153) for exceptions.

verb, it can be a directional verb with a fully compositional interpretation.

Both determinerless PPs as well as VPCs create paradigms in which open class lexemes interact with closed class lexemes (particle, preposition). In contrast there are paradigms which consist of open class lexemes only. We shall discuss one of them, namely MWEs that consist of a verb and a determinerless noun as a direct object. As with the previous types of paradigmatic MWEs, there is a continuum of fully compositional up to opaque verb object combinations. There is a broad class of (usually countable) nouns that can occur without article in the direct object slot of a verb: locational nouns. This class has been extensively studied in Stvan (1998). Although he is mostly concerned with determinerless PPs, he also did tests on bare nouns in direct object position[77] (Stvan 1998: 25). He tested the acceptability of nouns occurring in the template sentence *He planned to finish it before leaving* ___ on a set of locational nouns. The test was done with native speakers and the results were reinforced by a corpus search. Leveraging on these results we would like to note that not all locational nouns are able to occur as bare nouns in the direct object slot. While (32a) is perfect, (32b) seemed questionable to the informants and (32c) unacceptable[78]. Interestingly, these nouns are accepted in the determinerless PP paradigm ((32d) and (32e)).

(32)
a. ... before leaving school / home / town.
b. ?... before leaving bed / stage.
c. *... before leaving country / table / hall / kitchen.
d. in country / hall
e. at table

Whatever theory will be worked out to account for these differences in terms of semantics and/or pragmatics, the point is that if the system does not have a corresponding means of differentiation, it will have to rote-learn this behavior lexeme by lexeme. Moreover it is challenging to set these cases apart from the opaque idioms, such as the ones listed in (33)[79]:

[77] He also tested the availability of the subject position, which is not of relevance for our discussion.

[78] For demonstration we are citing only a few of the nouns tested.

[79] The examples and paraphrases are taken from Hornby (2005). The verb object combination is underlined by us.

(33)
a. The pillars gave way and a section of the roof collapsed.
to break or fall down
b. The film festival takes place in October.
to happen, especially after previously being arranged or planned
c. I lost count and had to start again.
forget the total of sth before you have finished counting it

Although it might be possible for humans to relate the idiomatic meaning to the meaning of the simplex words, this will not be possible for a contemporary NLP system. Note that the MWEs are syntactically marked in two ways: a) the verbs take count nouns without article and b) they are semi-fixed expressions: passive voice blocks the idiomatic reading and no modification is possible.

Deviant Semantics/Pragmatics

Semantic and/or pragmatic idiosyncrasies are usually discussed in the context of idioms or idiomaticity. An MWE is semantically/pragmatically idiosyncratic or non-compositional when the meaning or use of the MWE cannot be entirely predicted by the meaning or use of its parts together with the principles underlying their combination. This shall not include mere lexical selection by virtue of collocational preference (section 2.6). Moon (1998: 19) separates 'problems of pragmatics' from 'problems of semantics'. While the first deal with formulae (simple formulae, sayings, proverbs and similes) the second concern metaphors (transparent, semi-transparent and opaque).
Non-compositionality is a sufficient condition for MWE-hood (cf. the Stanford MWE project) and a necessary condition for idioms (Nunberg et al. 1994: 492 use the term 'conventionality'). Apart from that, MWEs can vary along multiple dimensions, which are neither sufficient nor necessary conditions for idioms or MWEs in general: inflexibility (cf. the paragraphs on morpho-syntactic idiosyncrasies above), figuration (metaphor, metonymy, hyperboles), proverbiality, informality, affect and situatedness[80].

Nunberg et al. (1994) argue at length that many (if not most) of the idioms are 'compositional' in the sense that 'the phrasal meaning, once known, can be analyzed in terms of the contributions of the idiom parts' (page 498). This is why the notion 'non-compositional' is perhaps misleading, which is

[80]These criteria except the last one were discussed by Nunberg et al. (1994). The Stanford MWE project added the last one, with the intention that the MWE 'is associated with a fixed pragmatic point' (*good morning*, *all aboard*).

why we prefer 'semantic/pragmatic idiosyncrasy'. This type of idiom is called *idiomatic combination* in Nunberg et al. (1994: 496) and *idiosyncratically decomposable MWE* in Baldwin et al. (2003a). Examples are `spill the beans` (divulge a secret) and `pull strings` (exploit personal relationships), in which parts of the idioms 'take over' parts of the intended meaning: *spill* ∼ *divulge*, *the beans* ∼ *secrets* or *pull* ∼ *exploit*, *strings* ∼ *personal relationship*, respectively. Idioms that refuse this kind of analysis are called *idiomatic phrases* in Nunberg et al. (1994: 497) and *non-decomposable MWE* in Baldwin et al. (2003a). A prototypical example is `kick the bucket`. This means that semantic/pragmatic idiosyncrasy is a dimension of MWE-hood that goes from fully compositional over decomposable up to unanalyzable. Nunberg et al. (1994) make the point that it is the analysability that accounts for the observable differences regarding inflexibility. While `kick the bucket` cannot be passivized without destroying the idiomatic meaning (cf. the discussion of semi-fixed expressions above), the idiomatic combinations can undergo quite a couple of syntactic processes, as exemplified in (34) (all examples taken from Nunberg et al. 1994).

(34)
a. Reinventing and Tilting At the Federal Windmill.
b. Pat got the job by pulling strings
that weren't available to anyone else.
c. We could...pull yet more strings.
d. Those strings, he wouldn't pull for you.
e. I was worried that [the beans]$_i$ might be spilled,
but they$_i$ weren't __
f. The beans were spilled by Pat.
g. The cat seems to be out of the bag.

In the first example (34a) (a headline from *The Washington Post Weekly*) the idiom *tilt at the windmill* undergoes two variations: the verb is coordinated with another free verb (*reinvent*) and the noun is modified by an adjective. Modification by relative clause is also possible (34b). In (34c) the idiom undergoes quantification and in (34d) topicalization. Even coreference with a pronoun and VP ellipsis is possible in (34e). Passive is possible (34f) as well as raising (34g).

According to Nunberg et al. (1994) the syntactic variations available to the idiom must be due to its semantics. Also from a point of learnability - and this brings us back to the context of lexical acquisition - it seems awkward to assume syntactic arbitrariness of idioms: 'It is evident that speakers are never explicitly taught which idioms passivize and which don't' (Nunberg et al. 1994: 507), this statement appears to carry over to all the other

syntactic variations available.

The discussion extends to noun compounds to which we turn now, although the link to flexibility loses ground because parts of noun compounds cannot undergo syntactic processes. As the clear majority of noun compounds is binary[81] we restrict ourself to those. Noun compounding is an extremely productive process yielding compositional as well as fully non-compositional combinations. From this point of view the rule $N \rightarrow NN$ can be seen as a 'forge of lexemes'. Noun compounds are traditionally divided into (semantically) endocentric and exocentric combinations. Semantically endocentric means that the noun compound is a specialization of the head noun, hence a hyponym thereof (cf. Baldwin et al. 2003a). Exocentric combinations have been traditionally treated as non-compositional although this is not necessarily the case as pointed out by Benczes (2004)[82]. It depends on the semantic relationships that one wants to maintain between the parts of the compound and the thing denoted by it. While (35a) is endocentric[83] because an apple tree *is* a tree, (35b) is exocentric because a person is not a skin. However (35b) is not more intransparent than *spill the beans*. There is a simple *pars-pro-toto* relationship between the person (said to have reddish skin) and the reddish skin. Even the semantic contributions of the nouns *jail* and *bird* to the compound in (35c) ('person serving a prison sentence') can be explained in terms of a process called blending (roughly because of the metaphor that a prisoner is like a caged bird, cf. Benczes 2004). In contrast (35d) is fully non-transparent (disregarding all etymological considerations).

(35)
a. apple tree
b. redskin
c. jailbird
d. butterfly

Even in the endocentric case the interpretation of noun compounds is an extremely challenging task for an NLP system because of three reasons (Lapata 2002): a) the high productivity of the compounding process, b) the implicit semantic relationship between head and modifier, and c) they are influenced by contextual and pragmatic factors. There is even

[81] Kim and Baldwin (2005) estimate 'that 88.4% of NCs [noun compounds] in the Wall Street Journal section of the Penn Treebank and 90.6% of NCs in the British National Corpus are binary' (footnote 4).

[82] 'Since the vast majority of English compounds is endocentric [...], linguistic literature has a tendency to mention exocentric combinations only peripherally (if they are mentioned at all), and views these constructions as semantically non-transparent.'

[83] Note that we do not dwell on homographic differences whether a compound is written in solid form (*housewife*) or hyphenated (*house-builder*) or with spaces (*house wine*).

low agreement between human annotators for the interpretation task (Kim and Baldwin 2008). From this perspective noun compounds are semantically/pragmatically idiosyncratic given the definition at the outset because the meaning is *not* entirely predictable by knowing the parts. Automatic interpretation/acquisition of noun compounds has been tackled by a couple of researchers using different means: 'semantic scattering' (Moldovan et al. 2004), WordNet similarity (Kim and Baldwin 2005), kernel methods (Ó Séaghdha 2008) or web-mining (Nakov and Hearst 2008).
Before an endocentric interpretation of a noun compound can be exercised, an NLP system has to decide whether the compound is 'compositional' at all. In other words, if the compound is unknown to the system, it becomes a matter of lexeme detection. In one line of research an attempt is made to compare the semantic similarity of the constituents to the MWE. This strategy is not restricted to noun compounds. The rationale behind it is the hypothesis 'that where the similarity between the constituents of an MWE and the whole is sufficiently high, the MWE must be of simple decomposable type'[84] (Baldwin et al. 2003a). In their paper the authors use latent semantic analysis (LSA) as the similarity method based on WordNet hyponymy relation and semantic distance.

Probabilistic Effects

The last idiosyncrasy to be discussed is a statistical one. According to the MWE project, being an institutionalized or conventionalized expression is a necessary (but not sufficient) condition for an MWE. They define 'institutionalization' as a 'process of an expression becoming recognised and accepted as a lexical item, through consistent use over time'[85]. While this process is one of the qualities of MWEs, mere probabilistic effects are not. However, it is often claimed that institutionalization is accompanied by statistical idiosyncrasies: 'Institutionalized phrases are semantically and syntactically compositional, but statistically idiosyncratic' (Sag et al. 2002)[86]. If this is correct then this means that MWEs can be reliably detected by statistical inference. This kind of approach has flourished since the seminal work of Firth (1957). The question concerning appropriate statistical means, however, is still a matter of empirical studies despite the notorious difficulties of comparing the results

[84] By 'simple decomposable type' the authors refer to MWEs that are semantically composed of simplex senses, but nevertheless MWEs by virtue of their being institutionalized. These are collocations in our work.

[85] http://mwe.stanford.edu/reading-group.html

[86] In this context the authors refer to institutionalized and only institutionalized (hence compositional) phrases.

of different studies: 'The use of statistical measures is widespread in NLP but there is no consensus about how good these measures are for describing natural language phenomena. It is not clear what exactly they capture when analysing the data.' (Villavicencio et al. 2007: 1038)
A recent test reported in Pecina (2005) has compared 84 (!) different association measures on a collocation extraction task using annotated Czech text from the *Prague Dependency Treebank*. Among the basic association methods best performance was achieved by *pointwise mutual information* (cf. subsection 2.3.5). The performance was further enhanced by linearly combining the basic association methods using logistic linear regression. Another recent detailed work on the statistical aspects on cooccurrences and collocations can be found in Evert (2005). In one of his tests (Evert 2005: 142–143) log-likelihood outperformed MI and χ^2 (among others). In a different test on German ADJ-N pairs, χ^2 surprisingly outperformed log-likelihood. This again contrasts with the result of the study presented in Villavicencio et al. (2007), which suggest that χ^2 is not able to differentiate MWE from non-MWEs (but MI *inter alia* can). As a result we agree with the statement that 'collocation extraction is not a solved problem' (Evert 2005: 165) and it can be questioned whether it will ever be solved. There might be a general limit of the applicability of cooccurrence statistics on the problem of MWE identification.
One of the major issues for lexeme detection is that statistically significant cooccurrence does not guarantee non-compositionality. But at least it can make a contribution to the whole process which at the end has to be a hybrid one that comprises statistical, morpho-syntactical and semantical/pragmatical means.

2.7.3 Multiword Expressions in Learn$-\alpha$

In the context of Learn$-\alpha$ we propose to characterize MWEs along two dimensions: fixedness and transparency. The first dimension is of a morpho-syntactic nature, the second is related to semantics/pragmatics. **MWEs of different types are supposed to be acquired differently.** Further, leveraging on the discussion of the last subsection, it makes sense to segment both dimensions by means of binary features[87]. The first dimension (fixedness) varies from absolutely fixed up to free word combinations (cf. the paragraph about MWE paradigms on page 163). We fully acknowledge that MWE dimensions are continua rather than discrete properties. The proposed classification, nevertheless, is a heuristic that will indicate how various proto-

[87] These features are used for mere classification; they are not lexical features.

typical classes may be tackled in Learn$-\alpha$. For the first dimension the feature FIXED is used to denote whether there is some morpho-syntactic fixedness and the feature PARADIGMATIC is designed to account for those MWE types that seem to follow certain paradigms such as VPCs or determinerless PPs. Because semi-fixed expressions also seem to interact with paradigms (such as inflection or reflexive pronouns) it makes sense to characterize semi-fixed expressions as [+FIXED, +PARADIGMATIC], and the same also holds for noun compounds. The first dimension is then characterized as follows:

+FIXED	−PARADIGMATIC	fixed expression
+FIXED	+PARADIGMATIC	semi-fixed expression
−FIXED	+PARADIGMATIC	free paradigmatic expression
−FIXED	−PARADIGMATIC	free word combination

For the second dimension, based on the paragraph about deviant semantics/pragmatics (page 170), we propose the two features COMPOSITIONAL and DECOMPOSABLE with the following characterization:

+COMPOSITIONAL	+DECOMPOSABLE	fully compositional expression
−COMPOSITIONAL	+DECOMPOSABLE	idiomatic combination
−COMPOSITIONAL	−DECOMPOSABLE	idiomatic phrase
+COMPOSITIONAL	−DECOMPOSABLE	-

The combinations of the four features lead to the categorization of MWEs presented in table 2.5. The table also includes collocations for the sake of comparability.

Table 2.5: Categorization of Multiword Expressions and Collocations

	FIXED	PARAD.	COMP.	DECOMP.	Example
A.1	+	–	+	+	?
A.2	+	–	–	+	*kith and kin*, *good morning*
A.3	+	–	–	–	*by and large*, *ad hoc*
B.1	+	+	+	+	*by car*, *in school* *at seven o'clock*
B.2	+	+	–	+	*housewife*, *redskin*
B.3	+	+	–	–	*butterfly*, *red herring* *kick the bucket*, *give way* *go haywire*
C.1	–	+	+	+	*perform an analysis* *brief moment* *take a decision* *as former president*
C.2	–	+	–	+	*at (great) length*, *eat up* *take advantage*, *spill the beans*
C.3	–	+	–	–	*turn sth./sb. down*

This categorization allows us to indicate how Learn$-\alpha$ deals with MWEs. The class A.1 is hypothetical, because it is not clear whether there is any fixed expression that is fully compositional. Therefore it is assumed that all fixed, non-paradigmatic expressions are non-compositional. Ill-formed MWEs (A.3) are opaque by definition. Because they have to be listed a-priori neither form nor meaning are subject to lexical acquisition. Class A.2 is expected to show very high association scores and is therefore very susceptible to statistical inference. Being non-compositional each element of this class is treated as a separate lexeme automatically.
Group B comprises semi-fixed expressions. The compositional class B.1 has to be accounted for by rules in Γ reflecting the relatively fixed paradigms. Its elements are not subject to lexeme detection. The non-compositional classes B.2 and B.3 might be well captured by statistical inference - limited by the noise barrier. It is a matter of the implementation whether decompositionality is accounted for in any case. The prototype presented in this thesis does not account for this distinction.
Group C contains the most variable phrases. Class C.1 is not an MWE by definition because there is no idiosyncrasy (except preference) whatsoever. It is therefore accounted for along the lines developed in section 2.6. The remaining classes C.2 and C.3 are similar to B.2 and B.3 with the difference

that their elements are much more flexible. Again, the implementation may vary regarding decompositionality. And as with group B statistical inference can help to detect those MWE lexemes that are not swallowed by noise. One of the great challenges for accounts of MWEs is the accommodation of fixedness in the paradigmatic groups (B versus C). Different implementations will have different ways of acquisition of this property. For example, elements of B and C may be inserted / checked at different stages of the analysis. The non-compositional meaning of group C.2/3-MWEs might be added on top of the compositional meanings that are already computed in the course of analysis. Recent versions of ERG pursue this kind of approach along the lines discussed in Copestake et al. (2002: section 4).

In summary, this section is an attempt to homogenize an area which is extremely heterogeneous. We fully acknowledge that the admittedly simplified categorization proposed here cannot do justice to all the fine grained distinctions related to the field. It may be, however, a first valuable step in order to bring the various linguistic phenomena, whenever possible, under the same umbrella as other lexical features like subcategorization thereby making them tangible for Learn$-\alpha$.

2.8 A Probabilistic Account of Learn−α

This section develops the full-blown probabilistic account of Learn−α. As a prerequisite subsection 2.8.1 will first explicate the notion *learning down the hierarchy*, then subsection 2.8.2 will introduce the basic counters that are required for the statistics. The formulae required for the induction step are then presented in a series of theorems.

2.8.1 Learning Down the Hierarchy

In this section, we wish to illustrate how the maximal admissible feature determination is achieved. Consider the following example:

(36) He gnarfed to get rich.

Let us discuss the observation of lexical feature values for the unknown lexeme `gnarf`. The crucial question is: For which feature values can this utterance be a reference, given a maximum RDoA of n? This question can be restated formally:

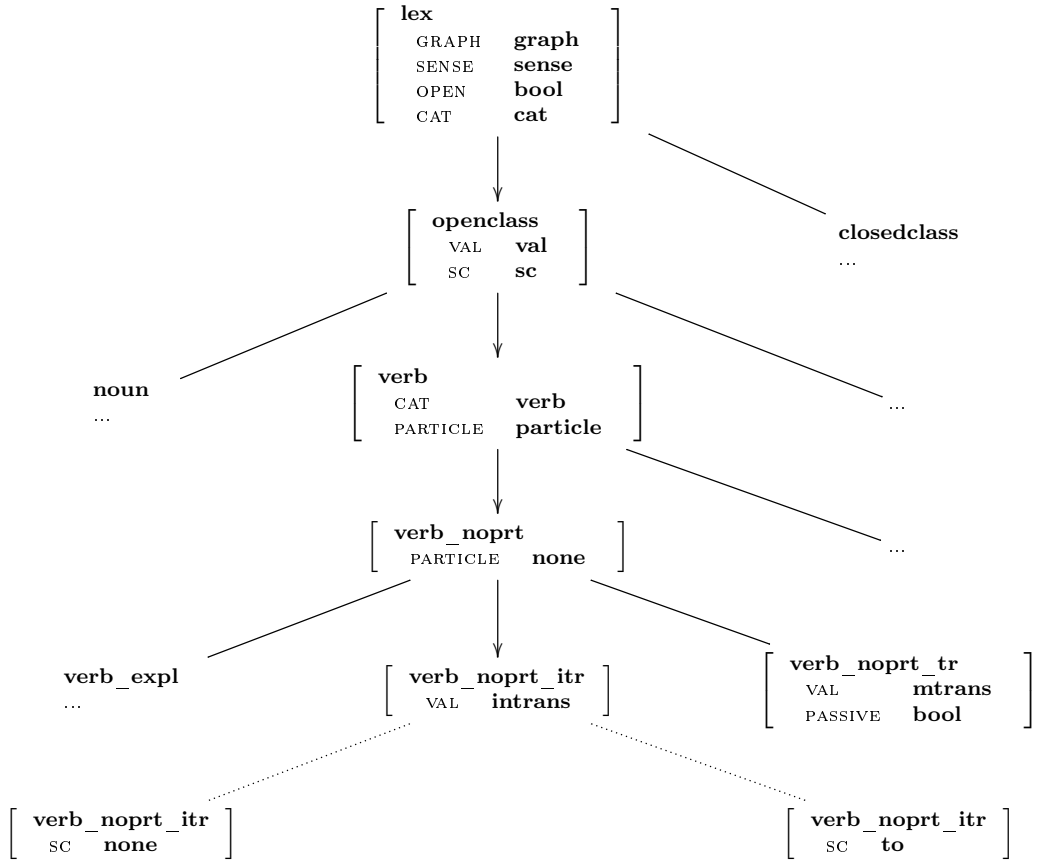
$$\textit{For which feature } \alpha \textit{ and value } \mathbf{v} \textit{ does } (36) \mapsto^{\rho=n} \Box\alpha^{gnarf}_{\mathbf{v}} \textit{ hold?} \qquad (2.54)$$

In order to determine this, let us go down the lexical type hierarchy (LTH) from the most general class *lex* to the most specific[88]. It is mere logic that forces us to follow this path as we have to first find out whether a feature is *appropriate* for this particular token of `gnarf` before we can determine the feature's value. The exemplary lexical type hierarchy in figure 2.12 will visualize where to go. This LTH comprises some of the exemplary features already introduced in section 2.5.1.

Starting with type *lex*, the LTH forces us to specify the lemma and the book-keeping feature OPEN. Let us assume that the system sets the value of this feature to + intrinsically the in the course of lexical look-up (cf. assumption 4 in section 2.1.5). The GRAPH feature value is derived morphologically by detaching the *-ed* suffix. At the same time, the CAT feature is set to **verb** because this is the only lexical category that licenses the suffix. Let us ignore the SENSE feature for ease of exposition. This will be discussed further below. Because of [OPEN +] and [CAT **verb**] we reach the type **verb**. A new feature, PARTICLE, appears on the scene (which would not be available if the

[88] We follow the way how Ann Copestake (Copestake et al. 1999b) depicts type hierarchies: the most general at the top - as opposed to how it is done in Carpenter (1991) (and in section 2.1.3).

Figure 2.12: Going Down the Lexical Type Hierarchy



token under consideration would be a noun, for example). The utterance is such that the value can be determined unambiguously: [PARTICLE **none**]. This allows us to follow the path down to type **verb_noprt** which denotes verbs without particles. It is reasonable to set them apart from VPCs, because these follow different laws. Further, the utterance allows the inference [VAL **intrans**] because it lacks a direct object. Ultimately, we reach the type **verb_noprt_itr**. Note that the feature PASSIVE (intended to specify whether passive is possible or not) is not in sight because it is not appropriate for any type reached.

We are still on unambiguous grounds (RDoA_{max}=1), which changes now when we consider the feature SC. Because *to get rich* can be an adverbial clause as well as a verbal complement, the feature's value cannot be determined under RDoA_{max}=1. So we get the following result:

$$(36) \mapsto^{\rho=1} \square \left\{ \begin{array}{l} \text{LEMMA}x, \texttt{gnarf} \\ \text{OPEN}x+ \\ \text{CAT}x\textbf{verb} \\ \text{PARTICLE}x\textbf{none} \\ \text{VAL}x\textbf{intrans} \end{array} \right. \tag{2.55}$$

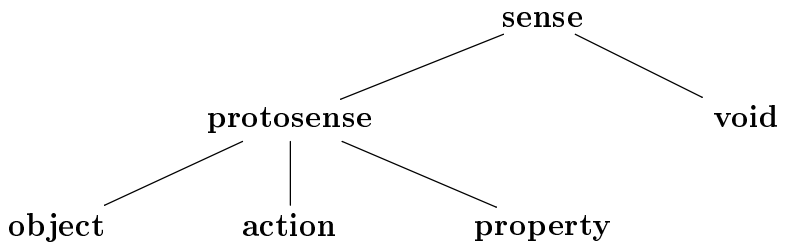
Suppose now that RDoA_{max}=2 (the dashed lines in the figure). In this case we would arrive at two lexical instances based on the two transitions:

$$(36) \mapsto^{\rho=2} \square \left\{ \begin{array}{l} \text{LEMMA}x, \texttt{gnarf} \\ \text{OPEN}x+ \\ \text{CAT}x\textbf{verb} \\ \text{PARTICLE}x\textbf{none} \\ \text{VAL}x\textbf{intrans} \\ \text{SC}x\textbf{none} \end{array} \right. \quad (36) \mapsto^{\rho=2} \square \left\{ \begin{array}{l} \text{LEMMA}x, \texttt{gnarf} \\ \text{OPEN}x+ \\ \text{CAT}x\textbf{verb} \\ \text{PARTICLE}x\textbf{none} \\ \text{VAL}x\textbf{intrans} \\ \text{SC}x\textbf{to} \end{array} \right. \tag{2.56}$$

Sense First?

In the LTH the SENSE feature is meant to specify the lexeme's semantic content according to a type hierarchy rooted in **sense**. If its appropriateness specification does not change within the LTH this feature is not a *type constraining feature*, which means that it will be orthogonal to all other lexical features. This might be desired in implementations with flat lexical semantics. It is worthwhile to point out, however, that Learn−α accounts for language models with rich lexical semantics, too. In this case the SENSE feature should be a type constraining feature and therefore part of the realization context. As a rule that could be called *sense first* it should even be part of the *ultimate realization context* (along with GRAPH, cf. section assumption 3 in section 2.1.5) and as a consequence the realization context of every lexical feature. This will require that the system learns the value of the SENSE feature before it acquires any other lexical feature. A slight modification of the LTH above is required to achieve this effect. Let us assume that the **sense** top level hierarchy looks similar to the example in figure 2.13. Here the hierarchy splits into **void**, a type that can be used to specify non-content lexemes, and **protosense**, the root of the prototypical semantic classes object, property and action (Croft 1991: 53).

Figure 2.13: Example Top Level Sense Hierarchy



It is natural to stipulate that the appropriate type of SENSE is **protosense** for all openclass lexemes, which means that all open class lexemes are content lexemes:

$$Approp(\text{SENSE}, \textbf{openclass}) = \textbf{protosense} \tag{2.57}$$

By virtue of this specification, SENSE automatically becomes part of the realization context for all features of the **openclass** type.

2.8.2 Feature Value Characteristics

In order to be able to induce the admission of a lexical feature value for a particular lexeme $\mathcal{L}$, the system has to remember how often $\mathcal{L}$ has been realized and how many cases it has observed in which the feature value has been realized. We will call all these counts and derived ratios the characteristics of a feature value. The two most salient characteristics are called *reference counts* and *number of learning opportunities*, respectively.

Reference Counts and Positive Evidence

For statistical considerations it is obvious that the learning system has to be able to count the number of utterances that prove a given feature value for a particular lexeme $\mathcal{L}$ (written as $u_i \mapsto \Box\alpha^{\mathcal{L}}_{\mathbf{v}}$). According to definition 36 in subsection 2.3.1, we call such utterances 'references for $\alpha^{\mathcal{L}}_{\mathbf{v}}$'. What is new here is that the *number* of all references for a feature value is called *reference count* (c_{ref}). To increase the precision of this notion, we also include information about the *maximum Relative Degree of Ambiguity* ρ that is allowed for the utterance to be referenced as positive evidence (cf. subsection 2.5.5). We mark the reference count accordingly:

Def. 37 (reference count) *Given is an experience set E, a lexeme $\mathcal{L}$ and a lexical feature α. The number of utterances in E that prove a given feature value $\alpha^{\mathcal{L}}_{\mathbf{v}}$ under a maximum RDoA $\rho = n$ is called reference count for $\alpha^{\mathcal{L}}_{\mathbf{v}}$ in E.*

$$c^{\rho=n}_{ref}(E, \alpha^{\mathcal{L}}_{\mathbf{v}}) \quad = \quad |\{u \in E \quad | \quad u \mapsto^{\rho=n} \Box\alpha^{\mathcal{L}}_{\mathbf{v}}\}| \tag{2.58}$$

The reference count can extend to a set of feature values that are proven simultaneously:

$$\begin{aligned} &c^{\rho=n}_{ref}(E, \alpha^{\mathcal{L}}_{1,\mathbf{v_1}}...\alpha^{\mathcal{L}}_{m,\mathbf{v_m}}) \quad = \\ &\quad |\{u \in E \quad | \quad u \mapsto^{\rho=n} \Box\alpha^{\mathcal{L}}_{1,\mathbf{v_1}} \;\&\; ... \;\&\; u \mapsto^{\rho=n} \Box\alpha^{\mathcal{L}}_{m,\mathbf{v_m}}\}| \end{aligned} \tag{2.59}$$

Learning Opportunities

Suppose we randomly choose one of the utterances in the corpus, an utterance u that realizes a given lexeme $\mathcal{L}$. As an example, let us take lexeme `look`. The utterance may be one of the sentences in (37):

(37)
a. Instinctively I looked at my watch.
b. He looked such an honest man.
c. All looks were centered on him.

In all utterances `look` is realized[89].
The lexeme might be realized as a noun (as in (37c)) or as a verb (37a and b). The question of concern is now: is (37a) for instance a learning opportunity for a given feature of `look`? The answer is yes and no. Yes, it is a learning opportunity for the feature that expresses verbal valency, for example. No, it is no learning opportunity for any nominal feature. This is simply because `look` is not realized as a noun in this utterance. In other words, whether an utterance u is a learning opportunity (LO) for a given lexeme, depends on the lexical feature in question. How does the system know whether an utterance is an LO? It knows this by virtue of the LTH which is part of its a-priori knowledge. The system knows for instance that before it can learn the value of the feature PARTICLE for a lexeme $\mathcal{L}$, it has to determine whether $\mathcal{L}$ is a verb. So the feature that encodes part-of-speech (CAT) constrains the applicability of the feature PARTICLE. In fact, more than one feature can be part of the constraint for a particular type which introduces a feature to

[89] We noted already that the system has to have some a-priori capabilities to detect this (e.g. morphology). In addition, all tokens shall belong to one and the same lexeme (lacking any sense differentiation).

be learned. According to the definition of lexical feature (def. 8) it is the realization context ψ that allows the inference of the type which introduces the feature. This means that ψ contains the feature value settings of the type constraining features. It follows that the number of LOs for a feature $\alpha = \langle \sigma, \psi, F \rangle$ is just the number of observed utterances in which ψ is realized.

Def. 38 (number of learning opportunities) *The number of learning opportunities for a given experience set E, a particular lexeme $\mathcal{L}$ and a lexical feature $\alpha = \langle \sigma, \psi, F \rangle$ is the number of references for α's realization context ψ, where ψx is a formula of the form $F_1 x\mathbf{v_1}$ & ... & $F_m x\mathbf{v_m}$.*

$$c_{LO}^{\rho=n}(E, \mathcal{L}, \alpha) \quad = \quad c_{ref}^{\rho=n}(E, \alpha_{1,\mathbf{v_1}}^{\mathcal{L}} ... \alpha_{m,\mathbf{v_m}}^{\mathcal{L}}) \tag{2.60}$$

Feature Value Distribution

Learning opportunity becomes an important notion because if the system relates the number of LOs to the number of retrieved references, it is able to estimate

a. whether the feature can be determined at all,

b. (if it can be determined) how efficient the learning procedure is and

c. the relative frequency of a given feature value.

If the system, for example, encounters 10000 LOs for `look` and generates no reference for [SC **to**] within, say, the intransitive frame, this could be due to different reasons:

a. the value **to** of feature SC is indeterminable

b. the realization of this value is very infrequent for `look` or at least it is quite difficult to detect

c. `look` simply does not allow for this subcategorization frame.

In order to make a reasonable judgment about option c), the system should gain some idea about a) and b), respectively. If the system *would* yield some references out of the LOs, option a) could be excluded. In this case it would be interesting to know how many references (c_{ref}) have been extracted out of the set of LOs. We will call this ratio *relative feature value frequency*:

Def. 39 (relative feature value frequency)
The relative feature value frequency of a lexical feature value $\alpha^{\mathcal{L}}_{\mathbf{v}}$ is the ratio of references to learning opportunities:

$$freq^{\rho=n}(E,\alpha^{\mathcal{L}}_{\mathbf{v}}) = \frac{c^{\rho=n}_{ref}(E,\alpha^{\mathcal{L}}_{\mathbf{v}})}{c^{\rho=n}_{LO}(E,\mathcal{L},\alpha)} \quad \textbf{if} \quad c^{\rho=n}_{LO}(E,\mathcal{L},\alpha) > 0 \quad \textbf{else} \quad 0 \quad (2.61)$$

In the ideal world the relative feature value frequency would be a perfect MLE for the real feature value realization probability $p(\alpha^{\mathcal{L}}_{\mathbf{v}})$. Due to lexical neutralization, however, the value for $freq^{\rho=n}(E,\alpha^{\mathcal{L}}_{\mathbf{v}})$ may be lower than the 'correct' MLE. There is always some 'loss' when we try to estimate $p(\alpha^{\mathcal{L}}_{\mathbf{v}})$. In the worst case of total neutralization, $freq^{\rho=n}(E,\alpha^{\mathcal{L}}_{\mathbf{v}})$ is zero, which means that (on ground of E) the determination of this feature value is impossible for the lexeme $\mathcal{L}$ - under the restriction of a given RDoA_{max}. Consequently, it is crucial to find out whether this is the case for all lexemes or for certain classes of lexemes, too.
This is achieved by summing up the reference counts (2.62) for a given feature value $\alpha_{\mathbf{v}}$ of all lexemes that a) belong to a particular class C and b) have at least one LO and dividing it by the corresponding number of LOs (given in 2.63). We call the result the *overall relative feature value frequency* $freq^{\rho=n}(C,E,\alpha_{\mathbf{v}})$ (2.64).

Def. 40 (overall relative feature value frequency)
The overall relative feature value frequency $freq^{\rho=n}(C,E,\alpha_{\mathbf{v}})$ is the average feature value frequency over all lexemes in a class C. Let $\overline{C} \subseteq C$ be the subset of all lexemes for which there is at least one learning opportunity.

$$c^{\rho=n}_{ref}(C,E,\alpha_{\mathbf{v}}) = \sum_{\mathcal{L}\in\overline{C}} c^{\rho=n}_{ref}(E,\alpha^{\mathcal{L}}_{\mathbf{v}}) \quad (2.62)$$

$$c^{\rho=n}_{LO}(C,E,\alpha) = \sum_{\mathcal{L}\in C} c^{\rho=n}_{LO}(E,\mathcal{L},\alpha) \quad (2.63)$$

$$freq^{\rho=n}(C,E,\alpha_{\mathbf{v}}) = \frac{c^{\rho=n}_{ref}(C,E,\alpha_{\mathbf{v}})}{c^{\rho=n}_{LO}(C,E,\alpha)} \quad \textbf{if} \quad c^{\rho=n}_{LO}(C,E,\alpha) > 0 \quad \textbf{else} \quad 0 \quad (2.64)$$

If the overall relative feature value frequency is 0 and the number of investigated lexemes reasonably high, then it is *very likely* that $\alpha^{\mathcal{L}}_{\mathbf{v}}$ is indeterminable for the lexeme class C. This would favor option a). We will call the set of (overall) relative feature value frequencies for a given feature: *(overall) feature value distribution*. For all the relative feature value frequencies the confidence intervals can be calculated along the lines of section A.3. We shall denote the lower bounds with $\underline{freq}$ and the upper bounds with $\overline{freq}$.

Neutralization Factor and Recoverability Factor

Because of systematic lexical neutralization the relative feature value frequency can be far lower than $p(\alpha_{\mathbf{v}}^{\mathcal{L}})$, which would be otherwise correctly MLE-estimated as, say, $\hat{p}(E, \alpha_{\mathbf{v}}^{\mathcal{L}})$. The relation of the two numbers can be expressed by a factor in the range $[0, 1]$: the *neutralization factor*, which may be considered a measure fore the *efficiency of learning* with regard to a particular feature value. This rather theoretical value is to be thought independent of the experience set. So we stipulate the following estimations for a sufficiently large experience set:

$$NF^{\rho=n}(\alpha_{\mathbf{v}}^{\mathcal{L}}) \quad \approx \quad \frac{freq^{\rho=n}(E, \alpha_{\mathbf{v}}^{\mathcal{L}})}{\hat{p}(E, \alpha_{\mathbf{v}}^{\mathcal{L}})} \tag{2.65}$$

It makes sense to approximate the factor of a whole class C of lexemes as well. In this case the MLE based estimation is an average over the lexemes in the class:

$$NF^{\rho=n}(C, \alpha_{\mathbf{v}}) \quad \approx \quad \frac{freq^{\rho=n}(C, E, \alpha_{\mathbf{v}})}{\hat{p}(C, E, \alpha_{\mathbf{v}})} \tag{2.66}$$

The lower the neutralization factor the more potential references 'slip away' due to neutralization. Unfortunately we cannot determine the individual efficiency of learning without knowing the 'real' frequency and vice versa. We would like to put this finding into a theorem:

Theorem 5 *The neutralization factor of a feature value $\alpha_{\mathbf{v}}^{\mathcal{L}}$ is indeterminable if the real frequency of realization of $\alpha_{\mathbf{v}}^{\mathcal{L}}$ is unknown. In turn, the real frequency of realization is indeterminable if the neutralization factor is unknown.*

Despite its indeterminability this theoretical notion is valuable because it allows one to express and to compare learning efficiency under a particular RDoA_{max}. It is desirable to enable the learning system to switch automatically to a more 'fuzzy' mode of $\text{RDoA}_{max} > 1$ if the factor $NF^{\rho=1}$ gets to low. Although the system looses its 'strength' in this case, it is desirable because this is the only chance to get a grip on these otherwise invisible feature values. This was the conclusion (3) we arrived at in subsection 2.5.4, which can be paraphrased to: There are features that only become visible if the sharpness of the system is reduced.
The situation, however, is not as bad as it appears at first sight. If we recall our claim (4) in subsection 2.5.6 that total neutralization usually happens on the way from RDoA=2 to RDoA=1, then RDoA_{max} can often be limited

to 2.
In order to take this finding properly into account, the system must determine whether a feature value is highly neutralized and a reduction in sharpness might be advisable. This is the case if the efficiency of learning under RDoA_{max}=1 is very low compared to the efficiency of learning under RDoA_{max}=2. We call this ratio the *recoverability factor* (RF) of a feature value. This value tells the system how much the learning efficiency can be increased when it switches from RDoA_{max}=1 to RDoA_{max}=2 or to put it differently: this measure tells us how much of the recoverability of a feature is lost when the system is restricted to unambiguous references only. In certain circumstances, however, the move from RDoA_{max}=1 to RDoA_{max}=2 is not sufficient because the feature value might stay indeterminable. In this case the system should switch to RDoA_{max}=∞. This motivates the definition of a first and a second recoverability factor ranging over a class C of lexemes and a feature value $\alpha_{\mathbf{v}}$:

Def. 41 (first recoverability factor)
The first recoverability factor of a feature value $\alpha_{\mathbf{v}}$ is the ratio of the two neutralization factors of $RDoA_{max}$=2 and $RDoA_{max}$=1, respectively:

$$RF_1(C,\alpha_{\mathbf{v}}) = \begin{cases} \frac{NF^{\rho=2}(C,\alpha_{\mathbf{v}})}{NF^{\rho=1}(C,\alpha_{\mathbf{v}})} & \textbf{if} \quad NF^{\rho=2}(C,\alpha_{\mathbf{v}}) > 0 \\ & \textbf{and} \quad NF^{\rho=1}(C,\alpha_{\mathbf{v}}) > 0 \\ 0 & \textbf{if} \quad NF^{\rho=2}(C,\alpha_{\mathbf{v}}) = 0 \\ \infty & \textbf{if} \quad NF^{\rho=2}(C,\alpha_{\mathbf{v}}) > 0 \\ & \textbf{and} \quad NF^{\rho=1}(C,\alpha_{\mathbf{v}}) = 0 \end{cases} \tag{2.67}$$

Def. 42 (second recoverability factor)
The second recoverability factor of a feature value $\alpha_{\mathbf{v}}$ is the ratio of the two neutralization factors of $RDoA_{max}$=∞ and $RDoA_{max}$=2, respectively:

$$RF_2(C,\alpha_{\mathbf{v}}) = \begin{cases} \frac{NF^{\rho=\infty}(C,\alpha_{\mathbf{v}})}{NF^{\rho=2}(C,\alpha_{\mathbf{v}})} & \textbf{if} \quad NF^{\rho=\infty}(C,\alpha_{\mathbf{v}}) > 0 \\ & \textbf{and} \quad NF^{\rho=2}(C,\alpha_{\mathbf{v}}) > 0 \\ 0 & \textbf{if} \quad NF^{\rho=\infty}(C,\alpha_{\mathbf{v}}) = 0 \\ \infty & \textbf{if} \quad NF^{\rho=\infty}(C,\alpha_{\mathbf{v}}) > 0 \\ & \textbf{and} \quad NF^{\rho=2}(C,\alpha_{\mathbf{v}}) = 0 \end{cases} \tag{2.68}$$

From the previous subsection it has become clear that we do not know the value of $NF^{\rho=n}$. The *ratios* can, nevertheless, be computed exploiting approximation 2.66:

$$RF_1(C, \alpha_{\mathbf{v}}) \approx \frac{NF^{\rho=2}(C, \alpha_{\mathbf{v}}) \cdot \hat{p}(C, E, \alpha_{\mathbf{v}})}{NF^{\rho=1}(C, \alpha_{\mathbf{v}}) \cdot \hat{p}(C, E, \alpha_{\mathbf{v}})} \approx \frac{freq^{\rho=2}(C, E, \alpha_{\mathbf{v}})}{freq^{\rho=1}(C, E, \alpha_{\mathbf{v}})} \tag{2.69}$$

This approximation holds if the denominators are greater than zero. The same holds for RF_2 accordingly. The Recoverability Factor signals the system whether a feature value is indeterminable (under $\text{RDoA}_{max} \in \{1, 2\}$):

Theorem 6 *The feature value $\alpha_{\mathbf{v}}$ is indeterminable under $RDoA_{max}=n$ for a class C of lexemes if and only if $freq^{\rho=n}(C, E, \alpha_{\mathbf{v}}) = 0$. It gets determinable under a higher $RDoA_{max}(1 \mapsto 2, 2 \mapsto \infty)$ if $RF_n(C, \alpha_{\mathbf{v}}) > 1$.*

2.8.3 Default Feature Values

It is possible and indeed the case in our implementation that a certain feature value will never be observed by virtue of how the grammar works. Take the boolean feature GRADABLE, for example, which might be used to denote gradability of adjectives and adverbs. Let us use [GRADABLE +] to admit for comparison of adjectives/adverbs and [GRADABLE −] to forbid it. It is easily imaginable that there are a couple of rules which set or check the plus value, but no rule which is sensitive to the minus value. Hence, [GRADABLE −] is never ever observed. It is indeterminable even under $\text{RDoA}_{max}=\infty$. On the other hand it would be a success for the system to induce [GRADABLE −] for adjectives/adverbs that never realize [GRADABLE +]. For those lexemes the only feature value observable is **bool**, the most general type of GRADABLE. So we propose to extend the LTH by a feature meta property: *default value* expressed by a function *Default* that maps feature names to types[90]:

(2.70)

i. $Intro(\text{GRADABLE}) = \mathbf{bool}$

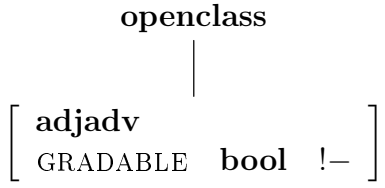
ii. $Default(\text{GRADABLE}) = -$

We tag this with an exclamation mark in the formulae and depictions, respectively as in figure 2.14.

The $Default$ function allows to specify the following induction rule:

[90] In our implementation, a boolean feature always has the minus value as its default.

Figure 2.14: Type Hierarchy with Default



Rule 1 Default Induction Rule
If $freq^{\rho=\infty}(E, \alpha^{\mathcal{L}}_{\mathbf{v}}) = 0$ *for a lexical feature* $\alpha = \langle \sigma, \psi, F \rangle$ *and all values* $v \in V(F) \setminus \{Intro(f)\}$ *although the number of learning opportunities for* α *and* $\mathcal{L}$ *is sufficiently high and if* $Default(f)$ *is defined then induce* $\alpha^{\mathcal{L}}_{Default(f)}$.

2.8.4 Sparse Data and Quantization Effects

Many lexemes tend to realize particular feature values with a very low relative frequency so that the realizations are rarely observed. This effect is usually referred to by the *sparse data* problem. It becomes a crucial question of how many references have to be observed so that statistical inference methods become reliable. We cannot give a general answer to this problem, but the study of *quantization effects* in Evert (2005) in the context of collocations provides a very valuable clue. This study starts with the following thought experiment: 'Imagine a population consisting of 500 high-frequency pair types [pair of co-occurring words], each one of which occurs once every two thousand tokens ($\pi = 5{\cdot}10^{-4}$), and 750000 low-frequency pair types, each one of which occurs once in a million pair tokens ($\pi = 10^{-6}$). [...] Assume further that all component types [the individual words of the pair] occur once in a thousand tokens ($\pi_1 = \pi_2 = 10^{-3}$), so that the null probability of any pair type under the independence hypothesis H_0 is $\pi_1 \cdot \pi_2 = 10^{-6}$. Thus, the low-frequency types are random combinations of their components (since they satisfy the null hypothesis $\pi = \pi_1\pi_2$), while the high-frequency types show strong positive association ($\pi \gg \pi_1\pi_2$).' (Evert 2005: 119)
In a sample of size $N = 2000$ taken from this hypothetical population most of the high-frequency pairs will show up exactly once or not at all. It can be shown (given certain simplifications that do not invalidate the argument) that '[...] even a single instance of a pair type in the sample is considered significant evidence for a positive association'. (Evert 2005: 120) The computed p-value based on an assumed Poisson distribution is ≈ 0.002 showing

a high degree of significance. High-frequency pairs cannot be distinguished from low-frequency pairs based on single observations. Evert (2005: 120) traces back this problem to three effects: '(i) the very large number of low-frequency pair types in the population, (ii) the different statistical properties of single events vs. classes of events, and (iii) the quantisation of frequency counts. [...] Although the occurrence probability is fairly small for each individual type, the large number of low-frequency types causes some of them to "leak through" into the sample.' As a result, 'Quantisation effects allow the influence of the shape of the population to become dominant for lowest-frequency data, especially the hapax legomena.' (ibid.)

We believe that the thought experiment carries over to the acquisition of lexical features. Suppose there is a lexical feature α with a set of appropriate values in $V(F)$ and a real feature value distribution that is skewed: the value $\mathbf{v_1}$ is realized by 90% of the time, the remaining, say, 20 values are realized with equal probabilities of 0.005. Together they share the residual probability mass of 10%. In a sample of 100 utterances, for example, there are approximately 90 realizations of $\mathbf{v_1}$ and 10 realizations of the remaining 20 values (one for each) that incidentally happened to occur in the sample. For them the MLEs for the relative frequencies double the real ones and the estimated frequency distribution is highly distorted. The situation will be different and much more reliable with a sample of 1000 utterances.

The study of various sophisticated population models in Evert (2005) showed that these models are not able to solve the issue and finally it is concluded that 'it is impossible to correct for quantisation errors unless better population models become available. [...] For the time being, however, we must assume that probability estimates and p-values for the lowest-frequency types are distorted in unpredictable ways. Fortunately, the influence of quantisation effects and the specific shape of the population is minimal for frequency classes m >= 5, so that statistical inference is accurate. Taken together, these conclusions provide theoretical support for frequency cutoff thresholds. Data with cooccurrence frequency f < 3, i.e. the hapax and dis legomena, should always be excluded from the statistical analysis. On the other hand, the shape of the population has little effect for f >= 5 and the data can safely be used.' (Evert 2005: 133)

For this reason we propose to stipulate a system parameter c_{ref}^{min}, called *minimum reference cutoff*, which specifies the minimum number of references that have to be observed for $\alpha_{\mathbf{v}}^{\mathcal{L}}$ before they enter the statistical inference procedures. Under this view the relative frequency $freq^{\rho=n}(E, \alpha_{\mathbf{v}}^{\mathcal{L}})$ will be set to 0 if there are less than c_{ref}^{min} references. It is reasonable to set c_{ref}^{min} in the range $[3, 4]$.

2.8.5 ANLE Algorithm

The estimation of the effective noise level is key to the probabilistic account of Learn−α. In section 2.4.7 the ANLE has been discussed but no algorithm was specified because of the lacking definitions of feature value characteristics. We now propose the following simple algorithm for the automatic noise level estimation of a given feature value $\alpha_{\mathbf{v}}$ presented as algorithm 1.
It is based on the heuristic and simplifying assumption that the overall relative feature value frequency of $\alpha_{\mathbf{v}}$ for a class of lexemes for which no systematic neutralization is happening *is* either an approximation of the average real realization frequency or of the effective noise level. In other words, the overall relative feature value frequency becomes the MLE of the real realization frequency or the effective noise level, respectively. Note that because each selected lexeme has at least one reference - otherwise neutralization could be assumed - no smoothing of the MLE is required.
Further, the approximated effective noise level and its confidence interval is computed for a given $\text{RDoA}_{max}{=}n$. In the case of $\text{RDoA}_{max}{=}1$ the estimated noise will by capped by ϵ_{max}. In addition, the noise level will be computed *relative* to a given lexeme $\mathcal{L}$, which means that all observed lexemes except $\mathcal{L}$ will be consulted for the estimation.
The algorithm computes a threshold $\hat{\epsilon}$ that excludes lexemes from being included in the estimation. The intuition is that lexemes with a relatively high relative feature value realization frequency may be those that *admit* the respective feature value. It is reasonable that $\hat{\epsilon}$ should not exceed the smallest lower bound observed relative feature value realization frequency f_{min} - at least those that are stable[91]. For a simple heuristic we choose $\hat{\epsilon} = \underline{freq}^{\rho=n}(C, E, \alpha_{\mathbf{v}})$ or $\hat{\epsilon} = f_{min}$, whichever is smaller.

2.8.6 The Statistical Model in the Non-ideal World

In section 2.3.5 it was outlined that for the statistical inference procedure (the decision function) a critical value C has to be specified so that the hypothesis $H : \square - \alpha_{\mathbf{v}}^{\mathcal{L}}$ is accepted when a) the actual observed number of references is zero and b) there is a sufficient number of learning opportunities. In the ideal, noise-free world (cf. the paragraph on page 119) C is zero. To account for noise, however, C will usually be greater than zero. Its lower bound is determined by the actual noise level and the maximum false positive error rate that will be admitted.

For the latter we now introduce the parameter β_{FP}, the counterpart of β_{FN}. By means of these two parameters the LE can control the system's

[91] We will motivate the distinction of stable versus non-stable beliefs further below.

Algorithm 1 Automatic Noise Level Estimation for a Feature Value $\alpha_{\mathbf{v}}$ and $\rho = n$

Require: Let E be the experience set, C be the set of all observed lexemes and $\mathcal{L}_\epsilon$ the lexeme relative to which the noise level is to be estimated.

$f_{min} \leftarrow \inf\{\mathcal{L} \in C : \underline{freq}^{\rho=n}(E, \alpha_{\mathbf{v}}^{\mathcal{L}})\}$

$\hat{\epsilon} \leftarrow \inf\{\underline{freq}^{\rho=n}(C, E, \alpha_{\mathbf{v}}), f_{min}\}$

$C^* \leftarrow \{ \quad \mathcal{L} \in C \quad | \mathcal{L} \neq \mathcal{L}_\epsilon \quad \& \quad freq^{\rho=n}(E, \alpha_{\mathbf{v}}^{\mathcal{L}}) \leq \hat{\epsilon} \quad \}$

if $n = 1$ **and** $freq^{\rho=n}(C^*, E, \alpha_{\mathbf{v}}) > \epsilon_{max}$ **then**

$\hat{\epsilon}_{eff}^{\rho=n}(\alpha_{\mathbf{v}}) \leftarrow \epsilon_{max}$

else

$\hat{\epsilon}_{eff}^{\rho=n}(\alpha_{\mathbf{v}}) \leftarrow freq^{\rho=n}(C^*, E, \alpha_{\mathbf{v}})$

end if

precision (β_{FP}) and recall (β_{FN}).
For the calculation of the minimum number of learning opportunities and the critical value in the non-ideal, noisy world a statistical model has to be developed which accounts for the maximally admitted error rates, the noise levels and the smallest feature value realization probability p_{min}^{fv} which determines the system's sensitivity. By virtue of this model two inequations can be formulated which reflect the constraints posed by the specification of the two maximal error rates.
From the considerations of the paragraph about the observed relative feature value realization frequencies (page 183) it has become clear that there is no direct access to the real realization probability $p(\alpha_{\mathbf{v}}^{\mathcal{L}})$. Let us therefore first assume that there is a probability $\theta^{\rho=n}$ which specifies the binomial distribution of intended (noise-free) references under $\text{RDoA}_{max}{=}n$. $\theta^{\rho=n}$ is also unknown, but is used in order to 'free' $p(\alpha_{\mathbf{v}}^{\mathcal{L}})$ which has a different meaning. Because the set of observed references may comprise intended as well as unintended ones, the distribution that accounts for its size is a complex distribution of two interfering binomial distributions: $X \sim f(x|\theta^{\rho=n}, \epsilon_{eff}^{\rho=n})$.
For the calculation of C let us assume that there is a sufficient number N of learning opportunities. C must then be such that the following two inequa-

tions become true[92]:

$$P_H[x > C] = \sum_{i=C+1}^{N} \binom{N}{i} (\epsilon_{eff}^{\rho=n})^i (1 - \epsilon_{eff}^{\rho=n})^{N-i} = 1 - F_{\mathbf{B}(\epsilon_{eff}^{\rho=n}, N)}(C) \leq \beta_{FP} \tag{2.71}$$

$$P_K[x \leq C] = \sum_{i=0}^{C} \binom{N}{i} (\theta^{\rho=n})^i (1 - \theta^{\rho=n})^{N-i} = F_{\mathbf{B}(\theta^{\rho=n}, N)}(C) \leq \beta_{FN} \tag{2.72}$$

As neither $\epsilon_{eff}^{\rho=n}$ nor $\theta^{\rho=n}$ are known, these parameters have to be fixed by virtue of reasonable assumptions. We first assume that $\theta^{\rho=n} \geq \epsilon_{eff}^{\rho=n} \cdot SNR$ along the lines discussed in subsection 2.4.6. Secondly, θ must not fall below p_{min}^{fv} in accordance with the considerations made in subsection 2.3.5, the latter being the worst-case assumption for the former. This system parameter was intended to relate the observed references to the number of utterances in which a lexeme $\mathcal{L}$ itself is realized. Put differently, the sample size was the number of utterances. This picture changes now given that the observed references are compared to the number of LOs, which becomes the sample size in that sense. Because LOs depend on their realization contexts they can be relatively spare. It would be unrealistic to assume that p_{min}^{fv} equally applies to that relation because the minimum probability related to the realization frequency of $\mathcal{L}$ could dramatically and unrealistically decrease down the LTH. To see how, consider the following example: let $p_{min}^{fv} = 0.001$ and the relative feature value frequency of the realization context ψ 0.01 which means that on average in 100 utterances there is one which 'has' the realization context. If the feature value $\alpha_{\mathbf{v}}$ depending on ψ was realized with no more than p_{min}^{fv} per realization context, this would mean that the lexeme would realize $\alpha_{\mathbf{v}}$ once in 100000 utterances $(= 1/(0.01 \cdot 0.001))$.
In order to keep the original semantics of the parameter p_{min}^{fv} intact, we propose to adjust the parameter per realization context. The adjusted parameter is symbolized by $p_{\psi}^{\rho=n}$. It also accounts for the different observations made on different RDoA_{max}-levels. In the example $p_{\psi}^{\rho=n}$ would be 0.1 which means that the sensitivity of the system related to the realization context ψ

[92] P_H is the probability of observing references in a world in which $H : \Box - \alpha_{\mathbf{v}}^{\mathcal{L}}$ is true. P_K is the probability of observing references in a world in which $K : \Box\alpha_{\mathbf{v}}^{\mathcal{L}}$ is true.

is such that it does not recognize a feature value realization that is below one in ten cases. On the other hand, it makes sense to define an upper bound of $p_{\psi}^{\rho=n}$, otherwise feature acquisition in some realizations contexts could become meaningless. We propose to let p_{med}^{fv}, introduced in subsection 2.4.7, take the role of the upper bound. The parameter $p_{\psi}^{\rho=n}$ is thus computed as follows. Let $E^{\mathcal{L}}$ be the set of utterances in which $\mathcal{L}$ is realized, then:

$$p_{\psi}^{\rho=n}(\alpha) = \inf\{p_{min}^{fv} \cdot \frac{|E^{\mathcal{L}}|}{c_{LO}^{\rho=n}(E, \mathcal{L}, \alpha)}, p_{med}^{fv}\} \tag{2.73}$$

The parameter $\epsilon_{eff}^{\rho=n}$ will be estimated according to the algorithm explained in subsection 2.8.5. As with $p_{\psi}^{\rho=n}$ the estimated parameter $\hat{\epsilon}_{eff}^{\rho=n}$ also accounts for the various RDoA_{max}-levels. With the help of these parameters the two inequations can be solved (if not analytically then at least by computation). The only problem remaining is the situation in which the ratio of the two parameters is below SNR. In this case the 'leading' parameter is $\hat{\epsilon}_{eff}^{\rho=n}$ and the worst case stipulation for $\theta^{\rho=n}$ will be increased accordingly.

$$\theta^{\rho=n} = \sup\{\hat{\epsilon}_{eff}^{\rho=n} \cdot SNR, p_{\psi}^{\rho=n}\} \tag{2.74}$$

The discussion so far has not yet accounted for the parameter c_{ref}^{min} introduced in subsection 2.8.4. We include it by demanding that $C \geq c_{ref}^{min}$.
With these assumptions we arrive at the final formulation of constraints regarding the critical value. The number of learning opportunities N is sufficiently high if there is a C such that the three inequations become true:

Theorem 7 *Let $\theta^{\rho=n}$ and $\hat{\epsilon}_{eff}^{\rho=n}$ be the parameters that specify the hypothesis test for the induction of $\alpha_{\mathbf{v}}^{\mathcal{L}}$ on the experience set E. Let $N = c_{LO}^{\rho=n}(E, \mathcal{L}, \alpha)$ be the number of learning opportunities. The critical region R for that test is the range $[C+1, \infty]$ such that:*

$$C \geq c_{ref}^{min} \tag{2.75}$$

$$1 - F_{\mathbf{B}(\hat{\epsilon}_{eff}^{\rho=n}, N)}(C) \leq \beta_{FP} \tag{2.76}$$

$$F_{\mathbf{B}(\theta^{\rho=n}, N)}(C) \leq \beta_{FN} \tag{2.77}$$

If N is too small it may be that no critical value C can solve the inequations. In this case R is not defined.

In order to get a feel for the number of LOs required depending on the various parameters, we compiled the following table[93] with $SNR = 5$:

[93]Note that the table suggests that $\beta_{FN} = \beta_{FP}$, which is not necessarily the case.

β_{FN}:	0.1	0.05	0.01	0.001
β_{FP}:	0.1	0.05	0.01	0.001
$\hat{\epsilon}_{eff}^{\rho=n} = 0.001$	1335	2100	3476	6247
$\hat{\epsilon}_{eff}^{\rho=n} = 0.005$	266	418	692	1242
$\hat{\epsilon}_{eff}^{\rho=n} = 0.010$	132	208	344	617

What if there is no Noise?

There is a special case to be mentioned in which there is no measurable noise ($\hat{\epsilon}_{eff}^{\rho=n}$=0). In this case the critical value simply amounts to $C = c_{ref}^{min} - 1$ and the hypothesis test is similar to the ideal world scenario.

2.8.7 Decisions and Strength of Belief

The induction of lexical features is a process that starts with the observation of the lexeme's behavior captured by the generalized observation schema on top of an experience set E and which ends with decisions. Decisions are threefold. Either the system decides to believe that either a) $\alpha_{\mathbf{v}}^{\mathcal{L}}$ or b) $-\alpha_{\mathbf{v}}^{\mathcal{L}}$ is optimal (entailed by the optimal model) or it concludes that c) no decision can be made because neither a) nor b) is sufficiently supported by E. The choice between a) and b) is grounded in a hypothesis test, whereas c) is opted for if the minimum number of learning opportunities is not included in E. The decision step will be captured by the next theorem:

Theorem 8 *Let $\theta^{\rho=n}$ and $\hat{\epsilon}_{eff}^{\rho=n}$ be the parameters that specify the hypothesis test for the induction of $\alpha_{\mathbf{v}}^{\mathcal{L}}$ on the experience set E under the system's focus n. Let N be the minimum number of learning opportunities for $\alpha_{\mathbf{v}}$ and C the critical value of the test. Let δ be the decision function that maps E to a belief state of the system. Then the decision step is characterized as:*

$$\delta^{\rho=n}(E) = \begin{cases} \Box\alpha_{\mathbf{v}}^{\mathcal{L}} & \textbf{if} \quad c_{LO}^{\rho=n}(E,\mathcal{L},\alpha) \geq N \quad \textbf{and} \quad c_{ref}^{\rho=n}(E,\alpha_{\mathbf{v}}^{\mathcal{L}}) > C \\ \Box - \alpha_{\mathbf{v}}^{\mathcal{L}} & \textbf{if} \quad c_{LO}^{\rho=n}(E,\mathcal{L},\alpha) \geq N \quad \textbf{and} \quad c_{ref}^{\rho=n}(E,\alpha_{\mathbf{v}}^{\mathcal{L}}) \leq C \\ \textit{undecided} & \textbf{if} \quad c_{LO}^{\rho=n}(E,\mathcal{L},\alpha) < N \end{cases} \tag{2.78}$$

Note that Learn−α is not restricted to binomial hypothesis testing: other statistical inference methods such as LRT (cf. section 2.3.5) or pointwise MI can be plugged in as well, provided that the critical region can be calculated.

The *strength of belief* related to the system's decision can be expressed using the p-value taken from the Fisherian view of hypothesis testing (cf. section 2.3.5). The p-value expresses how odd the experience of the system is in a world in which the null hypothesis H_0 holds. Note that the null hypothesis

is not restricted to hypothesis H only (although one often finds this equation in the literature). There is no theoretical reason that hinders us to state hypothesis K as the null hypothesis, which allows us to express the oddity of the experience in the light of a world in which K holds - thereby sustaining the decision to believe in H. Let therefore $\pi(B_s p)$ be the p-value expressing the oddity to believe in the statement p. The following theorem states the strength of belief as a function f_π that trends towards zero when the strength increases. From this view a value of $f_\pi = 0$ would express total conviction.

Theorem 9 *Let $B_s \,\square\, \alpha^{\mathcal{L}}_{\mathbf{v}}$ ($B_s \,\square\, -\alpha^{\mathcal{L}}_{\mathbf{v}}$) express that the system believes (does not believe) in the optimality of the feature value admission $\alpha^{\mathcal{L}}_{\mathbf{v}}$. Then the function $f_\pi(\alpha^{\mathcal{L}}_{\mathbf{v}}, x) \in [0,1]$ expresses the strength of belief in $\alpha^{\mathcal{L}}_{\mathbf{v}}$ ($-\alpha^{\mathcal{L}}_{\mathbf{v}}$) on the grounds of x observed references.*
The stronger the belief the smaller the value of the function. The function is determined by the p-value of the corresponding alternative hypothesis:

$$f_\pi(\alpha^{\mathcal{L}}_{\mathbf{v}}, x) = \pi(B_s \,\square\, \alpha^{\mathcal{L}}_{\mathbf{v}}) = \begin{cases} 1 - F_{\mathbf{B}(\hat{\epsilon}^{\rho=n}_{eff}, N)}(x-1) & \textbf{if} \quad x > 1 \\ 1 & \textbf{else} \end{cases} \tag{2.79}$$

$$f_\pi(-\alpha^{\mathcal{L}}_{\mathbf{v}}, x) = \pi(B_s \,\square\, -\alpha^{\mathcal{L}}_{\mathbf{v}}) = F_{\mathbf{B}(p^{\rho=n}_{\psi}, N)}(x) \tag{2.80}$$

Stability of Decisions

The strength-of-belief function allows the system to assess how *stable* a decision (or belief) is. This is motivated by the intuition that in the course of a potential revision process the system should keep to the decisions that are based on very clear evidence while it may give up beliefs that do not have a too high degree of empirical support. While the strength of belief is a continuum ranging from 0 to 1, the stability of belief shall comprise discrete categories. We propose the following categorization: i) *confirmed*, ii) *stable*, iii) *likely* and iv) *instable*. The first is the most stable category of decisions. It can be used by the LE to flag a system's decision as fixed and therefore irrevisable. Logically, it gains a similar status to all the entailments given a-priori. The second category shall be reserved for those (unconfirmed) decisions that have empirical support beyond reasonable doubt. A revision of such a decision should thus be disallowed, too. Decisions of the third and forth category however are candidates for revision if necessary. The forth category is reserved for the most 'shaky' decisions.
It is up to the implementation of Learn$-\alpha$ how to cast the strength of belief into categories of stability. Here is an example:

$$\text{believe } B_s \,\square\, \alpha_{\mathbf{v}}^{\mathcal{L}} \text{ is } \begin{cases} \text{stable} & \textbf{if} \quad f_\pi(\alpha_{\mathbf{v}}^{\mathcal{L}}, x) < 0.0001 \\ \text{likely} & \textbf{if} \quad 0.0001 \leq f_\pi(\alpha_{\mathbf{v}}^{\mathcal{L}}, x) < 0.001 \\ \text{instable} & \textbf{if} \quad f_\pi(\alpha_{\mathbf{v}}^{\mathcal{L}}, x) \geq 0.001 \end{cases} \tag{2.81}$$

The categorization of stability of decisions/believes can be modeled by the corresponding partitioning of the lexicon Λ into disjunct subsets: $\Lambda = \Lambda_{confirmed} \cup \Lambda_{stable} \cup \Lambda_{likely} \cup \Lambda_{instable}$. We will resume the discussion in section 2.9 which discusses the revision of belief.

2.8.8 Cascading Focus

The induction of realization contexts and lexical features takes place down the LTH. We call the level of RDoA_{max} under which a hypothesis test is performed the *system's focus* of the test. The three possible foci are $1, 2$ and ∞. For the sake of precision the system will try to perform the test with highest focus ($\rho = 1$). If the observation suggests that systematic neutralization happens so that the hypothesis H cannot be rejected on this level then it gives it another try by switching to $\rho = 2$. Again, if there is still evidence for systematic neutralization then the test is performed with the lowest focus $\rho = \infty$. We call this procedure *cascading focus*.
The algorithm sketched as algorithm 2 employs the feature value characteristics *overall relative feature value frequency*, *relative feature value frequency* and *recoverability factor* as well as theorem 6 (page 187) and theorem 8. Further, if the observed relative realization frequency falls below the noise barrier on all RDoA_{max}-levels the system will accept H if there is a sufficient number of learning opportunities as per the equation developed in the ideal world scenario (eq. 2.35 on page 119).

2.8.9 Example

The decision process shall be illustrated by the following example. Similar to the example of the lexeme `communicate` in 2.3.5 which discussed the strength of hypothesis that the lexeme opts for the subcategorization frame SCF104 (= intransitive + *that* complement), here the same hypothesis is considered based on the exemplary LTH given in figure 2.12 on page 179, slightly adapted to our needs. Let the conditions for the example be: SNR=5, $\beta_{FP} = 0.05$, $\beta_{FN} = 0.01$, $c_{ref}^{min} = 3$, $p_{min}^{fv} = 0.005$, and $\hat{\epsilon}_{eff}^{\rho=n} = 0.0015$ for all feature values and all n (for the sake of simplicity). Figure 2.15 displays the observed references for the relevant realization contexts. In the experience set there are 10000 utterances realizing the lexeme. 9000 of them

Algorithm 2 Cascading Focus

Require: Let E be the experience set and $\alpha_{\mathbf{v}}$ the feature value to be tested for lexeme $\mathcal{L}$.

if $freq^{\rho=1}(E, \alpha^{\mathcal{L}}_{\mathbf{v}}) > \hat{\epsilon}^{\rho=1}_{eff}$ **and** $\delta^{\rho=1}(E) = \Box\alpha^{\mathcal{L}}_{\mathbf{v}}$ **then**

decide for $\Box\alpha^{\mathcal{L}}_{\mathbf{v}}$

else
if $RF_1 > 1$ **and** $freq^{\rho=2}(E, \alpha^{\mathcal{L}}_{\mathbf{v}}) > \hat{\epsilon}^{\rho=2}_{eff}$ **and** $\delta^{\rho=2}(E) \neq \Box - \alpha^{\mathcal{L}}_{\mathbf{v}}$ **then**

decide for $\delta^{\rho=2}(E)$

else
if $RF_2 > 1$ **and** $freq^{\rho=\infty}(E, \alpha^{\mathcal{L}}_{\mathbf{v}}) > \hat{\epsilon}^{\rho=\infty}_{eff}$ **then**

decide for $\delta^{\rho=\infty}(E)$

else

$N \leftarrow \frac{log(\beta_{FN})}{log(1-p^{\rho=\infty}_{\psi})}$

if $c^{\rho=\infty}_{LO}(E, \mathcal{L}, \alpha) \geq N$ **then**

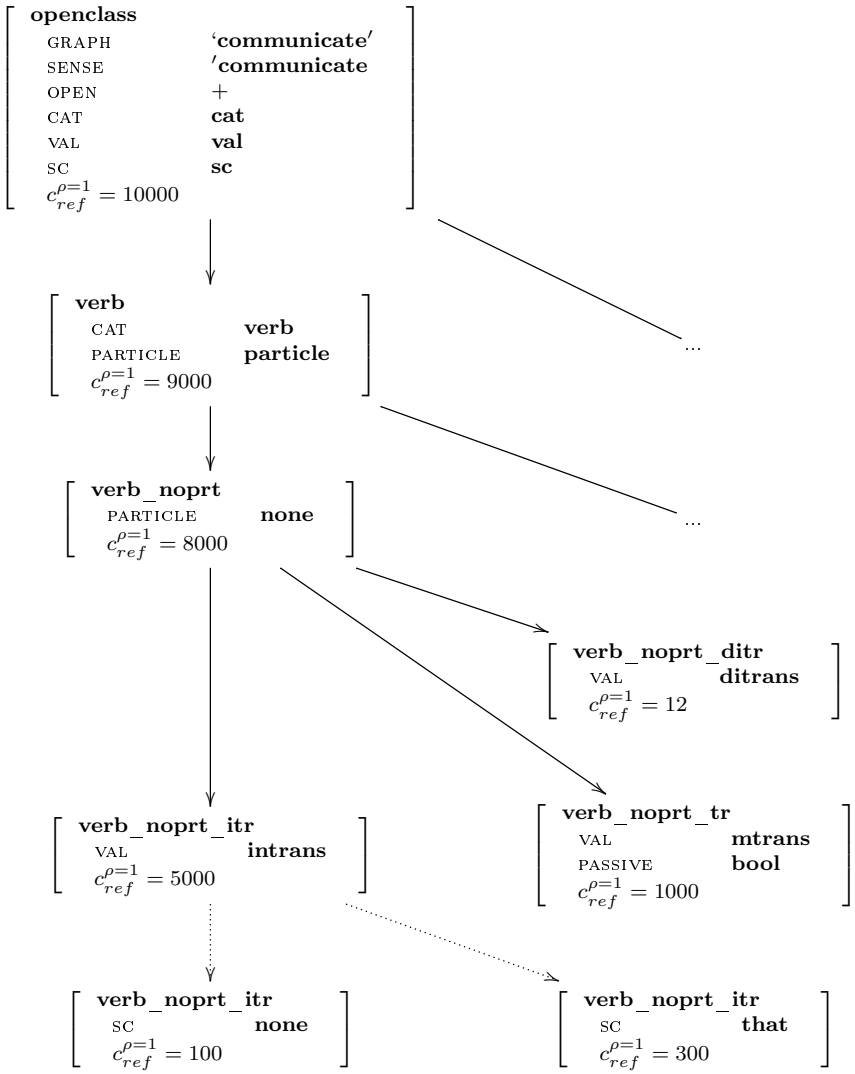
decide for $\Box - \alpha^{\mathcal{L}}_{\mathbf{v}}$

else
stay undecided
end if
end if
end if
end if

realize the lexeme unambiguously ($\rho = 1$) as a verb (with / without particle). Of those, 8000 unambiguously prove the realization of the lexeme as a verb without particle. Within this set, there are 5000 in an intransitive, 1000 in a transitive and 12 in a ditransitive frame. The remaining cases may be undecided under $\rho = 1$. Before we come to the hypothesis concerning SCF104, let us avail ourself of the opportunity to evaluate the strength of the hypotheses that the lexeme can be used a) intransitively, b) transitively and c) ditransitively. For all three, 8000 utterances make up the set of learning opportunities[94]: $c_{LO}^{\rho=1}(E, \texttt{communicate}, \langle \textbf{verb\_noprt}, ..., \text{VAL}\rangle) = 8000$. The adjusted minimal feature value realization probability for that context is $p_\psi^{\rho=1} = 0.005 \cdot 10000/8000 = 0.00625$. For this realization context, the hypotheses are specified by $\theta^{\rho=1} = \sup\{0.0015 \cdot 5, 0.00625\} = 0.0075$, so the test is fully determined by the estimated noise level. The minimally required number of learning opportunities for the test is 1939, so there are in effect enough data to perform the test. The critical value for the tests amounts to $C = 18$ as $1 - F_{\mathbf{B}(0.0015,8000)}(18) = 0.037 \leq \beta_{FP}$ and $F_{\mathbf{B}(0.0075,8000)}(18) \approx 1.7 \cdot 10^{-10} \leq \beta_{FN}$. For both a) and b) the number of references are far beyond the critical value so that hypothesis H will be rejected. For case c) the sample is within the acceptance region which indicates that the lexeme does not opt for the ditransitive frame and that all 12 references are due to noise only. The p-value shows that the feature value constraint is quite stable: $f_\pi(-\alpha_{\mathbf{v}}^{\mathcal{L}}) = F_{\mathbf{B}(0.00625,8000)}(12) \approx 1.2 \cdot 10^{-10}$.
Coming to SCF104, its test is based on the observation that there are 300 references out of 5000 learning opportunities. The adjusted minimal feature value realization probability for the realization context is $p_\psi^{\rho=n} = 0.005 \cdot 10000/5000 = 0.01$ so that the hypothesis test is bound to $p_\psi^{\rho=n}$ this time: $\theta^{\rho=1} = \sup\{0.0015 \cdot 5, 0.01\} = 0.01$. The minimum number of required LOs is 1157. The rejection region is $[13, \infty]$ and for the feature SC both values **none** and **that** (=SCF104) undoubtedly have to be specified in the lexical entry of the lexeme. The strength of belief for [SC **that**] is $f_\pi(\alpha_{\mathbf{v}}^{\mathcal{L}}) = 1 - F_{\mathbf{B}(0.0015,5000)}(299) \approx 0$, thus beyond any doubt.

[94] We will simplify the notation omitting the full specification of the realization context.

Figure 2.15: Example: Induction of SCF104 for lexeme `communicate`



2.9 Revision of Belief

In section 1.4.3 it was claimed that the ability of revision is one of the necessary conditions for being an intelligent system. The present section is dedicated to the logic of revisability in Learn$-\alpha$. In this context, revision happens if a statement ϕ is added to a current theory Θ with which ϕ is inconsistent under the requirement that the revised theory Θ' be consistent. In our theory there are two scenarios for revision. The first scenario is based on the ideal, noise-free world assumption in which every utterance u in the experience set E as well as in the challenge set C is grammatical. Even in this nicely behaving world the system can make 'mistakes' that deserve revision in the light of new information (which is part of the challenge set) - simply because the induction step happened 'too early'. For a proper account, we have to make a crucial distinction with respect to what can be a 'mistaken' decision when the system has fully determined a lexical feature in terms of definition 10 (page 92): a) the determination is a *constraint* with regard to a value ($-\alpha^{\mathcal{L}}_{\mathbf{v}}$) that should be admitted or b) it is an *admission* of a value ($\alpha^{\mathcal{L}}_{\mathbf{v}}$) that should rather be constrained.
In the first case it can happen that the constraint is not consistent with an utterance $u \in C$ so that parsing fails (no structural description can be assigned to u). Let us call this situation scenario Ia. In the second case (scenario Ib) there is no such obvious effect. What happens is that there will be u-models assigned to an utterance that are not intended. Under the assumption that u is grammatical, scenario Ia requires some sort of *immediate repair*. After parser failure the system will try to analyze the utterance with a revised theory. Because the feature system and grammar of the language model are fixed, revision can refer to the current lexicon Λ only. One option would be to drop all acquired lexical constraints and try to parse u again. If the analysis is possible after this amendment this would clearly prove that the parser failure in the first run was due to a mistaken lexical constraint. However, this method appears to be to drastic. It seems more plausible to give up the constraints in $\Lambda_{instable}$ first, provided that it contains any constraints on lexemes realized in u. Then the system shall repeat the analysis. If it succeeds this time then the by-product of the analysis is a revised sub-lexicon $\Lambda'_{instable}$. If it does not then the revision process should target Λ_{likely}, eventually resulting in two revised sub-lexicons $\Lambda'_{instable}$ and Λ'_{likely}. The question remains whether in the case of persistent failure the sub-lexicon Λ_{stable} should come into the focus of revision. We leave this question to the particular implementation of Learn$-\alpha$, remarking that in the light of the real, noisy world, it might be wiser to drop the assumption that u is grammatical and to regard Λ_{stable} as sacrosanct along with $\Lambda_{confirmed}$.

This leads to scenario II embedded in the real world which is afflicted with noise in E, C, Θ as much as with systematic lexical neutralization. In this setting the analysis of a single utterance will not decide on the actual revision of Θ, because the analysis cannot be blindly trusted. As described in section 2.8.7 decisions are made on ground of statistics and parameters that bridle the resulting errors. In this context revision is only justified if the actual decisions cease to satisfy the parameters when a part of C has been processed, so that the experience set E is replaced by E' - an extended experience set which contains more learning opportunities and thus a better foundation for even more precise decisions.

Definition 20 (page 101) introduces the doxastic operator B_s into the model system of a language model. It allows for the expression of belief states in which a system might actually be. A logical account of believe *revision* requires the move from such a static doxastic system to a dynamic one in which theories can undergo change operations. This gets its start in the seminal article of Alchourrón et al. (1985) to which most of the contemporary belief revision theories trace back (whether they agree with the conclusions or not). The theory set forth there is now consolidated under the title *AGM theory*, we therefore use the notion 'AGM operators' for the theory changing operators introduced in the AGM theory, which we now wish to introduce into Learn$-\alpha$ along with the postulates accompanying them, slightly adapted to our notation used so far.
Let T be a theory closed under logical consequence, which means that $Cn(T) = T$ if Cn is interpreted as the Tarskian consequence operator. Then $T' = T+\phi$ is an *expansion* of T if and only if $T' = Cn(T \cup \{\phi\})$. Expansion counts as the most unproblematic operation in belief revision, because nothing more special is happening than merely adding the statement ϕ to a theory with which the statement is 'hopefully consistent' (Alchourrón et al. 1985: 510). In Learn$-\alpha$ a stronger version is required: the initial theory Θ_0 is consistent (this is granted by the definition of a symbolic language model) and every expansion is so, too. Moreover, this is a requirement for all revision operations on Θ_0.
The removal of a statement ϕ from a theory T needs much more consideration. The removal process is called *contraction* and is denoted by the AGM operator $\dot{-}$, so that $T' = T\dot{-}\phi$ is a theory in which the statement ϕ is removed from T. The process of *revision*, denoted by $\dot{+}$, can be modeled by virtue of the so-called *Levi-identity*: $T\dot{+}\phi =_{df} Cn((T\dot{-} - \phi) + \phi)$. Hence revision is contraction followed by expansion. What makes contraction (and therefore revision) non-trivial is the circumstance that a statement ϕ may

be entailed by a couple of other statements so that these (or parts of them) have to be dropped in a way that T' ceases to entail ϕ. It is obvious that in general there might be more than one theory as a possible result of the contraction. In AGM it is assumed that there is a selection function γ that 'picks out a class of the "most important" maximal subsets' of T that cease to imply ϕ (Alchourrón et al. 1985: 511). Such a function is called *partial meet contraction*. It satisfies the so-called *basic Gärdenfors' postulates* (or basic AGM postulates):

($\dot{-}$ 1)	$T\dot{-}\phi$ is a theory whenever T is a theory	(closure)
($\dot{-}$ 2)	$T\dot{-}\phi \subseteq T$	(inclusion)
($\dot{-}$ 3)	If $\phi \notin Cn(T)$, then $T\dot{-}\phi = T$	(vacuity)
($\dot{-}$ 4)	If $\phi \notin Cn(\emptyset)$, then $\phi \notin Cn(T\dot{-}\phi)$	(success)
($\dot{-}$ 5)	If $Cn(\phi_1) = Cn(\phi_2)$, then $T\dot{-}\phi_1 = T\dot{-}\phi_2$	(preservation)
($\dot{-}$ 6)	$T \subseteq Cn((T\dot{-}\phi) \cup \{\phi\})$ whenever T is a theory	(recovery)

(2.82)

The postulates translate to Learn$-\alpha$ in the following way. Contraction in this context means the removal of the belief that a lexical feature admission/constraint is optimal: $\Theta' = \Theta\dot{-} \Box\, \alpha^{\mathcal{L}}_{\mathbf{v}}$ or $\Theta' = \Theta\dot{-} \Box\, -\alpha^{\mathcal{L}}_{\mathbf{v}}$, respectively. Postulate 1 states that Θ' is still a theory (which is closed under logical consequence) if Θ is a theory. This is obvious as either even the zero-lexicon model $\langle \mathbf{FS}, \Gamma, \emptyset \rangle$ counts as a theory (and nothing more can be contracted from it) or none of the expansions of it is a theory. Postulate 2 says that the set of statements in Θ' is a subset of Θ which is obvious, too. According to postulate 3 a removal of the belief from a theory that does not entail it is a vacuous operation. As per postulate 4 the selection function γ ensures that removals of non-tautologies are always successful: Θ' does not entail the belief any more. Postulate 5 may be interpreted as the circumstance of having two beliefs that have exactly the same consequences in the application of the language model, in which case the contraction of Θ by the one belief equals the contraction of Θ by the other. Postulate 6 is the most debated postulate of contraction (Fermé and Hansson 2001). The problem is that in the course of contraction of a belief B some beliefs depending on B may get lost that cannot be restored after adding B again. Suppose, for example that the system believes that it is optimal that lexeme $\mathcal{L}$ is a count noun ([COUNT +]). Removing the belief that $\mathcal{L}$ is a noun automatically removes the belief that [COUNT +]. This belief is not automatically restored when the belief that $\mathcal{L}$ is a noun is added again.

As mentioned in scenario Ia above belief revision must take into account

that there are some beliefs that are easier to give up than others. This can be modeled by employing a binary relation, called *epistemic entrenchment* (Gärdenfors and Makinson 1988) that controls contraction. It is said that $B_1 > B_2$ if belief B_1 is more entrenched (more stable) than B_2. This correlates directly with the *strength of belief* in the statistical inference part of Learn$-\alpha$ (section 2.8.7) as well as the definition of *realization context*. The relation can be employed in the definition of *entrenchment based contraction*:

$$\phi_1 \in T\dot{-}\phi_2 \quad \text{iff} \quad \phi_1 \in T \quad \text{and either} \quad \phi_2 \in Cn(\emptyset) \quad \text{or} \quad \phi_2 < (\phi_1 \vee \phi_2) \tag{2.83}$$

According to Gärdenfors and Makinson (1988) epistemic entrenchment satisfies 5 postulates, among which we want to emphasize the second, called *dominance*: if $\phi_1 \vdash \phi_2$ then $\phi_1 \leq \phi_2$. The justification for this postulate is that if ϕ_1 entails ϕ_2 and one of the two have to be retracted, then it is a 'smaller change' (Gärdenfors and Makinson 1988: 89) to give up ϕ_1 and retain ϕ_2 than vice versa. In the context of the exemplary count noun above this translates to the circumstance that being a count noun entails being a noun. Hence it is a smaller change to give up the claim that $\mathcal{L}$ is a count noun than that it is a noun. In other words being a noun is more entrenched than being a count noun. Consequentially feature admissions/constraints are given up easier than their related realization contexts. From that perspective the LTH stamps itself on the entrenchment relation.

Theorem 10 *If $\alpha = \langle \sigma, \psi, F \rangle$ is a lexical feature and $\alpha^{\mathcal{L}}_{\mathbf{v}}$ then:*

$$((\exists x)\ \mathcal{L}x\ \&\ \psi x) \geq \alpha^{\mathcal{L}}_{\mathbf{v}} \tag{2.84}$$

The forth AGM postulate (success) has been attacked by Fermé and Hansson (2001) because 'actual doxastic agents are known to have beliefs (of a non-logical nature) that nothing can bring them to give up' (page 85). They argue for an operation of contraction that treats certain not necessarily tautological beliefs as irretractible: *shielded AGM contraction*. Let $\mathcal{R}$ be the set of retractable statements then shielded contraction is defined as $T \ominus \phi = T\dot{-}\phi$ if $\phi \in \mathcal{R}$, $T \ominus \phi = T$ otherwise. The revision operation has to be adapted accordingly. Shielded contraction perfectly supports the distinction of $\Lambda_{confirmed/stable}$ versus $\Lambda_{likely/instable}$ if it is postulated that nothing from the confirmed or stable lexicons can be retracted:

Theorem 11 *Let $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ be the current consistent symbolic language model and Λ be partitioned into the disjunct sub-lexicons $\Lambda = \Lambda_{confirmed} \cup \Lambda_{stable} \cup \Lambda_{likely} \cup \Lambda_{instable}$. Let the function $f_\pi(\phi) \in [0, 1]$ express the strength*

of belief in ϕ. Let $\Theta \dot{-} \phi$ denote shielded contraction with the set of retractable statements in $\mathcal{R}$ and $\geq$ (including $>$) its entrenchment relation. Then the following postulates hold:

$$\begin{array}{ll} \#1 & \textit{if } \Lambda_{confirmed} \cup \Lambda_{stable} \vdash \phi \textit{ then } \phi \notin \mathcal{R} \\ \#2 & \textit{if not } \phi_2 > \phi_1 \textit{ by virtue of theorem 10} \\ & \textit{then from } f_\pi(\phi_1) < f_\pi(\phi_2) \textit{ follows that } \phi_1 > \phi_2. \\ \#3 & \textit{for all } \Lambda_{likely} \vdash \phi_1 \textit{ and } \Lambda_{instable} \vdash \phi_2\textit{: } \phi_1 > \phi_2 \end{array} \tag{2.85}$$

The original AGM theory does not come with a modal-theoretic semantics. It was Grove (1988) who first showed how the AGM theory and its postulates can be recast in a system of possible worlds, which is called the *system of spheres*. Applied to Learn$-\alpha$'s model system $\mathcal{M} = \langle [\Gamma \cup \Lambda_0], R_\Box, R_{B_s} ..., \mathcal{V} \rangle$ in which the a-priori theory $\Theta_0 = \Gamma \cup \Lambda_0$ is the basis of the set of possibilities $W = [\Theta_0]$ (section 2.2.4) this means that there is a set of worlds $X \subseteq W$ that forms the center of a set of sets of worlds that is totally ordered by $\subseteq$ with X as the minimal and W as the maximal element (conditions S1 - S3 in Grove 1988). This set of sets is the system of spheres centered on X. Its elements are sets of worlds that 'are nested like matryoshka dolls or the layers of an onion' (Fritz 2009: 3). A system of spheres represents the belief state of the system with the most plausible beliefs in the center of the system. The crucial idea is that for any statement $\phi \in T$ if there is any sphere that intersects $[\phi]$ then there is a *smallest* sphere intersecting $[\phi]$ (condition S4). This allows for a definition of a function f_S that selects the 'closest worlds' in the system of spheres S in which ϕ holds. This system imposes a structure on the accessibility relation R_{B_s}, the relation of doxastic alternatives: it becomes a relation $\leq$ that is connected and transitive and that ensures that there is always a 'best' ($\leq$-minimal) set of worlds in which a proposition holds. The relevance for the revision operation becomes obvious: the relation $\leq$ 'may be related as a measure of how "compatible" alternative worlds are with our current beliefs' (Grove 1988: 160). Let $t(V)$ denote the set of all statements that hold in the worlds $V \subseteq W$ and let S be any system of spheres centered on $[T]$, then revision is defined as $T \dot{+} \phi = t(f_S(\phi))$. The function $t(f_S(\phi))$ corresponds to the selection function γ of the original AGM model. It ensures that the changes on T are minimal when T is revised by ϕ. The system of spheres model warrants the AGM postulates of revision (and by virtue of the Levi identity those of contraction, too) and corresponds to entrenchment-based contraction[95]. Although Grove (1988) supplies AGM

[95] According to Spohn (1988) the system of spheres model falls short of accounting for iterated revision. In the article he develops an alternative model of belief revision that is known as *ranking theory*.

with a model theory, he does not introduce an object language containing a modal belief operator. For a proposal of how this can be achieved see Segerberg (1995).

The theorems 10 and 11 support the handling of the two scenarios Ia and II exposed at the outset of this section. The first accounts for the LTH given a-priori, the latter for the significance of experience. Immediate repair (scenario Ia) of the analysis of an utterance u that cannot be parsed under the current theory Θ has to follow both theorems. However, the implementation can choose between postulate 11#2 and 11#3. The former allows for a more fine-grained revision (contraction) process in which all feature constraints of all lexemes realized in u are relaxed in the reversed order of entrenchment until the analysis becomes possible. The latter indicates a kind of more 'crisp' strategy that relies on the partitioning of the lexicon proposed in subsection 2.8.7. All lexical constraints within one partition are equalized and temporarily released altogether during immediate repair.
Revision in scenario II will happen once the new experience set fails to maintain the parameters β_{FP} and β_{FN}. For performance reasons it suffices to monitor the feature admissions/constraints entailed in the likely and instable sub-lexicons only, following postulate 11#1.

The problem with incremental learning is that mistaken decisions introduce more noise into the system and distort the observation capabilities and therefore the basis of proper revision. This is especially true for the feature values that are systematically neutralized. Once a wrong decision for such a feature value has been induced, the bifurcations in the lexical acquisition space turn into single mistaken paths, which block potential references that prove the alternative. To circumvent this effect, it might be interesting to perform a second analysis of every utterance in which the instable lexical constraints are temporarily released. While the output of the parsing system will be the first analysis (provided it does not require an immediate repair anyway), the second one - which might be called a *shadow analysis* - might be used to generate further references. We propose to call a system that does not really 'trust' a certain portion of its current theory a *skeptical learner*:

Def. 43 *(***Skeptical Learner***)*
Let $\mathbf{M} = \langle \mathbf{FS}, \Gamma, \Lambda \rangle$ *be the current consistent symbolic language model of a learning system* S*. Let* $\Lambda_{instable} \subseteq \Lambda$ *be the sub-lexicon that entails the beliefs whose strengths fall below a certain threshold.* S *is called a skeptical learner if it applies the generalized observation schema under* $\mathbf{M'} = \langle \overline{\mathbf{FS}, \Gamma, \Lambda \setminus \Lambda_{instable}} \rangle$

The exploration of this idea is left for further research.

2.10 Learn−α Revised

Towards the end of the first chapter (section 1.4, page 55) we proposed a set of requirements for an intelligent NLP system which is supposed to exploit its experience to improve by inferring lexical knowledge in an autonomous manner. Apart from being too vague, this set along with the preliminary version of Learn−α has a couple of difficulties as revealed in the present chapter. Vagueness has now been disposed of by the exact explication of the related notions (features, consistency, lexical knowledge, strength of belief, optimality, revision etc.) and the general observation and induction schemata (2.47 on page 147 and 2.20 on page 109, respectively). So what is left for this chapter is a sharpened formulation of the requirements, which are first reviewed briefly before the chapter concludes with a final version of Learn−α.

2.10.1 Reviewing the Requirements

Autonomous and unsupervised lexical acquisition is not a 'nice-to-have' feature of an intelligent and robust parser component. The inherent openness and the complexity of the lexicon make it a necessary condition, albeit not a sufficient one (there might be many additional requirements that are not considered in this work). We claimed that an intelligent system must neither be ignorant nor stubborn. The label 'ignorance' appears to be appropriate for a system that does not exploit the learning opportunities given or that would infer beliefs that are dramatically inconsistent with its experience - ignoring the supposed grammaticality of the input utterances. Further ignorance can be supposed if the system does not account for systematic neutralization (section 2.5) - ignoring a fundamental aspect of language.
On the other hand, stubbornness must be alleged if the system sticks to decisions that appear to be unwarranted on grounds of new experience. Learning is induction and the nature of induction is its fallibility. Hence revisability (section 2.9) must be considered at design-time of lexical features as much as learnability.
Apart from that, it has become clear that the optimal model - the target of the learning process - is a theoretical construct, whose reachability is prevented by noise and neutralization. But it makes sense to ask for the best and fastest approximation. This adds a third - more gradient - criterion, namely a good *learning performance*, defined in terms of accuracy and efficiency, which are two intertwined requirements, as an early induction (for the sake of efficiency) might be at the cost of a lower accuracy and a too low accuracy might be at the cost of unintended references and thus a lower efficiency. Thus, the best learning performance is achieved by finding an op-

timal balance of accuracy and efficiency.
In a word, the design of the learning system should enable it to arrive at the best approximation of $\mathbf{M}_{opt}$ in the shortest possible time and thus should enable it to gain maximum information with a maximum of precision out of the provided learning opportunities and use it to infer proper conclusions that lead to an improved performance, conclusions that might well be subject of later revision.

2.10.2 The Final Version

Having reviewed the necessary conditions for intelligent NLP systems, namely a design that properly accounts for the inherent complexity and openness of the lexicon, and having formalized the related notions of lexicon, features, experience, consistency etc., the final version of Learn−α can now be formulated. Generally, while the preliminary version of Learn−α demanded to 'learn a feature if you can', we now claim that learnability should be a necessary condition for the stipulation of a lexical feature.
Learning is seen as an interplay of induction (including observation) and revision for the purpose of continuous improvement. To approximate the optimal model as close as possible the system must find the optimal moment of induction - for every lexeme and feature. Hence, it must determine whether the number of learning opportunities in the experience is sufficient or if the induction step should be deferred. This behavior is controlled by the parametrization of the system. It has been shown in section 2.8 how a small set of basic parameters (such as β_{FP} and β_{FN}) controls test size, testing power and the strength of belief.
Further, accounting for noise in the experience, the demand for complete consistency of a feature determination with the experience is of course to be dropped. It is replaced by the claim that consistency with most of the experienced utterances has to be maintained. In a similar vein the correctness of grammar and lexicon is no longer taken for granted. In favor of an autonomous acquisition capability, section 2.8.5 proposes a procedure of automatic noise level estimation.
Additionally, we require that the system should target at an optimal learning performance, balancing out the conflicting aims of accuracy and efficiency. The latter is accounted for by avoiding any unnecessary procrastination of decisions. Note that the criterion of efficiency also subsumes the weak version of the preliminary version of Learn−α, namely that learning should show some effect. The criterion of accuracy is accounted for by demanding that first, RDoA=1 should be the default which can only be overruled for otherwise indeterminable features and second, RDoA_{max} should be as low as

possible (strategy of *cascading focus* developed in section 2.8.8).
Revisability is achieved by the requirement that those decisions that are not beyond any reasonable doubt are constantly reviewed and verified or falsified, respectively. This concerns both unlikely feature values that a lexeme is thought to admit as well as feature constraints that lead too an unmaintainable inconsistency with the experience.
All these considerations are summarized in the final version of the Learn−α design rule:

(2.86)

The Learn−α Design Rule (final)

The design of a natural language parser system shall enable it

i. to **induce** every defined lexical feature α of every lexeme $\mathcal{L}$, provided that there is a sufficient number of learning opportunities for $\langle \mathcal{L}, \alpha \rangle$ in the experience set. Learning shall
   (a) take place so that the setting of α is *consistent* with most of the utterances in the experience set and
   (b) be as *efficient* as possible and
   (c) be as *precise* as possible

ii. to **revise** the determination of a feature α if further experience
   (a) shows a too high degree of inconsistency with α or
   (b) proves a too low likelihood that $\mathcal{L}$ admits a value of α

Chapter 3

Analyze, Learn, Reduce

The theory of Learn$-\alpha$ fertilizes a family of *in vivo* lexical acquisition frameworks. While members of this family may vary in different aspects such as grammar formalism, parser, and others, they all have to satisfy the Learn$-\alpha$ design rule (subsection 2.10.2 on page 207). Common themes of all the potential implementations will be the generalized observation schema, the induction schema, automatic noise level estimation, strength of belief and other building blocks of the theory developed in the last chapter.
This chapter describes a framework called ANALYZE-LEARN-REDUCE (ALR) along with the prototype which implements it. The prototype is embedded in the Heart of Gold (HoG) system (Callmeier et al. 2004, Schäfer 2007) developed by the Deutsche Forschungsgesellschaft für künstliche Intelligenz (DFKI). It uses the ERG grammar and PET (Callmeier 2001) as the parser and facilitates the experiments reported in chapter 4. This was an attempt to develop an implementation of ALR on top of one of the state-of-the-art open source NLP frameworks with the least effort, motivated by the idea of turning such a framework into a learning system with as few modifications as possible. For example, the integrity of the grammar was left untouched, some few exceptions notwithstanding.
The resulting implementation is intended to be a proof-of-concept that pursues two related questions: a) how can Learn$-\alpha$ be put into practice and b) how effective and efficient is such a realization finally? While the latter question will be the scope of the following chapter, the present chapter investigates the former. It is important to emphasize that ALR and the resulting prototype is not claimed to be the best realization of Learn$-\alpha$ that can be anticipated, but it is one that is compatible with one of the most prominent broad-coverage grammars of English (the ERG) underpinned by one of the mainstream grammar formalisms, HPSG. This choice invokes some important limitations, two of which should be addressed at the outset: firstly,

HPSG was not designed for lexical acquisition by underspecification in the first place. The underspecification of SUBCAT, for example, results in non-terminating parsing. This is of course not the case for every feature, but an issue that has to be addressed at design-time. Another shortcoming from the perspective of Learn$-\alpha$ is the circumstance that there is no such notion as a 'lexical feature' in HPSG. In principle every feature can be 'lexical' as long as there is a corresponding constraint specified in the lexicon. This leaves the lexical acquisition space quite unstructured. Secondly, the ERG is by far not (yet) adequate in terms of definition 16 (page 96). As any large-scale modification of this complex grammar would deserve a thesis of its own, it was a conscious decision to keep the rules unchanged apart from the few adaptations required to realize Learn$-\alpha$ as such. The impact of the grammar's quality on the overall acquisition performance will be a topic of chapters 4 and 5.

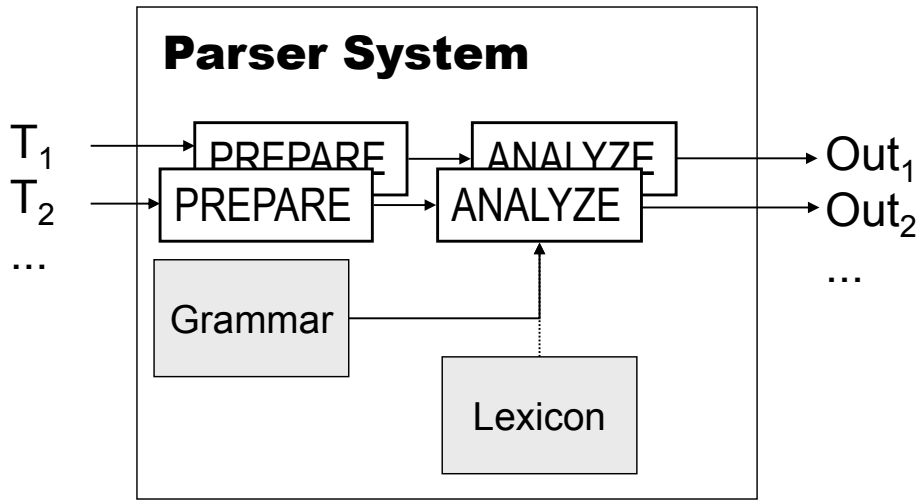
ALR is presented as follows. The first section describes the overall architecture by contrasting a parser system without ALR versus a system provided with ALR. Additionally it will serve as a functional specification of the framework, still independent of a particular grammar or formalism. The development of the prototype starts with section 2 which is dedicated to the most central concept of the implementation: the *lexical seeds*. Lexical seeds can be seen as the formal language in which the lexicon is specified. The type hierarchy generating the lexical seed instances is in fact a realization of the LTH described in subsection 2.1.3 (page 86).
The three prototyped modules ANALYZE, LEARN and REDUCE are described in sections 3.3 to 3.5.

3.1 Architecture of ALR

3.1.1 Overview

ALR is a framework that turns a parser system (just 'system' henceforth) into a learning system which satisfies the Learn$-\alpha$ design rule. To understand its mode of action, consider how a system without ALR may be schematically described (see figure 3.1). Although contemporary systems like LKB or PET are extremely complex, it suffices for this discussion to assume that there is a preparatory step followed by an analysis step (the parsing in the narrower sense). Preparation may consist of all necessary processes required to provide the ANALYZE module with tokenized and annotated utterances in a format that the latter can understand. In the following the preparation module is

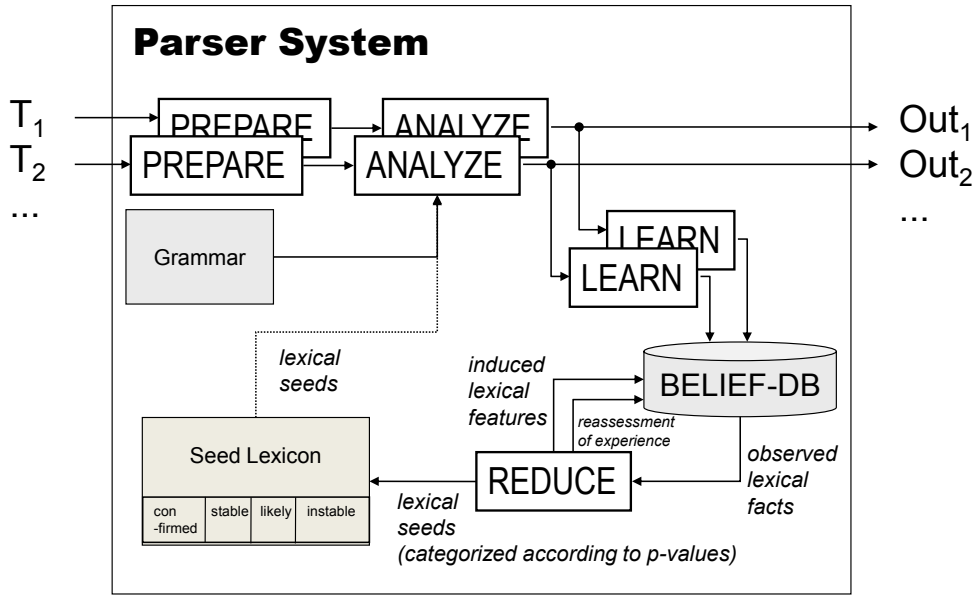
Figure 3.1: Parser System without ALR



completely taken for granted and out of discussion.
The ANALYZE module maps the incoming prepared utterances to linguistically motivated descriptions by virtue of the grammar and the lexicon. The latter is thought to correspond to Λ in Learn$-\alpha$ so that it does not contain anything else than open-class lexical entries, i.e. the feature value admissions ($\alpha^{\mathcal{L}}_{\mathbf{v}}$) and implicitly the feature value constraints ($-\alpha^{\mathcal{L}}_{\mathbf{v}}$) given by a *closed-world assumption*. The grammar component thus contains everything else: Γ (including morphology), the feature system **FS**, and all the closed-class lexical entries.
Because we are considering scalability of ALR at design-time we mention that parsing of separate texts T_i may happen in parallel.
In the ALR framework both the ANALYZE module and the lexicon get modified and two modules (LEARN and REDUCE) along with a database, called BELIEF DB, are added (figure 3.2).

There are two changes the ANALYZE module must undergo. Firstly, because in ALR it is lexical underspecification that makes lexical acquisition possible, the module has to be adapted in such a way that it can cope with

Figure 3.2: Parser System with ALR



underspecified lexical entries[1]. In the framework these potentially underspecified lexical entries are called *lexical seeds*. They are listed in the *seed lexicon*, which replaces the original lexicon. In this work the seed lexicon - containing the current entries - is sometimes referred to by the *current lexicon*. The lexical entries are classified as confirmed, stable, likely and instable along the lines explained in section 2.8.7 (page 195).
Secondly, it may be enabled for the capability called *immediate repair* (section 2.9 on page 200). The module should be able to work in a mode (controlled by a parameter) in which the grammaticality of the input can safely be assumed. This mode should be enabled for input texts that have some 'linguistic authority'. In this mode the analysis of an utterance that cannot be parsed using the current lexicon is tried with relaxed lexical constraints (giving up the instable first, then the likely, depending on the implementation). Further the behavior of a 'skeptical' system (definition 43 on page 205) may be implemented in which the output of the system is not identical with the input of the LEARN module, but where the result of a shadow analysis

[1] If the original system already has this capability this change is of course not required.

is fed into it using the current lexicon without the instable lexical constraints (not depicted).

The lexical seeds that feed the analysis during *lexical lookup* may be enriched with further information during derivation. As it is the case with the original system, the lexical entries put onto the chart form the hypothesis space of the parser. Parsing is the decision which of the hypotheses 'survive' the derivations. Underspecification in ALR just enlarges the hypothesis space. All the 'surviving' hypotheses correspond to lexical feature values realized by some of the available readings. The LEARN module 'harvests' the enriched lexical seeds that are contained in the derivations. Enrichment means binding lexical features with values that are specified in the grammatical rules applied during derivation. See section 1.5 for examples. The module considers the seeds of all readings[2] of an utterance and extracts the maximal information by applying the *learning down the hierarchy* procedure illustrated in section 2.8.1 on page 178. It reads off the seed's feature values under RDoA_{max}=1, 2, and ∞, respectively, which can be seen as an implementation of the *generalized observation schema*. The observed lexical feature value realizations are stored in the BELIEF database along with the ID of the corresponding utterance, which is used for later reference (cf. definition 23 on page 104). Note that these retained realizations are not yet the 'beliefs' of the system, they act more as the justification thereof. This is the starting point of learning, hence the name of the module.
As with the ANALYZE module, the LEARN module may work in parallel threads straightforwardly because the observations of different utterances (in different texts T_i) are mutually independent.

The *induction schema* has its counterpart with the REDUCE module. This is the place where statistical inference takes place. Basically, the module performs two different reductions: firstly, it reduces the wealth of observed lexical feature value realizations to single belief statements that are stored in the BELIEF database and that get assigned strength values plus categories of stability[3].
Secondly the module executes an algorithm which we call *reassessment of experience*. To understand its function consider that the newly induced belief states are not restricted to influence the disambiguation of the next utterances to come. They can be applied retroactively to the utterances in the

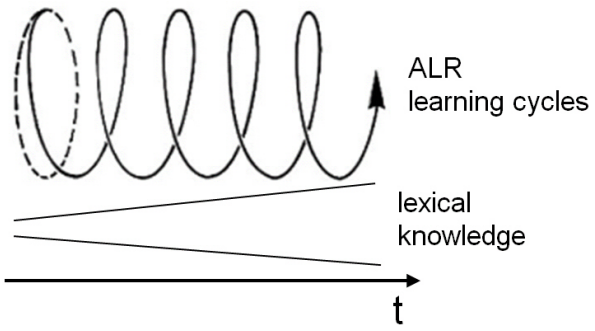
[2]May be up to a certain threshold, introduced later on.

[3]The belief states encode the formulae $B_s((-) \,\square\, \alpha^{\mathcal{L}}_{\mathbf{v}})$. The strength values and categorizations correspond to theorem 9 on page 195 and to the exemplary equation 2.81 on page 196, respectively.

experience set with the effect that some of the readings permitted previously are now ruled out. In general the reduction of admissible readings increases with the number of new belief states induced. As a desired side effect the new disambiguations can lead to further inductions leading to the rejection of even more readings and so forth. This loop is ended when no further inductions are possible. The interesting feature of the algorithm is that it can perform the retroactive disambiguations without repetitive parsing of the utterances thus being very efficient. The end of the loop is the point where the belief states are exported to the seed lexicon in forms of related lexical entries. This marks the end of one ALR *learning cycle*. The experience set can no further be exploited at this stage.

Incremental learning consists of the iterated execution of ALR learning cycles and may be visualized as a helix that moves on in time and that leads to increased lexical knowledge.

Figure 3.3: Incremental Lexical Acquisition



3.1.2 Functional Specification: ANALYZE

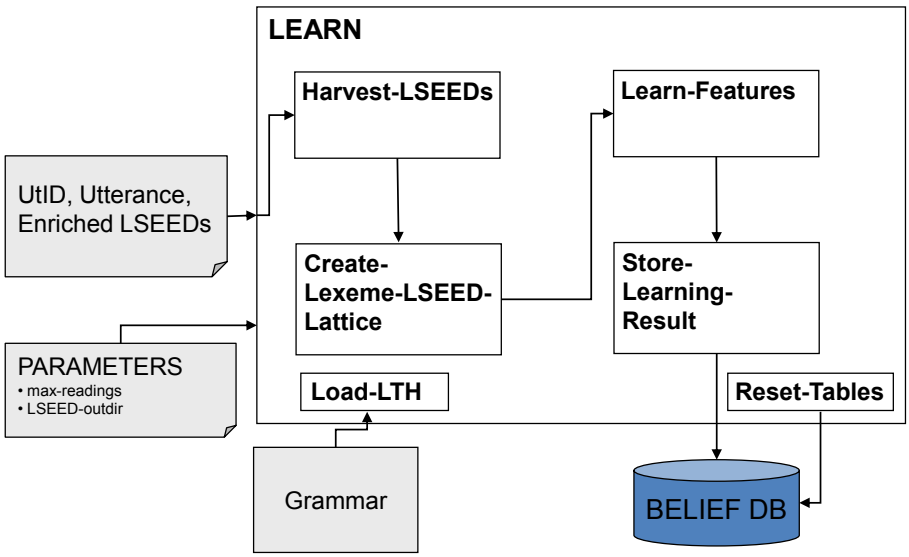
The ANALYZE module performs one main function only, the Analysis-Function. Five points are functionally important: firstly, the utterances must be annotated with a unique ID (UtID) so that the LEARN and REDUCE modules can unequivocally refer to them. Secondly, whatever the output of the analyzer should be in the first place, the derivation in form of feature structures must be exported so that the LEARN module can read off the enriched lexical seeds. This should be suppressed if the parameter *trust-grammaticality* is set to false, because learning should take place from linguistically 'trustworthy' texts only (cf. assumption 1 on page 60). Thirdly,

the enriched lexical seeds must somewhere encode the base forms (lemmas) of the lexemes. Fourthly, the enriched lexical seeds must be unequivocally related to particular token in the utterance. This can be achieved by using characterization with the features CFROM and CTO[4]. Lastly, the grammar employed must have an LTH (including default values for features) to which the lexical seeds are related and because learning must not be indifferent to the input ('effective learning', cf. page 57) the lexical seeds must meaningfully interplay with the grammatical rules.
Optionally the module should be able to perform *immediate repair* (section 2.9)[5].

3.1.3 Functional Specification: LEARN

The architecture of the LEARN module is presented in figure 3.4. The input of the module is the utterance annotated with its UtID and the related set of enriched LSEEDs. During startup the module loads the grammar or at least its LTH.

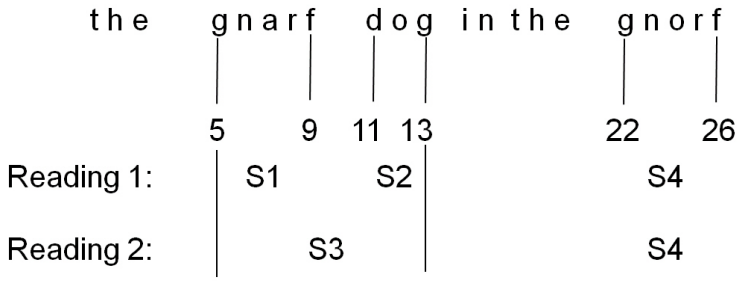
Figure 3.4: Architecture of the LEARN Module



[4]See http://wiki.delph-in.net/moin/RmrsSpan.
[5]This function is not yet implemented in the prototype.

Basically, the module executes four main functions in sequence when it processes one utterance: firstly, the utterance, its ID and the related enriched lexical seeds are read by the function *Harvest-LSEEDs*. If the UtID can be already found in the BELIEF database then processing has to be canceled in order to ensure that every utterance is processed only once. The function produces a set of unique tuples $\langle RBV, CFROM, CTO, EnrichedLSEED \rangle$. The enriched LSEED encoded in one tuple belongs to the token or tokens spanning from $CFROM$ to CTO and is read off from the readings specified in RBV. 'RBV' stands for *reading bit vector*. This technique allows for an extremely fast processing of set operations on reading sets. Each reading is represented by a bit in the RBV. Union of reading sets can then be realized by using bitwise AND operators. Because RBVs can become very long we propose to store them as strings of arbitrary length in the database.
The next function turns this tuple set into a structure that we call *lexeme LSEED lattice*, which is a crucial prerequisite for the *learning down the hierarchy* procedure. The lattice encodes the information whether a particular lexeme is unambiguously realized in a particular span $\langle CFROM, CTO \rangle$ or not. Every lexeme is uniquely represented by its lemma, which means that the function has to read off the lemma from the related LSEED. Consider the LSEED lattice in figure 3.5 for an illustration.

Figure 3.5: Example LSEED Lattice



Here the utterance *the gnarf dog in the gnorf* has (at least) two readings. While in both readings the span $\langle 22, 26 \rangle$ contains one lexeme `gnorf`, the material in the span $\langle 5, 13 \rangle$ is ambiguous. In reading 1 the two lexemes `gnarf` and `dog` are realized, in reading 2 the compound `gnarf dog` is filling the span. This means that even the identification of lexemes within this span is ambiguous and that the bifurcation in lexical acquisition space already starts at the identification step. As a consequence, RDoA_{max} cannot be better than 2 for the feature values in the structures connected to the lex-

emes in this span. This restriction is specified in a variable called $RDoA_{min}$ stored in the lexeme LSEED lattice. This lattice consists of a set of tuples $\langle lemma, span, RDoA_{min}, \{..., \langle EnrichedLSEED, RBV\rangle, ...\}\rangle$ which in case of the example results in the following:

$$\langle \texttt{gnarf}, \langle 5, 9\rangle, 2, ...\rangle$$
$$\langle \texttt{dog}, \langle 11, 13\rangle, 2, ...\rangle$$
$$\langle \texttt{gnarf dog}, \langle 5, 13\rangle, 2, ...\rangle$$
$$\langle \texttt{gnorf}, \langle 22, 26\rangle, 1, ...\rangle$$

Now consider the NP *the gnorfs* as a second case in point and suppose that both `gnorf` and `gnorfs` are unknown lexemes. Here the ambiguity is slightly different because there is no competing multiword lexeme involved. It is the uncertainty of whether the related lemma is *gnorf* or *gnorfs* which gives rise to the ambiguity. In other word the bifurcation is due to inflectional morphology. The resulting lexeme LSEED lattice would be:

$$\langle \texttt{gnorf}, \langle 5, 10\rangle, 2, ...\rangle$$
$$\langle \texttt{gnorfs}, \langle 5, 10\rangle, 2, ...\rangle$$

The lattice is fed into the *Learn-Features* function which can be considered the core of the module. This is the function that realizes the *learning down the hierarchy* concept. For every lexeme $\mathcal{L}$ in the lattice it creates a set of tuples $\langle \mathcal{L}, span, \psi, F, V, RDoA_{max}, RBV\rangle$ for all lexical features $\alpha = \langle \sigma, \psi, F\rangle$ and value V that can be observed under RDoA_{max}. RBV is the reading set to which the observation belongs. The span is included in order to distinguish multiple realizations of one lexeme in the same utterance properly. The realization contexts (ψ) and feature values are read off from the enriched lexical seeds stored in the lattice. The value V must be different from the most general value of the type σ in order to be included into the result set (because learning of this particular fact would not be effective, cf. section 1.4.3 on page 57).

The last function *Store-Learning-Results* writes the result into the four LEARNER tables in the BELIEF database[6], the structure of which is given in table 3.1:

For reasons of performance lexemes, realization contexts and utterances are represented by unique IDs: LEXID for lexemes, CID (context ID) for

[6]Note that the module must provide a function with which the LEARNER tables can be reset.

Table 3.1: LEARNER Tables

Table	Structure
lexemes	LEXID (integer,unique) LEMMA (string, unique)
utreferences	UID (integer,unique) UTID (string,unique) UTTERANCE (string)
rcontexts	CID (integer) CONTEXT (string, unique)
learntfeatvalrefs	UID (integer) LEXID (integer) CFROM (integer) CTO (integer) CID (integer) FEATURE (string) VALUE (string) RDOA (string) ACTIVE (integer) RBV (string)

realization contexts and UID for utterances. The latter is an internal ID, whereas UtID is the external identifier of an utterance. Whenever a new (potential) lexeme is encountered, an entry in the *lexemes* table is created with an incremented LEXID. This similarly holds for the realization contexts (table *rcontexts*) and utterances (table *utreferences*). Utterances for which no feature values are observed need not be retained. All the observed lexical facts are stored in table *learntfeatvalrefs* with the convention that mere realization contexts are stored with empty FEATURE and VALUE. The column RDOA is a string with three possible values {'1','2','inf'}. The column ACTIVE is used to indicate whether the observed feature value is (still) valid. If yes, then its value is 1 otherwise 0. Other values are temporarily used in the course of *reassessment of experience*. The LEARN module always stores the records with ACTIVE=1. An alternative to the deactivation of records (ACTIVE=0) would be their deletion. Retaining them as deactivated records, however, gives the LE the option to better trace the incremental learning process.

Querying the LEARNER Tables

The relational structure of the BELIEF database is part of the functional specification. It allows for a series of interesting Structured Query Language (SQL) queries that may serve the LE for a review of the learning process. Here are some examples:

1) Show everything that is observed about a particular lexeme:

```
SELECT DISTINCT CONTEXT, FEATURE, VALUE, RDOA
FROM
 LEXEMES L,RCONTEXTS R, LEARNTFEATVALREFS F
WHERE
 ACTIVE=1 AND LEMMA=... AND F.LEXID = L.LEXID AND F.CID=R.CID
```

2) Show everything that was observed from a particular utterance:

```
SELECT DISTINCT LEMMA, CONTEXT, FEATURE, VALUE, RDOA
FROM
 LEXEMES L,RCONTEXTS R, LEARNTFEATVALREFS F, UTREFERENCES U
WHERE
 ACTIVE=1 AND UTID=... AND F.UID = U.UID AND F.CID=R.CID
```

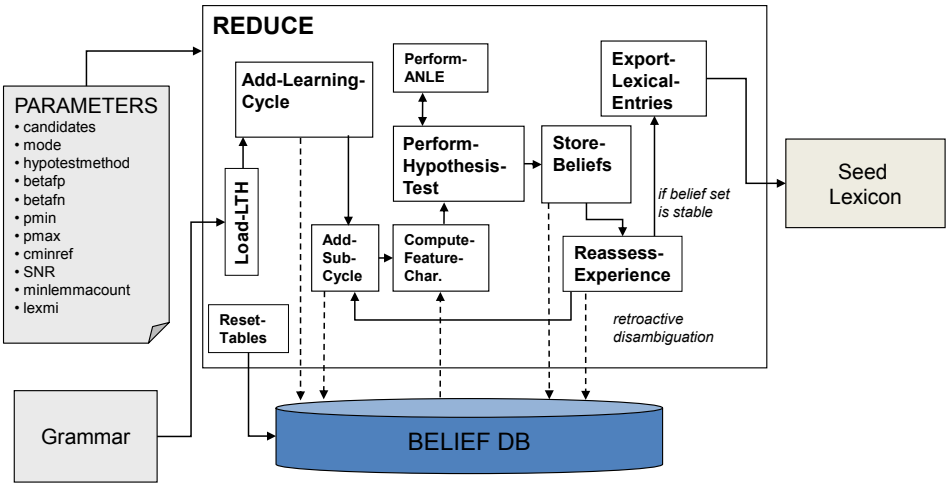
In a similar vein the number of references for a particular lexical feature and lexeme can be retrieved using the SQL's COUNT(*) construction.

3.1.4 Functional Specification: REDUCE

The architecture of the REDUCE module (figure 3.6) is slightly more complex than that of the previous module.

The module reads its input directly from the LEARNER tables. It can be customized using various parameters:

Figure 3.6: Architecture of the REDUCE Module



Parameters of the REDUCE Module	
candidates:	list of candidate lexemes that should be processed; if empty then all lexemes will be processed
mode:	can be used to run the module in different modes; one such mode is *forced* which means that induction takes place even if the recommended number of learning opportunities is not available (useful for testing)
hypotestmethod:	can be used to run the module with different hypothesis testing methods such as BHT, MLE, or LRT
betafp:	β_{FP}, see page 190
betafn:	β_{FN}, see page 119
lexmi:	pointwise MI threshold MI_{min}, see page 122
pmin:	p^{fv}_{min}, see page 113
pmax:	p^{fv}_{max}, see page 131
cminref:	c^{min}_{ref}, see page 188
SNR:	signal-noise-ratio, see page 129
minlemmacount:	the minimum number of references in which a lexeme candidate must have been observed before it enters the statistical procedures (also used as frequency cut off for the detection of single word lexemes)

The module uses the three tables shown in table 3.2 as its persistence layer. The first two are book-keeping tables that allow the LE to trace back which features have been induced in which learning cycle or sub-cycle. A learning cycle starts with the invocation of the module and ends with the export of the new lexical entries to the seed lexicon. It consists of $n \geq 1$ sub-cycles. During a sub-cycle hypothesis tests are performed for all candidate lexemes as much as the current experience set allows it. The last step of a sub-cycle is the reassessment of experience to which we come further below. The third table stores the current belief states of the system along with the category of stability, the strength of belief (cf. subsection 2.8.7) and the maximal RDoA under which the decision was made. The last column just serves the purpose of analysis by the LE, it is not crucial for the process as such. The BELIEVE column encodes the actual belief state: 1 (feature value admission), -1 (feature value constraint), 0 undecided. Keeping the undecided states serves a later reassessment by the LE and is not crucial, either.

Table 3.2: REDUCER Tables

Table	Structure
learningcycles	LCID (integer,unique) ACTIVE (boolean) STARTTIME (string or datetime)
subcycles	LCSUBID (integer,unique) LCID (integer) ACTIVE (boolean) STATUS (string) ENDTIME (string or datetime)
inducedfeatures	LCSUBID (integer) LEXID (integer) CID (integer) FEATURE (string) VALUE (string) BELIEVED (integer) STABILITY (string) STRENGTH (float) RDOAMAX (string)

As with the LEARN module during startup the grammar (or at least

the LTH) must be loaded. After that it deactivates all records of the table *learningcycles* (ACTIVE=false) and creates a new *learning cycle* by adding a new record to the table with ACTIVE=true and the current date/time. Then a new sub-cycle is added to the *subcycles* table for the current learning cycle ID (LCID). The record is stored with ACTIVE=true; all other records are deactivated (ACTIVE=false). The STATUS will be set to 'initial'. Then the processing of the sub-cycle starts. First the feature value characteristics (cf. 2.8.2) are computed by reading the *learntfeatvalrefs* table. Based on this the hypothesis tests for all the candidate lexemes will be conducted using the *cascading focus* algorithm 2 on page 197 and the default induction rule given on page 188. The effective noise levels required are computed on demand via the ANLE algorithm 2.8.5 on page 190. The induced beliefs are then stored in the *inducedfeatures* table and used to reassess the experience set by deactivating (ACTIVE=0) all records in the *learntfeatvalrefs* table that are (now) inconsistent because of non-existing lexemes or lexical feature constraints. We also call this process *retroactive disambiguation*. If any record is changed then a new sub-cycle is started. If no record is changed then this indicates that no further exploitation of the experience set is possible. In this case the final function is invoked which maps the current beliefs to lexical entries to be stored in the seed lexicon.

Assembly of Lexical Entries

The assembly of lexical entries is performed using the recursive algorithm 3. It adds logical formulae that represent the lexical entries to the global set ξ, which has to be empty at the first call of the procedure. Its input is the lexeme $\mathcal{L}$ for which the lexical entries are to be created and the top context, which is the ultimate context at first call of the procedure. The idea behind the algorithm is simple: First, all contexts that are *immediately* subsumed by the top context are considered and collected along with the corresponding believed feature values (lines 1-6). Then for all the selected contexts the algorithm is applied recursively (line 9). This ensures that the entries will be assembled from the bottom of the LTH up to the ultimate context. The remaining lines construct the logical formulae. They are processed as long as there are valid solutions (indicated by line 10). This will create all permutations of believed feature values applicable to the current context in the outer for-all loop. The condition in line 17 ensures that there are no redundant entries. This is why the contexts lower in the hierarchy are processed first: it allows for checking whether a logical formula that results from the belief states is already entailed by another formula.

Algorithm 3 Creation of Lexical Entries out of Belief States

Require: Let $\mathcal{B}$ denote the set of beliefs $B_s \,\Box\, \alpha^{\mathcal{L}}_{\mathbf{v}}$, T denote the top context and $\mathcal{L}$ be the lexeme for which the lexical entry feature structures have to be created. Let ξ be the global set of all lexical entry formulae.

Let $\mathcal{B}_{next} = \{\langle \sigma, \psi, F, \mathbf{v}\rangle |$

$(B_s \,\Box\, (\exists x)\ \mathcal{L}x\ \&\ \psi x\ \&\ \mathrm{F}x\mathbf{v}) \in \mathcal{B}$ and

σ is the type corresponding to ψ and

the realization context ψ entails T and

there is no other $(B_s \,\Box\, \mathcal{L}x\ \&\ \overline{\psi}x\ \&\ \overline{\mathrm{F}x\mathbf{v}}) \in B$

so that $\overline{\psi}$ is entailed by $\psi\}$

Let $\mathcal{C} = \{\psi \mid \langle \sigma, \psi, F, \mathbf{v}\rangle\}$

for all Contexts $\psi \in \mathcal{C}$ **do**

  Create all feature structures from $\mathcal{B}$ for $\mathcal{L}$ and top context ψ and add them to ξ.

  Let $\phi = \psi$

  for all possible ϕ **do**

    for all features F appropriate for type σ **do**

      if $\langle \sigma, \psi, F, \mathbf{v}\rangle \in B_{next}$ **then**

        Add ' & $Fx\mathbf{v}$' to ϕ

      end if

    end for

    if ϕ is not entailed by any logical formula in ξ **then**

      Add ϕ to ξ

    end if

  end for

end for

3.2 Lexical Seeds

We now turn to the description of the ALR prototype. As already stated, the aim was to keep the ERG and the parser component as untouched as possible. Turning them into a learning system without any modification, however, was not possible. Two types of modifications to the ERG have been made. Firstly, a sub-layer was built in that realized an LTH along the lines described in subsection 2.1.3: the *lexical seed layer*, which is the topic of the present section. Secondly, some minor changes to ERG rules have been made, which will be explained in section 3.3.

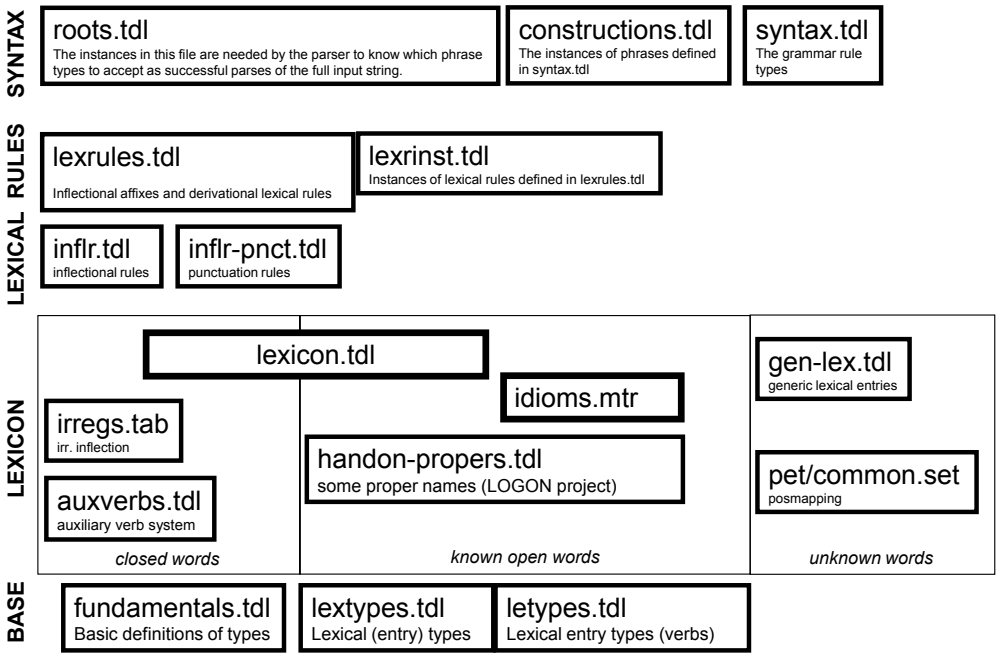
3.2.1 Structure of the ERG

In order to understand the operation of the lexical seed layer, it is worthwhile to consider how the ERG is structured. The version of ERG employed for the prototype is the one of February 2008; all comments to the structure of the grammar are related to this release. The description here aims to outline how the grammar is embedded into Learn$-\alpha$, but it is by no means a complete documentation of how the grammar works or how it is modularized. ERG files can be broadly grouped into four layers (figure 3.7): the BASE layer, the LEXICON layer, the LEXICAL RULES layer and the SYNTAX layer. The grammar and lexicon is specified in type description language (TDL, Krieger and Schäfer 1994). The BASE defines the most basic types including the lexical types and lexical entry types. Lexical types define the many word classes distinguished in the grammar. Lexical entry types (those ending with '_le') are those that are used in the lexicon. The LEXICON layer can be segmented into closed words, known open words and generic types for unknown words. The *lexicon.tdl* file includes both closed and open word classes and has more than thirty thousand entries. The *handon-propers.tdl* file contains proper names used for the HandOn project[7]. This file has been excluded from the prototype. The *idioms.mtr* file implements identification of idioms along the lines discussed in Copestake et al. (2002). Unknown word handling is realized by defining generic types in the file *gen-lex.tdl* to which PoS tags of unknown words will be mapped. The mapping for the PET parser is defined in the *pet/common.set* file.
Parsing with ERG is done by applying lexical and syntactic rules. The lexical rule instances are defined in *lexrinst.tdl* (e.g. dative shift) and the corresponding types in *lexrules.tdl*. Inflectional rules and punctuation rules are

[7] http://www.emmtee.net/index.php?page=7.

specified in *inflr.tdl* and *inflr-pnct.tdl*, respectively. Syntactic rule instances (also called 'grammar rules') are defined in *constructions.tdl*, their types in *syntax.tdl* (e.g. the head-complementizer rule). The 'start symbols' of the grammar are provided in *roots.tdl*.

Figure 3.7: Structure of the ERG



3.2.2 The Lexical Seed Layer

The introduction of the lexical seed layer pursues four goals. Firstly, it provides the ERG with an LTH which would otherwise be missing. In this regard it furnishes the grammar with a more concise structure of lexical facts (at least for open-class words) giving both the observation step and the induction step some guidance. Secondly, it allows for nearly arbitrary underspecification. The only feature that cannot be underspecified is LSSTEM, corresponding to Gx**v** in the theory (see section 2.1.5). While in the standard ERG the parser will be trapped in an endless loop if SUBCAT of a lexeme is not specified, in the prototype a lexical seed entry can be just **lseed** covering all open

word classes, for example, or **lseedbasicverb** covering all verb classes including all possible subcategorization frames with or without verbal particle. Lexical seeds thus perfectly facilitate the idea of partial lexical information. Thirdly, the lexical seed type system can be seen as a fine-grained formal 'language' of the open-class lexicon, the *seed lexicon*. Subcategorization frames are broken up into smallest elements such as valency, PP-arguments, sentential complements, voice etc. It is very straightforward to translate the belief set of the system into lexical entries formulated in this language. In order not to overload the prototype, however, some of the very rarely used lexical types in ERG cannot be expressed as a lexical seed. The fourth goal is encapsulation: the lexical seed layer separates the huge complexity of the grammar from the acquisition problem[8].

The lexical seed layer (figure 3.8) has become part of the lexical rule machinery of the ERG. Basically it consists of two main components: the lexical seed type hierarchy (the prototype's LTH, outlined in subsection 3.2.3) and the lexical seed mapping engine (subsection 3.2.4).
During lexical lookup the tokens[9] of an utterance are inserted as instances of the type **lseedentry**, a subtype of **lseedsign** which in turn is a subtype of **sign_min** (cf. section 3.3). An auxiliary boolean feature --LSEED has been introduced in order to distinguish an LSEED sign from a 'normal' sign[10]. The LSEED sign introduces the feature LSEED which is the root type of the lexical seed type system. The LSEED entry additionally carries the token span (CFROM, CTO) and the STEM. Before any other lexical rule (inflection etc.) can be applied, the LSEED entries must be mapped to the usual lexical types of the ERG. In course of the mapping the LSEED feature structure is copied to a new feature MAPPEDLSEED. After the lexical seed layer has been passed, parsing continues as usual with the addition that the feature structure rooted in MAPPEDLSEED is propagated up to the final nodes and enriched with further information during the application of the lexical and syntactic rules (cf. section 1.5 for some illustrations). The MAPPEDLSEED structures will be harvested by the LEARN module.

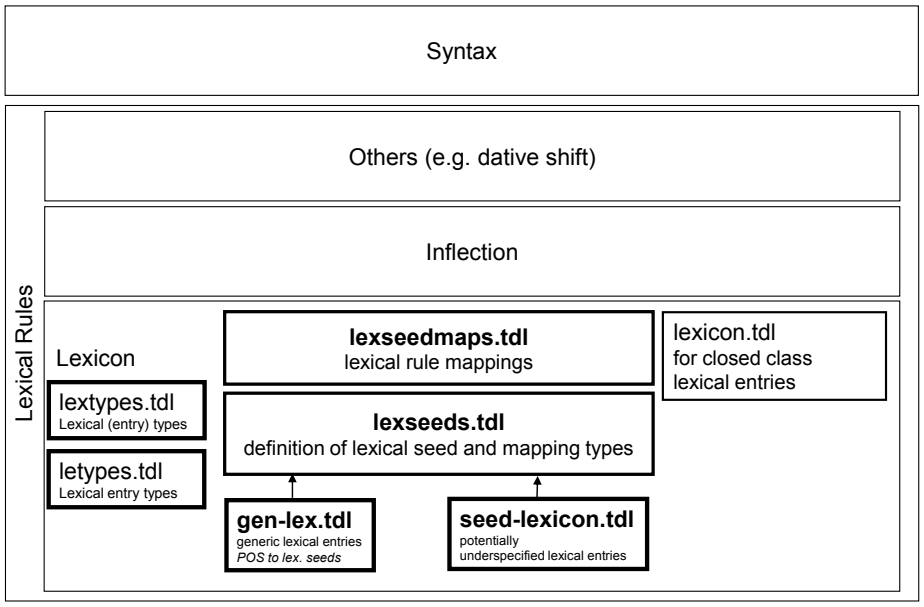
Technically the lexical seed layer is realized in the two new Type Description Language (TDL) files *lexseeds.tdl* and *lexseedmaps.tdl*. The former contains lexical seed and mapping types, the latter instantiates the mapping rules. Lexical seed entries are stored in the third additional file *seed-*

[8] After having intensively worked with the ERG, we find it hard to imagine a prototype that meaningfully interplays with the whole grammar.

[9] Only tokens of open word classes; closed-class lexical types remain in *lexicon.tdl*.

[10] In general, we mark abstract and/or auxiliary features with a preceding double hyphen.

Figure 3.8: Lexical Seed Layer



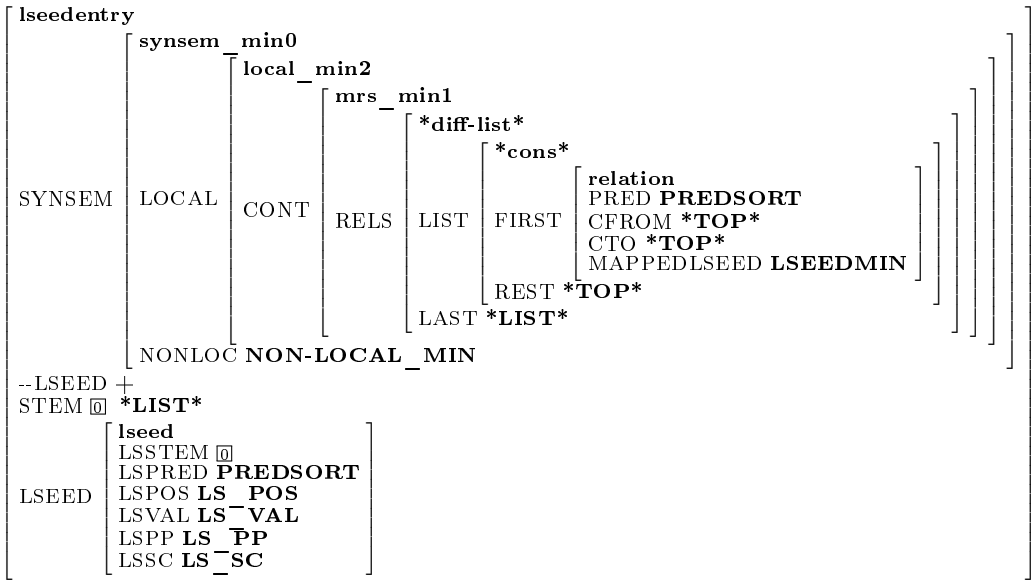
lexicon.tdl (cf. the example entries discussed in subsection 3.2.5).

3.2.3 The Lexical Seed Type Hierarchy

The lexical seed type hierarchy is the implementation of the LTH assumed by the Learn−α theory. As stated above, its root type **lseed** is part of the LSEED entry, whose type definition is presented in figure 3.9 (some details omitted). In addition it can be seen that the MAPPEDLSEED feature is part of the ERG's **relation** type[11]. This ensures that the feature will be properly propagated to the top nodes. This is the place where the token span is encoded, too, which simplifies harvesting. The **lseed** type contains the following features:

[11]This type is part of the minimal recursion semantics (MRS) component of ERG, cf. Copestake et al. (1999a) and Copestake et al. (2005). Note that the MAPPEDLSEED had to be added to the 'ignored semantic features' in *lkb/mrsglobals.lsp* and *pet/mrs.set*.

Figure 3.9: Lexical Seed Layer: Type **lseedentry**



Features Introduced by lseed	
LSSTEM:	corresponds to the basic form (lemma) of the lexeme
LSPRED:	encodes the sense of the lexeme
LSPOS:	part of speech
LSVAL:	valency (intransitive, monotransitive, ditransitive)
LSPP:	subcategorized prepositions
LSSC:	sentential complements

Further lexical features are introduced by subtypes of **lseed**. Figure 3.10 displays the top level of the type hierarchy.

Figure 3.10: Lexical Seed Type Hierarchy (Top Level)

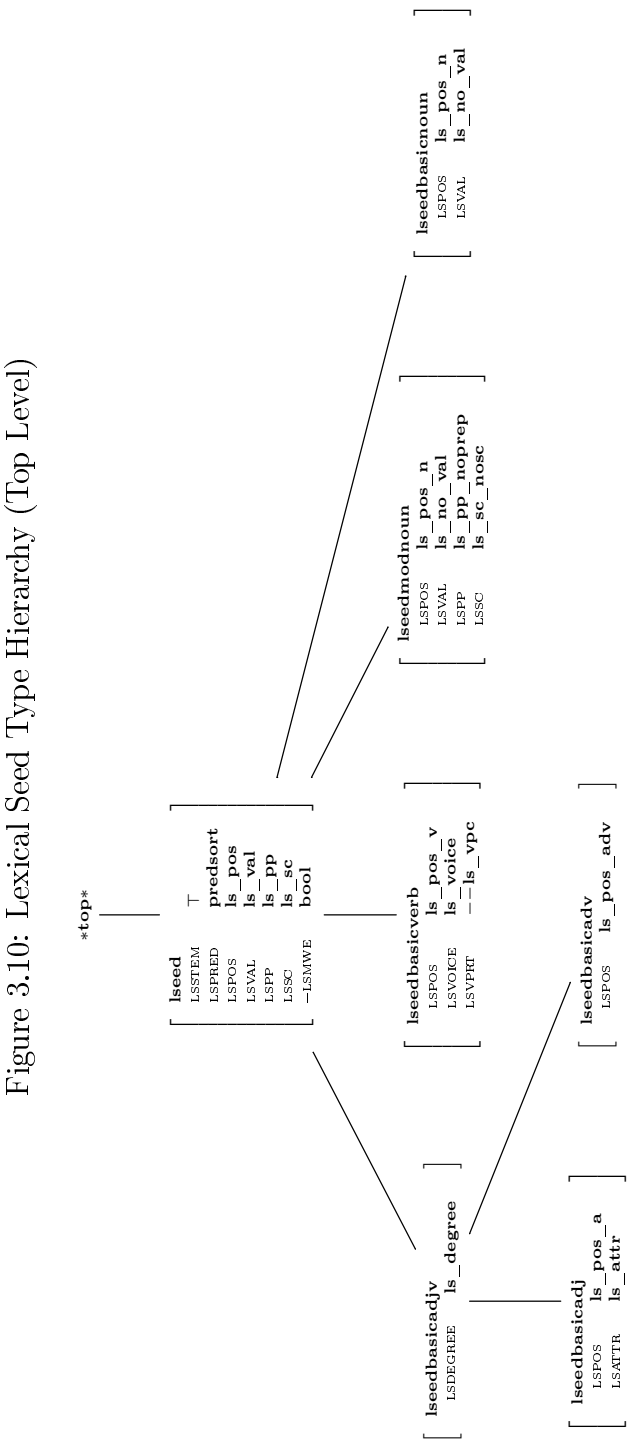
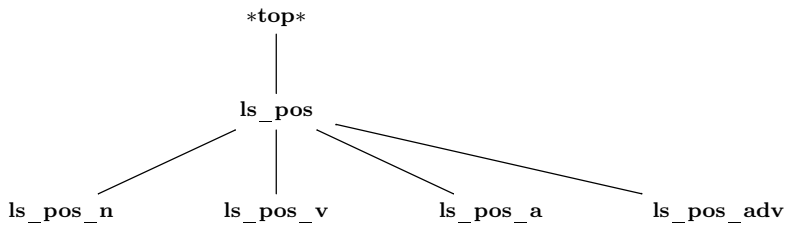


Figure 3.11: Lexical Seed Types (ls_pos)

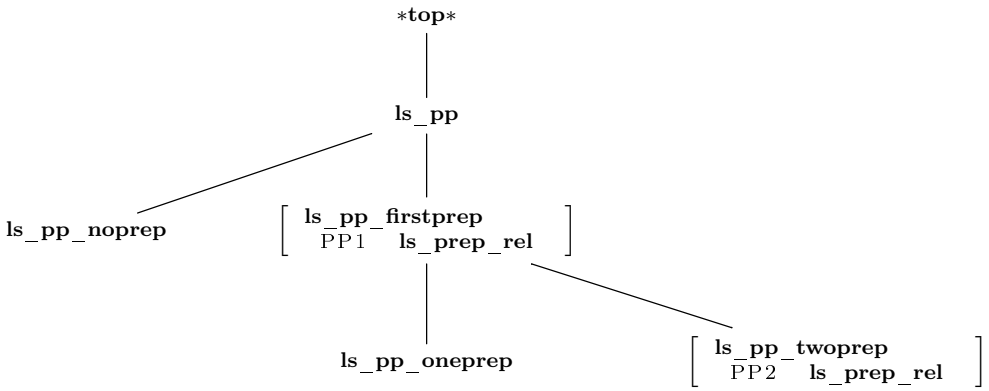


Most of the type and feature names are self-explanatory. A brief description of the complete hierarchy is compiled in the appendix E. Lexical seeds are subdivided into **lseedbasicnoun** for nouns that do not modify verbal phrases, **lseedmodnoun** for modifying nouns like `way`, **lseedbasicverb** for verbs and **lseedbasicadjv** for adjectives and adverbs. The latter type splits up further into **lseedbasicadj** for adjectives and **lseedbasicadv** for adverbs. The hierarchy employs some of the standard ERG types, such as **predsort** (for sense/predication), **bool** or the absolute root type ***top***. For some feature values new types have been created. The types **ls_pos** and **ls_val** (cf. E.1) are the roots of atomic type hierarchies, which means that neither they nor any of their subtypes introduce features. The types **ls_pp** and **ls_sc**, in contrast, are the roots of non-atomic hierarchies.
For the sake of illustration, one atomic and one non-atomic hierarchy will be discussed. Consider first the tiny atomic hierarchy in figure 3.11.

It shows the values appropriate for the part-of-speech encoding feature LSPOS. The subtypes are used for the open classes nouns, verbs, adjectives and adverbs (from left to right).
Figure 3.12 shows the non-atomic hierarchy employed for the LSPP feature.

This hierarchy is used to specify whether a lexeme subcategorizes for no PP (**ls_pp_noprep**), one PP (**ls_pp_oneprep**), or two PPs (**ls_pp_twoprep**). Moreover the features introduced here are used to encode *which* preposition is subcategorized (PP1 and PP2).
The lexical seed hierarchy does not only introduce new lexical features. It also sometimes constrains features. For example, the type **lseedmodnoun** constrains the feature LSSC to the value **ls_sc_nosc**, because modifier nouns do not subcategorize for sentential complements. Here it is important to emphasize that a noun like `way` has multiple lexical entries: one for **lseedmodnoun** and at least one for **lseedbasicnoun** or subtypes thereof. The mentioned constraint only applies to the first, not to the second. In a similar vein,

Figure 3.12: Lexical Seed Types (ls_pp)



prepositions subcategorized for by a lexical seed entry are not optional[12]. If a PP is optional, then this requires two entries in the seed lexicon, one using **ls_pp_noprep** and one using **ls_pp_oneprep**.

3.2.4 Lexical Seed Mapping

The types that make up the mapping engine of the lexical seed layer are subtypes of **lex_rule_lseed**, shown in figure 3.13, some details again omitted. The daughter (DTR) of this lexical rule type is of type **lseedentry**, because this is the type which is the input of the mapping. The type itself has [--LSEED −] indicating that it is no LSEED sign anymore. The span features are propagated from the daughter to the mother, LSSTEM maps to STEM and LSPRED maps to PRED. The feature GENRE is set to **formal** in order to make sure that lexical acquisition is not hampered by a too lax and overgenerating grammar. This feature is propagated to the top nodes and will be checked by the root-types defined in *roots.tdl*.

[12]This means that the ERG standard feature OPT is not employed in the prototype.

Figure 3.13: Lexical Seed Layer: Type **lex_rule_lseed**

For each ERG lexical entry type that has to be covered there must be a lexical seed mapping type defined, such as the following (specified in TDL):

```
lseedmap_v_pp_le := lex_rule_lseed_pp1 & v_pp_le & [
   DTR.LSEED lseeditrverb &
 [ LSPP ls_pp_oneprep &  [ PP1 prep_rel ],
   LSSC ls_sc_nosc
 ]
].
```

First of all, **lex_rule_lseed_pp1**, a subtype of **lex_rule_lseed**, is used for all mappings that involve a lexical type with one subcategorized PP. The type makes use of the --COMPKEY mechanism of ERG which controls verbal particle and preposition subcategorization (Villavicencio and Copestake 2002). It links the --COMPKEY value to the value of PP1 in the lexical seed. In addition, and this is crucial, it links itself to the ERG lexical entry type **v_pp_le** which is used for non-phrasal intransitive verbs with one PP[13]. Figuratively, a lexical seed entry mapped by this rule 'leaves' the lexical seed layer as a 'v_pp_le lexeme' subject to the normal ERG machinery. By virtue of the constraints specified only LSEED entries of type **lseeditrverb** having one PP and no sentential complement can be mapped by this rule. In cases where the LSEED entry is underspecified for LSPP or LSSC, respectively, the mapping will bind the feature values accordingly (learning by unification). Likewise, any entry that is of a supertype of **lseeditrverb**, such as **lseedbasicverb** or **lseed**, will be picked up by this rule because it unifies with it.

3.2.5 Lexical Seed Entries

This section concludes with a couple of exemplary lexical seed entries. It will be demonstrated how these entries go through the mapping engine before they undergo the usual ERG processing. The first one is a maximally underspecified entry, which even does not specify part-of-speech:

```
gnarf := lseedentry & [
  stem <"gnarf">, LSEED lseed &
  [ LSPRED "_gnarf_pred" ]
].
```

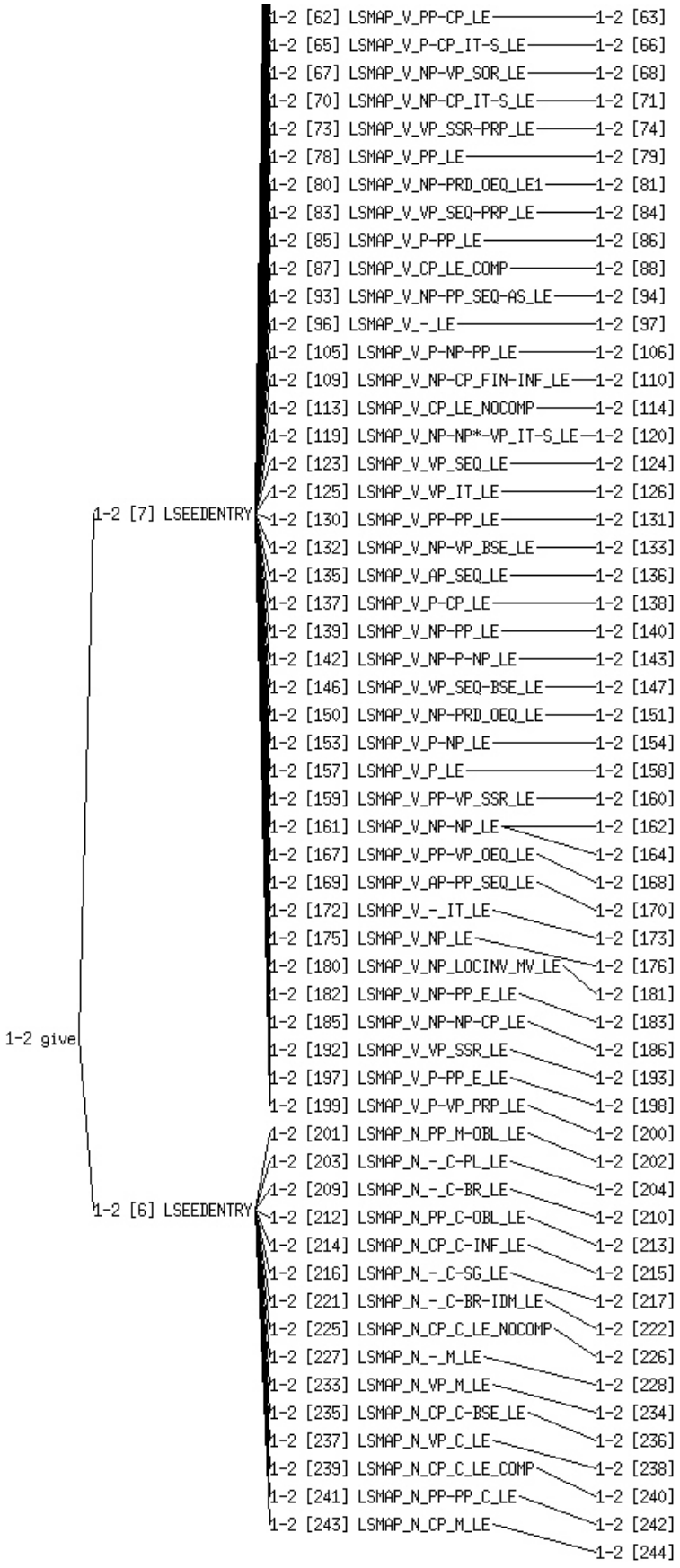
[13] See http://wiki.delph-in.net/moin/ErgLeTypes for a list of lexical types in ERG along with examples.

Note that this example is quite theoretical. In the experiments the initial lexicon specifies at least part-of-speech. Every token 'gnarf' or any token that can be morphologically related to the base form 'gnarf' will be picked up by all the 88 mappings defined in *lexseedmaps.tdl* resulting in a large disjunctive hypothesis of 88 lexical types.
The next entry shows a highly underspecified specification of the lexeme `give` as it is used in the bootstrapping experiments (chapter 4):

```
give_n := lseedentry & [
  stem <"give">, LSEED lseedbasicnoun &
  [ LSPRED "_give_n_pred" ]
].
give_v := lseedentry & [
  stem <"give">, LSEED lseedbasicverb &
  [ LSPRED "_give_v_pred" ]
].
```

By virtue of this definition, all mapping rules for adjectives and adverbs are incompatible with the lexical specification of `give`. Apart from this still a lot of mapping rules are compliant with these entries. Figure 3.14 shows a cutout of the chart of an utterance containing 'give'. (The mapping rules all start with 'lsmap_' and they usually contain the name of the lexical type they map to).

Figure 3.14: Chart for Lexical Seed Mapping of `give`



The entries for *unknown words* are similar. Lexical seed types for idiom parts, however, are excluded. This is how the generic entry for nouns in *gen-lex.tdl* is specified[14]:

```
generic_noun := lseedentry & [
   STEM < #base >,
   TOKEN.+TNT.+TAGS.FIRST <"NN", "NNS" >,
   TOKEN.MORPHBASE #base,
   LSEED lseedregularnoun & [ LSPRED "unknown_noun_pred"  ]
].
```

The entries for verbs, adjectives and adverbs fully work in analogy. The original generic lexical types are not required for the prototype. As a side-effect, because not all possible frames are realized with the original unknown word management, the coverage increases by virtue of these highly underspecified entries.

The last example illustrates the encoding of one of the verbal SCFs defined by Briscoe and Carroll (1997), namely monotransitive frame with one subcategorized preposition (here `for`): SCF31.

```
buy_v := lseedentry & [
  stem <"buy">, LSEED lseedmtrverb &
  [ LSPRED "_buy_v_pred",
    LSPP ls_pp_oneprep & [ PP1 _for_p_rel],
    LSSC ls_sc_nosc
  ]
].
```

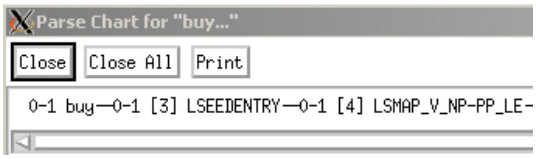
The lexeme of this entry is still not fully specified, for example the value of VOICE is unspecified, which is correct given that the verb can be used both in active and passive voice. There is only one mapping possible as demonstrated in figure 3.15.

3.2.6 Default Feature Values

A last word concerning default feature values is in order. In section 2.8.3 an argument was put forward for the introduction of a feature meta property that specifies a default value for a feature which is invoked during induction

[14]Note that the prototype still works with the pre-2009 unknown word management. The new chart-mapping (Adolphs et al. 2008) is not employed.

Figure 3.15: Chart for Lexical Seed Mapping of buy



in case no other value is observed. In the prototype, however, this meta property has not been introduced. Instead, a rule that is implicitly used in the REDUCE module has been setup: all boolean features automatically have '-' as default.

3.3 Module ANALYZE

The ANALYZE module, as opposed to the other two modules, has not been developed from scratch. Rather, it is the outcome of a set of modifications to the natural language processing environment at hand. These modifications serve the fulfillment of the requirements stated in subsection 3.1.2, briefly repeated here for convenience:

i. looping through of the utterance IDs (UtIDs)

ii. export of the feature structures containing the enriched lexical seeds (MAPPEDLSEED)

iii. enriched lexical seeds must contain the base forms of the lexemes

iv. enriched lexical seeds must be associated with token span

v. the grammar must have an LTH and must interplay with the lexical seeds (learning by unification)

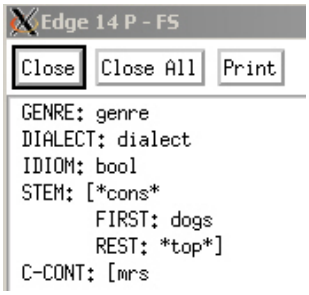
The prototype is based on the HoG with the ERG as the grammar. The last three requirements involve modifications to the ERG and the first three are realized by slightly adapting the NLP infrastructure. This section first outlines the changes to the grammar (subsection 3.3.1), then it summarizes the architecture of the NLP infrastructure, as far as it concerns ALR (subsection 3.3.2). This gives a better understanding of the changes to the PET parser (subsection 3.3.3) and to HoG (subsection 3.3.4).
Note that the prototype does not implement the capability of *immediate repair*.

3.3.1 Requirements 3-5: Modifications to the ERG

The most drastic change of the ERG is the introduction of the lexical seed layer described in section 3.2. The association of the enriched lexical seeds stored in MAPPEDLSEED with the token span (requirement #4) is achieved by hosting this feature in the **relation** type, which already encodes the token span (CFROM, CTO). The new layer takes account of requirement #5 because it provides the LTH and by the mapping engine the lexical seed features reflect the various word classes distinguished by the grammar. In a few cases, the interplay with the lexical seeds has to be built in directly to the grammatical rules. Before we turn to these, a solution for requirement #3 is outlined.

Feature MORPHBASE

As it may come as a surprise, the original grammar does not consistently encode the morphological base forms of words. This may be the case because this is not really required for its functionality. The STEM feature is used to encode the base form in lexical entries, but, despite its name, it may also contain the fully inflected form as exemplified in the screen shot showing the partial feature structure of 'dogs' after application of the inflectional rule PLURAL_NOUN_ORULE:



This is especially problematic for unknown word management: here the STEM feature is bound to the token and not to its base form. To solve this issue, we introduced a new feature MORPHBASE at type TOK_MIN and made PET bind the base form to it (see also subsection 3.3.3). It is this feature which is then used in the generic lexical entries (cf. the TDL fragment in subsection 3.2.4). In order to make the feature available for instances of **lseedentry**, the feature TOKEN has been moved up from **word** to **sign_min** (this being a supertype of **lseedentry**). The solution ensures that requirement #3 is fulfilled.

In order to make STEM available for **lseedentry**, the feature has been moved up from **sign** to **sign_min**.

Feature VOICE

The lexical feature VOICE is used to put constraints on diathesis alternations. Its value is set to **ls_pas** (for passive voice) in the lexical rule *basic_passive_verb_lr*.

Feature LSCPSUBJ

Not all verbs allow for sentential subjects (such as 'That Kim slept bothers Sandy', taken from *syntax.tdl*). Whether they can is controlled by the boolean lexical feature LSCPSUBJ. Its value is set to '+' in the rule *cp_subj_phrase*, the one that builds phrases with CP subjects. The feature is a case in point for the application of the *default induction rule* (see subsections 2.8.3 and 3.2.6), because its value is never set to '-' (apart from the setting in the a-priori given type **lseedexplverb** for verbs with expletive subject). This means that the '-' value can only be induced via default induction.

Feature LSDEGREE

The lexical feature LSDEGREE, introduced by the lexical seed type **lseedbasicadjv**, controls degree of adjectives and adverbs. If comparative or modification by degree specifiers is allowed then its value is **ls_graded**. This value is directly set in the type **basic_degree_spec_synsem**. Comparatives and superlatives, however, are not accounted for in the prototype.

Multiword Expressions

From the classification of MWEs in subsection 2.7.3 four classes have been accounted for in the prototype: noun compounds (group B.2/3), determinerless PPs (group B.1), VPCs (group C.3) and VN idioms (group B.3/C.2). The solution around VPCs fully leverages on the ERG standard and needs no modification. See subsection 1.5.2 for illustration. Noun compounds are easily handled in the prototype: the rule *basic_n_n_cmpnd_phr* responsible for noun compounding is enhanced by the following elements:

```
basic_n_n_cmpnd_phr := ...
   ...
   SYNSEM.LOCAL.CONT.RELS.LIST.FIRST.MAPPEDLSEED lseedbasicnoun &
     [ LSSTEM #stem, LSPRED "unknown_noun_compound" ],
   STEM #stem & <#stem1 . #stem2>,
```

```
NH-DTR.SYNSEM.LOCAL.CONT.RELS.LIST.FIRST.MAPPEDLSEED.LSSTEM #stem1,
HD-DTR.SYNSEM.LOCAL.CONT.RELS.LIST.FIRST.MAPPEDLSEED.LSSTEM #stem2
...
```

When a noun compound is created, then, by virtue of the additional specification, the (otherwise empty) MAPPEDLSEED feature of the compound is bound to a feature structure of type **lseedbasicnoun** which can later be harvested by the LEARN module.
Its LSPRED is set to "unknown_noun_compound" and the LSSTEM is just a concatenation of the daughter's stem features. The LEARN module has the capability to extract the lemma of the compound out of these arbitrarily complex list structures.

In order to understand the constructions around the remaining two MWE classes consider how determinerless PPs and VN idioms are handled in the ERG. The grammar ensures that whenever there is an idiomatic element in a phrase, the phrase is marked as idiomatic by virtue of the boolean feature IDIOM which is then set to '+' (otherwise it remains **bool**). This is the case for bare nouns used in determinerless PPs (realized by the lexical entry type **n_-_c-br_le**) and for VN idioms (types **v_nb_idm_le** and **n_-_c-br-idm_le**). In course of parsing [IDIOM +] phrases will be checked using the MRS transfer rule mechanism (MTR) along the lines motivated in Copestake et al. (2002: section 4). Transfer rules for idioms are specified in *idioms.mtr*. As an example consider the entry for the VN idiom `give way`:

```
give+way := v_nbar_idiom_mtr &
  [ INPUT.RELS.LIST < [ PRED "_give_v_i-way_rel" ],
                      [ PRED "_way_n_i_rel" ], ... > ].
```

Licensing of idioms takes place on MRS level, which means that only features like PRED are available for checking. In the example, the predicates of the idiomatic words `give` and `way` are checked. Other nouns like `guard` ('stand guard') would be ruled out: **give guard*. This mechanism works if all the idiomatic combinations are known to the system, hence a problem for Learn$-\alpha$.
In our solution to this problem, two generic idiom-MTR rules have been introduced, one for VN idioms and one for determinerless PPs:

```
generic_v_idiom := v_nbar_idiom_mtr &
  [ INPUT.RELS <! [ PRED predsort ],
                  [ PRED predsort ],
                  relation !> ].
```

```
generic_detless_pp_idiom := detless_pp_idiom_mtr &
  [ INPUT.RELS <! [ PRED ls_prep_rel ],
                  [ PRED predsort ],
                  [ PRED idiom_q_i_rel ] !> ].
```

Because all PRED values of the nouns and verbs, respectively, fit these rules, the constraints must be encoded elsewhere. The availability of building idiomatic phrases is now controlled by a new auxiliary feature --LSGOV introduced by **sign_min** (being thus available for lexical seed entries and normal signs). Its value is of type **ls_gov** with the following definition:

```
ls_gov := *top* & [
        LSGOVHEAD head,
        LSGOVPRED predsort
].
```

This structure encodes the information about the governing head, its part-of-speech (given with **head**) as well as its predicate. A few changes to the basic rule types in *syntax.tdl* ensure that the --LSGOV feature is always properly bound. A lexical entry can now put constraints on --LSGOV thereby selecting with which words it can form idiomatic phrases. This mechanism is employed in the following lexical seed mapping rules:

```
lseedmap_n_-_c-br_le  := lex_rule_lseed & n_-_c-br_le & [
     SYNSEM.LOCAL.CAT.HEAD.LSMWE +,
     --LSGOV.LSGOVPRED #pred,
     --LSGOV.LSGOVHEAD prep,
     DTR.LSEED lseednoun_detless_pp & [ LSGOVPREP #pred ]
].

lseedmap_n_-_c-br-idm_le := lex_rule_lseed & n_-_c-br-idm_le & [
     SYNSEM.LOCAL.CAT.HEAD.LSMWE +,
     --LSGOV.LSGOVPRED #pred,
     --LSGOV.LSGOVHEAD verb & [ LSMWE + ],
     DTR.LSEED lseednoun_v_idiom & [ LSGOVVERB #pred ]
].
```

Regarding the nominal branch in the LTH, the two idiom types are accounted for by two lexical seed types: **lseednoun_detless_pp** for nouns that occur in determinerless PPs and **lseednoun_v_idiom** for VN idioms. The two rules above map them to the standard ERG types **n_-_c-br_le** and **n_-_c-br-idm_le**, respectively. In addition, the lexical seeds specify with which predicate they may combine: LSGOVPREP for the prepositions and LSGOVVERB for the governing verb. The mapping rules also constrain the part-of-speech category of the governing head.

In order to make sure that these nouns combine with MWE items only, a further boolean feature was introduced at type **head**: LSMWE. The lexical seed mapping engine ensures that non-idiomatic lexemes specify [LSMWE −].

Interface to the Prepositional and Verbal Particle System

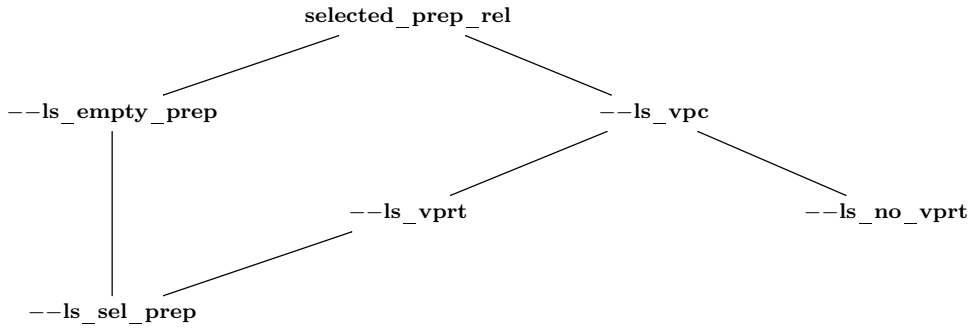
Some linguistic generalizations that may help in the acquisition process are only implicitly given in the grammar or lexicon, respectively. One case in point are verbal particles and prepositions. The ERG distinguishes particles/prepositions of type **selected_prep_rel** for VPCs and semantically bleached prepositions on the one hand and full prepositions of type **prep_rel** on the other hand. The first type can thus be used to mimic phrasal and prepositional verbs in the sense of Quirk et al. (1994) (cf. our discussion on page 167). A phrase $[_V$ V^0 P NP] may be projected from a) a verb subcategorizing for a non-empty preposition, b) a verb subcategorizing for an empty preposition (Type I/II Prepositional Verb) or c) from a Type II Phrasal verb. Consider the examples taken from the ERG lexicon in (38):

(38)	a.	account for sth.	**selected_prep_rel**
	b.	brace for sth.	**prep_rel**
	c.	break down sth. / sth. down	**selected_prep_rel**

While the preposition `for` employed in (38a) is of type **_for_p_sel_rel**, a subtype of **selected_prep_rel**, its type in (38b) is **_for_p_rel**, a subtype of **prep_rel**. So, theoretically, the first version could act as a particle in a VPC, an option which could be excluded upfront (see the list of particles in Quirk et al. 1994: 1151). That there is no VPC with `for` is implicitly encoded in the lexicon: there is simply no such VPC lexical entry.
The downside, however, is that during acquisition such an entry may be hypothesized. In order to restrict this we introduced a small 'interface' to the original prepositional and verbal particle system in terms of lexical seed types. For verbal particles the lexical seed type hierarchy does not directly use the original types, but employs the interface types instead. The type hierarchies related to them are depicted in figure 3.16.

There is one atomic type per structural alternative mentioned above: **--ls_vprt** for Type I/II Phrasal Verbs, **--ls_empty_prep** for semantically empty prepositions and the original type **prep_rel** for prepositions with semantic content. Particles that can be used as an adverb (*fall down*) as well as a preposition (*fall down the hill*) are of type **--ls_sel_prep**. The modification is actually done in *fundamentals.tdl* where the particle types are defined. In this context we also added a few missing verbal particles.

Figure 3.16: Interface Types for Prepositions and Verbal Particles



Minor Corrections

Two minor corrections have been applied to the ERG. First, lexical types **arg123_lt** and **arg123_double_pp_lt** have been added for transitive double-PP verbs, and the type **np_trans_double_pp_verb** adapted accordingly, because the originally used type **arg12_double_pp_lt** enforced agreement of the direct object NP and the object of the first PP.
Secondly, the type **pro_wcomps_plur_synsem**, which is responsible for phrases such as 'we/you/us humans', has been changed to avoid overgeneration of 'us that' and the like.

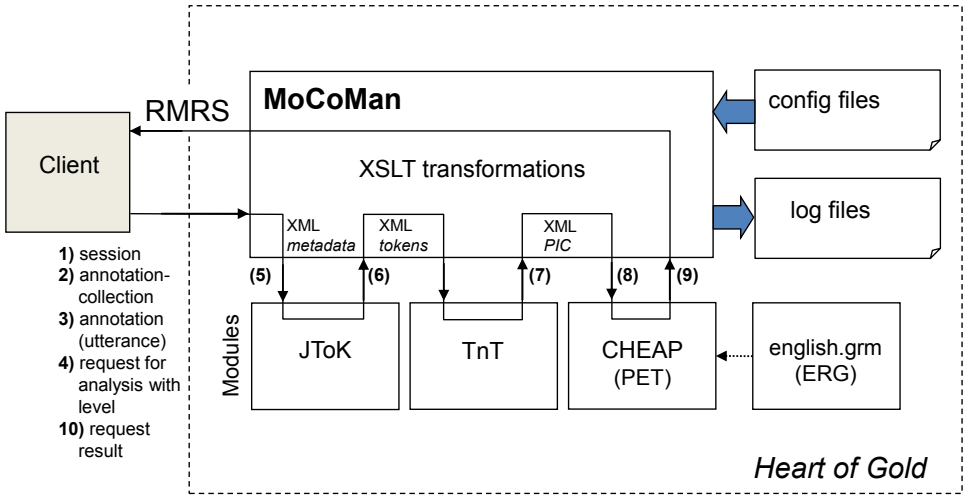
3.3.2 Natural Language Processing Infrastructure

The HoG system[15] has grown out of the project 'DeepThought'[16] which succeeded in combining linguistically shallow processing with deep processing with the goal 'to provide a system that combines different types of linguistic processing and that can be used for applications of different aims in a flexible way' (Callmeier et al. 2004). The driving factors are robustness and efficiency on the one hand and linguistically precise analysis on the other. 'Shallow processing enriches a text with Extensible Markup Language (XML) annotations (PoS, phrases, named entities, simple relations). Deep processing is only called at places where shallow analysis hypothesizes relevant relations but cannot detect or select the correct relations.' (ibid.)
Different NLP tasks require different levels of precision and it is extremely valuable to have one common infrastructure that provides the user or calling

[15] http://heartofgold.dfki.de
[16] http://www.dfki.de/deepthought

application with this flexibility.
The client of the HoG system can control how 'deep' the analysis should go and HoG invokes only those processing modules that are required to reach the required level. Moreover, HoG is an infrastructure that mediates between various NLP modules[17]. Which language is processed and by which module and level can be controlled by parametrization. Our prototype makes use of three modules only: JToK for tokenization (provided by the DFKI), TnT (Brants 2000) for stochastic PoS tagging and PET (Callmeier 2001) for the highly efficient HPSG-parsing using the ERG. It is technically straightforward to extend the prototype to other languages than English, provided there are HoG modules and grammars available.
The process flow at design time is relatively simple: the grammar is developed and functionally tested in LKB. Then it is compiled into a *.grm* file by the PET program *flop*. PET uses its own binary representation of the grammar including the lexicon for performance reasons. The process flow at run time is depicted in figure 3.17.

Figure 3.17: *Heart of Gold* Process Flow without ALR



The controlling process within HoG is the 'MoCoMan' server (Module Communication Manager). It is the interface to the client and it mediates between the linguistic processing modules. It converts the modules' output via extended style sheet transformations (XSLT) (Clark 1999) if required.

[17]It also provides an interface to XML databases that can be used for retaining annotations. This functionality is not used for the prototype.

A common result scheme is RMRS (Robust Minimal Recursion Semantics, Copestake 2007) to which some of the shallow outputs can be converted using XSLT, for example[18]. The configuration of the modules is in the control of this server, too. During processing the server writes protocol entries into various log files.
First the HoG client opens a session at the server (1), then it creates a so-called annotation-collection (2). Utterances are fed into HoG by creating 'annotations' (3). Analysis of the annotation starts when the client requests it (4). This is the time when the level of precision is passed to the server. HoG comes with a sample client written in Python[19]. Each annotation gets represented by an XML that contains some metadata and the outcome of the module that processed it (5). The metadata contain session and collection IDs, creation date/time, processing time and status information (diagnosis, error). The first module JToK tokenizes the utterance (6). The output is then sent to the TnT module which PoS-tags the utterance. TnT passes the result back (7) in a format which can be interpreted by PET: PET Input Chart (PIC, Waldron et al. 2006). The MoCoMan performs an XSLT to filter out metadata and passes the PET XML input chart (PIC) to *cheap* (8), the parsing program of PET. This program computes the RMRSs (9), which can be requested by the client (10).

3.3.3 Requirements 2-3: Making PET Ready for ALR

In order to satisfy requirement #2 PET's *cheap* program has to store the results of the enriched lexical seed feature structures in a format that can be processed by the LEARN module. Note that these feature structures are not included in the RMRS output. To implement a solution, we added a C++ printer class that saves feature structures into an XML file in an ISO format proposed by the Text Encoding Initiative (TEI) (see also Romary and TC 37/SC 4/WG 2 2006 and Lee et al. 2004). This class can be invoked by using various options:

[18] For RMRS on shallow grammars see Frank et al. (2004).
[19] www.python.org

Additional *cheap* (PET) Options	
TEI-mode:	0: MAPPEDLSEED and span only 1: all feature structures of top edges 2: all feature structures of all edges (good for debugging)
TEI-directory:	output directory (operating system path)
TEI-max:	maximum number of readings to be exported (default 1000)
TEI-gz:	save file in gzipped format (see www.gzip.org)

The options have to be specified in HoG's *pet.cfg* configuration file, for example:

```
pet.options=... -TEI-directory=/tmp/AILF -TEI-mode=0 -TEI-gz
```

The effect will be that *cheap* writes out one compressed (gzipped) XML file per utterance into the directory /tmp/AILF with maximum 1000 readings per file. The files will contain the features MAPPEDLSEED, CFROM and CTO only, which is sufficient for ALR. Note that compressing saves a considerable amount of disk space (and input/output operations) because the XML files are quite inflated. Figure 3.18 shows the first part (containing the first reading) of such a file for the utterance 'all my gnarfs'.

The root tag of the XML file is *<result>* with further attributes specifying the maximum number of readings (max), the actual number of readings (nr) and the export mode (given by option TEI-mode). Every reading is associated with a sub-XML rooted by *<r>* with the reading number as attribute. The feature structures included follow the TEI standard.
The name of the XML file is constructed in such a way that it can be related to the utterance (and the UtID) to which it belongs. It contains the session ID and the name of the annotation collection. To make the association with the utterance unequivocal there must be only one collection per utterance. The filename is derived from a variable that we call 'global context'. It is passed to *cheap* within the PIC input stream (cf subsection 3.3.4).

Requirement #3 demands that the base form of a token is made available to the feature structures during lexical lookup. We achieved this with by adding some C++ code to the lexical lookup routines and by adding the following line to the *pet/common.set* grammar configuration file:

```
morph-base-path := "TOKEN.MORPHBASE".
```

This entry has the effect that the morphological base form is stored under the TOKEN.MORPHBASE feature path.

3.3.4 Requirement 1: Making HoG Ready for ALR

According to requirement #1 the utterance ID (UtID) has to be looped through the infrastructure so that the LEARN module can associate enriched lexical seeds with the utterance from which it stems. In order to do as few modifications as possible the solution leverages on the circumstance that the session ID and annotation collection ID is already internally encoded in a HoG-specific URI (uniform resource identifier) following the pattern HoG://<Session ID>/<Collection ID>/<InputAnnotation>. So we just copied the XSLT file that is invoked for computing the PIC input for *cheap* and modified it so that the URI is integrated into the PIC XML. The new XSLT file *combinepixmlAILF.xsl* replaces the old one in the *pet.cfg* configuration file:

```
pet.combinestylesheet=xsl/pic/combinepixmlAILF.xsl
```

The generated PICs now look similar to the following fragment:

```
<?xml version='1.0'?>
<pet-input-chart
   context="hog://08082009-015823-6953.usecases.ailf/collection1/TnTpiXML">
...
```

Because both session ID and collection ID are stored as metadata in the resulting XML files, the file name of the related exported lexical seed files can easily be reconstructed. This solution is sufficient for a prototype and may be redesigned in future implementations.
The session ID can be specified by the calling HoG client. This is what the program *AILFanalyzeAll* does which we developed for experimenting with the prototype. It reads an input file which contains one utterance per line, prefixed with the UtID. The process flow of the modified system (the prototype) is shown in figure 3.19. The HoG client *AILFanalyzeAll*[20] creates a session with an ID that encodes the current date and time as well as the input file name (1). It then loops over the utterances in the input file: for every utterance it creates an annotation collection (2) and feeds the utterance (without the prefix) into it (3). Afterwards it requests analysis (4) with the highest level to ensure that PET is invoked (which counts as the deepest level). The utterance is processed as usual with the two differences that the session and collection IDs are encoded in the PIC as the 'global context' of the utterance (8) and that the feature structures are exported by *cheap* (9) before the RMRSs are passed back to the MoCoMan (10). The HoG client

[20]Note that multiple instances of *AILFanalyzeAll* can be started in parallel in order to scale well on multi-processor machines.

requests the result and stores it as XML file with a name that includes the UtID (11). This file contains the metadata that is later on required to locate the related XML file in the TEI output directory.

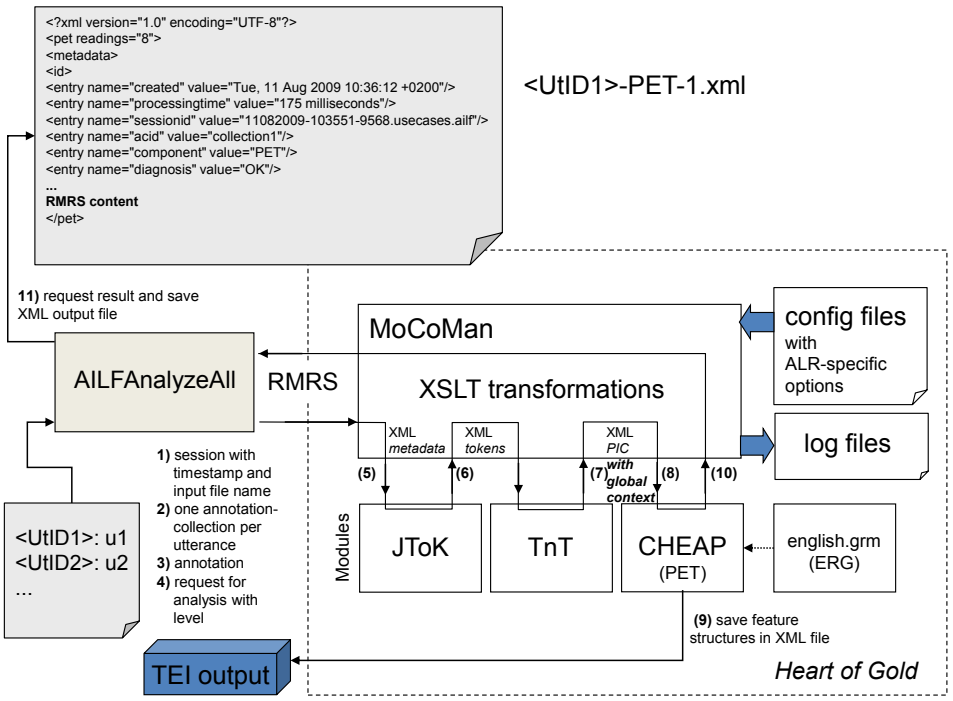
3.3.5 Accounting for Punctuation

Two further modifications have been done to enhance the overall quality. Firstly, in the preliminary tests we found that a) the enclitic *'d* and b) the contracted form *don't* was not properly handled by the default rules of the JTok tokenizer. We therefore added the regular expression '[Dd] to the specification in *en_ clitics.xml* and removed the expression '[Tt] from the same place.
Secondly, HoG does not account properly for ERG's treatment of punctuation. In ERG punctuation symbols are treated like suffixes. This mechanism does not work if punctuation symbols are separate tokens. As a preliminary modification we therefore slightly changed the HoG code which integrates the TnT tagger. By this change punctuation is still a separate token (which is per default ignored by PET) but in addition it is re-suffixed to the preceding word so that the ERG mechanism applies.

Figure 3.18: Example TEI Export


```
– <result max="1000" nr="8" mode="0" xml:id="hog://08082009-053041-11550.usecases.ailf/collection1/TnTpiXML">
  – <r nr="1">
    – <fs type="reg_nom_relation">
      – <f name="CTO">
          <string>"13"</string>
        </f>
      – <f name="CFROM">
          <string>"8"</string>
        </f>
      – <f name="MAPPEDLSEED">
        – <vLabel name="1">
          – <fs type="lseedmassnoun">
            – <f name="LSPRED">
                <string>"unknown_noun_pred"</string>
              </f>
            – <f name="LSSTEM">
              – <fs type="*cons*">
                – <f name="REST">
                    <symbol>*null*</symbol>
                  </f>
                – <f name="FIRST">
                    <string>"gnarfs"</string>
                  </f>
                </fs>
              </f>
            – <f name="LSPOS">
                <symbol>ls_pos_n</symbol>
              </f>
            – <f name="LSVAL">
                <symbol>ls_no_val</symbol>
              </f>
            – <f name="LSSC">
                <symbol>ls_sc_nosc</symbol>
              </f>
            – <f name="LSPP">
                <symbol>ls_pp_noprep</symbol>
              </f>
            – <f name="LSCOUNT">
                <symbol>-</symbol>
              </f>
            </fs>
          </vLabel>
        </f>
      </fs>
    </r>
```

Figure 3.19: *Heart of Gold* Process Flow with ALR



3.4 Module LEARN

Many of the details about the LEARN module have already been presented in section 3.1.3. The current section concerns the implementational details and will give some illustration of how to work with the module.
LEARN is fully implemented in SWI Prolog[21] and has been tested on openSUSE[22] Linux (the exact versions can be found in the appendix). The BELIEF DB is hosted in MySQL[23] to which SWI Prolog programs can be connected via the Open Database Connectivity (ODBC) interface. LEARN employs the SWI port of the ALE program (Carpenter and Penn 2001), which is a highly efficient attribute-value engine. LEARN reads grammar files written in TDL and converts the TDL specifications into ALE format. When the enriched lexical seeds (in MAPPEDLSEED) are loaded, then they are internally represented in ALE format.

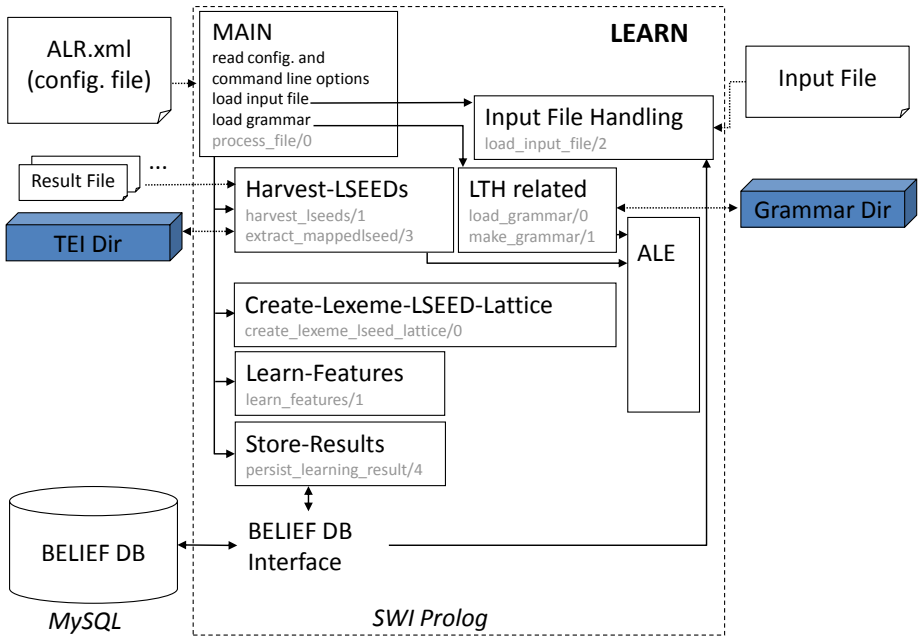
3.4.1 The Layout

The layout of the LEARN module is presented in figure 3.20, which contains some of the Prolog predicates in gray for reference into the source code. Note that it includes neither the create-table nor the display-references function. The module is controlled by the MAIN predicate. At startup it loads the configuration and command line arguments (if the module is run as an executable program, see subsection 3.4.3). Then it reads the input file that contains the UtIDs to be processed (which can be the same file as the one that is the input for *AILFanalyzeAll*). UtIDs that are already in the BELIEF DB are filtered out. It is assumed that the result files (produced by *AILFanalyzeAll*) are located in the same directory. Then it loads the grammar. For the sake of efficiency it suffices to read those TDL files of the ERG that contain the LTH and related type definitions (*fundamentals.tdl* and *lexseeds.tdl*). Because the compilation into ALE format is time consuming the compiled version is stored in a grammar directory (the path being a configuration parameter). It is only recompiled if at least one of the relevant TDL files is newer than the compiled version. For every utterance (every line) in the input file the module reads the corresponding result file and reads its metadata. Then it harvests the feature structures contained in the related XML file in the TEI directory (the directory path being a configuration parameter). Once the file has been successfully harvested, it

[21] http://www.swi-prolog.org/
[22] http://de.opensuse.org
[23] http://www.mysql.de

is renamed to <old file name>.learnt. This ensures that the file will not be read again by mistake. Note that the file contains all the relevant information: the reading numbers, the token spans, the MAPPEDLSEED content. The reading numbers are mapped to reading bit vectors (RBV). The tuples $\langle RBV, CFROM, CTO, EnrichedLSEED \rangle$ are temporarily asserted to the Prolog database. They will be retracted after the utterance is processed. The create_lexeme_lseed_lattice/0 predicate maps these tuples to a lattice as outlined in section 3.1.3. The result is again temporarily asserted to the Prolog database. The learn_features/1 predicate implements *learning down the hierarchy*. Its results are written to the BELIEF DB.

Figure 3.20: LEARN Program Stack



3.4.2 Interactive Mode

For testing and experimentation, the module can be run interactively in the SWI Prolog interpreter. The following dialog illustrates how one can interact with it[24]. First the SWI Prolog interpreter (*pl*) is started. Then the BELIEF

[24]Note that here and henceforth word-wraps have been inserted manually for better readability.

DB is mounted and the LEARNER tables are created. After that the LTH of the grammar is loaded. The trace messages (starting with 'T') tell what the system is actually doing.

```
$ pl -G100m -s /var/AILF/build/learn.pl
...
?- module(learn).
true.

learn:  ?- connect_to_beliefdb.
learn T080: CONNECT TO BELIEF DB
true.

learn:  ?- create_tables.
learn T082: INITIALIZING LEARNER TABLES
true.

learn:  ?- load_grammar.
% /var/AILF/grammar/ailf.cale compiled into ale 4.33 sec,
  19,625,108 bytes
true.
```

The module is now ready to process an input file. Here we load a file *demo* that contains one utterance only: 'all my gnarfs' (UtID is 'USECASE.1').

```
learn:  ?- load_input_file(demo,0).
learn T043: input file loaded
true.
```

Then we invoke the harvester (here and henceforth some of the output is masked with three dots for better readability). The predicate returns a triple that contains the UtID, the utterance and a boolean flag encoding whether the maximum number of readings has been reached.

```
learn:  ?- harvest_lseeds(U).
learn T060: Start processing of USECASE.1 at POS 0
learn T053: extracting from file ...11082009-103551-9568.usecases...
learn T000: TEI file loaded
learn T054: RN 1: asserting from 8 to 13: lseedmassnoun(...)
learn T054: RN 2: asserting from 8 to 13: lseednoun_with_plural(...)
...
learn T066: LSEEDs harvested
U = u('USECASE.1', 'all my gnarfs', false).
```

The asserted Prolog terms can be displayed using the *listing/1* predicate. The first argument of the term is the actual RBV, which can be translated to a Prolog list using *rbv_to_list/2*:

```
learn:  ?- listing(mappedlseed).
:- dynamic mappedlseed/4.

mappedlseed(34, span(8, 13), [gnarfs], lseedmassnoun(...)).
mappedlseed(80, span(8, 13), [gnarfs], lseedcountnoun(...)).
mappedlseed(132, span(8, 13), [gnarfs], lseednoun_with_plural(...)).
mappedlseed(264, span(8, 13), [gnarf], lseednoun_with_plural(...)).

true.

learn:  ?- rbv_to_list(264,L).
L = [3, 8].

true.
```

It can be seen that the lexeme with lemma `gnarf` is realized in readings #3 and #8. Now we create the lattice:

```
learn:  ?- create_lexeme_lseed_lattice.
learn T400: lattice created
true.

learn:  ?- listing(lseedlattice).
:- dynamic lseedlattice/4.

lseedlattice([gnarfs], span(8, 13), 2,
       [lseed(lseedmassnoun(...), 34),
        lseed(lseedcountnoun(...), 80),
        lseed(lseednoun_with_plural(...), 132)]).
lseedlattice([gnarf], span(8, 13), 2,
       [lseed(lseednoun_with_plural(...), 264)]).

true.
```

The lattice has combined the first three mappedlseed/4 terms (because of the same lemma `gnarfs` and same token span). The number '2' at the third position of the lseedlattice terms says that RDoA can not be better than 2. This is because two different potential lexemes occupy the same token span. The next step is 'learn-features':

```
learn:  ?- learn_features(u('USECASE.1', 'all my gnarfs', false)).
learn T200: start learning with MaxReached = false
learn T201: learning features for "[gnarfs]", span: span(8, 13), RDoAmin: 2
learn T204: result: [
  learnt(2, (lseedregularnoun,
              LSCOUNT:bool, LSPOS:ls_pos_n, LSPP:ls_pp_noprep,
              LSPRED:a_(string("unknown_noun_pred")),
              LSSC:ls_sc_nosc,
```

```
                LSSTEM: (*cons*, FIRST:a_(string("gnarfs")), REST: *null*),
                LSVAL:ls_no_val),
          246),
  learnt(inf, (lseedmassnoun, ...
  ...
]
learn T201: learning features for "[gnarf]", span: span(8, 13), RDoAmin: 2
learn T204: result: [
   learnt(2, (lseednoun_with_plural,
                LSCOUNT: +, LSNUMBER:ls_pl, LSPOS:ls_pos_n,
                LSPP:ls_pp_noprep, LSPRED:a_(string("unknown_noun_pred")),
                LSREGPL: +, LSSC:ls_sc_nosc,
                LSSTEM: (*cons*, FIRST:a_(string("gnarf")), REST: *null*),
                LSVAL:ls_no_val), 264),
   ...
learn T205: learning done
true.
```

The traces show that lemma `gnarfs` is processed first, then `gnarf`. The term with the functor 'learnt' has three arguments: RDoA_{max}, the feature structure that is learned and the RBV. Because **lseednoun_with_plural** is the only one found in the readings in which `gnarf` is realized, its RDoA_{max} equals 2 (which is the minimal RDoA according to the lattice). The situation for `gnarfs` is different. There are multiple different realization contexts (**lseedmassnoun** and others) found whose common denominator under RDoA_{max}=2 is **lseedregularnoun**.

The last step writes the learned feature values into the BELIEF DB. Feature values that are the most general within the context are not stored to avoid redundancy:

```
$ mysql -uailfuser -p... ailf

mysql> select * from lexemes;
+-------+--------+
| lexid | lemma  |
+-------+--------+
|     1 | gnarfs |
|     2 | gnarf  |
+-------+--------+
2 rows in set (0.00 sec)

mysql> select * from rcontexts;
+-----+-------------------------+
| cid | context                 |
+-----+-------------------------+
|   1 | lseedregularnoun        |
```

```
|   2 | lseedmassnoun            |
|   3 | lseedcountnoun           |
|   4 | lseednoun\_with\_plural  |
+-----+--------------------------+
4 rows in set (0.01 sec)

mysql> select * from learntfeatvalrefs;
+-----+-------+-------+-----+-----+---------+----------------+------+--------+------+
| uid | lexid | cfrom | cto | cid | feature | value          | RDOA | active | RBV  |
+-----+-------+-------+-----+-----+---------+----------------+------+--------+------+
|   1 |     1 |     8 |  13 |   1 |         |                | 2    |      1 | 246  |
|   1 |     1 |     8 |  13 |   2 |         |                | inf  |      1 | 34   |
|   1 |     1 |     8 |  13 |   3 |         |                | inf  |      1 | 80   |
|   1 |     1 |     8 |  13 |   4 |         |                | inf  |      1 | 132  |
|   1 |     1 |     8 |  13 |   1 | LSPP    | ls\_pp\_noprep | 2    |      1 | 246  |
...
```

3.4.3 Executable Program: Options

LEARN can be compiled to an executable program *ailf_learn* so that it can be invoked from the UNIX shell. It understands the following set of command line options:

LEARN Executable Program Options		
-I	-initdb	initialize LEARNER tables
-f	-file	learn from given input file
-p	-position	start at given input file position (useful for restarts)
-u	-unmark	unmark processed TEI files
-t	-trace	run with given trace level
-R	-rdoa	show references with given RDoA
-i	-inactive	show inactive references
-L	-lemma	show references for given lemma (only with -R)
-C	-context	show references for given context (only with -R)
-F	-feature	show references for given feature (only with -R)
-U	-utid	show learned context/features for given UtID (only with -R)

The following calls would have the same effect as the interactive dialog of the last subsection[25]:

```
ailf_learn -I

ailf_learn -uf demo -t 3 2> learn.log
```

[25] Trace messages are written to *stderr* and can be directed into a log-file.

The -R command line option is useful to query the LEARNER tables. The output is a semicolon-separated table. If nothing further is specified then all references are exported:

```
ailf_learn -R 2
Lemma;Context;UtID;Utterance
gnarf;lseednoun_with_plural;USECASE.1; all my gnarfs
gnarfs;lseedregularnoun;USECASE.1; all my gnarfs
```

3.5 Module REDUCE

Within the ALR framework, the REDUCE module is the decision maker. It is the one that is responsible for the creation of lexical entries derived by induction. Because induction involves a lot of calculation it is ALR's 'number cruncher', too. Technically, it is quite similar to LEARN: it is implemented in SWI Prolog, it reads the same configuration file and it uses ALE as its attribute value engine. As with LEARN, it runs both interactively (subsection 3.5.2) and as an executable program (subsection 3.5.3).

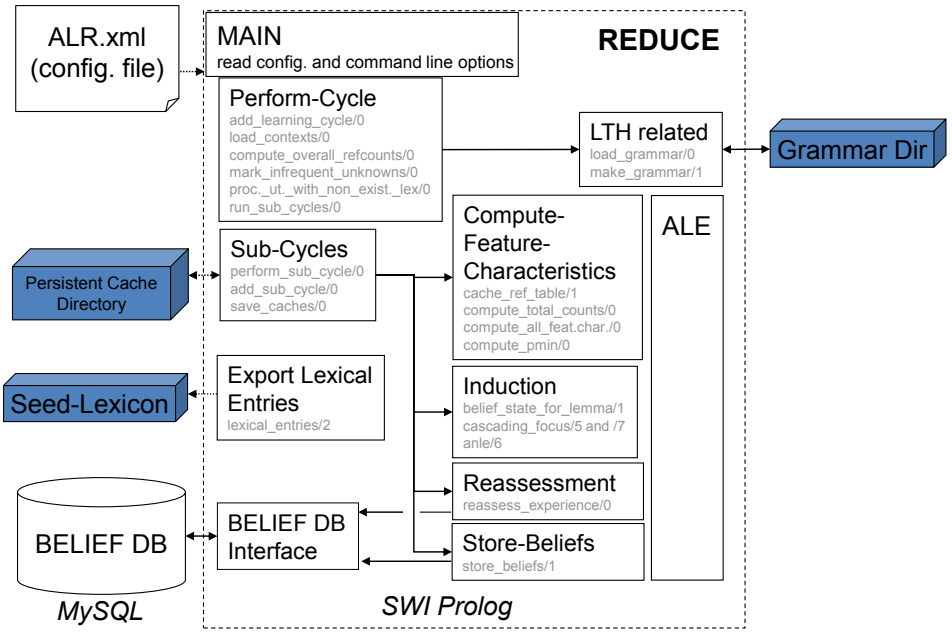
3.5.1 The Layout

The REDUCE functions (and Prolog predicates, respectively) are grouped according to figure 3.21. As with LEARN, main control is done by MAIN. Unlike LEARN, however, there is no input file. The module reads its data directly from the BELIEF DB. Because reading of the *learntfeatvalrefs* table as well as the calculation of reference counts is time consuming, REDUCE caches all data. For the cache the Prolog database (assert/1, retract/1) is used. The cache can be saved to disk ('persistent cache') and reloaded in subsequent sessions. This makes testing of the module more comfortable because it saves time and the cache files can be inspected with usual editors. The functionality of REDUCE is best demonstrated in an example dialog presented in the next subsection.

3.5.2 Interactive Mode

The following interactive dialog with REDUCE is based on the data generated by LEARN in course of the first bootstrapping cycle. The *learntfeatvalrefs* has around 130000 records by that time. At startup a couple of predicates are invoked that belong to the initialization group, e.g. *load_grammar/0*, *connect_to_beliefdb/0* and *load_context/0*:

Figure 3.21: REDUCE Program Stack




```
$ pl -G512m -L100m -s /var/AILF/build/reduce.pl
...
reduce T001: configuration candidates: [...]; mode: ;
hypotestmethod: bht; betafp: 0.05; betafn: 0.05;
pmin: 0.005; pmax: 0.5 pmed: 0.05; phigh: 0.158114;
cminref: 3; SNR: 5 minlemmacount: 5; lexmi: 2
% reduce.pl compiled into reduce 0.27 sec, 2,860,216 bytes
true.

?- module(reduce).
true.

reduce:  ?- load_grammar.
% /var/AILF/grammar/ailf.cale compiled into ale 4.66 sec, 19,624,972 bytes
true.

reduce:  ?- connect_to_beliefdb.
reduce T080: CONNECT TO BELIEF DB
true.

reduce:  ?- load_contexts.
```

```
reduce T020: create context hierarchy for lseed
reduce T027: lseed -> lseedbasicverb
reduce T020: create context hierarchy for lseedbasicverb
...
true .
```

During loading of the realization contexts the *rcontexs* table is loaded into the cache (implemented as a Prolog database). Afterwards the hierarchical relationships are computed and cached, too. We now create the REDUCER tables and add the first learning cycle and sub cycle to the BELIEF DB:

```
reduce:  ?- create_tables.
reduce T082: INITIALIZING REDUCER TABLES
true.

reduce:  ?- add_learning_cycle.
reduce T010: NEW LEARNING CYCLE: 1
true.

reduce:  ?- add_sub_cycle.
reduce T011: NEW LEARNING SUB CYCLE: 1
true.
```

Then the counters are calculated. At first for each lexeme (lemma) the number of utterances in which it was realized (*compute_ overall_ refcounts/0*) is computed. In the sequel all lemmas that are observed in less then n (default 10) utterances will be ignored. Then the active records of the *learntfeatvalrefs* table are read into the cache, which is a prerequisite for the subsequent calculation of the reference counts. The next step is the calculation of the so-called *total counts*: for each lexeme the number of utterances in which it is realized under a given RDoA_{max} is determined. The actual reference counts for all observed lexemes are then computed by *compute_ all_ feature_ characteristics/0*. Then the $p_{\psi}^{\rho=n}$ values (cf. section 2.8.6) are adjusted throughout the context hierarchy. Finally we save the cache to disk.

```
reduce:  ?- compute_overall_refcounts.
reduce T090: COMPUTE OVERALL REF COUNTS
true.

reduce:  ?- cache_ref_table.
reduce T091: CACHING LEARNT FEATURE VALUES
reduce T092: reading learntfeatvalrefs starting with UID=1
...

reduce:  ?- compute_totalcounts.
```

```
true.

reduce:  ?- compute_all_feature_characteristics.
...
true.

reduce:  ?- compute_pmin.
reduce T024: RDoA=1, lseedbasicverb PMIN: 0.00254681
reduce T024: RDoA=1, lseedregularnoun PMIN: 0.00245688
reduce T024: RDoA=1, lseeditrverb_noprt PMIN: 0.0289154
...
true.

reduce:  ?- save_caches.
reduce T070: saving cache: lcache
reduce T070: saving cache: realization contexts and hierachies
reduce T070: saving cache: totals
reduce T070: saving cache: feature characteristics
reduce T071: all caches saved
true.
```

For demonstration we may inspect some of the values computed for lexeme `time`. It can be seen that there is a quite smaller number of utterances in which the lexeme is unambiguously realized (RDoA_{max}=1) compared to the overall count:

```
reduce:  ?- lexid(time,ID).
ID = 84.

reduce:  ?- overall_count(84,Count).
Count = 3416.

reduce:  ?- totalcount(84,'1',Count).
Count = 681 .

reduce:  ?- totalcount(84,'2',Count).
Count = 2318 .

reduce:  ?- totalcount(84,'inf',Count).
Count = 3416.
```

The induction of lexical features starts with *belief_states_for_lemma/1*:

```
reduce:  ?- belief_states_for_lemma(time).
...
reduce T056: time: lseedmassnoun  believe 0 (stable)
reduce T057: time: lseedmassnoun  LSPP/ls_pp_oneprep, 'PP1':'_into_p_sel_rel'
```

```
                                    believe_not     0.0193882 (instable)
...
true
```

The results can be exported into a comma-separated-value (csv) file and written to the BELIEF DB:

```
reduce:  ?- export_beliefs('/tmp',time).
true.

reduce:  ?- store_beliefs(time).
true.
```

For completion of the sub-cycle the belief states for all lexemes would have to be computed and the experience reassessed. This should be done in batch mode, however.
Finally we inspect the corresponding lexical entries:

```
reduce:  ?- lexical_entries(84,LEs),
            msc_format_list(LEs,List), write_ln(List).
...
time_3 := lseedentry & [ stem <"time">, LSEED lseedmodnoun &
                         [LSPRED "_time_mod_n_pred"]].,
time_4 := lseedentry & [ stem <"time">, LSEED lseednoun_detless_pp &
                         [LSGOVPREP _on_p_rel,LSPRED "_time_n_pred"]].,
...
time_9 := lseedentry & [ stem <"time">, LSEED lseednoun_v_idiom &
                         [LSGOVVERB "_take_v_pred",LSPRED "_time_n_pred"]].,
...
time_11 := lseedentry & [ stem <"time">, LSEED lseednoun_with_plural &
                         [LSPP ls_pp_noprep,LSPRED "_time_n_pred",
                          LSREGPL +,LSSC ls_sc_nosc]].,
...
time_25 := lseedentry & [ stem <"time">, LSEED lseedmassnoun &
                         [LSPP ls_pp_noprep,LSPRED "_time_n_pred",
                          LSSC ls_sc_nosc]].
...
```

3.5.3 Executable Program: Options

As an executable program REDUCE can be started with the following command line options:

REDUCE Executable Program Options		
-I	-initdb	initialize REDUCER tables
-t	-trace	run with given trace level
-L	-lemma	induce features for the given lemma only

Chapter 4

A Bootstrapping Experiment

This chapter is dedicated to the presentation of a medium-scale bootstrapping experiment conducted with the prototype described in the last chapter, which is designed to facilitate various kinds of experiments on lexicon acquisition. The experiment comprises more than 10 lexical features of around 50 high-frequency lexemes tested with 170 thousand utterances.
The chapter is organized as follows: Section 4.1 outlines the motivation and goals of the experiment and section 4.2 its design. Some of its basic parameters had to be estimated in preliminary tests reported in section 4.3. The parameter values and lexemes chosen for the test are justified in section 4.4. The basic numbers and characteristics of the experiment, such as number of parsed utterances, number of beliefs etc., are summarized in section 4.5. Evaluation starts with section 4.6 which reports the results regarding lexeme detection. The subsequent sections 4.7 to 4.12 present the evaluation of the experiment with regard to the particular lexical features. The acquisition of some selected features is discussed in-depth. Section 4.13 concerns the question of whether learning in the experiment was effective. The final section 4.14 draws the conclusions from the observations.

4.1 Motivation and Goals

In general, experiments may vary in parametrization, candidate lexeme sets, investigated features, experience sets (corpora), the gold standard and the language model (cf. the discussion in subsection 1.3.7). The number of interesting test settings that may be realized within the implementation is vast and in principle allows for a thorough investigation of the quality of ALR. It must be noted, however, that systematic tests on all of the implemented

lexical features would be extremely time consuming (conduction as well as evaluation) and would require a book of its own. There are three further reasons why a comparison with extant work, particularly with *in vitro* accounts such as reported in Korhonen (2002), would be problematic and questionable at present: firstly, our approach differs from much of the related work in a crucial aspect: we do not try to solve the sparse data problem, quite the contrary. The Learn$-\alpha$ design rule does not demand that a system has to be able to infer lexical features from sparse data; it assumes instead that there is a sufficient number of learning opportunities available. The verbal subcategorization acquisition 'benchmark' published on the Internet[1], for example, has verbs in the test set for which there are not enough data in the BNC, the corpus that was chosen for our experiments[2]. This is partly due to the high underspecification of the lexicon we start with, which makes it too inefficient to analyze utterances with a token length greater than 12 (see section 4.2 below). Secondly, not all of the SCFs used in that research can be represented with the ERG and vice versa[3]. Thirdly, a mere comparison of precision/recall metrics would not foster a robust understanding of what is actually happening in the ALR system. This includes the exposition of references (cf. definition 23 on page 104) on the basis of which the system induces feature admissions. To gain an appropriate level of understanding, tests which analyze the actual feature value characteristics, noise scenarios, estimated noise levels and decisions made by the system are in order. Once this is achieved and potential shortcomings of the system are removed and more data and system resources are made available and there is a agreement on how to compare acquisition performance, it becomes interesting and worthwhile to work on large-scale comparisons. In summary, we think that it is far too early at this point of time for a meaningful comparison.

The experiments reported here aim at providing insights into the potentials and shortcomings of the Learn$-\alpha$ theory and ALR framework, respectively. The tests try to cover a broad range of lexical phenomena throughout the verbal, nominal and adjectival classes: lexeme detection (single words and MWEs), subcategorization, countability and number. They are driven by a fundamental question posed in the introduction (section 1.2): Can the lexicon be a natural fall-out and logical consequence of the system's a-priori

[1] http://www.cl.cam.ac.uk/~alk23/subcat/subcat.html

[2] According to Leech et al. (2001) the verb `slice`, for example, has a frequency of 15 in a million words.

[3] We agree with the remark in Schulte im Walde (2009) that '[...] one should only compare the outcome of approaches whose target frame types are sufficiently similar, and whose evaluation methods are comparable to a large extent.'

knowledge taken together with its experience?
In order to keep the number of assumptions regarding the content of the initial lexicon to a minimum, we start with a lexicon that does not contain more than part-of-speech classification of verbs, nouns and adjectives. The specifications for adverbs are taken over from the original ERG lexicon, because acquiring them is out of scope of this thesis[4]. The massive underspecification means that the experiment is done with a system that is still in the bootstrapping phase. Note that a completely empty open-class lexicon would make the experiments too costly, if not impossible. The PoS classification is taken from WordNet which suggests an extremely high level of recall for this feature.
We believe that for this first experiment a comparison with entries of the OALD is sufficient. Most of the features investigated here have formal correspondences in this dictionary[5]. Because we are testing with high-frequency lexemes, we expect a high coverage by the OALD, at least sufficient for evaluation. The situation in a broad-scale mass experiment with many lexemes would be probably different. Our tests may reveal that certain specifications in the gold standard are missing. On the other hand detailed analysis may show deficiencies in the grammar, too. Both would be interesting side effects indicating that ALR could also be used to improve the quality of a dictionary and the quality of a computational precision grammar.

In summary, this first experiment aims at assessing the quality of the ALR prototype and at shedding light on the question whether such a system can learn meaningfully and effectively despite massive lexical underspecification.

4.2 Design of the Experiment

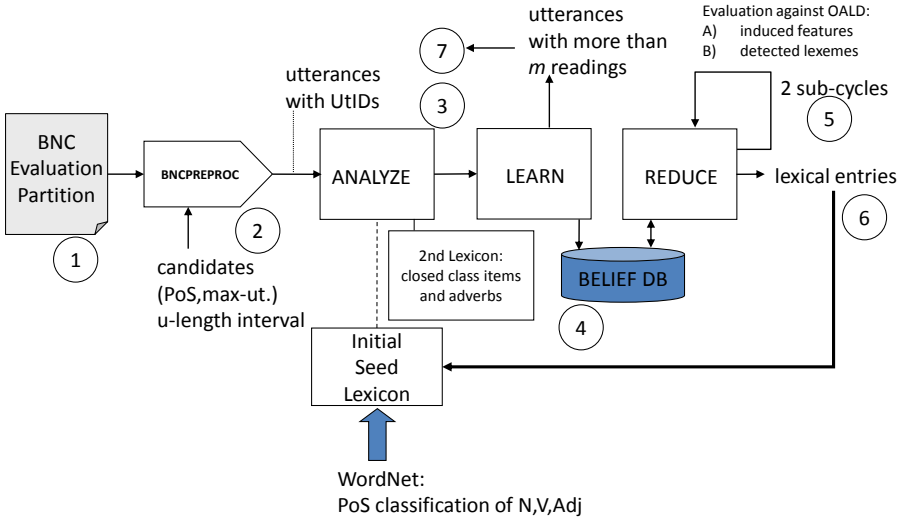
The basic layout of the experiment along with the sequence of action items is depicted in figure 4.1.

The corpus of the test is the XML-version of the BNC. It is divided into a test partition (1/5) for the development of the ALR prototype and an evaluation partition (4/5) for the experiment (step 1). We developed a program called *bncpreproc* with which utterances tagged with their unique ID (UtID) can be extracted out of the BNC. We print the UtID in this thesis in the form ⌊BNC-Text-ID:Number of utterance⌉. The program accepts a list of

[4]This includes some specifications for nouns that can be used adverbially, see section 4.2.

[5]Subsection 1.3.2, however, mentions the issue that the use of a dictionary as a gold standard is not ideal.

Figure 4.1: Layout of the Bootstrapping Experiment



lexeme candidates along with required PoS and an u-length interval $[a, b]$ as input. We use the notion *u-length* for the number of tokens in an utterance. Additionally a threshold M can be specified which limits the number of utterances per candidate. Moreover extraction of *headlines* can optionally be prevented. We made use of this option in all tests. An utterance is extracted if a) it belongs to the evaluation partition, b) it has a minimum number a and a maximum number b of words[6] and c) at least one of the candidate lexemes for which less then M utterances are extracted is realized in the PoS specified in the input. The actual parametrization (subsection 4.4) has to be done in such a way that there are a) enough learning opportunities and b) there are not much more utterances than required, which would lead to unnecessarily long run times.

The resulting list of UtID-utterance pairs is fed into the ANALYZE module (step 2). This module employs a lexicon that consists of two components: a lexicon with closed-class items and adverbs as in the original ERG lexicon and an initial seed lexicon. Nouns that can be used adverbially, such as in *I did it this way*, have corresponding entries in the first lexicon[7]. 144 of the

[6] We acknowledge that restricting the length of an utterance may statistically discount certain syntactic realizations such as sentential complements.

[7] These are two entries of *lseedmodnoun* for `time` and `way` and a couple of entries for

ERG adverbs were removed for the sake of the lexeme detection test reported in section 4.6.4. For the generation of an initial seed lexicon out of WordNet (version 3.0) we developed a program called *initial_seed_lexicon.pl*. This program exploits the PoS argument of the predicates listed in WordNet's file *wn_s.pl* in order to create seed entries of the following form:

```
...
car_n := lseedentry & [ stem <"car">, LSEED lseedbasicnoun & [ LSPRED "_car_n_pred" ]].
...
nice_a := lseedentry & [ stem <"nice">, LSEED lseedbasicadj & [ LSPRED "_nice_a_pred" ]].
...
go_v := lseedentry & [ stem <"go">, LSEED lseedbasicverb & [ LSPRED "_go_v_pred" ]].
...
```

Note that no synset information is used for the test. Adverbs, words in upper case, compounds, plural forms and closed-class items as well as *day*- and *day-part*-words are *not* mapped into the initial seed lexicon. The reason that compounds are not included is that we want the prototype to detect them.

The enriched LSEEDs exported by ANALYZE are fed into the LEARN module (step 3) which stores the results in the BELIEF DB (step 4). Utterances with more than m readings (a parameter of ANALYZE, see subsection 4.4) are filtered out and kept separately. Afterwards two sub-cycles are performed with the REDUCE module (step 5) and the belief states evaluated against the OALD.
Finally we update the seed lexicon with the exported lexical entries of the candidate lexemes (step 6) and run the ANALYZE module again (step 7) with the utterances that had too many readings in step 3. If learning is effective then a significant amount of utterances will have less than m readings.

4.3 Preliminary Tests

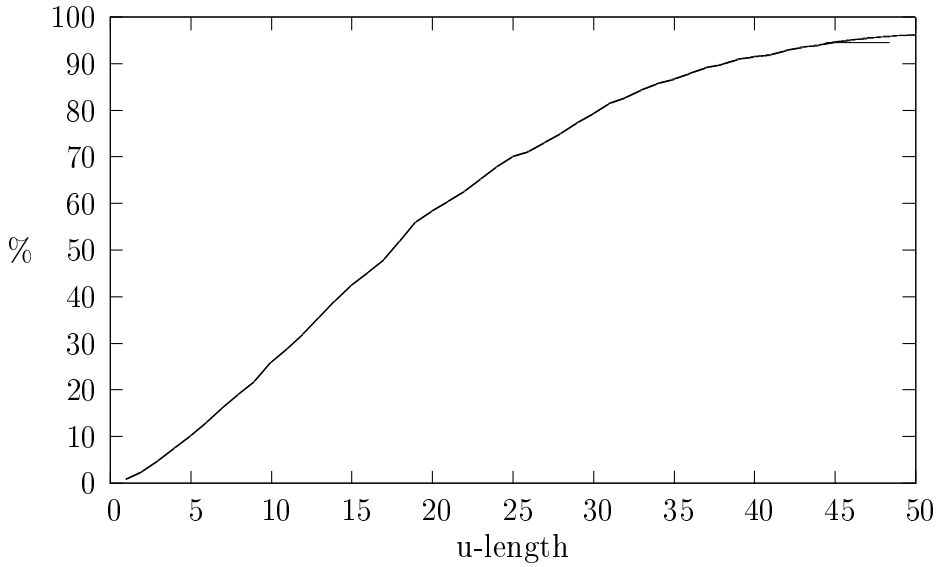
For the sake of a meaningful and efficient experiment its parameters must be chosen with care. This section concerns the outcome of a couple of preliminary tests which helped determine an appropriate parametrization. Most of the tests were done with a random sample of 1000 utterances extracted from the BNC test partition to which we refer by S1000$[a, b]$ meaning that the utterances are in an u-length interval of $[a, b]$.

day-nouns such as `day`, `week` and *day-part*-nouns such as `afternoon`.

4.3.1 BNC Basic Statistics

According to the BNC User Reference Guide the written portion of the corpus contains roughly 5 million s-units (corresponding to what we call utterances) and 88 million w-units (words)[8]. 90% of the s-units stem from written books and periodicals. The publication dates of 98% of the corpus texts are from 1960-1993, the remaining dates being unknown. The evaluation partition (4/5, no headlines) contains roughly 3.8 million utterances and 73 million words with an average of 19 words per utterance. Because the u-length has to be restricted in the bootstrapping scenario, it is valuable to know the distribution of utterances over their length. Figure 4.2 shows the distribution based on a S1000$[1, \infty]$ sample and up to an u-length of 50.

Figure 4.2: BNC Distribution of Utterances over Length



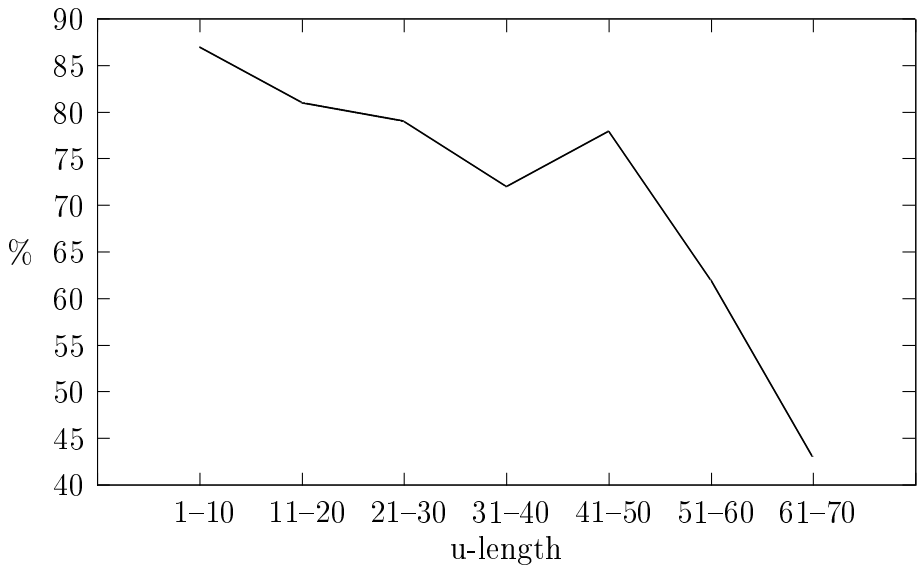
4.3.2 Parser/Grammar Performance

During the development of the prototype we experimented with different versions of the ERG and found that the version of February 2008 showed the

[8]The guide defines these notions as follows: an s-unit 'contains a sentence-like division of a text', a w-unit 'represents a grammatical (not necessarily orthographic) word'. Punctuation marks are called c-units: a c-unit 'contains a significant punctuation mark as identified by the CLAWS tagger'.

best performance in the context of the intended experiment[9] which is why it became the grammar employed in the prototype. In the first performance test with unrestricted u-length 79% of the utterances could be parsed. The average runtime was 1.7 sec. Later versions in the test showed considerably higher run times. Figures 4.3 and 4.4 show the parser success rate over u-length and processing time over u-length, respectively[10]. While the success rate is still above 70% up to 50 words, the average processing time crosses the 2 sec. boundary already after 30 words per utterance.

Figure 4.3: Parser Success Rate over U-length: Unmodified ERG



A test on a S1000[1, 30] sample yielded a parser success rate of 82.8% and an average processing time of 0.58 sec (approximately 6000 u/h).

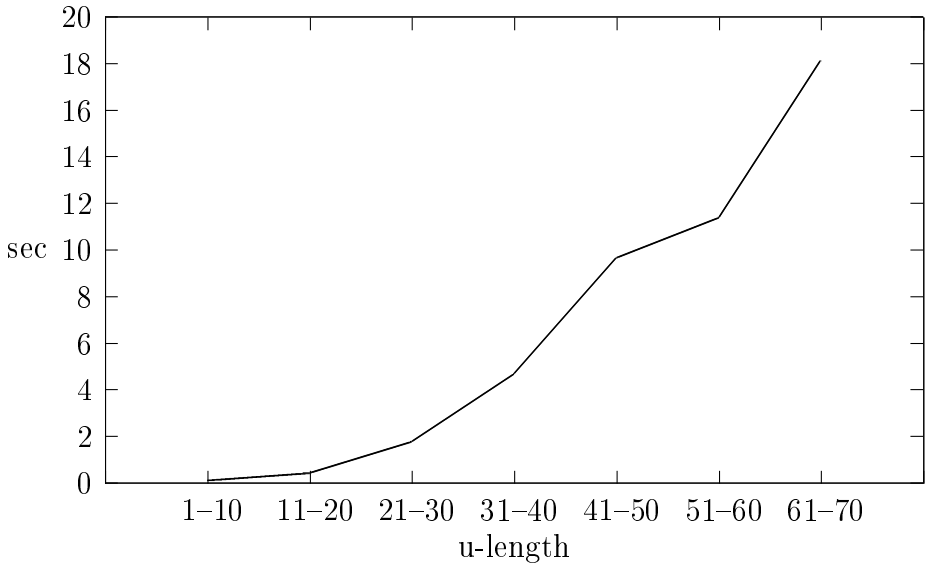
Because of the relatively minor coverage of the ERG lexicon (Baldwin et al. 2004, Zhang and Kordoni 2006) the remarkably high parser success rates could not be achieved without activating unknown word management[11]. In a test done with the BNC (1.8 million of sentences with maximally 20 words) and ERG version of June 2004, Zhang and Kordoni (2006) report

[9] In these tests the PET parser was configured for: max. 100k edges, mode: 20 best solutions, packing enabled, default types for unknown words.

[10] We measure the processing time as the sum of the processing times of all parsed utterances divided by the number of parsed utterances.

[11] This is done by running the parser with the -default-les option.

Figure 4.4: Parser Processing Time over U-length: Unmodified ERG



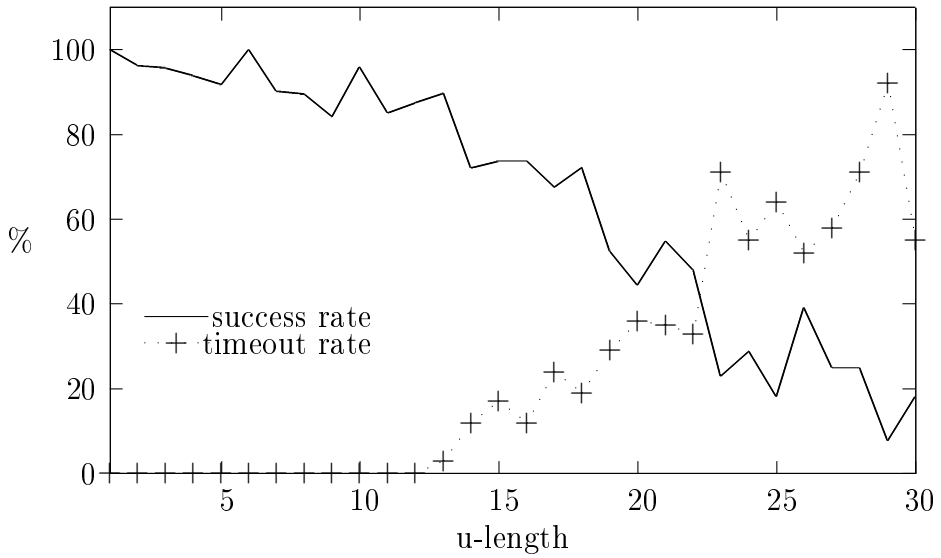
that roughly 70% of the input could not be parsed due to missing lexical entries.

We used the same S1000[1, 30] sample to measure the performance of the modified ERG with the initial seed lexicon. The result is shown in figures 4.5 and 4.6, respectively. This time the 2 sec. crossover point is already at 10 words/utterances and first timeouts occur after 12 words/utterances (the parser timeout was set to 30 sec.). The parser success rate was 68.2%; 20.8% of the utterances lead to timeouts. The average processing time was 2 sec.

Based on this result and because we did not want to feed too short utterances into the learning system we restricted all subsequent tests with the modified ERG to an u-length interval of [3, 12]. Note that according to figure 4.2 this interval still covers around 30% of the corpus.

In preparation for the bootstrapping experiment a test was done with a S9700[3, 12] sample. This time the parser was configured to compute up to 1000 readings. The success rate reached 91%; 84 utterances failed with timeout (0.9%). The average processing time was 1.9 sec. and 23.5% of the parsed utterances exceeded 1000 readings. Inspection of the learning performance during development revealed that the grammar has a too high degree of overgeneration during bootstrapping if the GENRE feature is left

Figure 4.5: Parser Success Rate over U-length: Modified ERG and Initial Seed Lexicon



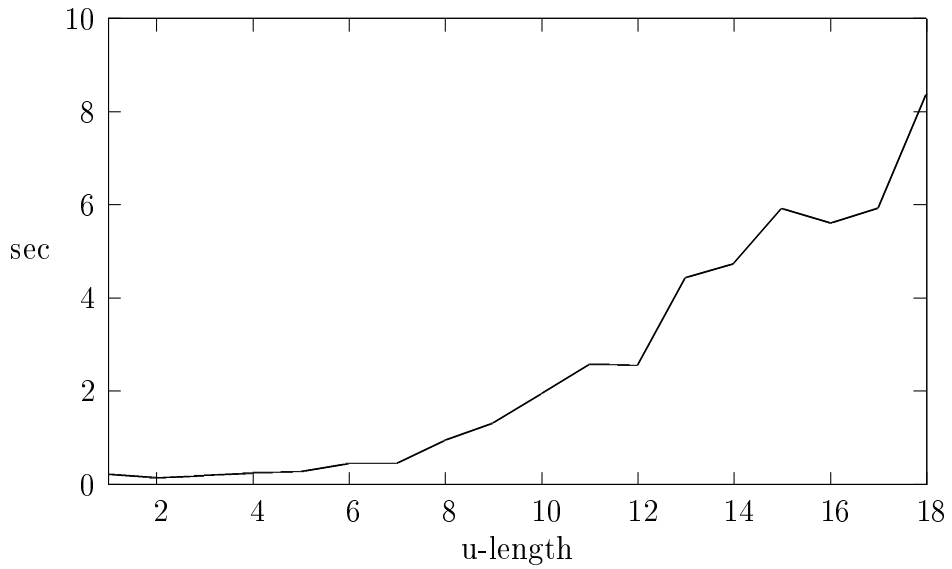
underspecified. For this reason we set [GENRE **formal**] in the lexical seed layer (subsection 3.2.4). In addition, for the sake of precision, we modified the ANALYZE module so that punctuation is no longer ignored (subsection 3.3.5). This lead to a decreased parser success rate of approximately 86%, but to a smaller number of utterances exceeding the 1000-readings threshold, namely just 10%. Processing is also faster: the average processing time was 1.1 sec. The results discussed so far are summarized in the following table. For comparison, performance measures are re-calculated for the u-length interval $[3, 12]$.

Table 4.1: Parser Performance in Preliminary Tests (U-Length = $[3, 12]$)

	Parser Success Rate	Avg. Proc. Time	# Ut. > 1000 readings
Unmodified ERG:	84%	84ms	-
Modified ERG / Initial Lexicon:	91%	1.9sec	23.5%
[GENRE **formal**] / punctuation:	86%	1.1sec	10%

Neutralization of lexeme determination (subsection 2.5.3) makes it worth-

Figure 4.6: Parser Processing Time over U-length: Modified ERG and Initial Seed Lexicon



while to estimate the number of references in which a given lexeme is unambiguously (RDoA_{max}=1) realized. Let us call the ratio of the number of RDoA_{max}=1 references divided by the number of input utterances the *overall yield* of the test. The theoretical minimum number of LOs divided by the overall yield gives the number of input utterances required per lexeme. In the course of development we found that approximately 80-90% of the references for a given lexeme are references for the unambiguous determination of the lexeme. The overall worst-case yield of the last preliminary test is thus $0.86 \cdot 0.9 \cdot 0.8 = 0.62$.

4.4 Parameters and Test Lexemes

This subsection will discuss the set of test lexemes and main parameters chosen for the experiment. The base for most of the decisions is the choice of β_{FP} and β_{FN}. We set both parameters to 5% because this is a threshold which is often used in the literature and because, as we found during development, more ambitious levels would require more data than we get out of the BNC. Two further crucial parameters are p^{fv}_{min}, p^{fv}_{max} and SNR (signal-to-noise ratio) which determine the sensitivity of the system. We found that 0.5%, 50% and 5, respectively, are reasonable values. The latter means that the 'signal' (the observed feature value realizations) is expected to be at least 5 times the corresponding noise level. Most of the 'rare' feature values are in the order of 0.1% to 1%. We want to emphasize again that p^{fv}_{min} is *not* a threshold. It is a parameter which influences the setup of the hypothesis test (cf. the paragraph on page 113 and subsection 2.8.6). The value for ϵ_{max} amounts to 3.16% (cf. subsection 2.4.7).
Given these parameters the minimum number of learning opportunities is 2100 (cf. subsection 2.8.6). The actual number of input utterances must be greater because there is a) a portion of utterances that cannot be parsed, b) a portion of utterances that have too many readings so that they will be ignored by LEARN and c) a portion of utterances in which the realization of a given lexeme is ambiguous. As preliminary tests (section 4.3) revealed that the overall yield is around 60%, this means that at least 3500 utterances per lexeme are to be extracted from the BNC evaluation partition. This leads to the restriction that only high-frequency lexemes come into consideration for this experiment.
Word frequencies based on the BNC have been reported in Leech et al. (2001). Because relative frequencies vary with the u-length, however, we have counted the occurrences of the most frequent lexemes in the evaluation partition and compiled the following list of test candidates[12]:

[12]Note that we only consider the main part-of-speech of the candidates.

PoS	Test Lexemes
Nouns:	area, case, child, hand, house, life, man, number, part, people, place, problem, thing, time, way, woman, work, world, year
Verbs:	ask, become, come, feel, find, get, give, go, know, leave, look, make, mean, need, put, say, see, seem, show, take, tell, think, use, want, work
Adjectives:	good, great, new, old, other

These are 48 lexemes (25 verbs, 19 nouns and 5 adjectives, `work` being member of two PoS classes). The main parameters are listed in the following tables:

Module/Program	Parameter	Value
bncpreproc	*u-length*:	[3,12]
	min. ut. per lexeme:	3500
ANALYZE	*PET timeout*	30s
	max-readings	1000
REDUCE	*hypotestmethod*:	BHT
	betafp:	0.05
	betafn:	0.05
	pmin:	0.005
	pmax:	0.5
	lexmi:	2
	cminref:	3
	SNR:	5
	minlemmacount:	5

4.5 Basic Numbers

This section presents the basic results that came out of the experiment, such as processing times and the overall yield. The numbers can be directly related to the process depicted in figure 4.1 on page 265.
The initial lexicon, to begin with, was populated with 70946 entries (60% nouns, 28% adjectives and 12% verbs). From the original ERG lexicon 27415 definitions have been removed and 4913 retained.

The *overall yield* discussed in section 4.3.2 resulted as follows: from the 1.1 million utterances with the u-length interval of $[3, 12]$ 169767 (15.2%)

were extracted by *bncpreproc*. 145889 (86%) of these could be parsed and 16261 (11%) of the parsed utterances had more than 1000 readings. 129504 utterances (76.3% of the input set) were stored in the BELIEF DB. The ratio of references under RDoA_{max}=1 to all references was 88.6%. The overall yield was thus 60%.

The LEARN module stored 58504 (potential) lexemes (lemmas) in the *lexemes* table of the BELIEF DB and populated the *learntfeatvalrefs* table, which stores the single observed feature values, with 7132968 rows (55 rows per utterance on average). In the course of disambiguation by deactivating readings with infrequent unknown lexeme candidates 118753 rows (1.7%) were deactivated affecting 26261 utterances (20%). Further, LEARN stored 411 realization contexts in the *rcontexts* table.

The recoverability factors (cf. the paragraph on page 185) listed on page 275 inform about the lexical neutralization with regard to selected realization contexts. Recall that RF_1 represents the neutralization of $\rho = 1 \rightarrow 2$ and RF_2 the neutralization of $\rho = 2 \rightarrow \infty$. The higher the values the higher the degree of neutralization. If the value is 1 then no neutralization takes place. A value of ∞ means that no observation is made on the more precise RDoA_{max} level. A value between 1 and 20 can usually be seen as a moderate degree of neutralization.
Three examples should illustrate how this table is to be interpreted. Firstly, there is a moderate degree of neutralization for countability and number. Most of the neutralization takes place for mass nouns because this context has to be inferred syntactically whereas countability and number can usually be detected on the morphological level. Many syntactic contexts are ambiguous with regard to the mass/count distinction (cf. the paragraph on page 138).
Secondly, there is not much neutralization for verbal transitivity.
Thirdly, regarding VPCs, there is only moderate neutralization of transitive VPCs with *up* whereas the intransitive counterpart is totally neutralized. This is because the readings of the intransitive VPC always competes with the directional/adverbial reading of the adverb.

Within the REDUCE learning cycle 2 sub-cycles were executed in order to measure the effect of retroactive disambiguation. The statistics of the *learntfeatvalrefs* gives information about this effect. We simply count the number of active rows depending on the respective RDoA_{max}-level:

Table 4.2: Recoverability Factors of Selected Realization Contexts

Realization Context	Super Context	RF_1	RF_2	Section
lseedregularnoun	lseedbasicnoun	1.0	1.1	-
lseedcountnoun	lseedregularnoun	1.5	1.4	4.7
lseedmassnoun	lseedregularnoun	15.1	2.2	4.7
lseedhoun_with_plural	lseedcountnoun	0.7	1.8	4.7
lseednoun_detless_pp	lseednoun_mwe_part	∞	1.9	4.6.5
lseedverb_nosc, [LSVAL **ls_intr**]	lseedverb_nosc	1.9	1.9	4.11
lseedverb_nosc, [LSVAL **ls_trans**]	lseedverb_nosc	1.0	1.4	4.11
lseedmtrverb	lseedverb_nosc, [LSVAL **ls_trans**]	1.4	1.3	4.11
lseedditrverb	lseedverb_nosc, [LSVAL **ls_trans**]	2.1	1.7	4.11
lseedverb_prt	lseedmweverb	∞	1.0	4.10
lseedverb_prt_nosc	lseedverb_prt	1.0	1.0	4.10
lseeditrverb_prt	lseedverb_prt_nosc	3.1	1.9	4.10
lseedmtrverb_prt	lseedverb_prt_nosc	0.4	2.1	4.10
lseeditrverb_prt, [LSVPRT **_up_p_sel_rel**]	lseeditrverb_prt	∞	0.7	4.10
lseedmtrverb_prt, [LSVPRT **_up_p_sel_rel**]	lseedmtrverb_prt	5.2	1.1	4.10

Sub-cycle	$\rho = 1$	$\rho = 2$	$\rho = \infty$	Total	Ratio Active/All
1	1261578 (21%)	1429455 (23%)	3442662 (56%)	6133695	86.0%
2	1275243 (22%)	1356247 (23%)	3156948 (55%)	5788438	81.2%

Three observations can be made: firstly, more than half of the observations are on the $\text{RDoA}_{max}=\infty$ level, a sign of the high indeterminacy during bootstrapping. Secondly, after sub-cycle 1 already 14% of the table rows were deactivated, after the second sub-cycle around 4%. Thirdly, because of disambiguation there is a moderate shift to the more precise levels.
During reassessment of experience not only disambiguation takes place but also entire utterances can be considered ungrammatical (lacking any reading). This happens if there are $\text{RDoA}_{max}=1$ instances which are considered noise. Consequentially, there is some 'loss' in the experience set. The following table shows the actual quantities of utterances with active readings:

Sub-cycle	Before Reassessment	After Reassessment	Loss
1	129504	126130	2.6%
2	126130	123665	2.0%

After the second sub-cycle there is a total loss of 123665/129504=4.5%.

Coming to the number of induced beliefs, it can be firstly noted that there are more constraints then admissions and secondly that the number of beliefs decreases during the learning cycle:

Sub-cycle	Constraints	Admissions	Undecided	Total
1	20226	6589	7453	34268
2	4232	4338	1866	10436

The reduction is due to the effect that constraints are stored only once, which means that feature values that are ruled out in sub-cycle n do not show up in the statistics of sub-cycle $n+1$. The induction of constraints also leads to a reduction of admissions due to the smaller number of admitted realization contexts.
The table on page 278 shows that the majority of inductions is made on the $\text{RDoA}_{max}=\infty$ level and that during *reassessment of experience* inductions shifted towards the $\text{RDoA}_{max}=1$ level, both for constraints as well as for admissions. Moreover, it can be seen that feature value admissions get more

stable whereas the stability of constraints does not change much.
The numbers also show that no constraint is imposed under $\rho = 2$: Once the focus of the system is relaxed, constraints are induced on the $\rho = \infty$ level (see the procedure of *cascading focus* in subsection 2.8.8).

The run times of the various modules were as follows: 4090 u/h for ANALYZE running on 2 CPUs, 9100 u/h for LEARN running on 2 CPUs and 13000 u/h for REDUCE.

4.6 Lexeme Detection

This section discusses the performance of the system with regard to lexeme detection (see also section 2.7). Lexeme detection is the process of including a previously unknown single or multiword item into the updated lexicon. The detection of *single*-word lexemes is based on the integration of PET's unknown-word handling (cf. subsection 3.2.5). Multiword lexemes are detected by harvesting enriched lexical seeds parallel to feature acquisition.
A new multiword item is not necessarily semantically idiosyncratic. Because ALR lacks the capability of inferring lexical semantics the system has to assume a separate sense for each detected multiword lexeme.
The relevant system parameters are *lexmi*=2 for noun compounds and *minlemmacount*=3 as the frequency cut off for single word lexemes. Lexemes are determined with RDoA_{max}=2.

4.6.1 Detection of Nouns

The detection of nouns comprises single word nouns and noun compounds. We discuss single word nouns first. Because of the broad coverage of WordNet, it can be expected that there will be only a few single noun true positives. This is indeed borne out. The following table lists the acquired nouns grouped by various categories. Frequencies are given in brackets for selected categories. Apart from the few true positives, which have entries in the OALD, there are some orthographic variants of nouns listed in WordNet, one unclear case, a set of named entities, a couple of abbreviations, some nouns that have been missed to be stored in the initial lexicon[13] and some missing closed class items. There is relatively little noise: some forms that may be called deviant English and words of other PoS classes. Abbreviations can be found in multiple PoS classes (see next subsection), which indicates

[13]For example the named entity `Jesus` is listed as a noun in OALD and WordNet.

Table 4.3: Statistics of Induced Beliefs

Sub-cycle	$\rho = 1$	$\rho = 2$	$\rho = \infty$	Constraints $\rho = 1$	$\rho = 2$	$\rho = \infty$	Admissions $\rho = 1$	$\rho = 2$	$\rho = \infty$
1	1738 (5%)	3657 (11%)	28873 (84%)	285 (1.4%)	0	19941 (98.6%)	1315 (20.0%)	2076 (31.5%)	3198 (48.5%)
2	1723 (17%)	1448 (14%)	7265 (70%)	172 (4.1%)	0	4060 (95.9%)	1416 (32.6%)	928 (21.4%)	1994 (46.0%)

				stable	likely	instable	stable	likely	instable
1				17 (0.08%)	13603 (67%)	6606 (33%)	1249 (19%)	4389 (67%)	951 (14%)
2				29 (0.69%)	2838 (67%)	1365 (32%)	1370 (32%)	2582 (60%)	386 (9%)

that unknown word management for them may be difficult.

Category	Total	Lexemes
True Positives:	2	media (21), carer (5)
Orthographic variants:	12	no-one (30), co-operation (13), grown-up (11), summat (11), dressing-gown (8), set-up (8), build-up (6), T-shirt (5), vice-president (5), steering-wheel (5), co-operative (5), back-up (5)
Missing irreg. form:	1	grandchildren
Unclear (not in OALD):	1	stepdad (5)
Named Entities:	85	1930s, 1970s, 1970, 1980s, 1980, Jesus (noun in OALD), Jesu, Profitboss, Menzy, Menzie, Menzies, Luke, Leed(s) German, Germans, Japanese, American, Americans, British, French, Chinese, ...
Abbreviations:	14	pp, PC, UN, sq, Q.B(.), incl, hon., mps, mp, eqn, IUD, BR, HIV, ph / pH
Missing in Lexicon:	8	whole (304), cry (112), front (72), amount (47), Jesus (35), ma'am (9), con (8), amounts (7)
Deviant English/foreign words:	17	(')ere, nothin('), goin('), (')im, dae, fe, fer, dat, yu, hir (?), comin('), y'know, 'ave, thoo, gonna, nowt, Monsieur
Other PoS:	9	hardest, happiest, gloomily, overleaf, testily, uncertainly, twenty-three (14), twenty-five (17), twenty-eight (6)

Because named entity recognition is not built in yet, it would be unfair to calculate precision. Leaving named entities and the unclear case out of the statistics and counting abbreviations, deviant forms and words of other PoS classes as false positives, the precision is: 23 / 63 = 37%.
In order to test the impact of RDoA_{max} we tested lexeme detection with $\rho = 1$. While the lexemes `hardest`, `gloomily` and `testily` were ruled out one of the TPs (`carer`) was ruled out, too. This is because the plural forms lead to ambiguity: `carers` could be a potential lexeme. Here are the references for this lexeme:

⌊B0W:573⌉	- Informal carer became ill - 1	$\rho = 1$
⌊CGD:1269⌉	There seems to be a changing pattern among carers.	$\rho = 2$
⌊EA1:198⌉	Setting up services that carers don't want is obviously wasteful.	$\rho = 2$
⌊B0W:844⌉	The kind of things the carers said were:	$\rho = 2$
⌊CAP:167⌉	Carers and families will also save time.	$\rho = 2$

As a consequence, a more precise focus seems to decrease recall.

Unlike single word nouns, noun compounds have been fully excluded from the initial lexicon, which means that they belong completely to the realm of unknown and potential lexemes. Three factors make up the acquisition of multiword nouns: firstly, new lexical seeds are created when the ERG rule responsible for noun-compounding is applied (see subsection 3.3.1, page 239). These are harvested by LEARN and stored in the BELIEF DB. Secondly, REDUCE selects those lexeme candidates that have a frequency greater than *cminref*, which equals 3 in the experiment. Thirdly, a pointwise MI score is computed for the selected candidates. Those candidates with a score less than 2 (*lexmi*) are removed. As with the other lexemes, it is not tried to acquire the lexical semantics of the candidates nor the semantic relation between the compound parts. We are merely interested in the question of how many of the retrieved candidates are actually listed in the gold standard (the OALD) and whether a setting of *lexmi*=2 is an effective choice. It must be noted that in preliminary experiments we found pointwise MI outperforming LRT.
In the experiment REDUCE selected 393 candidates out of a total of 15545 (potential) compounds. 80 of them were listed in OALD (precision of 20.4%). Inspection of the candidate set revealed that many candidates are familiar combinations like *tourist attraction*. Only 3 of the candidates consisted of more than 3 nouns: *like old time* (not a noun), *good old days*, *country house hotel* (both not listed in the OALD). The highest MI score of the 2-word compounds was 8.1.
Here are examples for illustration. The MI scores are given in brackets:

Category	Total	Lexemes
True Positives:	80	magic wand (8.1), nursery rhyme (7.8), killer whale (7.3), senior citizen (7.1), fire extinguisher (7.0), detention centre (6.8), prime minister (6.7), estate agent (6.5), opinion poll (6.3), ..., council house (2.3), open air (2.2), little finger (2.1), life insurance (2.1), old age (2.0), shelf life (2.0)
False Positives:	313	communist novelist (8.3), liberal democrat (8.2), hon. gentleman (8.1), newsletter envelope (7.8), pink champagne (7.3), ..., lot worse (2.0), sensitive area (2.0), whole thing (2.0), great pity (2.0), see recipe (2.0)

To get a feel for the impact of the MI threshold on precision, we simulated lexeme detection with higher thresholds. This yielded the following results:

MI Threshold	Recalled Lexemes (OALD)	Precision
2	80	20.4
3	62	26.2
4	40	32.8
5	22	33.8

The table shows that a threshold higher than 2 leads to increased precision at the price of significant drop of recalled lexemes. A threshold greater than 4 does not significantly increase precision.

Many non-lexeme compounds (e.g. *pink champagne*) stem from one text only. So we tested whether precision can be increased by imposing the constraint that a lexeme must occur in at least three different texts. The result was a slight increase of precision (22.8%) at the price of 6 fewer true positives.

4.6.2 Detection of Verbs

Similar to the single word nouns it was not expected to find many true positives of unknown verbs. From 23 lexeme candidates, one lexeme has been detected: `cry`, because it has been missed to be included in the initial seed lexicon. The contracted form *shan't*, which is missing in the closed class lexicon has been found 9 times. Some of the candidates are deviant English (*goin'*), the majority is noise.

4.6.3 Detection of Adjectives

As with single word nouns and verbs, not too many new adjectives have been expected with two exceptions. Firstly, because upper case words have not been mapped into the initial lexicon, geographical adjectives belong to the unknown lexemes. Secondly, as comparative and superlative forms are not productive in ERG (they are directly listed) and because WordNet does not contain them, they were missing, too. Apart from this no new adjective was detected. The next table lists the various categories of lexeme candidates:

Category	Total	Lexemes
New adjectives:	0	-
Orthographic variants:	4	o.k. (13), grown-up (12), non-existent (6), co-operative (5)
Inflected forms:	21	brighter, calmer, colder, easier, finest, happiest, hardest, heavier, highest, newer, newest, oldest, poorer, safer, safest, simplest, slower, smallest, stronger, weaker, youngest
Upper case forms:	23	American, African, Arab, Asian, Australian, British, Chinese, Dutch, English, European, French, German, Greek, Indian, Irish, Italian, Japanese, Jewish, Masai, Olympic, Romanian, Russian, Scottish
Abbreviations:	4	UN, pp, eqn, IUD,
Missing in Lexicon:	3	whole (304), former (31), sometime (12)
Deviant English/foreign words:	7	tae, (')im, goin('), comin('), y'know, fer, Monsieur
Other PoS:	24	Jesus, no-one, Gazzer, Luke, London, summat, uncertainly, Jinny, ma'am, set-up, dressing-gown, Vern, Ricky, Elsie, T'ang, Gurder, gloomily, Ave, overleaf, dryly, back-up, vice-president, twenty-three, twenty-five

If the abbreviations, the deviant forms, and the words that belong to other PoS classes are considered noise (false positives) then we arrive at a precision of $51/86 = 59\%$.

4.6.4 Detection of Adverbs

The test of adverb detection differs from the previous tests, because this PoS class is not taken from WordNet. Instead, many adverbs are supposed to be unknown given the coverage of the ERG. Moreover, 144 adverbs of the

original lexicons were removed for the sake of a recall-based test. With the configuration chosen for the experiment, the system detected 62 lexeme candidates, of which 19 were false positives. This yielded a precision of 72%. For the recall-based test, we increased the system's focus to $RDoA_{max}$=1 and decreased the frequency cutoff to 3. This resulted in 123 true and 13 false positives, a precision of 90%. One inflected form has been retrieved: `highest`. The following table lists the selected lexeme candidates:

Category	Total	Lexemes
Not in ERG:	80	everywhere, sometime, somewhere, anywhere, accusingly, aerobically, agitatedly, aloud (adjective in ERG), blankly, bluntly, brusquely, chronically, confidently, contemptuously, conversationally, crossly, defensively, defiantly, delicately, diagrammatically, disconcertingly, disproportionately, drastically, dutifully, enquiringly, expectantly, fearfully, foolishly, frighteningly, frightfully, gleefully, gloriously, grudgingly, hesitantly, highest, hugely, hurriedly, idly, incredulously, indefinitely, independently, intently, intermittently, irritably, keenly, ludicrously, minutely, persistently, prematurely, proportionately, protectively, psychologically, quizzically, ruefully, sarcastically, sheepishly, shrewdly, shyly, slyly, sorely, sparingly, sternly, succinctly, suspiciously, sympathetically, terminally, testily, tirelessly, triumphantly, truthfully, uncertainly, uncharacteristically, uncontrollably, undeniably, understandably, vehemently, visibly, warily, woefully, wryly
In ERG, but removed:	43	adequately, amiably, annually, appropriately, conceivably, consistently, curiously, currently, desperately, expressly, fondly, graphically, grimly, happily, heavily, ideally, impeccably, innocently, lazily, locally, loosely, narrowly, neatly, oddly, painfully, passionately, positively, potentially, proudly, publicly, realistically, remarkably, scornfully, sensibly, severely, socially, solemnly, successfully, traditionally, tremendously, uneasily, unquestionably, wisely
False Positives:	13	Bernice, Crilly, Emily, Hoomey, Jinny, Markby, Otley, nt (?), nae, no-one, Peggy, Ramsay, Ricky

80 (65%) of the acquired adverbs are not listed in the original ERG lex-

icon[14]. From the 144 adverbs removed from the ERG lexicon, 43 have been recalled (30%).

4.6.5 Detection of Determinerless PPs

Determinerless PPs form one class of *paradigmatic MWEs* discussed in subsection 2.7.2. Members of this class can have a fully compositional meaning (recall the example of *at sea* in Baldwin et al. 2003b) or an idiomatic interpretation, which is why we cover the related tests under lexeme detection. The OALD lists these PPs as idioms regardless of their compositionality[15]. Note that they must be carefully set apart from larger non-PP idioms such as *man to man*[16], *year by year*, PP idioms such as *by way of* or verbal idioms such as *fall/slot into place* because the determinerless PP is restricted to the particular idiom. They must also be distinguished from the productive and complex PP construction *from* $\overline{N}$ *to* $\overline{N}$[17]. In addition, it must be emphasized that the OALD does not only list idioms that are morpho-syntactically marked, but also those idioms that are not, for example (*by numbers*, *at work* or *in time*). This is important because this dictionary is the gold standard here. As ALR may disfavor the idiomatic interpretation of the unmarked determinerless PPs there may be a principle mismatch. In other words the setup of the gold standard depends on the interpretation of the lexical feature which represents the noun's option of being used determinerless in a PP.

In the current LTH determinerless PPs are represented by the lexical seed type **lseednoun_detless_pp**. A noun is licensed to be projected into the structure [P $\overline{N}$] if it is listed with this type and if the feature LSGOVPREP introduced by this type takes the value P.

With regard to the outcome of the experiment the following observations were made. Firstly, the system retrieved far fewer references than we intuitively estimated. A closer inspection of the data revealed that this is due to the problem that the feature LSGOVPREP is only unified with the governing preposition in certain syntactic contexts, a bug which has to be removed in

[14]Note that the first 4 lexemes in the row 'Not in ERG' are nouns in ERG, but adverbs in OALD.

[15]Cf. OALD's idiom entry for *at sea*. We cannot make a statement about the coverage of determinerless PPs. All the forms we are aware of, however, briefly checked with the test nouns, are listed.

[16]Because `man` is a count and mass noun, this phrase is not morpho-syntactically marked.

[17]The construction does have an idiomatic interpretation in some cases, see, for example the OALD entry for *from time to time*.

later versions of the prototype. As a consequence, many potential references (about 80%) were lost. Nevertheless there were enough references to perform a number of correct inductions.
Secondly, the system did indeed rule out all morpho-syntactically unmarked cases. After the harvesting of the enriched LSEEDs, all references for **lseed-noun_detless_pp** were done under $\text{RDoA}_{max}>1$ because of total neutralization. In the course of the first sub-cycle it first separated the mass- from the pure count nouns (see section 4.7). During *reassessment of experience* all the references for the pure count nouns were 'upgraded' to $\text{RDoA}_{max}=1$ because the ambiguities were removed. The system thus switched into a 'precision mode' in which the features are acquired under $\rho = 1$. This way the system 'decided' for the interpretation of morpho-syntactically marked determinerless PPs.
Thirdly, compared to the gold standard, there were a couple of true and false positives retrieved for the mere count nouns for which we first list the gold standard data[18]:

Pure Count Noun	Determinerless PPs (OALD)
area	-
case	in case
child	-
hand	by hand, at hand, in hand, on hand, out of hand, to hand
house	-
number	-
people	-
place	in place, out of place
problem	-
thing	-
way	under way
woman	-
world	-
year	-

The false positives were: *from year*, *from hand*, *from place* (because the grammar does not account for the productive *from* $\overline{N}$ *to* $\overline{N}$ construction), *in year*, *(more) than year* and *into place*. False negatives were: *by hand* (only two references), *out of hand* (only one reference), *to hand* (few references and neutralization with [to V]), *under way* (the noun was categorized as a mass noun in the first sub-cycle) and *in case*. The latter does not count because it is already listed as a closed word item in ERG. True positives were: *at hand*, *in hand*, *on hand*, *in place* and *out of place*.

[18]For the sake of consistency, we also list the nouns that are false negatives with regard to being mass-nouns, see next subsection.

We list a few references for illustration (including false negatives):

⌊C9X:1259⌉	Making beautiful books by hand is Tessa Fantoni's speciality.
⌊CH7:235⌉	Or a rum and blackcurrant to hand while he played dominoes.
⌊FU3:643⌉	Before you begin ensure that you have the following to hand:
⌊AM8:799⌉	A training revolution is under way in Britain.
⌊AT4:467⌉	They've got a jumble sale under way already.
⌊CBC:13355⌉	HELP could soon be at hand for Britain's 500,000 stammerers.
⌊EBS:1368⌉	However, help is at hand from other sources.
⌊CKF:2920⌉	Ewen had himself in hand again.
⌊FAJ:775⌉	Casting is in hand in Paris and Los Angeles.
⌊A7P:1143⌉	And there are highly qualified instructors on hand to advise you.
⌊CH5:4919⌉	Fergie's on hand as Eugenie gets the high-flying treatment from Johnny
⌊FEH:750⌉	Remember that anything out of place is incorrect.
⌊H8B:795⌉	He's a bit out of place here.
⌊A58:59⌉	He also put in place a long-term programme of staff development.
⌊A70:2022⌉	Hold it in place with a rubber band.

4.7 Countability and Number

The mass/count distinction is reflected by the two lexical seed types **lseedmassnoun** and **lseedcountnoun**, respectively. The latter introduces the feature LSNUMBER with the values **ls_sg** for nouns that have singular forms and **ls_pl** for those which occur in plural. This type subsumes a further lexical seed type **lseednoun_with_plural** for which LSNUMBER is constrained to **ls_pl** and which is required for the introduction of the feature LSINFLPL which takes the value '+' for nouns with inflectional plural and '-' for bare plurals. This fragment of the lexical seed type hierarchy forms the language for the specification of the following basic noun classes:

	Class	Specification
A	mass nouns	**lseedmassnoun**
B	sg. nouns	**lseedcountnoun**, [LSNUMBER **ls_sg**]
C	infl. pl. nouns	**lseed_with_plural**, [LSINFLPL +]
D	bare pl. nouns	**lseed_with_plural**, [LSINFLPL −]

Most of the count nouns have entries for B and C. Plural nouns like `scissors` have D-entries only, singular nouns like `grasp` opt for B only. Nouns that have both count- and mass-interpretation will be stored with entries of A-C. `people(s)` would have entries for B-D.

These distinctions are accounted for in the OALD with the symbols [C] (countable) [U] (uncountable), *sg.* and *pl.* In some clear cases where the noun is countable only, the markings are suppressed. The gold standard data for the test nouns are as follows[19]:

	Class	Nouns
A	mass nouns	area, case, house, life, man, number, part, time, work
B	sg. nouns	all test nouns
C	infl. pl. nouns	all test nouns
D	bare pl. nouns	people

Some of the mass nouns seem to have very restrictive usage: for `area` the OALD cites *The room is 12 square metres in area.* It is not clear to us whether this should be better treated as a determinerless PP. `house` is uncountable in the sense of *house music.* `case` and `number` are uncountable

[19] Also cf. subsection 4.8 for a discussion of the lexeme `part`.

in the context of *grammar*.

The first observation made about the test results is that the system conducted all inductions of the related realization contexts in the first sub-cycle and on the RDoA_{max}=1 level. All nouns have been identified as count nouns. Most of them have a relative realization frequency $> p^{fv}_{max}$ and are unproblematic. This is due to the fact that countability comes along with precise cues (morphology and determiners). Because of the high overall relative realization frequency the estimated noise level was limited to ϵ_{max}. Compared to the overall usage the noun `work` is rarely used in a countable sense but its relative frequency (6.1%) is above the noise barrier. The feature value that represent count nouns is best described by noise scenario 3A (cf. the table on page 133). The corresponding beliefs for all test nouns were stable and thus have not been revised during subsequent sub-cycles.

The system further inferred that all test nouns can be used in plural. Again all related beliefs were stable and the noise scenario was again 3A. For most of the nouns [LSINFLPL −] has been induced, which means that they are treated as non-bare plural nouns. For some nouns the situation keeps undecided because the minimum number of LOs has not been reached. For a subset of these the correct decision was made in the course of subsequent sub-cycles. The noun `people` was correctly identified as bare plural noun, but an incorrect decision (FN) was made regarding [LSINFLPL +] (*peoples*) due to an insufficient number of references (only 12 out of 556 LOs).

The identification of mass nouns worked reasonably well. The system estimated a noise level of $\hat{\epsilon}^{\rho=1}_{eff} = 0.5\%$ in the first sub-cycle which increased to 0.8% in the second. The noun `way` is the only false positive in the first sub-cycle, but turns into a true negative in the second. The nouns `area`, `house`, `case` and `number` are false negatives due to their restricted usage as mass nouns.
Figure 4.7 shows the relative frequency spectra. Arrows with the label 'SC2' indicate movements in the second sub-cycle.

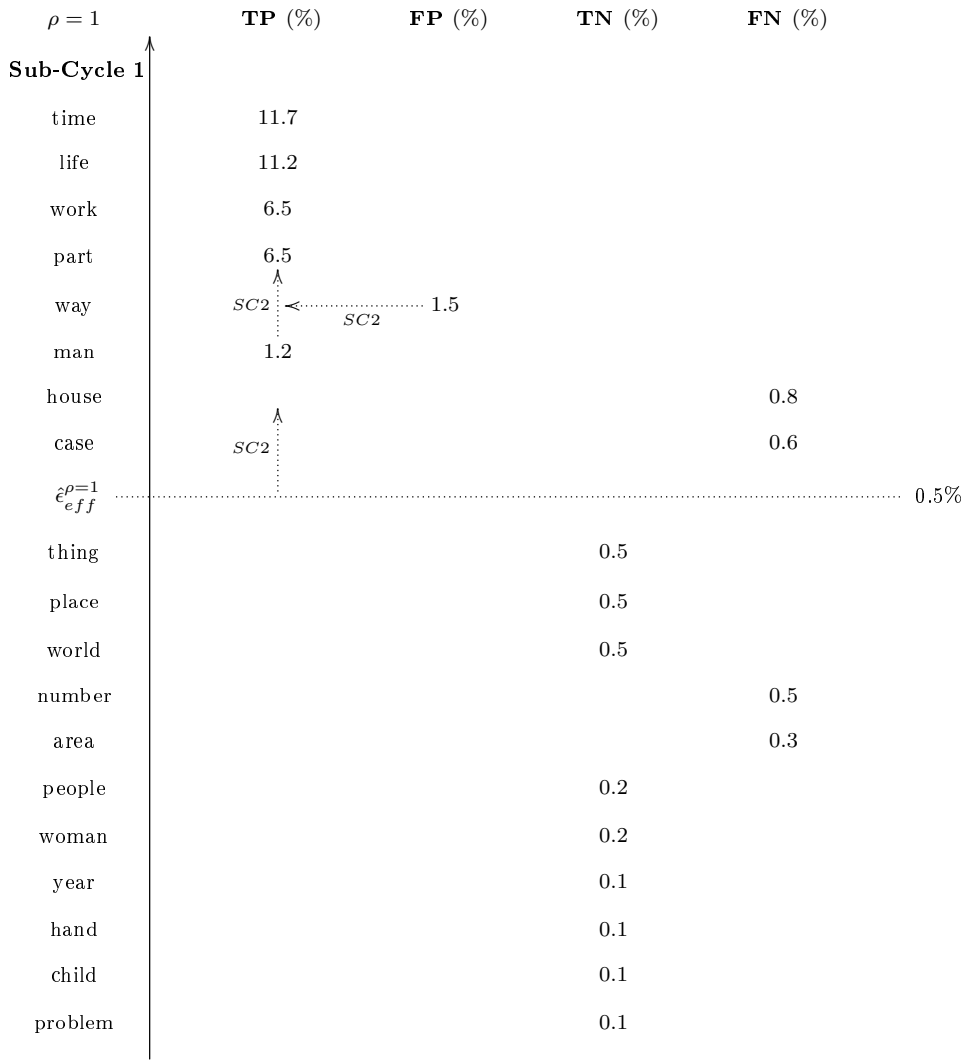
A couple of references found by the system will be listed here for illustration.

MASS	⌊FB9:1007⌉	Life's the thing, isn't it?
MASS	⌊HS2:435⌉	Work has already started.
MASS	⌊A15:1015⌉	Time, in fact, had ceased to have meaning.
MASS	⌊ABD:1404⌉	And enforcing his policies is only part of the problem.
COUNT/sg.	⌊K5D:11624⌉	Alcohol also plays a part.
COUNT/sg.	⌊H45:874⌉	It's a great life.
COUNT/sg.	⌊K55:5327⌉	It was a great time.
COUNT/sg.	⌊EFD:913⌉	They were a resourceful and talented people.
COUNT/pl.	⌊AKJ:215⌉	Eardley's work comes in two parts.
COUNT/pl.	⌊K1C:2220⌉	They'll remember it as a tremendous part of their lives.
COUNT/pl.	⌊CKV:1050⌉	New works by him are at Paula Cooper.
COUNT/pl.	⌊K5C:2558⌉	People are coming to us in great distress.

This section concludes with a few remarks on the example utterances listed above. Utterance ⌊FB9:1007⌉ is an unambiguous (RDoA_{max}=1) reference for the lexeme `life` being a mass noun because the lexeme is projected to a determiner-less DP that agrees with the copula in singular (so that a bare plural noun analysis is excluded)[20]. This is similar to utterance ⌊HS2:435⌉ where the determiner-less DP agrees with `have`.
Utterance ⌊A15:1015⌉ is more interesting because it can be used to illustrate the dynamics of the learning system. It is an ambiguous reference for the lexeme `time` being a mass noun or alternatively a bare plural noun ([LSINFLPL +]). So, after the first sub-cycle it is treated as a RDoA_{max}=2 reference. During the second sub-cycle, however, the system was able to induce that the lexeme is *not* a bare plural noun. As a consequence, the corresponding reading of the utterance was then deactivated so that the utterance was 'upgraded' to a RDoA_{max}=1 reference for `time` as a mass noun. The process starts similarly for utterance ⌊ABD:1404⌉ and lexeme `part`. The reference, however, stays ambiguous because the system does not fix the value for LSINFLPL in sub-cycle 2.
The next four utterances are unambiguous references for the respective lexemes being count nouns that can occur in singular.
The remaining four utterances are all unambiguous references for the respective lexemes being count nouns that can occur in plural. In the first three of them it is plural inflection that gives the clues to the system. In the last example the plural noun agrees with the auxiliary in plural.

[20] As pointed out in section 2.5.3 the system is limited to distributional properties and it does not reflect semantic particularities of the mass/count distinction.

Figure 4.7: Relative Frequency Spectrum of **lseedmassnoun**



4.8 Nominal Selection of Prepositions

Nominal postmodification by prepositional phrases can be productive (*a man with a beard*) or lexicalized (*time for something*). The latter is reflected by the feature LSPP in the nominal realization contexts. Its values are subtypes of **selected_prep_rel** plus the value **ls_no_prep** which indicates that the noun can also be used without preposition in the realization context. Nouns which require a PP lack this feature value. One case in point is probably[21] `part` in the realization context of **lseedmassnoun**.
Lexicalized cases are listed in the OALD. In the case of `time`, for example, the preposition `for` is specified as an optionally selected preposition using the key '(~ for sth)'.
For the evaluation we focus on the countable realization context and indicate major differences in the other contexts whenever they occur. We also restrict the evaluation to single PPs because double-PPs such as *the way from X to Y* are generally too infrequent in the set of test sentences. This may be a general problem or may be due to the restricted u-length.

The gold standard data are as follows:

Noun with lex. PP	Prepositions
area	of
case	for, against
hand	in
number	of
place	in
time	for
way	of, from, to

The first interesting result is that there is no false negative, which means 100% recall. The only exceptions are the two prepositions of `case` when the noun is used in plural. This lines up with the related remark in the OALD that *case against/for* is 'usually sing.'
The high recall is at a price of a considerable amount of false positives. It can be noted, however, that there are many true negatives, too, because many potential PP candidates did not pass the hypothesis test. The false positives are usually a) due to a cohesion between noun and the preposition which does not (yet[22]) give rise to a lexicon entry or b) to a systematic ambiguity

[21] In the first section that treats `part` as uncountable noun [U], the OALD specifies that `of` is mandatory, but there is another [U]-paragraph which does not. The example in this paragraph, however, is: *You need to be able to work as part of a team.*.

[22] Some noun/preposition combinations may be on their way of becoming lexicalized.

sustained by the indeterminacy during bootstrapping. The false positives are listed below. Prepositions that are false positives in the countable context only are subscripted with C.

Noun	False Positives	Precision
area	for	50%
case	at, in, of, with	33%
child	with	-
hand	across, against, at, down, from, into, on, over, through, to, towards, up, with	7%
house	at, by, down, for, from, in, of, on, to, with	-
life	after, as, at, for, in, of, on	-
man	at, behind, by$_C$, from, in, on, to$_C$, with	-
number	on, to	33%
people	at, in, with	-
part	at$_C$, in, of, to$_C$	-
place	as, at, for, of, on, to, with, without	11%
problem	about, at, for, in, of, with	-
thing	about, as, for, in, to, with	-
time	at$_C$, in, of, to, without	17%
way	about, across, along, around, as, down, in, into, of, out of, through, to, towards, up, with	17%
woman	from, in, on, to, with	-
work	as, by, for, in, of, on, out, with	-
world	around, as, by, from, of	-
year	after, for, in	-

In order to illustrate how the automatic noise level estimation performed, we discuss two relative frequency spectra, one for the high-frequency preposition `in` and one for the low-frequency preposition `about`[23].
The relative frequency spectrum for `in` presented in figure 4.8 shows that the estimated noise level is too conservative. It also demonstrates how difficult it is from a statistical point of view to separate the true positives from the high-frequency false positives. An interesting case is `part`. In the majority of references (50%) the idiom *play a part in sth.* is realized. A similar situation is *case in point* which accounts for 38% of the references for `case`. The same holds for *way of* or *part to play* and further instances. This shows how important it is to set these larger idioms apart. We leave this as an action item for future research. Other high frequency combinations such as *house in* or *place in* are perfectly productive. The effective noise level is thus relatively high and the acquisition performance is subject to the noise scenario 1A.

[23]Relative frequencies are given for RDoA$_{max}$=∞ and in brackets for RDoA$_{max}$=2.

The relative frequency spectrum displayed in figure 4.9 shows that the majority of test nouns does not occur with the preposition `about`. The estimated noise level is therefore at the zero-level; two of the FPs are related to instable beliefs. The related noise scenario is 1B or 2B, depending on the nouns that are really lexicalized with the preposition. According to the OALD, none of the test nouns do so. The very high frequency of `thing` is thus striking. The references, of which we list a few here, indicate that there is a certain amount of idiomaticity and it may be asked whether it should better be listed in the dictionary[24].

⌊A0L:773⌉	Jay had a big thing about birthdays.
⌊CCA:618⌉	I've got this thing about beetles.
⌊A18:657⌉	We learn not a thing about him.
⌊A2U:70⌉	Which is probably the most offensive thing about her.
⌊A68:265⌉	Gollancz was the notorious thing about Repton in 1918-19.
⌊FRH:1168⌉	ABBERLEY: I don't remember a thing about you.
⌊ASV:827⌉	For this was the extraordinary thing about Pipeline.
⌊CBC:2433⌉	But then isn't that the greatest thing about Christmas?
⌊ASS:374⌉	The great thing about poisoners is their control.

Two further remarks should be made. Firstly, the system correctly infers a constraint on [LSPP **ls_pp_noprep**] for `part` as a mass noun. This does not only demonstrate how valuable it is to differentiate feature values by the various possible realization contexts but also that it is possible for the learning system to make use of it.
Secondly, systematic ambiguity with [N to V] (*time to watch*) leads to false positives of noun plus `to`. The reason is that during bootstrapping many verbs that are homograph with nouns are analyzed as mass nouns or nouns with determinerless PP in this syntactic context. Affected are all nouns that frequently occur with a *to*-clause. This effect could be avoided if the ranking suggested by the tagger would be taken into account, as taggers show a very high performance at part of speech.

[24] The OALD accounts for a series of idioms of which *thing about* is a part: *have a thing about sb/sth* (realized in the first reference), *not know, etc. the first thing about sth/sb*, realized in one of the references not listed here, *make a (big) thing of / about sth* and *the thing (about / with sth/sb) is* (also realized in one of the references). The references and most of all their high frequency, however, suggest that *thing about* is more flexible.

Figure 4.8: Relative Frequency Spectrum of **lseedcountnoun** and [LSPP **in_p_sel_rel**]

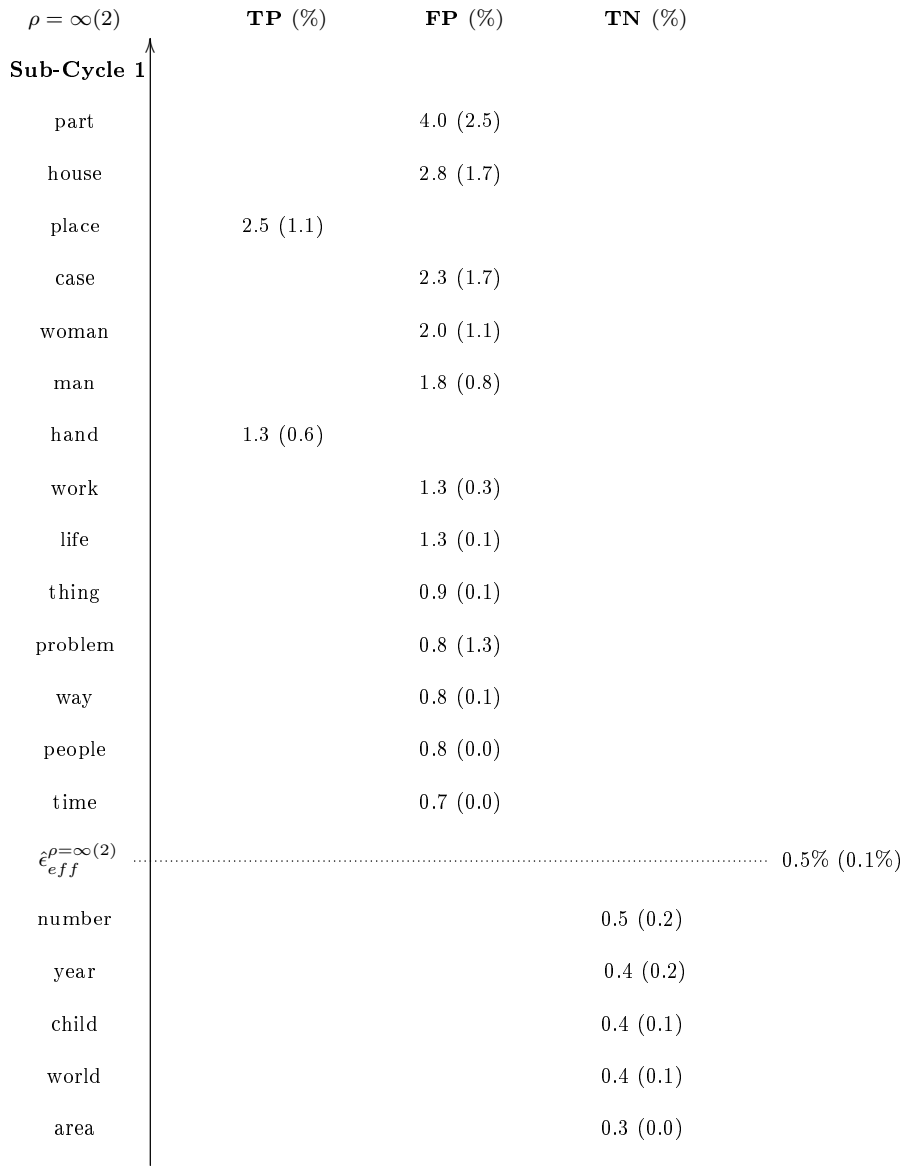
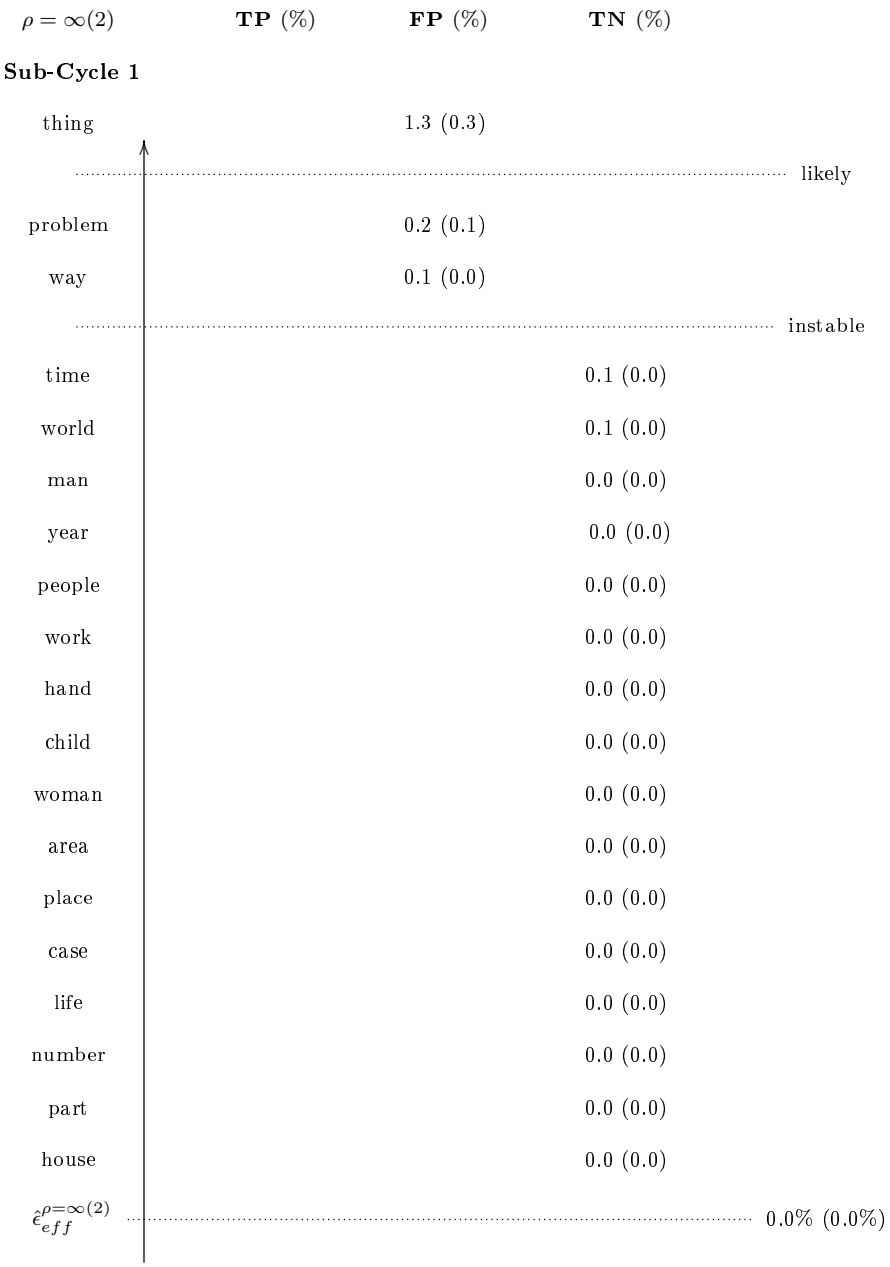


Figure 4.9: Relative Frequency Spectrum of **lseedcountnoun** and [LSPP **about_p_sel_rel**]



4.9 Noun Postmodification by SCs

Similar to noun postmodification by PPs there are cases in which postmodification by sentential complements is lexicalized, in which case the feature LSSC has a value which encodes the type of the complement[25]. Here we are interested in four possible values which are accounted for in the OALD. Because the complex feature values are somewhat unwieldy, we will use short keys for them:

	Type of Complement	Value of LSSC	Short Key
A	no complement	**ls_sc_nosc**	[LSSC **none**]
B	*that*-clause	**ls_sc_compl**, LSSCSF:**prop**	[LSSC **that**]
C	complementizer-less clause	**ls_sc_nocompl**, LSSCSF:**prop**	[LSSC **nocompl**]
D	*to*-clause	**ls_sc_to**	[LSSC **to**]

The classes B and C are intended to account for nouns that allow for an appositive clause in the sense of Quirk et al. (1994: 1260) (as in *the fact that he wrote a letter to her*, ibid.). They are usually constructed with overt complementizer, but the covert variant is also possible[26].
The class D is reserved for appositive clauses with *to* and for infinitival relative clauses in which the head noun has an adverbial role in the clause and must be contrasted with productive relative clauses. The examples in (39) are intended to illustrate the difference:

(39)
a. the man to help you [...] (Quirk et al. 1994: 1266)
b. the man (for you) to see [...] ibid.
c. the time at which to go is ... ibid.
d. the time to go
e. [...] the will to win (Quirk et al. 1994: 1272)

The examples in (39a) and (39b) are productive relative clause (RC) constructions in which the head noun is argument of the verb in the RC. The head noun can also take an adverbial role as in (39c), which is productive when the relative pronoun is overt. The covert variant in (39d) is restricted to certain nouns and must therefore be accounted for by the lexicon. The

[25] In the current prototype this feature is used for all kinds of non-PP complements, so the notion 'sentential' is used very lax here. This might be changed in later versions of the LTH.

[26] As in *B has a feeling C fell.*, taken from http://wiki.delph-in.net/moin/ErgLeTypes.

RCs are to be set apart from the appositive clause in (39e) which is lexical, too.
The constructions realized in (39d) and (39e) have different semantics; this difference, however, is conflated in the OALD. Both are specified with the key '~ (to do sth)'. This mirrors the specification in the prototype's LTH which does not differentiate them either. Because not too many nouns allow for sentential complements the gold standard data for the test nouns are relatively spare:

Noun with lex. SC	Complement (Key)
case	**that**, **nocompl**
time	**to**
way	**to**

Regarding the outcome of the test we start with the unproblematic feature value **ls_sc_nosc**. This one is straightforward because nouns occur without sentential complement most of the time. The prototype thus had no problem of handling it correctly.
As in the last subsection we restrict the discussion to the countable realization contexts and indicate deviation in the other contexts. First of all, it turned out that the system could not acquire the feature value for `case` because there was only one reference. On the other hand the complementizer-less case (C) appeared to be extremely noisy because it is afflicted with a broad range of potential ambiguities. Due to that the system decided for 7 false positives (37% of the test nouns). In the mass noun context there is only one false positive: `time` in category B.
Regarding the remaining class (D) the system inferred both cases of the gold standard. Because of the overlap with isomorphic structures, discussed above, there is systematic neutralization and thus a high degree of indeterminacy. It may be worthwhile to analyze how the system coped with that. This is illustrated with the help of figure 4.10, which allows for a series of observations. Firstly, there is considerable noise both on the RDoA_{max}=2 and RDoA_{max}=∞ levels. The estimated noise level of the latter is around 0.3% which lines up with the majority of nouns which do not admit the feature value. The estimated noise level for the former is very low (0.08%). This is because of a very small overall relative feature value realization frequency of 0.13% with the effect that decisions for feature value admissions are mainly done on RDoA_{max}=2. The two TPs are among the exceptions, which means that the noise scenario for the RDoA_{max}=∞ is here more comfortable than for the other level.

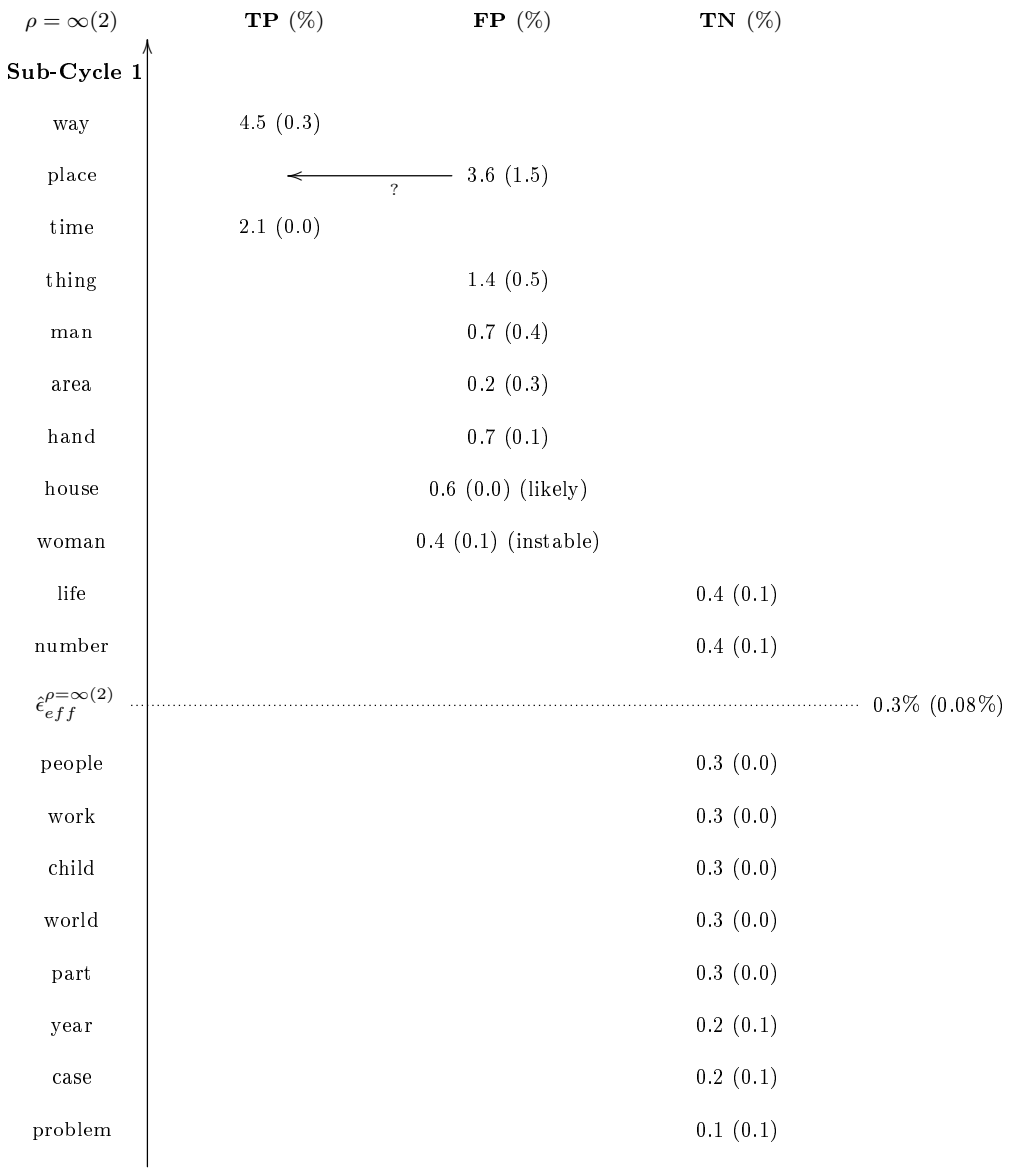
Secondly, it may be questioned whether `place` is indeed a false positive. There are many references that show the usage of `place` in analogy to the example (39d) above, some of them are listed here:

⌊A7A:2423⌉	My God, Mecken, what a place to live!
⌊A7D:1888⌉	Jadis is in London Road, a good place to begin.
⌊CH0:430⌉	It was his place to remember the ancestors.
⌊AMD:874⌉	It is the ideal place to spend a leisurely holiday.
⌊ECG:413⌉	We looked for a place to sit.
⌊ED9:322⌉	It's an elegant and cultivated place to be.
⌊AS3:970⌉	What a place to spend a birthday.
⌊FP1:1779⌉	I can think of no better place to be married.

The data show that this noun can be used similar to `time` and `way`, so we propose to specify this usage in the dictionary, too.

A final remark with regard to the other two realization contexts is in order. Firstly, there was only one false positive (`work`) in the mass noun context. Secondly, in the plural context (which shows a behavior similar to the singular context discussed above) the system inferred that `times` is not in category D. It is not clear from the OALD whether this is a false or a true negative.

Figure 4.10: Relative Frequency Spectrum of **lseedcountnoun** and [LSSC **to**]



4.10 Verb Particle Constructions

Verb particle constructions are a prominent case of multiword verbs (for a detailed analysis cf. subsection 2.7.2; for a list of publications of related research see subsection 1.3.1). In the lexical seed type hierarchy VPCs are represented by the feature LSVPRT for which subtypes of **selected_prep_rel** are appropriate values (see the paragraph on page 242 for details). The lexical seed types **lseeditrverb_prt** and **lseedmtrverb_prt** differentiate intransitive from transitive VPCs without sentential complements. Together with the feature mentioned they form realization contexts of their own, the feature being a type constraining feature. This makes it possible to learn further features like LSPP in the various VPC contexts.
VPCs are listed in the OALD with the key [PhrV] (for *phrasal verbs*). Care must be taken, however, because the key also comprises prepositional verbs. It is nevertheless straightforward to filter out VPCs out of the OALD. In the experiment we focused on intransitive and transitive VPCs without sentential complement. For reasons of limited space we cannot discuss all the aspects of acquisition here and limit ourself to a few general remarks, the recall/precision metrics, a few illustrative references, and a presentation of selected relative frequency spectra.

The performance in terms of precision/recall is as follows:

	Intransitive VPCs	Transitive VPCs
True Positives:	49	46
False Negatives:	27	28
False Positives:	12	73
Precision:	80.3%	38.7%
Recall:	64.5%	62.2%

The observations allow for a couple of conclusions: Firstly, there is less recall than with the features discussed so far. This is mainly due to the very rare frequency of certain VPCs attested in the OALD. Secondly, while the recall values are similar the precision of the acquisition of transitive VPCs is much smaller than the one of the intransitive VPCs. This is mainly due to the effect that the former are often acquired on the RDoA_{max}=2 level (cf. the recoverability factors on page 275) for which in many cases the number of learning opportunities is not sufficient in order to make a decision (note that undecided cases count as positives because they will not lead to constraints in the lexicon).

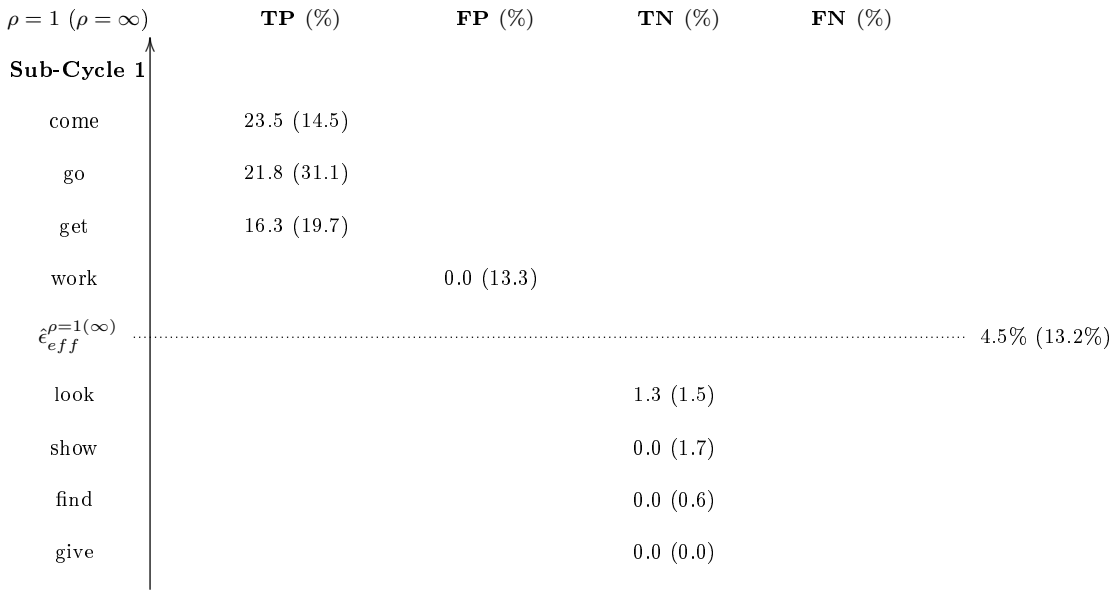
The following (intended and unintended) references demonstrate how ALR differentiates unambiguous from ambiguous cases. All references are related to transitive VPCs with the particle *up*. The next table lists three intended and one unintended reference. The last column reflects the number of references found similar to the respective example.

UtID	Ref. for Transitive VPC	RDoA_{max}	c_{ref}
⌊ANP:360⌉	They are made up of chopped leaves and grass cuttings.	$\rho = 1$	5
⌊AT7:2253⌉	Let's get him up.	$\rho = 2$	1
⌊HTS:3422⌉	He looked up a number.	$\rho = 2$	12
⌊CFJ:1495⌉	Time to get up!	$\rho = \infty$	31

The first reference is an example for the rare intended references under RDoA_{max}=1. These are typically cases in which the VPC is realized in passive voice. Usually more frequent instances are references under RDoA_{max}=2. The actual yield, however, may vary. There is one reference for `get` and there are 12 references for `look`. The feature for the former lexeme is therefore acquired under RDoA_{max}=∞. The last reference is unintended because it is a reference for the intransitive VPC rather than the transitive. It is ambiguous because the noun `time` *could* be the direct object of the VPC.

For an illustration of the feature value characteristics involved in the induction process we pick out two particles in the context of intransitive VPCs: the particle `on` with a relatively good acquisition performance (100% recall, 75% precision) and the particle `about` with a lower performance (33% recall and 100% precision). Because the relative frequencies are related to the realization context **lseeditrverb_prt** it is worthwhile to look at the verbal LTH (cf. figure E.6) as a whole to better relate the figures to the verb classes: All figures are based on the first sub-cycle, if not stated otherwise. The VPC related types emerge within the context of MWE verbs (**lseedmweverb**). The overall relative frequencies of this type compared to basic verbs are 0.3% (RDoA_{max}=1), 14.2% (RDoA_{max}=2), and 22.7% (RDoA_{max}=∞), respectively. This already indicates that the system usually induces the corresponding features with a more fuzzy focus. The next type under **lseedmweverb** is **lseedverb_prt**. This type is a candidate for induction under RDoA_{max}=2, because the overall relative frequency for RDoA_{max}=1 is 0%, for RDoA_{max}=2 it is already 58.7%. This changes after the first sub-cycle because of the disambiguation during the reassessment of experience: the overall relative frequency for RDoA_{max}=1 is then 87.9%. The next type down the hierarchy is **lseedverb_prt_nosc**. Here the overall relative frequencies are beyond 90% which indicates that there is not

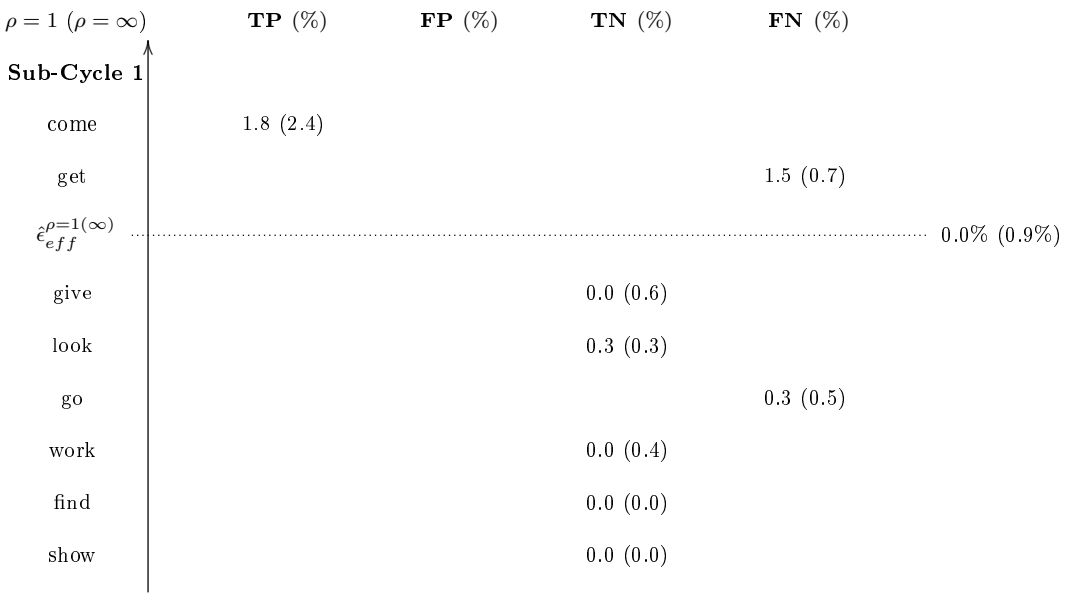
Figure 4.11: Relative Frequency Spectrum of **lseeditrverb_prt** and `on`



much loss of learning opportunities. The next type is already the realization context **lseeditrverb_prt** in scope of this discussion. Its overall realization frequencies are 5.6% (RDoA_{max}=1), 17.2% (RDoA_{max}=2) and 32.5% (RDoA_{max}=∞).
The yield of references for VPCs can be calculated using some absolute numbers: in total there are 59386 references for **lseedbasicverb** for RDoA_{max}=1, 84448 for RDoA_{max}=2, and 93364 for RDoA_{max}=∞. For the type **lseeditrverb_prt** there are 19 references for RDoA_{max}=1 (0.03%), 1611 for RDoA_{max}=2 (1.9%) and 4872 for RDoA_{max}=∞ (5.2%). The relative frequencies presented in the relative frequency spectra in figures 4.11 and 4.12 are related to the few learning opportunities available for the realization context. Note that a couple of verbs have already been excluded from the induction process of this realization context because they have dropped out on hypothesis tests higher up in the LTH (e.g. `say` or `know`).

It must be finally noted that the performance of *in vitro* methods (as reported in Baldwin 2005b) is better. This may be traced back to the fact that state-of-the-art taggers have the excellent capability to differentiate particles from adverbs, a capability which is not (yet) exploited in ALR.

Figure 4.12: Relative Frequency Spectrum of **lseeditrverb_prt** and `about`



4.11 Verbal Valency

This section evaluates how the system performed at the acquisition of verbal valency. We restrict this notion here to the classification into intransitive, transitive and ditransitive verbs on the grounds of mere syntactic criteria. These categories are represented by the three lexical seed types **lseeditrverb**, **lseedmtrverb** and **lseedditrverb**, respectively. Hence realizations with sentential/adverbial complements are excluded (cf. the LTH in E.3 in the appendix).
The gold standard as per the OALD is as follows[27]:

Class	Nouns
intransitive	ask, come, feel, find, get, give, go, know, leave, look, say, see, seem, show, take, tell, think, work
transitive	all test verbs
ditransitive	ask, find, get, give, leave, make, show, take, tell

First of all, all test verbs have been correctly categorized as *transitive* verbs in the first sub-cycle and with RDoA_{max}=1. This means 100% recall and precision. The overall relative feature value frequency was 55.7% (far beyond ϵ_{max}) and the estimated noise level was capped to ϵ_{max} (from an uncapped estimated level of 30.5%), cf. the relative frequency spectrum in figure 4.13. The noise scenario is 3A (cf. the table on page 133) and the lexeme that comes closest to the capped noise level is `go`. This lexeme is rarely realized in a transitive frame. Closer inspection of the (alleged) references reveals that only a few were really intended. Most references were unintended due to noise in the grammar and/or initial lexicon[28]. The next tables show some of the intended[29] and unintended references under RDoA_{max}=1:

[27] The OALD specifies these categories with the keys [V], [VN] and [VNN]. Linking verbs are marked with [V-N] and [VN-N], respectively, but are not differentiated here from the superficially isomorph structures so that [V-N] is conflated with [VN] and [VN-N] with [VNN], respectively.
The intransitive usages of `use`, `mean`, and `want` are marked as informal by the dictionary and will be ignored here. For convenience and because some categorizations may appear counterintuitive at first glance we have compiled a list of examples from the OALD in section C in the appendix.

[28] The situation with `come` is even worse. Nearly all references were unintended.

[29] Note that these usages seem to be unproductive. The phrase 'next door' in the second sentence might be analyzed as a PP, cf. 'the girl next door' or 'he was waiting next door' but *'he went next building'. This alternative, however, is not covered by the grammar, hence the phrase is analyzed as the direct object of `go`.

UtID	Intended reference for `go` in transitive frame
⌊ACK:1114⌉	Why do they just go ding-dong?
⌊HNJ:2537⌉	he put down the instrument and went next door.

UtID	Unintended reference for `go` in transitive frame
⌊CEU:3197⌉	Thing, we can't go!
⌊ANK:1895⌉	Thus Wednesday 12 February went much the usual affable way
⌊CEP:9244⌉	His place goes to Wales international Scott Gibbs.
⌊APM:667⌉	Isn't it time you went?

The differentiation of *intransitive* verbs was more difficult for the system. The overall relative feature value frequency was 18.2% (again far beyond ϵ_{max}), the estimated noise level was therefore again capped to ϵ_{max}. Most of the time the decisions were made during the first sub-cycle and under RDoA_{max}=1. For a couple of verbs (e.g. `make`) the system set the focus to RDoA_{max}=∞ and induced a false positive. There was no true negative, but one false negative, namely `give`, because its relative realization frequency was below the estimated noise level (cf. the relative frequency spectrum in figure 4.14). Type precision was 70.8% and recall 94.4%. Note that for some of the verbs the intransitive context is required for [V PP] subcategorization (for example `feel` or `ask`)[30].
An interesting case is the false positive verb `mean` for which a relatively high relative frequency can be observed. This is mainly due to the idiom *I mean* which is classified as informal in the OALD. A similar case is the false positive `become` which is due to the relative frequent idiom *what became, has become, will become of sb/sth?*.

This section concludes with the discussion of the *ditransitive* valency frame. Again the overall relative frequency was beyond ϵ_{max}: 10.8% in sub-cycle 1. And again the estimated noise level was capped to ϵ_{max}, cf. the relative frequency spectrum in figure 4.15. As with the transitive frame all decisions were made during the first sub-cycle and under RDoA_{max}=1. Note that the ditransitive valency frame type **ls_ditrans** in the LTH is a subtype of the transitive **ls_monotrans** (cf. figure E.1 in the appendix), which means that the realizations in the transitive frames become the learning opportunities for the ditransitive. All relative frequencies are thus relative to the transitive frame. The system inferred 10 true negatives. There was no false negative (100% recall), but a couple of false positives: `go`, `think`, `come`, `use`, `put`, `become`, so the precision was 60%. The verbs `go` and `come` occur

[30]This means that the system has the chance to exclude the [V] usage by virtue of induction of the LSPP feature. Unfortunately the system left this undecided.

infrequently in the transitive frame so that there were only an insufficient number of learning opportunities for the ditransitive frame and the system left these cases undecided (to be judged as false positives). An interesting case is `think`: compared with OALD it counts as a false positive. There are, however, a couple of references that suggest a ditransitive (or more precisely a linking verb) analysis, some of them are listed below:

UtID	Intended reference for `think` in ditransitive frame
⌊A68:1165⌉	He thought a procession a bore.
⌊ABJ:1669⌉	Several members of the Council think this a good idea.
⌊B20:1428⌉	I thought her a trollop.
⌊FS5:1469⌉	thought Africa no place for that grubby
⌊G1L:882⌉	The Gnomes thought this a fine old idea.
⌊BP9:592⌉	They all thought it a great joke.

Figure 4.13: Relative Frequency Spectrum of **lseedmtrverb**

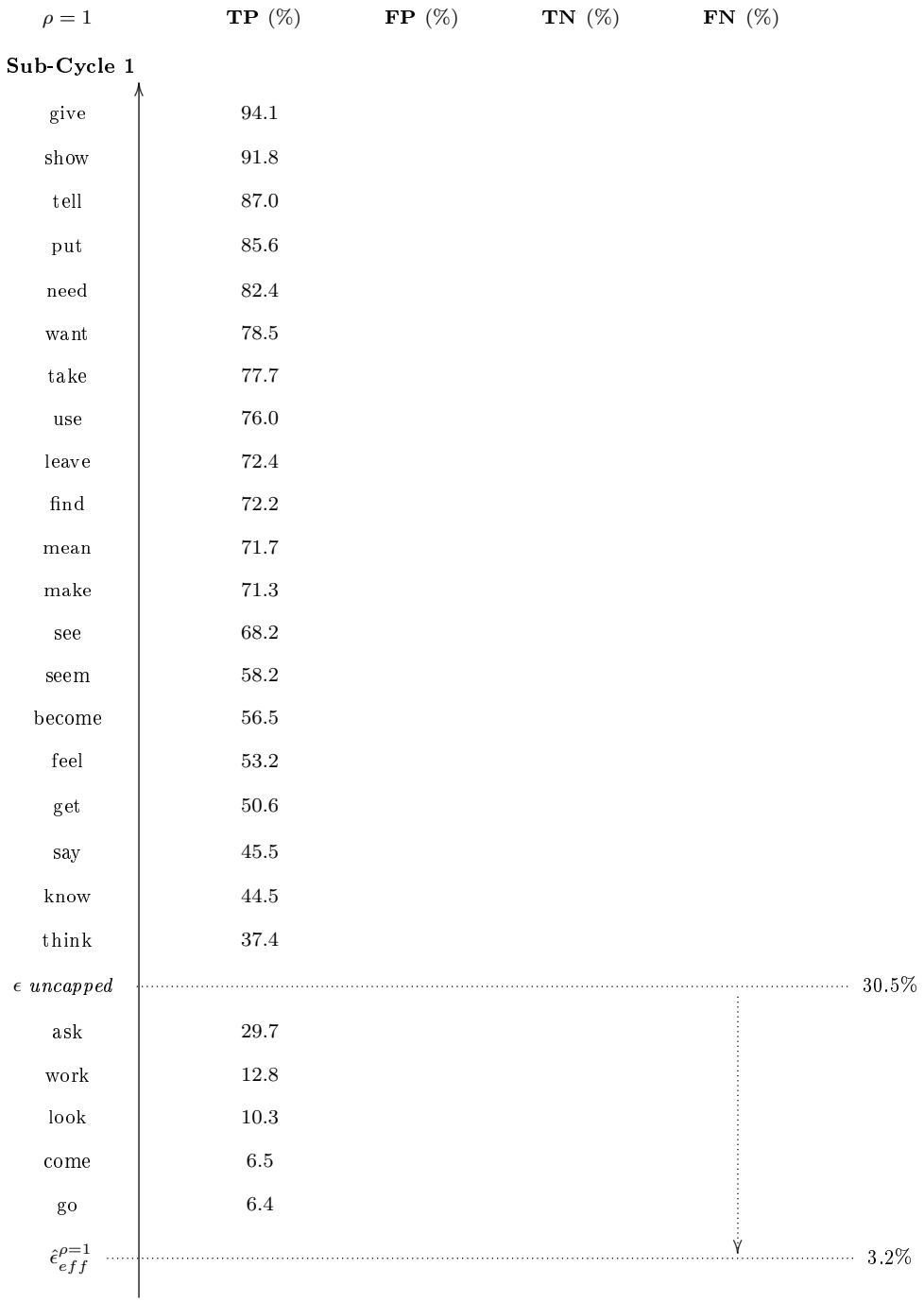


Figure 4.14: Relative Frequency Spectrum of **lseeditrverb**

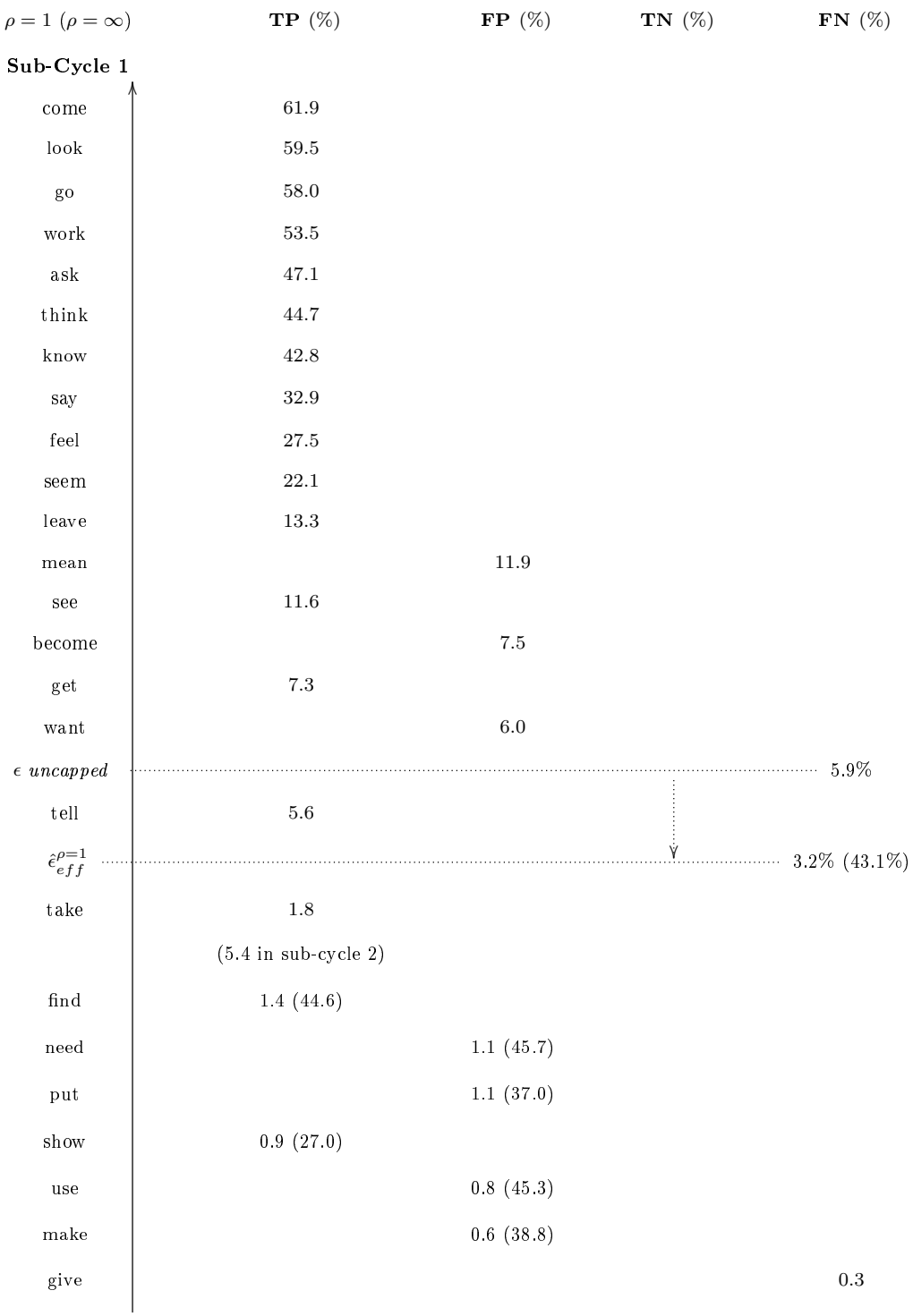
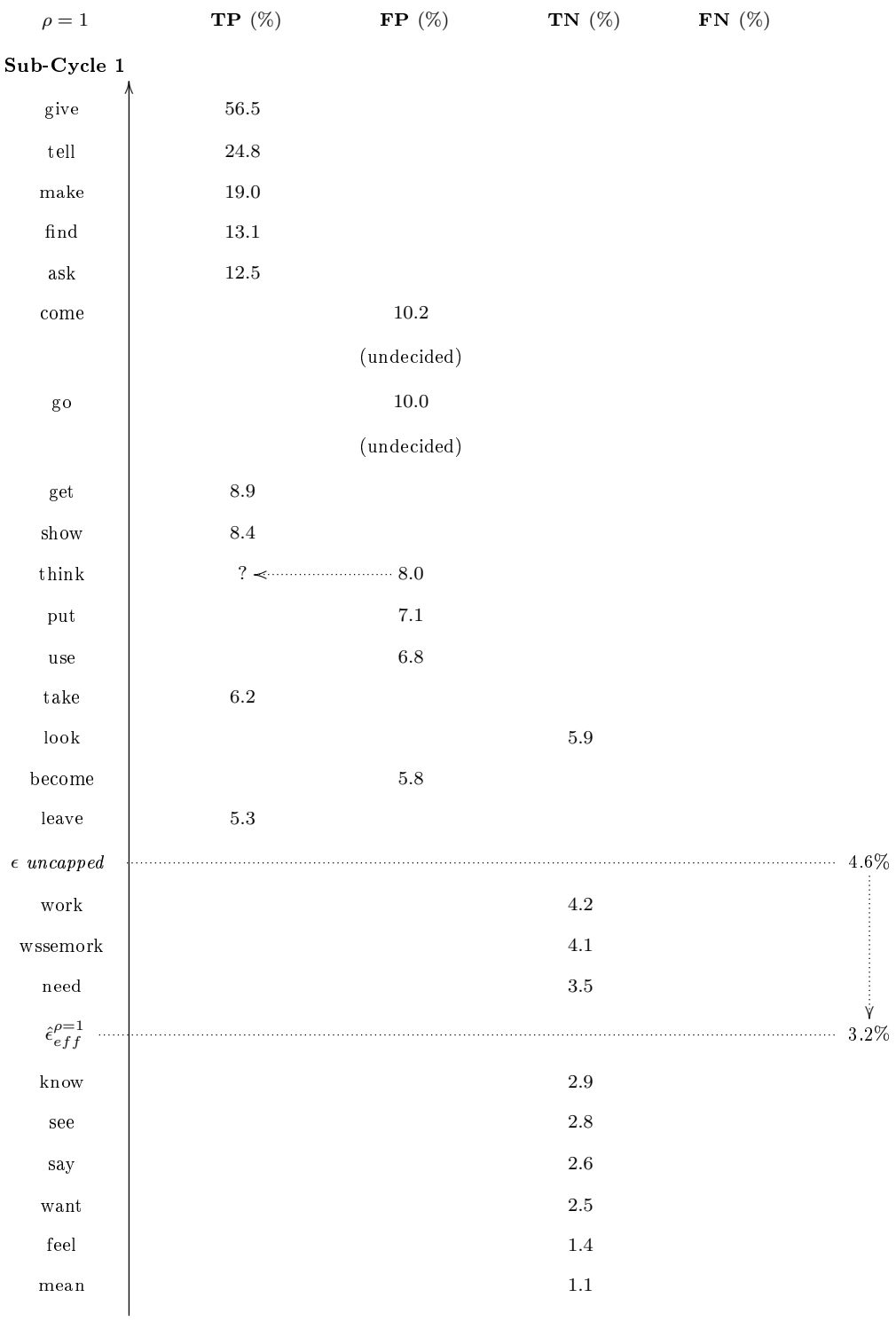


Figure 4.15: Relative Frequency Spectrum of **lseedditrverb**



4.12 Attributive vs. Predicative Usage of Adjectives

In the lexical seed type hierarchy the feature LSATTR is used to specify whether an adjective can be used attributively (**ls_attr**), and/or predicatively (**ls_prd**). For the five adjectives `other`, `good`, `new`, `old`, and `great` we did a quick check (without recourse to the OALD as a gold standard because the correct settings are obvious) whether a) this feature was correctly induced and b) which was the RDoA_{max} level of decision.
First of all it can be stated that there is 100% recall for both feature values and the 5 adjectives. All feature values have been induced during the first sub-cycle and with RDoA_{max}=1. The relative frequencies are well beyond ϵ_{max} (which is why we do not include any relative frequency spectra in this section): For **ls_attr** the highest value was 74.3% (lexeme `new`), the lowest was 12.1% (lexeme `old`). For **ls_prd** the highest value was 82.5% (lexeme `old`), the lowest was 4.5% (lexeme `other`).
The relatively high value for `other` is responsible for the induction of **ls_prd** for the lexeme, which is a false positive given that it cannot be used predicatively (**she is other*). A look at the data showed that this was due to two different effects. Firstly, the idiomatic construction *other than* could not be analyzed but predicatively. From the perspective of the system the decision was therefore correct. The corresponding unintended references made up 25%. Here are some of them:

⌊CBU:4209⌉	Shares other than non-equity shares.
⌊CBU:4248⌉	Entities other than companies may adapt the terminology as appropriate.
⌊CBX:3073⌉	There is no use for the material other than the project.
⌊CCV:1555⌉	Many children in our schools speak languages other than English.
⌊CD2:1649⌉	Could she play other than Shakespeare?

Secondly, the system was misled by the overlap with the closed-class item `each other`, which accounts for 37% of the unintended references. This shows that the LEARN module must take such overlaps into account, which will be an action item for future versions.

4.13 Was Learning Effective?

In order to investigate whether the acquired features show some significant effect, we conducted two tests with the seed lexicon updated with the lexical entries created by REDUCE. These were generated for the test lexemes only. The module exported around 51 seed entries per lexeme mainly due to the false positives of the LSPP feature.
The first test concerned the average processing time of the parser motivated by the question of whether the restricted u-length interval may be widened after the learning cycle for the sake of more learning opportunities. We therefore processed 1000 utterances with an u-length interval of $[13, 20]$ both with the initial and with the updated system. There was a measurable difference in favor of the updated system, which had 24% less timeouts. The processing time, however, increased slightly by 4% because the additionally parsed utterances which had timeouts in the initial system increased the average processing time of the updated system. In addition the performance gain by disambiguation appears to be outweighed by the increased number of lexical seed entries. This moderate increase notwithstanding the outcome suggests that subsequent learning cycles may be processed with a moderately widened u-length interval.
In the second test we fed 10000 of those utterances into the updated system which previously had more than 1000 readings – based on the idea that if learning shows effect then there will be a certain amount of utterances for which the updated system will compute less than 1000 readings. This was borne out. Approximately 17% of the re-analyzed utterances had more then 0 and less then 1000 readings. (2.7% had 0 readings, most of them due to timeouts).

Based on the results of the two tests we conclude that learning was indeed effective and lead to a significantly increased disambiguation performance.

4.14 Conclusions Drawn from the Outcome

This chapter concludes with a brief review of the major experimental findings. The bootstrapping experiment and the preliminary tests covered both functional and non-functional aspects.
Starting with the question of the system's non-functional performance, it can be stated that the impact of underspecification on the processing time was better than we initially expected. In preliminary tests (subsection 4.3.2) it was shown that the average processing time of utterance with an u-length

interval of $[3, 12]$ increases by factor 22.6 when the ERG is equipped with the lexical seed layer and the initial seed lexicon. Setting the GENRE feature to **formal**, which appeared to be a reasonable choice given the linguistic quality of the corpus, this factor went down to 13.1. A prerequisite for that was the proper handling of punctuation (subsection 3.3.5). The implementation proved to be very robust, provided that there is a sufficient amount of RAM (cf. the specification in D). Apart from a very few hick-ups during the ANALYZE step, the tests performed smoothly.

The functional evaluation starts with the performance of lexeme detection (subsection 4.6). The detection of single-word lexemes makes use of PET's capability of unknown-word management. As outlined in subsection 2.7.1 single-word lexemes are notoriously difficult to detect when lexical semantics is ignored. Moreover, evaluation hinges on the definition of what constitutes a lexeme. These principle issues notwithstanding, we were interested in how much noise will enter the lexicon when detection is done with a simple frequency cut-off (in later versions this may be replaced by relative frequency thresholds). Because of WordNet's coverage we did not expect to find many 'new' (common) nouns, verbs or adjectives. The true positives retrieved were mainly orthographic variants of the lexemes stored in that resource. In addition adjectives that are spelled with capital letter (e.g. geographic adjectives) as well as inflected forms (comparatives, superlatives) were subject to lexeme detection, because these have not been mapped into the initial seed lexicon. The system found 23 TPs of the former and 21 of the latter.
As an instance of a word class that is only moderately covered, we tested the detection of adverbs: the system detected 80 adverbs not listed in the ERG. Overall, there was less noise than we initially assumed. The majority of true negatives were orthographic representations of deviant English.
The detection of multiword lexemes were tested on noun compounds and determiner-less PPs. The precision of the former was around 20%. The false positives were usually familiar combinations or collocations with productive semantics. The detection of determinerless PPs was accompanied with a moderate amount of unattested cases, if one abstracts away from the cases that are due to the productive *from* $\overline{N}$ *to* $\overline{N}$ construction which should be accounted for by the grammar.
From these observations we conclude that ALR's capability of lexeme detection can be used to enhance the system's lexicon. The price, however, is a considerable amount of lexical entries which do not constitute 'real' lexemes, but this capability may well be used to feed a component which focuses on lexical idiosyncrasies based on the distribution of the lexeme candidate in order to filter out productive MWEs.

Regarding the lexical acquisition performance, lexical feature values can be put on a scale starting with those which are absolutely unproblematic up to those that are hardly learnable, at least during bootstrapping. Generally unproblematic cases were feature values that reflect count/mass distinction and number (section 4.7). This is a very positive sign for the general acquisition performance because the count/mass distinction is crucial for disambiguation. Some infrequent cases, however, resulted in false negatives (e.g. *peoples*, which will therefore be treated by the system as a lexeme of its own).
The acquisition of prepositions selected by nouns (section 4.8) showed 100% recall at the cost of relatively low precision. This is due to the fact that PP-selection is highly neutralized on the surface of the utterance, given that the system has no account for idiosyncratic senses. As a consequence, a considerable amount of prepositions that have a certain degree of cohesion with the noun accounts for the false positives. The related feature values may be ranked in the middle of the scale of learnability. The same holds for prepositional verbs (which are not discussed in this chapter) and for the differentiation of intransitive versus transitive verbs. The analysis of the acquisition of ditransitive verbs revealed that the verb `think` might deserve an extra entry in the OALD as a linking verb.
The acquisition performance for nouns selecting *to*-clauses (section 4.9) is in this middle range, too. There was again 100% recall with a considerable amount of false positives. As a side effect, the data suggest that the noun `place` should be specified for this feature in the dictionary.
VPCs form a class on their own because here the system showed less recall than with most of the other features.
Complementizer-less complements proved to be very difficult because they are afflicted with a high level of noise.

Finally, it was demonstrated that the acquired features had a significant effect on disambiguation.

Chapter 5

Summary, Conclusions and Outlook

This final chapter provides a summary of the assumptions, proposals and findings made in this thesis, followed by the conclusions that can be drawn from our work. Section 5.1 starts with a review of the challenges and questions that motivated our research. Section 5.2 presents the theoretical part in a nutshell and section 5.3 briefly reviews the implementation based on it. The experimental results are then discussed in section 5.4 and the conclusions drawn from it in section 5.5. The chapter concludes with suggestions for further research in section 5.6.

5.1 Three Questions

A parser system for deep linguistic analysis of natural language employs a precision grammar and a lexical component which provides it with all the fine-grained lexical information required for linguistic processing. Consider, for instance, which lexical distinctions have to be accounted for in order to produce a deep analysis of utterances such as (40):

(40) John tried to look up the meaning of green ideas.

Required lexical knowledge comprises at least part-of-speech, phrasal verbs, countability, availability of plural, morphological information, attributive vs. predicative usage of adjectives as well as verbal, adjectival and nominal argument structures including transitivity, subcategorization of prepositions and sentential complements. Because openness is an inherent property of the lexicon (section 1.1), any instantiation of it is incomplete from the

outset, which means that the required resources are only *partially* available. The call for robustness makes it necessary to build in devices that take care of words in the system's input that are not covered by the current lexicon. These can be named entities like the subject of the matrix clause in (40), nonce words, neologisms or even common lexemes that have not been accounted for during the development of the lexical component. The English Resource Grammar (ERG) as of version of July 2003, for example, defines around 10500 lexical items as opposed to the Oxford Advanced Learner's Dictionary (OALD) (7th edition) which comprises more than 180000 entries[1]. Openness, however, is only one side of the issue. The other one is defectiveness (section 1.1). Firstly, manual introspection does not guarantee that all properties of all lexemes are detected. Secondly the set of distinctions established at the design-time might miss out a property that comes into scope later on, calling for an expensive revision process. This goes in line with the relativity of the lexicon (section 1.1). For example, many of the distinctions made by ERG are not covered by the OALD[2].

In summary, state-of-the-art deep processing systems such as PET plus ERG leave the potential user behind with a non-trivial problem.

The automatic or semi-automatic detection of lexical properties has therefore become a line of research in computational linguistics, the various facets of which have been discussed in chapter one. A very broad but useful distinction separates accounts in which lexical information is acquired using components of the target system itself (*in vivo*) or completely isolated from it (*in vitro*) (Baldwin 2005c). In this thesis we argue for a radical position: deep lexical acquisition (DLA) should not only be *in vivo*, it should be fully integrated into the parser system and considered at the design-time. Ideally the introduction of a (new) lexical feature into the grammar should be accompanied with either a reliable procedure of detecting it or, even better, should be proven to be a logical by-product of the system's linguistic processing of text. This must include lexeme detection (cf. section 2.7). There should not be a principle difference between missing ('unknown') lexemes or missing properties of known lexemes. Everything that is missing should be able to be induced automatically, first during training and later on 'on the job', accounting for the claim that lexicon acquisition based on stationary resources has to give way to more dynamic systems (Briscoe 2001). Our claim is formulated in terms of the Learn$-\alpha$ Design Rule (section 2.10), which says

[1] One entry in the OALD does not exactly correspond to one lexical item in the ERG, but the difference is striking.

[2] Of course, the OALD is not an NLP lexicon. The point is that even if it were somehow transformed into a deep lexical component, some important information would not be included.

that the *design* of a natural language parser system should enable it a) to induce *efficiently* and *precisely every* lexical feature α of *every* lexeme based on the learning opportunities it encounters and b) to revise decisions if they cannot be further maintained in the light of new experience.

The design rule raises (at least) three questions, which are the driving force of this work. Firstly, can the lexicon be a natural fall-out and logical consequence of the system's a-priori knowledge (its grammar and initial lexicon) taken together with its experience (section 1.2)? The answer to this question is difficult to explore due to its generality. In this thesis a first attempt is made to pursue it with a much more narrowed, but still challenging second question, which in addition incorporates the intended application task: how can *machine-readable unannotated texts* of *English* be exploited *automatically* to acquire missing information about *morphological, syntactic and semantic* aspects of the *lexical component* of an natural language processing (NLP) system in order to improve its *disambiguation performance*? The answer comprises two complementary sets of statements: all the assumptions made and considerations explored in the theory of Learn−α on the one hand (chapter 2 and section 5.2 below) and the blueprint of a practical implementation on top of a contemporary NLP system on the other (chapter 3 and section 5.3 below). Finally, in case of a positive answer, a third question is of high interest: how well does the automatic induction of lexical features work in practice? This includes considerations of acquisition performance and quality especially in the bootstrapping phase as well as the identification of general limits.

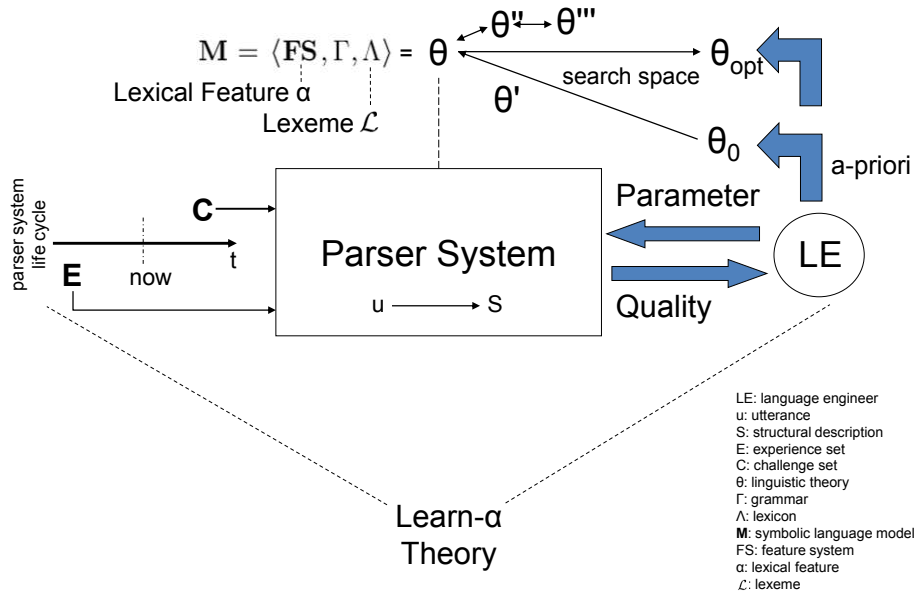
5.2 Learn−α

The theory of Learn−α developed in chapter two concerns the *logic* of lexical acquisition. Learning is unsupervised as the input is raw text. The theory is not bound to any particular grammar formalism. Its scope and general layout is depicted in figure 5.1: In the center of the theory stands a parser system that maps incoming utterances (u) to linguistically motivated structural descriptions (S). Along the life cycle of the system the utterances fall into two complementary sets: the experience (E) and the challenge (C). The structural descriptions are derived from the application of the system's model of language **M** (D6, pg. 90[3]), which comprises the feature system **FS** (D2,

[3]Here and in the following we refer to elements of Learn−α in the format '*En*, pg. *page*' where E is 'D' for definitions, 'A' for assumptions and 'T' for theorems and n is the element's number. We hope to provide a more comfortable way of providing links back to

pg. 81), the grammar Γ and the lexicon Λ. Logically, the union of these two sets of statements forms a theory Θ. Because the system is able to vary Θ it makes sense to speak of the system's *current* theory. The feature system is the theory's ontology and includes, among others, the definitions of lexical features α (D8, pg. 91).

Figure 5.1: Learn−α's Scope



It is assumed (A2, pg. 87) that lexical features are arranged hierarchically to express *is-a* relationships such as 'a count noun is a noun', 'a phrasal verb is a verb' etc. The 'backbone' made up of these subsumptions is called lexical type hierarchy (LTH). Note that in the theory lexical features are marked as such by being subsumed by a most general lexical type which is the root of the LTH. A lexical feature is defined as a triple $\alpha = \langle \sigma, \psi, F \rangle$ specifying a feature name F, the type σ to which the lexical feature belongs and the *realization context* ψ in which it is embedded. By this context-sensitive definition it is possible to use feature names in different contexts. Subcategorization of prepositions, for example, can be introduced as a lexical feature on the top level of the LTH splitting into distinct lexical features down the hierarchy: PP-subcategorization of phrasal verbs versus PP-subcategorization of nouns

the theoretical chapter.

etc. In this way the interpretation of a lexical feature hinges on the type of the particular lexeme to which it refers.
For every lexeme $\mathcal{L}$ the lexicon Λ entails the lexical type(s) it belongs to and for every type it entails the admitted feature value combinations (an admitted feature value is notated as $\alpha^{\mathcal{L}}_{\mathbf{v}}$, see page 91). By a *closed-world assumption* values that are left out are considered disallowed (*lexical constraint*) with the exception that a lexical feature (for a particular lexeme) may be fully underspecified, which means that no value, apart from the most general value appropriate for the corresponding type, is specified and no corresponding lexical constraint is imposed. In the most extreme case even the *ultimate realization context* (A3, pg. 92), the feature responsible for the lexeme's spell-out, is underspecified, which means that the lexeme is unknown to the system as a whole. It is underspecification that fosters lexical acquisition. It is further assumed that lexical acquisition is restricted to open word classes (A4, pg. 93). Because Γ is kept constant, variation of Θ concerns the amendment of the lexicon Λ in course of *incremental acquisition* (section 2.3.2). All possible type/feature settings for all lexemes that come into existence during the system's life cycle make up the search space (*lexical acquisition space*, page 106), which in addition to the current theory encompasses two further salient theories: the initial theory Θ_0 and the optimal theory Θ_{opt}. While the former is the most underspecified of all theories in the acquisition space, comprising a-priori knowledge only, the latter includes the least number of underspecifications possible, yielding maximal precision. A-priori knowledge encompasses both the grammar and the *initial lexicon* and is provided by the language engineer (LE) who is the system's ultimate authority regarding linguistic truths[4] (section 2.2). The optimal model is a theoretical artifact that is intended by the LE but not necessarily in reach. It should be obvious that lexical acquisition cannot make up for unintended effects that have their root cause in Θ_0, hence the system is allowed to assume initial linguistic correctness of the initial model (A5, pg. 101 and T1, pg. 96). In an *ideal world* scenario (without noise) the system may further assume the grammaticality of the input utterances with regard to the initial model (A6, pg. 102). This assumption needs to be relaxed for the real world. In general, however, noise must not exceed a certain level, otherwise sensible learning may be questioned completely (A1, pg. 130).
The discussion of figure 5.1 concludes with the remark that the LE is the one who parameterizes the system and who judges its parser *and* acquisition quality.

[4]The LE is to be considered an abstract element of the theory, not a single person.

In Learn−α the parser system is upgraded to a lexical acquisition system. Lexical feature values are induced by the application of two reasoning schemes: the *generalized observation schema* (pages 105 and 147) and the *induction schema* (page 109). The first allows the system to exploit lexical information entailed in single utterances to the maximum extent - by harvesting feature value settings in the related structural descriptions. The utterance may thus become a *reference* (D23, pg. 104) for a particular $\alpha^{\mathcal{L}}_{\mathbf{v}}$. Observations of the lexeme's feature value realizations (D25, pg. 111) are categorized regarding the degree of ambiguity. The higher the degree of ambiguity, the weaker the evidence provided by the particular utterance motivating the notion *Relative Degree of Ambiguity (RDoA)*. During exploitation of an utterance the system works itself down the LTH to find out which feature values can be observed under which RDoA_{max} (*learning down the hierarchy*, subsection 2.8.1). Because lexical feature value realizations are often neutralized at the surface (section 2.5), ambiguity is one of the major obstacles to lexical acquisition. In the worst case it prevents feature values from being unambiguously detected at all (*total neutralization*, D35, pg. 136 and T3, pg. 145).
In an ideal, noise-free world without total neutralization the detection of one utterance which unambiguously proves a particular feature value realization of a particular lexeme already suffices to incorporate this finding into the lexicon. To prove that a lexeme admits a certain feature value, however, is not the point. What matters is the induction of lexical constraints: the fact that a given lexeme does *not* admit a particular value for a given (appropriate) lexical feature. Because the system's input lacks negative evidence, the system must be enabled to make decisions based on statistical inference on grounds of the induction schema. It will then induce the constraints based on the *absence* of evidence in a set of *learning opportunities* (LOs, D38, pg. 183) which must be large enough to be significant. This invokes a last - and critical - assumption: lexical feature values are realized with statistical significance (A9, pg. 146). If they are not, the system will not detect them. For the formulation of a *null hypothesis* the system has to assume that every lexical feature is at least realized with a probability p^{fv}_{min} (section 2.3.5), a parameter provided by the LE. In extreme cases it might even be that none of the appropriate values might be observed despite a sufficiently large number of LOs so that a default value may safely be induced (default induction rule on page 188). The theory requires from the implementation that it keeps track of all the reference counts (D37, pg. 181) and related numbers subsumed under the heading *feature value characteristics*.

Because the real world is noisy (section 2.4) the system might be misled by a significant number of *unintended references* (D31, pg. 129) which means

that hypothesis testing must take the effective noise level (D32, pg. 129) into account. This affects assumption 9 mentioned above: lexical features must not only be 'statistically significant', they need to be realized with a probability significantly higher than the noise level. Otherwise they get swallowed by the *noise barrier* (subsection 2.4.6). If the system is expected to learn autonomously then it has to estimate the noise level on its own based on the observed feature value characteristics. Subsection 2.8.5 suggests an algorithm for the automatic noise level estimation (ANLE).

The Learn−α design rule demands the highest precision possible with the consequence that any implementation has to induce lexical features under RDoA_{max}=1 if feasible. Because the structure of the grammar (encoded in Θ_0) can, however, render certain lexical features indeterminable given this restriction, the demand that *every* feature must be inducible leads to the requirement that those features invisible under RDoA_{max}=1 have to be learned with relaxed restrictions. The Learn−α theory therefore introduces the *system's focus* which is automatically adapted according to the characteristics of the feature. During hypothesis testing induction of a feature value $\alpha^{\mathcal{L}}_{\mathbf{v}}$ is tried with highest precision (RDoA_{max}=1) first. If the result is negative, the feature value gets 'another chance' under relaxed conditions (RDoA_{max}=2 followed by RDoA_{max}=∞)[5] - if the system has evidence that relaxation makes totally neutralized values visible (T6, pg. 187). This procedure is called *cascading focus* (section 2.8.8). Testing with RDoA_{max}>1 usually leads to an increased noise level. 'Noise' is then no longer restricted to mistakes but additionally refers to alternative feature values in ambiguous situations, which means that the feature value must be realized significantly more often than it appears to be realized on the average. This increased noise level is estimated by the ANLE algorithm, too. Again, assumption 9 comes into focus: this time the realization of the feature value in question must be statistically more significant than the average realization frequency of the values that neutralize it.

The statistical inference and decision-theoretic part of Learn−α is presented in a series of theorems (T7, pg. 193 - T9, pg. 195). Induction of a lexical feature α starts when the system's experience set contains a sufficient number of learning opportunities for it. The threshold is calculated based on the feature value characteristics and the two system parameters β_{FP} and β_{FN} which determine the size and power of the hypothesis test. Hypothesis testing is not restricted to binomial hypothesis testing (BHT); other testing methods such as likelihood ratio test (LRT) can be plugged in, too. In any

[5] See T4, pg. 149, for the reasoning behind choosing RDoA_{max}=2 as a salient constraint.

case, the p-values related to the particular tests are retained as metadata of the induced feature values. They represent the *strength of belief* related to the decisions made by the system. Decisions can thus be classified as 'stable' (beyond doubt), 'likely', 'instable' etc. and it is reasonable to partition the lexicon accordingly (section 2.8.7).

Section 2.9 explains how belief *revision* theory is incorporated into Learn$-\alpha$. Two scenarios of revision are discussed: the first covers the situation in which a particular utterance (that is considered grammatical) cannot be parsed. Because this failure *can* be due to an incorrectly induced lexical feature value, the parser may try the analysis again, this time with temporarily relaxed lexical feature constraints. If parsing is successful this is a proof that at least one of the inductions is mistaken (*immediate repair*). It is suggested that relaxation takes the stability of decisions into account, which can then be considered *entrenchment-based contraction*. Stable parts of the lexicon may even be considered irrevisable (*shielded AGM contraction*).
In the second scenario the system considers the whole experience set and tests whether values of fully underspecified lexical features can be induced, which leads to an *expansion* of Θ or whether induced feature feature value admissions or constraints need to be revised enforcing *contraction*, which should be entrenchment-based in general (T10, pg. 203). For reasons of efficiency stable decisions might be shielded here, too. In addition, we suggested the implementation of a *skeptical learner* (D43, pg. 205). Such a system may be considered 'skeptical' because it distrusts some of its own decisions. It will apply the general observation schema based on 'shadow analyses' that are produced ignoring the instable feature values. This might limit the danger of introducing noise into the system due to shaky decisions.

5.3 The Prototype

In chapter three we developed the architecture and specification of a framework that satisfies the Learn$-\alpha$ Design Rule: ANALYZE-LEARN-REDUCE. This framework is tailored to unification-based grammars (cf. *learning by unification*, subsection 1.4.5), but is still independent of a particular formalism. The prototype implemented on top of it, however, uses the Head-driven Phrase Structure Grammar (HPSG) precision grammar ERG. It demonstrates how a state-of-the-art NLP system - the *Heart of Gold* - can be upgraded to a lexical acquisition system. The module's devision of labor is aligned with the cornerstones of Learn$-\alpha$: the ANALYZE module parses the incoming utterances as in the original system (D7, pg. 90), but additionally ensures that the structural descriptions are made available for subsequent

lexical acquisition. The LEARN module realizes the *generalized observation schema* by harvesting the enriched lexical feature structures (*lexical seeds*) tapped from the ANALYZER's output. See section 1.5 for illustration. The REDUCE module realizes the *induction schema* by implementing the statistical inference theorems mentioned above and exports new lexical entries to the lexicon. It is also able to disambiguate utterances in the experience set retroactively in order to reach maximal exploitation (*reassessment of experience*, cf. subsection 3.1.1). All relevant information is retained in a relational database, called 'BELIEF DB', which enables the LE to trace how the induction of lexical features is actually performed.
It was argued that HPSG is not optimal for the implementation of Learn$-\alpha$, which indicates that the Learn$-\alpha$ design rule should already be accounted for at the design-time of a computational grammar formalism. To make ERG compliant to the theory, we introduced the *lexical seed layer* (section 3.2) as a new base layer of the grammar. This layer specifies the LTH and can be considered the 'language' of lexical entries which may be *arbitrarily* underspecified, thereby facilitating the bootstrapping experiments reported in the fourth chapter. Lexical seeds, the building blocks of the lexicon, are mapped to ERG lexical types directly after lexical look-up. The layer encapsulates the complexity of the grammar from the lexical acquisition problem with the effect that only a very few changes to the grammar, apart from the new layer, are necessary.
The *unknown word* handling of the PET parser is fully integrated, too. Unknown words are mapped to extremely underspecified lexical seeds in the course of processing: nothing more than the part-of-speech is encoded in them.

5.4 Experimental Results

The prototype was evaluated in a medium-scale experiment (chapter 4) with more than 10 features and around 50 lexemes on a subset of the British National Corpus (BNC). As the initial seed lexicon did not comprise more than the part-of-speech (PoS) classification of nouns, verbs and adjectives listed in WordNet, the system under test was in a bootstrapping phase. Adverbs were treated like closed-class items and partially taken over from the ERG's original lexicon.
The experiment pursued two goals, one from a theoretical and one from a practical perspective (cf. section 4.1). The theoretical side attempted to find answers for the question of whether there are sufficient cues in the grammar combined with the closed-class lexicon for the system to learn lexical

features in an effective manner. The outcome gives hints on the theoretical issue of whether such a system has chances to escape the vicious circle of lexical unknownness. The other question of concern is how much a statistical, hypothesis-testing approach accounts for the detection of lexicalized items. From the engineering perspective it was an effort to stress-test the prototype in a close-to-worst-case situation in order to test non-functional criteria such as resource consumption and robustness under hard conditions. Related to this was the constraint of supplying the system with utterances limited to a maximum length of 12 tokens. Utterances of this length, however, already make up 30% of the s-units in the BNC.
Apart from this, due to the radical position adopted by the Learn$-\alpha$ the experiment was motivated more by theoretical considerations than by practical questions so that effectiveness takes priority over efficiency.

The experiment revealed that ALR is able to update the lexicon with new (unknown) single-word as well as multi-word lexemes (section 4.6). Not too surprisingly, this update is accompanied with noise (false positives) mainly due to the lack of a component that accounts for lexical semantics. The noise level, however, is lower than we have initially expected.
The induction of feature values is of varying quality: it ranges from unproblematic feature values (e.g. count/mass noun categorization, transitivity) over those that have some difficulties (PP selection, verb particle combinations) up to those that are hardly learnable (e.g. complementizer-less sentential complements), at least during bootstrapping. Here the account is clearly outperformed by *in vitro* methods with hand-made cues. The experiment demonstrates that with the algorithms developed here recall generally takes priority over precision. It also gives a feel for the potentials and limits of a mere morpho-syntactic and statistical account of lexicon acquisition.
Finally it was shown that the learning cycle had a significant effect: around 17% of the utterances that exceeded the 1000-readings threshold before learning could be analyzed with less than 1000 readings after the lexicon was updated.

We conclude this section with a list of observations that show how the various ideas suggested in ALR and implemented in the prototype hold water in a practical test:

realization contexts: the fine-grained nature of features embedded in realization contexts was exploited in some cases (the noun `part`, for example, must be used with a preposition in the uncountable context, cf.

section 4.8)

retroactive disambiguation: disambiguation during the *reassessment of experience* lead to higher precision (see the example of `way` in section 4.7)

system's focus: the system automatically adapted its focus when the contexts became more precise after retroactive disambiguation (see section 4.5)

ANLE: the automatic noise level estimation worked reasonably well and was responsible for many of the true negatives

references: references cited by the system help investigate whether the decisions are made on ground of correct examples or by mere coincidence (cf. subsection 2.1.6); they also form a good device for the detection of systematic deficiencies (e.g. the case of `other` in section 4.12)

5.5 Conclusions

In this thesis we argued for a radical position expressed by the Learn$-\alpha$ design rule: NLP systems should be designed in a way that they autonomously acquire information missing in the lexicon. To our knowledge no such system has been developed yet.
The three major obstacles for the automatic induction of lexical features are sparse data, noise and lexical neutralization. In this research we focused on the last two challenges hoping that the first issue can be tackled with the use of ever larger online resources (e.g. *Web as Corpus*).
The design rule is supported by a general theory of automatic lexicon acquisition which outlines the logic and assumptions underpinning it. The theory starts with the definition of *lexical feature*, defines an observation schema which accounts for the maximum exploitation of learning opportunities, exposes a UMP-based statistical account for induction and ends with the embedding into belief revision theory.

The theory is implemented in the new NLP framework called ANALYZE-LEARN-REDUCE (ALR) which turns parser systems into lexical acquisition systems. This framework facilitates experiments for the investigation of dynamic lexicons. As a side effect it can also be used to improve dictionaries and to detect systematic deficiencies in the grammar employed.
A first medium-scale bootstrapping experiment conducted with an ALR prototype demonstrated that the system can effectively learn despite massive

lexical underspecification. It also shows limits of a mere morpho-syntactical account.

Both the theory and the ALR framework are under active development. We think that this thesis is a first step on the roadmap to autonomously learning NLP systems. The next and final section suggests an agenda for future research.

5.6 An Agenda for Future Research

We conclude this thesis with a few remarks on what we think could be some promising topics on the agenda of future research.

In one line of research the theory of Learn$-\alpha$ should be extended by additional, may be analogy-driven observation schemas. It will be worthwhile to analyze whether successful *in vitro* methods can be generalized and integrated. A case in point is the exploitation of the excellent PoS categorization of state-of-the-art taggers (cf. the final remark concerning the acquisition performance of verb particle combinations in section 4.10).
One promising aspect will be the setting of defaults based on the overall behavior of semantic classes - in analogy to the work of Korhonen (2002).

In another line of research deep lexical acquisition should be combined with acquisition by shallow analysis. In analogy to robust minimal recursion semantics (RMRS) which is the common semantic representation schema for the various Heart of Gold (HoG) modules a common LTH could be developed so that different modules (shallow, deep) can contribute to the same lexicon.

The effects of incremental knowledge acquisition are particularly worth exploring. How will the feature value characteristics change over time? How will the REDUCE module account for these changes? These experiments require multiple learning cycles.
Will a system that is skeptical (cf. definition 43) outperform a system that is not?

We would finally like to suggest further research topics along the dimensions proposed in the introduction:

linguistic precision: The improvement of the language model is certainly an ongoing process. It is not only noise in the grammar that challenges the constructed lexicons, but the language model in its current version is also a bit too lax and allows for many spurious ambiguities which

may be avoided. We expect that a higher precision will lead to less neutralization and a higher learning efficiency. In addition it would be interesting to follow up with the suggestion to take the ranking of the tagger into account as suggested on pages 293 and 302, respectively.

linguistic phenomenon: Here we see the most promising line of research, namely the integration of semantics and pragmatics into the system. How will the system perform at the acquisition of collocations, for example? How should idioms be dealt with? How can neutralization be decreased by exploiting semantic and may be even world knowledge? Can semantics/pragmatics ease the sparse data problem? How can word sense disambiguation be integrated?

parser system component: Can the ALR system be modified in such a way that not only the lexicon but also grammatical facts can be acquired?

investigated language(s): How does ALR work for other languages?

investigated sublanguage: How will ALR perform for other text sorts?

language source: How will ALR perform on speech input? Can we supplement the classic corpus approach with a scenario in which the system proactively and in a self-organized manner looks out for learning opportunities in the world wide web? For example, the system might constantly monitor lexeme frequencies as well as the associated learning efficiency metrics (as a function of utterance complexity) and, based on that, compile training sets on its own. It will be also interesting to see whether the results of *partial parsing* can be meaningfully exploited without introducing too much of noise.

initial knowledge: What is the impact of additional initial knowledge? Is there an optimum with regard to the cost/performance ratio?

statistical model: The ALR framework may be tested with alternative statistical models in order to find optimal solutions for different lexicon acquisition tasks.

method of evaluation: Can we think of more adequate evaluation procedures? When features are put to the test which have no counterpart in any available resources, how is the quality of acquisition to be measured?

The pursuit of all these questions will gain valuable insights into the automatic induction of lexical features.

Appendix A

Probabilistic Preliminaries of Learn$-\alpha$

In order to arrive at a probabilistic account of Learn$-\alpha$ we have to define the *probability space* $\langle \Omega, \mathcal{F}, P \rangle$ that properly models the probabilities of observing particular references $\mathbf{M} : u \mapsto \Box\alpha^{\mathcal{L}}_{\mathbf{v}}$ as the basis of the induction step. Ω is called the *sample space*, $\mathcal{F}$ is the σ-field of *events* and P the *probability function* that maps events in $\mathcal{F}$ to probabilities. Recall that both the observation step and the induction step take into account one particular combination of lexeme $\mathcal{L}$ and feature α at the same time. This means that the probability space is to be defined for such a particular combination. It has to model the probability of seeing a particular observation in the system's experience set $E^{\mathcal{L}}_{\alpha} \subseteq E$ which is thought to be the set of all experienced utterances in which $\mathcal{L}$ realized one of α's values. An *experiment* (or *trial*), a fundamental notion for the definition of the probability space, can be seen as the process of picking out one $u \in E^{\mathcal{L}}_{\alpha}$ at random and observing which feature value $v_i \in V(F)$ for $\mathcal{L}$ and α has been realized. Let us say that v_i is the *basic outcome* of the experiment. If there is only one trial, then Ω is simply equal to V(F). However, we are interested in events that are made of sequences of trials, especially when the whole set $E^{\mathcal{L}}_{\alpha}$ is considered, which can be seen as the sequence of $n = |E^{\mathcal{L}}_{\alpha}|$ experiments. In this case, Ω is the set of all possible permutations of $v_i \in V(F)$, for example, if n=2 and $V(F) = \{v_1, v_2\}$ then $\Omega = \{v_1v_1, v_1v_2, v_2v_1, v_2v_2\}$. If we choose $\mathcal{F} = 2^{\Omega}$ then each subset of Ω is an event. Very often we are interested in the question how probable it is to observe v_i exactly k times. Using the example above, if we want to know the probability of seeing v_1 exactly 1 time, this amounts to the question of P(A) with $A = \{v_1v_2, v_2v_1\}$. If the distribution of the basic outcomes in the example's sample space would be uniform then the probability could be easily determined analytically: $P(A) = |A|/|\Omega| = 2/4 = 0.5$.

However, Learn$-\alpha$ does not tell us the probabilities of the basic outcomes. On the contrary, during the induction step the learning system should *infer* probabilities of certain events from the experience. In the ideal, noise-free world the system should for example be able to state with a certain level of confidence and based on E that $\mathrm{P}(\mathrm{A}) = 0$ for all A that include an occurrence of a particular feature value. This process, called *parameter estimation*, is dealt with in *statistical inference*, which will be the topic of this section and of section 2.3.5 in the main text.

A.1 Relative Frequencies and Maximum Likelihood

Central to the frequentist's tenet of probability is the claim that the observed *relative frequency* of an event will approximate the probability of the event and that the approximation will get better with an increasing sample size. Moreover, the event's relative frequency of an infinite sample will be exactly the probability of that event. However, on a finite sample there is always some random fluctuation. The relative frequency (rf) of a feature value realization in our account is simply $rf(v_i) = c(v_i)/n$ where $c(v_i)$ is the number of observed realizations of v_i and n the size of the sample $E^{\mathcal{L}}_{\alpha}$. Based on a maximum likelihood estimate (MLE), we can state that it is exactly $\hat{p}(v_i) = rf(v_i)$[1] that is the best estimate of the underlying probability rendering the observation of the related sample most likely. If we repeat the experiment with a different sample $\overline{E^{\mathcal{L}}_{\alpha}}$, all else being equal, we *expect* v_i to occur $\hat{p}(v_i)n$ times. But because of random fluctuation the observed value $c(v_i)$ might deviate from the expected value. Hence it is interesting to know how likely a particular deviation actually is. This can be perfectly modeled with a *random variable*. A random variable is a function $X : \Omega \mapsto \mathbb{R}$ that assigns real valued numbers to basic outcomes of the experiment[2]. The value of the random variable is determined by counting the number of occurrences of v_i in the outcome of the experiment. Let us use X_i to denote the random variable related to v_i[3]. Here, X is called *discrete*, because it maps the sample space to a countable subset of $\mathbb{R}$. That is, the random variable numerically represents the possible outcomes of the sample $E^{\mathcal{L}}_{\alpha}$. In general, a sample is a vector $n\langle X\rangle$ of n independent trials all based on the same distribution. n is called the *size of the sample*. In order to comply with this definition let us state

[1] We will use $\hat{p}$ for estimated probabilities.

[2] Or more generally $X : \Omega \mapsto \mathbb{R}^n$. We restrict n to 1 in this discussion.

[3] So in the example above, $X_1(v_1v_1) = 2$ etc.

that $\Omega = V(F)$ and that the possible events are simply the basic outcomes (the feature value realizations)[4]. If $|V(F)| = |\Omega| = 2$ the n experiments leading to the sample are called *Bernoulli trials* and X is called *Bernoulli random variable*. Otherwise X is a *categorical random variable*, which is the general case for Learn$-\alpha$.

A.2 Binomial, Multinomial Distribution and Parameter Estimation

The distribution of a discrete random variable is characterized by its *probability mass function* (pmf). In the case of a *univariate distribution* the pmf yields the probability that the variable takes a certain value: $f_X(x) = Pr(X = x)$. We use the notation $X \sim D$ to express that X is D-distributed. In the binomial case ($|V(F)| = |\Omega| = 2$) the variable has a *Bernoulli distribution*. The sum of n Bernoulli random variables is a random variable with a *binomial distribution*, characterized by the underlying probability p and n (we write $\mathbf{B}(p, N)$):

$$f_X(k; p, n) = Pr(X = k) = \binom{n}{k} p^k (1-p)^{n-k} \quad \textbf{if} \quad X \sim \mathbf{B}(p, N) \qquad \text{(A.1)}$$

The cumulative distribution function (cdf) $F_X(k)$ yields the probability that the variable takes the value k or any value below k and is simply the sum $\sum_{i=0}^{k} f_X(k; p, n)$.
If the random variable has a categorical distribution, its input is a vector of d basic outcomes $\mathbf{x} = \langle x_1, \ldots, x_d \rangle$ and the variable is a random vector $\mathbf{X}$ of dimension $d > 2$, which is the case in *multivariate distributions*. Transformed to a variable that represents the outcome of n trials, the analog of the binomial distribution is called *multinomial distribution*. It is characterized by a tuple of probabilities $\mathbf{p} = \langle p_1, \ldots, p_d \rangle$, where each p_i is the probability of observing one of the possible (categorical) values in the sample space, subject to the condition that $\sum_i p_i = 1$. Let us denote this distribution with

[4]Note that in a binomial setting it makes no difference whether there is only one random variable representing the complex events (permutations of n basic outcomes) or n binomial random variables with values 0 and 1. The reason for this is that the *sufficient statistics* for the sample in the latter case is just the sum over all observed values of the variables in the sample, cf. Weerahandi (1995: 51). The same argument holds for the multinomial case.

$\mathbf{M}_d(\mathbf{p}, N)$. In the case of lexical features, each $v_i \in V(F)$ has a probability $p_i^{\mathcal{L}}$ (or p_i for short, if the context is given) of being realized by the lexeme $\mathcal{L}$. The pmf is:

$$f_{\mathbf{X}}(\mathbf{x}; \mathbf{p}, n) = Pr(\mathbf{X} = \mathbf{x}) = \frac{n!}{\prod_{i=1}^{d} x_i!} \prod_{i=1}^{d} p_i^{x_i} \quad \textbf{if} \quad \mathbf{X} \sim \mathbf{M}_d(\mathbf{p}, N) \tag{A.2}$$

Probability distributions are used to model processes or state of affairs that inhere random components. One of the *parameters* of these models are the underlying probabilities. Often these parameters are unknown (as is the case in Learn$-\alpha$), in which case those that are of a particular interest have to be estimated. The other parameters, called *nuisance parameters*, have still to be accounted for during estimation. The parameter θ is element of the *parameter space* Θ. During *parameter estimation* the model's formulae are treated as functions of their parameters θ given a certain observation $\mathbf{x}$. These functions are the *likelihood functions* of a model given $\mathbf{x}$. In the binomial case, for example, the pmf $f(x; p, n)$ can be turned into the likelihood function $L(\theta|x, n)$, where θ is the underlying probability of the distribution and x the observed value. In *maximum likelihood estimation* a parameter value $\hat{p}$ is computed that makes the observed data most likely. It can be shown that the maximum likelihood estimation x/n is the best estimation for θ in the binomial setting (cf. the discussion at the outset of section A.1). The same holds in analogy for the estimation of the multinomial probabilities (for a proof see Lehmann and Romano 2005: 515):

$$\hat{p}_i = \frac{x_i}{n} \tag{A.3}$$

A.3 Confidence Intervals and Confidence Regions

The estimated parameter value of a statistical model can vary from sample to sample, hence it is a random variable by itself. In order to indicate how reliable a certain estimation is, a confidence interval (CI) can be computed. The estimated value is then said to lie within the interval with a certain

confidence level $1 - \alpha_s$[5].

For binomial parameter estimation based on an observed x and n trials (with MLE $\hat{p}$) there are multiple ways to compute the CI. One of the most popular methods, the *Wald interval*, is known to perform poor if the estimated probability is near 0 or 1 or when n is small (Brown et al. 2001: 101). Even worse than that the coverage probability shows erratic (oscillating) behavior even when n is not so small[6]. Borrowing the notation used in Brown et al. (2001), we abbreviate the $1-\alpha_s/2$ percentile of the normal distribution with the letter κ. The standard (Wald-) CI is:

$$CI_{Wald} = \hat{p} \pm \kappa n^{-1/2}(\hat{p}(1-\hat{p}))^{1/2} \tag{A.4}$$

For sample sizes ≤ 40, Brown et al. (2001) recommend the *Wilson interval* based on Wilson (1927), given in A.5 or alternatively the *equal-tailed Jeffreys prior interval* given in A.6. The last interval uses the alpha quantile $B(\alpha_s, m_1, m_2)$ of the Beta distribution $\mathbf{Beta}(m_1, m_2)$.

$$CI_{Wilson} = \frac{x + \kappa^2/2}{n + \kappa^2} \pm \frac{\kappa n^{1/2}}{n + \kappa^2}(\hat{p}(1-\hat{p}) + \kappa^2/(4n))^{1/2} \tag{A.5}$$

$$CI_{Jeffreys} = [B(\alpha_s/2, x+1/2, n-x+1/2), B(1-\alpha_s/2, x+1/2, n-x+1/2)] \tag{A.6}$$

For larger sample sizes, the authors recommend the *Agresti-Coull interval* based on Agresti and Coull (1998):

$$\begin{aligned} &CI_{Agresti-Coull} = \tilde{p} \pm \kappa(\tilde{p}(1-\tilde{p}))^{1/2}\tilde{n}^{-1/2} \quad \textbf{with} \\ &\tilde{n} = n + \kappa^2, \quad \tilde{p} = (x + \kappa^2/2)/\tilde{n} \end{aligned} \tag{A.7}$$

All these alternative CI formulae have been demonstrated to show less oscillation and to yield more reliable intervals.

[5]Usually the significance level is denoted with α. With the subscript we explicitly denote the significance level, see section 2.3.5.

[6]cf. the demonstration for $25 \leq n \leq 100$ and $p = 0.02$ in Brown et al. (2001: 104). In another impressive experiment with $p = 0.005$ the coverage probability seems to increase normally before it starts to oscillate at $n = 591$.

The multinomial case is more complicated because the estimated probabilities $\hat{p}_i$ are interdependent and so their confidence intervals cannot be computed in isolation, rather a *confidence region*, a multi-dimensional generalization of the CI, has to be calculated instead. A confidence region[7] is a subset of the parameter space Θ so that on average it covers the estimated parameter in $1 - \alpha_s$ percent of the repetitions of the experiment. It is modeled geometrically by a sets of points in a d-dimensional space, subject to the condition that the point coordinates add up to 1, hence a $d - 1$ *simplex*.
Sison and Glaz (1995) report approximations to confidence regions that appear to work reasonably well, which means that the coverage probability in simulations is close to the prescribed confidence level (0.95) and that the volume of the confidence region is small[8], which are the two most prominent indicators of confidence region quality. They outperform methods such as those provided by Goodman (1965) or Fitzpatrick and Scott (1987). While the first shows poor coverage in some simulations, the second seems to be too conservative resulting in a higher volume[9].
In any case, as with the binomial case, asymptotic approximations are less reliable for small sample sizes. A solution for this problem is proposed in Chafaï and Concordet (2009). This method guarantees a prescribed lower confidence bound while keeping the volume of the confidence region as small as possible, but is computationally more complex. We leave it to the implementation of Learn$-\alpha$ which method is to be chosen.

[7] In the literature one also finds the notion *simultaneous confidence intervals*.

[8] See also Hou et al. (2003) for calculations based on the Monte-Carlo technique.

[9] Because of their simplicity the formulae based on theorem 1 in Fitzpatrick and Scott (1987) are very attractive if a minimum volume is not required. The intervals are just: $\hat{p}_i \pm 1/\sqrt{n}$ for a simultaneous coverage probability of 0.90, $\hat{p}_i \pm 1.13/\sqrt{n}$ for 0.95 and $\hat{p}_i \pm 1.40/\sqrt{n}$ for 0.99, respectively.

Appendix B

Linguistic Background

B.1 Features: Applications and Development

Features and feature structures have their origin in the Prague school's work on phonology (Jakobson et al. 1952) and have subsequently been fine-grained in linguistics and its neighboring disciplines of logic and computer science (albeit sometimes put under different headings). We cannot give a complete survey on the history of feature theories, but we want to mention some important changes to the original concept of features along with some of the most relevant logical and ontological questions. In phonology, features were (and are still) used to distinguish different phonological segments using smallest descriptional units like ±VOICE, so-called distinctive features. Those features are used to define natural classes of sounds - classes of sounds that share at least one phonetic (acoustic and articulatory) property. Their distinctiveness is empirically based on word meaning. They are typically binary in the sense that there are two possible values: $V(f) = \{+, -\}$, with a straightforward interpretation, in our example: +VOICE means that the sound is voiced and -VOICE means that it is not voiced (or voiceless)[1]. A phoneme is then a bundle of phonological features, a so-called attribute value matrix or - more general - a *feature structure*. The employment of these structures has become wide-spread in phonology since Chomsky and Halle (1968).
In analogy, the concept of features has later been transferred to other levels of linguistic descriptions: semantics, syntax etc. and to linguistic entities much larger than sounds - so it became possible to describe the meaning of *sentences* based on feature structures (Katz and Postal 1964).
While at the beginning feature values were restricted to (binary) atomic

[1]Note that -VOICE does not mean that the feature VOICE is not appropriate for the sound in question. Also note that in this notation the feature value precedes the feature.

values, this restriction has been dropped and even feature structures were allowed to be values of features, leading to a recursive definition.
In the early days of feature-based theories, the logic, semantics and ontology underlying the feature structures was more implicit than explicit and many subsequent investigations contributed to a deeper understanding of these constructs, posing and answering questions that are vital for logical theories: proper semantics, axiomatizability, soundness, compactness and completeness. Feature structures have been related to first-order logic, modal logic and intuitionistic logic. For a survey, see Rounds (1997). Here we want to mention important properties and extensions of feature structures.

Description Language: A description language allows us to talk about feature structures in a systematic and logical fashion. Feature theories differ with respect to which elements or expressions are allowed in such a language. The interpretation of the structures varies accordingly. As an example, if ϕ and ψ are formulas of the language then $\phi \vee \psi$ is also a formula of that language if disjunction is allowed and $\vee$ is the disjunction operator. A description language is supplemented by its semantics. To stay with the example, we can say that $d \models \phi \vee \psi$ ($\phi \vee \psi$ is true for an entity d) if and only if $d \models \phi$ or $d \models \psi$. This kind of feature logic is often referred to as *Kasper-Rounds logic*, going back to the seminal work of Kasper and Rounds (1986)[2].

Feature Path: Features are functions, mapping an entity d of a domain onto an entity v of a set of appropriate feature values $V(f_1)$. If the value v can in turn be a feature structure, a further mapping of v to a subsequent value v_2 is defined by another function, say f_2. In other words, we can concatenate the functional mappings: if $f_1(d) = v$ and $f_2(v) = v_2$, then $f_2(f_1(d)) = v_2$. The transition from d to v_2 is called a (feature) path π and we usually write $d\pi = v_2$.
v_2 is called the value of the path.

Path Equations: A feature structure (and the according description language) can express that two path values are the same entity. The paths 'point' to the same entity and are said to be *structure shared* and the values are said to be *token identical*. The semantics of Kaspar-Rounds account for this possibility by virtue of the statement $d \models p \doteq q$ **if** $dp = dq$[3]. For example, to express that the agreement features (denoted by agr) of an NP and a verb V are identical, we might establish the path equation NP agr $\doteq$ V agr.

[2]The semantics of a (feature) logical formula in a Kasper-Rounds Logic is stated inductively by conditions under which a deterministic finite automaton $\mathcal{A}$ *satisfies* a formula.

[3]Here we use the condition as presented in Rounds (1997).

Negation and Inequations: Some feature logics allow negation, e.g. Johnson (1988), where it is assumed that $d \models -\phi$ if and only if $d \not\models \phi$. See Carpenter (1992) for drawbacks of this classical treatment of negation, where Carpenter introduces his alternative concept of inequations, reminiscent of the treatment of negation in constraint logic programming systems. 'The import of these inequations is that they ensure that the two terms that are stated to be different never become identical after the inequation is encountered.' (Carpenter 1992: 110).

Subsumption and Unification: As feature structures may contain partial information, a feature structure F_1 is said to *subsume* another feature structure F_2 if F_1 bears at least as much information as F_2. For a formal account, see Carpenter (1992: 40-45). Unification can be expressed in terms of subsumption. '[...] the unification of two feature structures is a feature structure representing neither more nore less information than is contained in the feature structures being unified.' (Carpenter 1992: 45). Unification of two feature structures is expressed by the operator $\sqcup$.

Sorts and Types: In the seminal work of Aït-Kaci (1984) and Carpenter (1992) a substantial improvement is made by introducing the concept of *inheritance* which is common in artificial intelligence. To achieve this, a partial ordering is defined over entities called sorts (Aït-Kaci) or types (Carpenter) which are associated to the entities the features apply to. This association might be best imagined if we think of feature structures as directed graphs (or deterministic finite automatons) in which a feature represent an arc going from one node q to another node q'. A particular swan, to resume the example in subsection 2.1.1, would be represented by a node q and its colour (white) by q'. The structure is a typed (or sorted) feature structure if both q and q' are associated with a type (or sort). For instance, q is associated with the type SWAN and q' with COLOUR. Types are ordered within a hierarchy of subsumption. A type σ subsumes (is more general) another type τ ($\sigma \sqsubseteq \tau$) if τ inherits information from σ. σ is also called a *supertype* of τ. Now the unification of feature structures is only possible if their types are compatible according to the type hierarchy. With the invention of types there comes another important concept of *appropriateness* - a concept that is used for the definition of lexical features in the main text. In the example, we might want to ensure that the feature Colour is only applied to concrete objects that can have a colour - not to abstract objects, for instance. In an appropriateness condition we might state that the feature Colour is appropriate for a type, say, CONCR_OBJECTS.

The type hierarchy might reveil that CONCR_OBJECTS $\sqsubseteq$ SWAN, so that Colour is appropriate for swans as well.

Extensionality: Carpenter (1992) notes that in 'standard treatments of feature structures' it is 'usually assumed that there can only be one copy of any atom in a feature structure, so that if we know that a path π_1 leads to an atom a and a path π_2 leads to the same atom a, then the two paths must be structure shared' (Carpenter 1992: 124). This is the so-called extensional view on features. Identical feature structures describe the same entity. Carpenter's feature structures, however, are to be treated intensionally. If extensional types are used (for instance we might need features that unequivocally identify individuals from a given universe) they have to be marked as such by virtue of an extensionality specification.

Partial Objects: Feature theories also differ with respect to whether feature structures are *total* or *partial* objects. They are called total objects if every feature that is defined on a given entity (or node) is specified in the feature structure. Head-driven Phrase Structure Grammar (HPSG) is of this kind. Otherwise they are called partial objects. The grammar formalism that we employed for our experiments uses partial feature structures because it makes heavily use of underspecification.

Incorporation of Phrase Structure: Linguistic theories differ with regard to the description of phrase structure. In some theories (like Lexical Functional Grammar (LFG)), feature structures (called f-structure in LFG) annotate nodes in a phrase structure tree (c-structure in LFG), which is an object of its own. HPSG, on the other hand, does not make an ontological distinction between phrase structure and other entities. Here, constituent structure is incorporated into feature structures by virtue of features like COMPLEMENT-DAUGHTERS.

Default Unification: Some linguistic theories make use of so-called defaults, information about entities which is valid until the default is revised. But the provision for defaults in feature structures is not without problems, as it turns a monotonic logic into a non-monotonic one. Note that the basic operation on features - unification - is a monotonic operation. See Carpenter (1993) for an account of default unification.

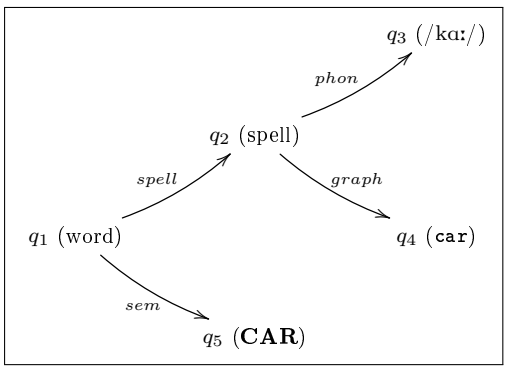
Feature structures are the *de facto* standard for unification-based accounts (Carpenter 1991) and must be seen as an integral component of many linguistic formalisms such as Functional Unification Grammar (FUG) (Kay 1979), Parse and Translate (PATR-II) (Shieber et al. 1983), Lexical Functional Grammar (Kaplan and Bresnan 1982), Head-driven Phrase Structure

Grammar (Pollard and Sag 1994) and Generalized Phrase Structure Grammar Gazdar et al. 1985, Categorial Unification Grammar (Uszkoreit 1986), and Generative Grammar (e.g. Standard Theory since Chomsky 1965, Government and Binding, Chomsky (1981), and Minimalism, Chomsky 1995), among others. Although these theories differ in the ways they formulate linguistic generalizations, we think that there is a common denominator in how they spell out linguistic properties: it is the functional nature of features captured by the definition of *feature systems* presented in subsection 2.1.1.

B.2 Representations

Superficially, the above mentioned theories differ with respect to the employed feature structures when we look at the different notions or graphical representations of these structures. But after all, all of these representations can be mapped to a finite directed graph as exemplified in figure B.1. This is an example of a possible typed feature structure of the word `car`[4]. The node q_1 is thought to be a token of this word. The types are given in brackets. The word is made up of two features, SPELL and SEM. The latter is thought to encode the semantics of the word, the first is mapping to a feature structure consisting of the two features PHON and GRAPH, encoding the word's phonological and graphical representation, respectively.

Figure B.1: A Finite Directed Graph Representing a Feature Structure



The same feature structure can be displayed in a more readable notation, called attribute value matrix (AVM):

[4]This is just an example, not intended to schematize a certain theory of words.

Figure B.2: Attribute Value Matrix (AVM) Representing a Feature Structure

$$word\begin{bmatrix} \text{SPELL} & spell\begin{bmatrix} \text{PHON} & /\text{kɑ:}/ \\ \text{GRAPH} & \texttt{car} \end{bmatrix} \\ \text{SEM} & \textbf{CAR} \end{bmatrix}$$

Note that the nodes representing the linguistic entities are not explicitly mentioned in the matrix. The types are written on the left down corner of the matrix.
Structure sharing is represented by identical nodes in directed graphs and boxes in AVMs. Suppose we want to express that the colour of a swan is not known, but we know that the head and the body must be of the same colour (while other features like size might differ). This example is depicted in figures B.3 and B.4, respectively. Here, we omit the specification of types.

Figure B.3: A Finite Directed Graph with Structure Sharing

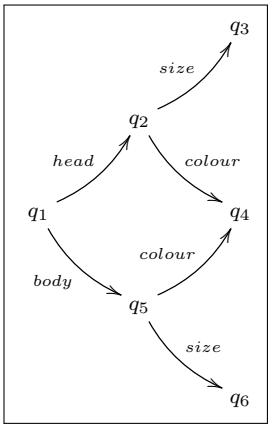


Figure B.4: Attribute Value Matrix (AVM) with Structure Sharing

$$\begin{bmatrix} \text{HEAD} & \begin{bmatrix} \text{SIZE} & \ldots \\ \text{COLOUR} & \boxed{1} \end{bmatrix} \\ \text{BODY} & \begin{bmatrix} \text{SIZE} & \ldots \\ \text{COLOUR} & \boxed{1} \end{bmatrix} \end{bmatrix}$$

B.3 Lexical Relativity

B.3.1 Examples of Lexical Entries in the Literature

To illustrate lexical relativity, we want to briefly discuss some different lexical entries that are used to exemplify the concept of the lexicon in different linguistic theories[5]. First, in the work of Chomsky (1965) (the so-called Standard Theory) the lexicon is an unordered list of lexical entries that supplements grammar (the rewriting rules) mediated by the application of a lexical rule. A lexical entry is a tuple $\langle D, C \rangle$ where D is a phonological matrix (the 'spelling' of a lexical formative) and C a complex symbol consisting of different sorts of features. Lexical formatives (as opposed to grammatical formatives like English *the*) are inserted into a grammar-generated preterminal string resulting in a terminal string according to the following *lexical rule*: 'If Q is a complex symbol of a preterminal string and (D,C) is a lexical entry, where C is not distinct from Q, then Q can be replaced by D.' (Chomsky 1965: 84) Lexical entries encode unpredictable properties of lexical formatives: 'In general, all properties of a formative that are essentially idiosyncratic will be specified in the lexicon. In particular, the lexical entry must specify: (a) aspects of phonetic structure that are not predictable by general rule [...]; (b) properties relevant to the functioning of transformational rules; (c) properties of the formative that are relevant for semantic interpretation [...]; (d) lexical features[6] indicating the positions in which a lexical formative can be inserted (by the lexical rule) in a preterminal string.' (Chomsky 1965:

[5] In this section we dedicate more space to the Chomskian concept of the lexicon than to other theories. This is due to historic reasons and shall by no means indicate that implementations of the Chomskian theory would be preferable to other grammatical theories to foster Learn$-\alpha$. Quite the contrary: we believe that the Learn$-\alpha$ Design Rule would be hard to realize for implementations of minimalist theories.

[6] Note that the notion of *lexical feature* in the Standard Theory differs from our notion.

87). The syntactic behavior of lexemes is determined by three aspects: a) the category label (which can be a 'complex' symbol consisting of features), b) rule features that indicate the applicability of transformational rules and c) subcategorization information and selectional restrictions. Take B.1 as examples for a) and c)[7]. B.1 i) and ii) show sets of rewriting and subcategorization rules, respectively (cf. Chomsky 1965: 68). B.1 iii) represents a set of lexical entries where the formative *sincerity* is associated with categorical features (Chomsky 1965: 85). Note that '[...] the italicized items stand for phonological distinctive feature matrices, that is, "spellings" of formatives' Chomsky 1965: 68, a practice common in many linguistic theories. The categorical features determine the distribution to terminal nodes of the rewriting rules. B.1 iv) is an example of a subcategorization (or *contextual*) feature (Chomsky 1965: 94). Here, +—NP means that the formative matches a contexted rewriting rule which is specified for this particular context, namely the symbol preceding an NP. Selectional restrictions are encoded in a similar way.

$$\begin{array}{ll} i) & \{\ldots, \mathrm{NP} \rightarrow \mathrm{N}, \ldots\} \\ ii) & \{\ldots, \mathrm{N} \rightarrow [+\mathrm{N}, \pm\mathrm{Common}], \ldots\} \\ iii) & \{\ldots, (\mathit{sincerity}, [+\mathrm{N}, \ldots]), \ldots\} \\ iv) & \{\ldots, (\mathit{eat}, [+\mathrm{V}, +\text{—}\mathrm{NP}]), \ldots\} \end{array} \tag{B.1}$$

Chomsky notes that a mere positive specification of a feature in the complex symbol of a lexical entry is insufficient: '[...] lexical entries must be specified negatively for features corresponding to contexts in which they may not occur'. Consequently, the formative *sincerity* in B.1iii) must be associated with [-V] to prevent the formative *sincerity* to replace a verbal terminal symbol. This is achieved either explicitly by listing the respective features or implicitly by conventions like the one proposed in Chomsky (1965: 165). Note that the same effect can be reached by some sort of *closed-world reasoning*: Every logical sentence that is not stated in a knowledge database and that cannot be inferred is false. This requires that *false that* $\langle D, [+F]\rangle$ is semantically equivalent to *true that* $\langle D, [-F]\rangle$ and that underspecification of [F] is equivalent with [-F], a possibly undesired consequence.

Although many details of lexical specification and especially the way how the lexicon interacts with the syntactic component[8] might have changed

[7] We will not discuss b) as 'abstract features triggering transformational rules came to be seen as ad hoc devices , and have fallen into disuse.' (Stowell 1992: 10).

[8] In Minimalism, the lexical rule of Chomsky (1965) is replaced by an operation called *select* that allows the computational component to pick out lexical items from a pre-built so-called *numeration* and add a corresponding syntactic object to any stage of derivation as long as spell-out has not been reached.

throughout the development of Generative Grammar, the view that lexical entries are phonological matrices associated with features encoding unpredictable properties is a constant. Chomsky (1995) takes a radical position on the lexicon being 'optimally coded' in the sense that really *nothing* that follows from universal or language specific principles must be lexically specified: 'It is the optimal coding of information that just suffices to yield the LF [Logical Form] representation and that allows the phonological component to construct the PF [Phonetic Form] representation; [...] One idiosyncratic property of *book* coded in the lexical entry is the sound-meaning relation. The lexical entry also either lists, or entails, that it has the categorical feature [N]; [...] But the lexical entry should not indicate that *book* has Case and ϕ-features; that follows from its being of category N [...] In some cases, such features might be idiosyncratic (e.g., the plural feature of *scissors*, or grammatical gender).' (Chomsky 1995: 235-236) On similar grounds, selectional restrictions of verbs that follow from semantic properties have to be excluded from the lexicon. Chomsky is not very specific how to implement such a lexicon, but if our interpretation is correct, some mechanism of default overwriting must be necessary to realize 'optimal coding'; for instance, the value of NUMBERx**v** is to be overwritten by the lexical specification of *scissors*, whereas the value of non-plural nouns is predictable from general principles.

Whereas Minimalism places most of the burden of feature instantiation on general principles (whatever the implementation might look like), other theories not only explicitly specify (morpho-)syntactic and semantic features in their lexical entries, they also equip the lexemes with information on how to link its semantic arguments to syntactic positions. In LFG, for instance, a lexical entry provides a) the category of the lexical item b) its meaning, c) its predicate-argument structure, d) grammatical functions associated with the arguments (either directly or by features like [-r] which indicates a patient-like role, features that are read-off during the process of linking) and e) further grammatical properties such as agreement with the subject or token identity of the verb's object with the subject of a subordinated clause (XCOMP). In a sense, a verbal entry in the LFG's lexicon is able 'to see' the subject of the clause into which the verb is projected so that restrictions on the subject can directly be encoded in the lexical entry via path equations - a concept that is diametrically opposed to Chomskian thinking. As an example, take the lexical specification of the inflected verb *walks* in LFG-format.

$$\begin{array}{lll} walks & Verb & \\ & (\uparrow \text{PRED}) = \text{`walk}\langle SUB\rangle' & \\ & (\uparrow \text{SUB NUM}) =_c \text{-PL} & \end{array} \tag{B.2}$$

The specification says that if *walks* is a leaf (of category V) in the c-structure, then the associated f-structure has a PRED feature that equals the predicate-argument structure 'walk$\langle SUB \rangle$' indicating that the verb licenses one argument that is bound to the subject of the c-structure. Further the NUM feature of the subject is restricted to be -PL ensured by the *constraining equation* expressed by the operator $=_c$ (on the distinction of defining vs. constraining equations, also cf. subsection 2.3.1).
B.3 shows the corresponding entry in HPSG notation as presented in Pollard and Sag (1994: 28).

$$walks \quad \left[\begin{array}{l} \text{CAT} \left[\begin{array}{ll} \text{HEAD} & verb[fin] \\ \text{SUBCAT} & \langle \text{NP}[nom]_{\boxed{1}[3rd,sing]} \rangle \end{array}\right] \\ \\ \text{CONTENT} \left[\begin{array}{ll} \text{RELN} & \text{walk} \\ \text{WALKER} & \boxed{1} \end{array}\right] \end{array}\right] \tag{B.3}$$

Here, the phonological appearance of *walks* is associated with a feature structure that bears syntactic (CAT) and semantic (CONTENT) information. As opposed to LFG, grammatical relations are not directly feature-encoded. By convention, $verb[fin]$ denotes finite verbs and $\text{NP}[nom]_{\boxed{1}[3rd,sing]}$ abbreviates 'a nominal phrase in nominative case the index of which is specified as 3rd singular'. Lists are displayed in $\langle\rangle$. So the first (and only) item of the subcategorization list of the verb is an NP that is restricted to appear in nominative case and whose index must be 3rd-singular. As with LFG, the mapping from (morpho-)syntax to semantics (linking) is already 'built-in' to the lexical entry of the verb - in this case realized by token identity of the NP's index and the value of the WALKER feature.

B.3.2 Relativity of the Lexicon

The three entries exemplify what we called the relativity of the lexicon (section 1.1), a statement that can be now recast more precisely by saying that the content and the structure of the lexicon varies with the structure and content of the computational component of the theory (e.g. transformational rules triggered by features encoded in the lexical entries versus general principles that predict features like NUMBER) and the linguistic background theory

(the ontology provided by the feature system). But relativity goes deeper. All entries reviewed in the last paragraph describe properties of entities that we called lexical types: sets of lexical tokens - idiosyncratic units of utterances rooted in some particular contexts. Being a type, the entity is by itself a generalization or abstraction and nothing prevents us to root it back to some 'more underlying' lexical entry from which it has been projected by the application of some rule as the one sketched out schematically in B.4:

$$\begin{bmatrix} \text{CAT} & \text{v} \\ \text{ACTOR} & + \\ \text{UNDERGOER} & - \\ \text{COMPL} & - \\ \text{SEM} & \mathbf{S} \end{bmatrix} \rightarrow \quad (\uparrow PRED) = \text{`}\mathbf{S}\langle SUB \rangle' \tag{B.4}$$

According to this 'rule', a lexical entry will contain the equation of the right side if it is compatible with the feature structure of the left side. It appears to be impossible to determine the 'best' structure of a lexical entry (and that this is not given empirically is clear anyway).

Appendix C

OALD Gold Standard for Verbal Valency

For the reader's convenience we compile here the Oxford Advanced Learner's Dictionary (OALD) gold standard for the experiment reported in section 4.11 along with examples taken from this dictionary. The first table lists the meanings of the OALD keys, the second provides the examples.

OALD key	Meaning
[V]	intransitive verb
[VN]	transitive verb
[V-N]	linking verb with one NP
[VNN]	ditransitive verb
[VN-N]	linking verb with two NPs

Verb	OALD key	Example
ask	[V]	He asked about her family.
	[VN]	Can I ask a question?
	[VNN]	She asked the students their names.
become	[VN]	Such behaviour did not become her.
	[V-N]	She became queen in 1952.
come	[V]	Your breakfast is coming soon.
	[VN]	We've come 50 miles this morning.
feel	[V]	I felt like a complete idiot.
	[VN]	He seemed to feel no remorse at all
	[V-N]	Standing there on stage I felt a complete idiot.
find	[V]	The court found in her favour.
	[VN]	Look what I 've found!
	[VN-N]	She finds it a strain to meet new people.
get	[V]	We got to San Diego at 7 o'clock.
	[VN]	I got a letter from Dave this morning.
	[VNN]	Did you get your mother a present?
give	[V]	They say it's better to give than to receive.
	[VN]	She gave a reading from her latest volume of poetry.
	[VNN]	Give your mother the letter.
go	[V]	This clock doesn't go.
	[VN]	We had gone about fifty miles when the car broke down.
know	[V]	What's the answer? I don't know.
	[VN]	Do you know his address?
leave	[V]	Come on, it's time we left.
	[VN]	I hate leaving home.
	[VNN]	I'm afraid you leave me no choice.
look	[V]	If you look carefully you can just see our house from here.
	[V-N]	That looks an interesting book.
make	[VN]	She makes her own clothes.
	[V-N]	She would have made an excellent teacher.
	[VN-N]	She made him her assistant.
mean	[VN]	What does this sentence mean?
need	[VN]	Do you need any help?
put	[VN]	I'll put it in my diary.
say	[V]	'When will it be finished?' - 'I couldn't say.'
	[VN]	Try to say this line with more conviction.
see	[V]	She will never see again.
	[VN]	He didn't see the joke.
seem	[V]	It seemed like a good idea at the time.
	[V-N]	He seems a nice man.
show	[V]	Fear showed in his eyes.
	[VN]	The map shows the principal towns and rivers.
	[VNN]	I'll go first and show you the way.
take	[V]	I need a shower - I won't take long.
	[VN]	I passed him the robe and he took it.
	[VNN]	Shall I take my family a gift?
tell	[V]	'That would be telling!'
	[VN]	Are you sure you're telling the truth?
	[VNN]	What did I tell you?
think	[V]	Are animals able to think?
	[VN]	Try to think yourself into the role.
use	[VN]	Can I use your phone?
want	[VN]	Do you want some more tea?
work	[V]	I can't work if I'm cold.
	[VN]	He works a large area.

Appendix D

Technical Specification of the ALR Prototype

Hardware:	AMD Opteron DUAL CPU 2.4GHz, 4GB RAM, 120GB disk space
Operating System:	OpenSuSE 11.1 with Linux Kernel Version 2.6.27.7
RDBMS:	MySQL 5.0.67-12.18.1
PROLOG Version:	SWI Prolog 5.6.59
Heart of Gold (HoG):	1.5
JTok:	1.01
TnT:	2.2
PET:	0.99.14svn
English Resource Grammar (ERG):	Feb. 2008
WordNet:	3.0

Appendix E

Lexical Seed Type Hierarchy

In this section we briefly describe the most important types and features of the lexical type hierarchy (LTH).

E.1 Lexical Seed Type Hierarchy: Top Level

LTH:	figure 3.10 on page 229
Feature	Description
LSSTEM	graphical representation of stem
LSSENSE	predicate
LSPOS	part of speech (open class lexemes), cf. figure 3.11, page 230
LSVAL	valency (subcaterization feature), cf. figure E.1
LSPP	subcategorization for PPs, cf. figure 3.12, page 231
LSSC	subcategorization for sentential complements, cf. figures E.3 and E.4
−LSMWE	indicates whether lexeme is a multiword expression
LSDEGREE	handles comparison of adjectives and adverbs
LSATTR	indicates whether an adjective can be used attributively and/or predicatively, cf. figure E.7
LSVOICE	specifies in which voice a verb may be used, cf. E.2
LSVPRT	used for the specification of verb particle combinations (VPCs), cf. figure 3.16, page 243
LSSCCTRL	handles control options (e.g. subject equi)
LSSCADJ	true if complement is adjective, false if complement is adverb
LSSCSF	specifies sentential force

Figure E.1: Lexical Seed Type **ls_val**

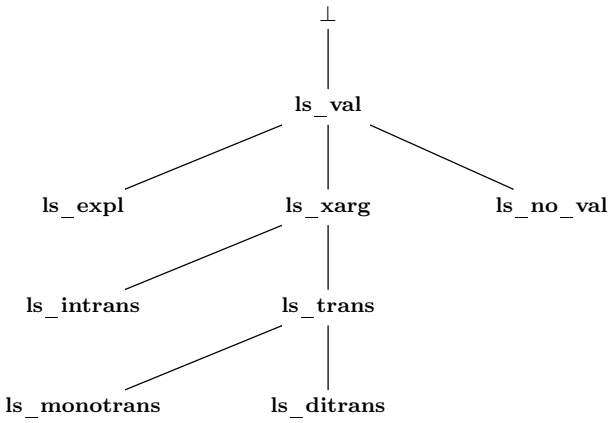
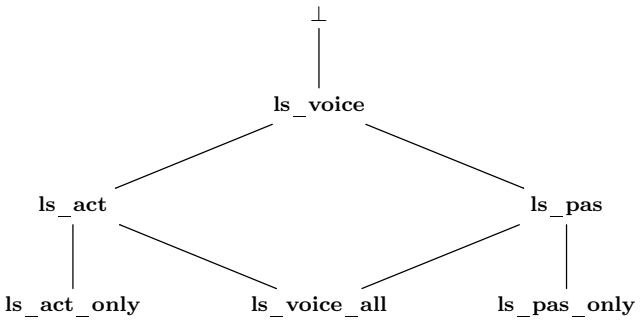


Figure E.2: Lexical Seed Type **ls_voice**



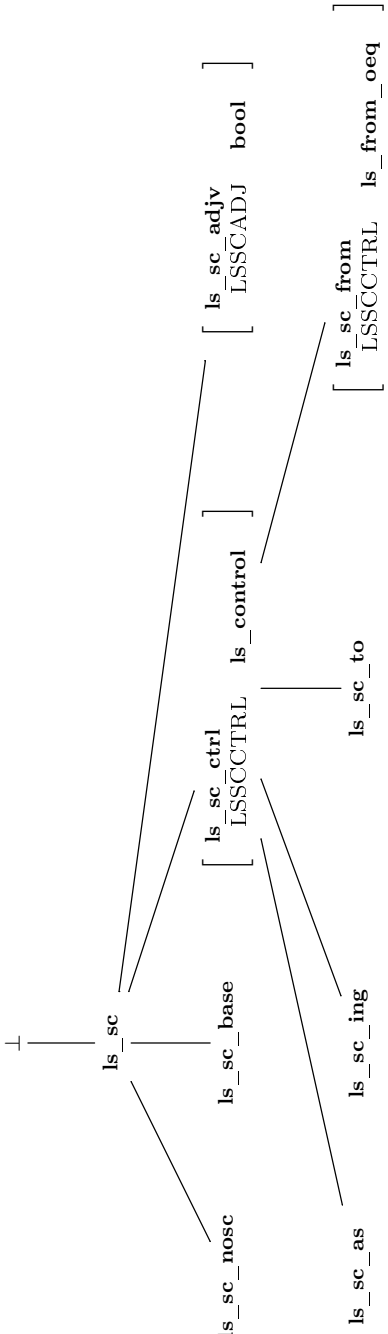


Figure E.3: Lexical Seed Type **ls_sc** - Part 1

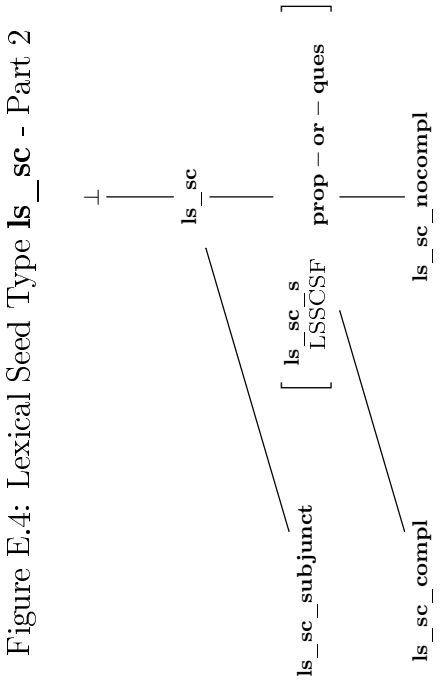


Figure E.4: Lexical Seed Type **ls_sc** - Part 2

E.2 Lexical Seed Type Hierarchy: Nouns

LTH:	figure E.5
Feature	Description
LSGOVPREP	governing preposition (for detless N)
LSGOVVERB	governing verb (for V N' idiom)
LSCOUNT	true if noun is count noun
LSNUMBER	specifies whether noun is inherently singular or plural
LSINFLPL	true if noun inflects for plural (false for bare plural nouns)

E.3 Lexical Seed Type Hierarchy: Verbs

LTH:	figure E.6
Feature	Description
LSIDIOMN	noun pred in V N' idiom (e.g. *give way*)
LSSSUBJ	true if sentential (CP/VP) subject is licensed
LSDATSHFT	true if ditransitive verb allows dative shift
LSPRTBEFN	true if V P NP is licensed ('particle before noun')
LSLOCINV	true if locative inversion is licensed

In addition to figure E.6 further verbal LSEED types are defined in the following table:

Verbal LSEED Type	Super Type	LSVAL	LSSC
lseedditrverb_prt	**lseedverb_prt_nosc**	ls_ditrans	
lseeditrverb_prt_adjv	**lseedverb_prt**	ls_intr	ls_sc_adjv
lseeditrverb_prt_s	**lseedverb_prt**	ls_intr	ls_sc_s
lseeditrverb_prt_ing	**lseedverb_prt**	ls_intr	ls_sc_ing
lseeditrverb_prt_to	**lseedverb_prt**	ls_intr	ls_sc_to
lseedmtrverb	**lseedverb_nosc**	ls_monotrans	
lseeditrverbsc_base	**lseedverb**	ls_intr	ls_sc_base
lseedmtrverbsc_base	**lseedverb**	ls_monotrans	ls_sc_base
lseeditrverbsc_adjv	**lseedverb**	ls_intr	ls_sc_adjv
lseedmtrverbsc_adjv	**lseedverb**	ls_monotrans	ls_sc_adjv
lseeditrverbsc_sub	**lseedverb**	ls_intr	ls_sc_subjunct
lseedmtrverbsc_sub	**lseedverb**	ls_monotrans	ls_sc_subjunct
lseeditrverbsc_as	**lseedverb**	ls_intr	ls_sc_as
lseedmtrverbsc_as	**lseedverb**	ls_monotrans	ls_sc_as
lseedmtrverbsc_from	**lseedverb**	ls_monotrans	ls_sc_from
lseeditrverbsc_ing	**lseedverb**	ls_intr	ls_sc_ing
lseedmtrverbsc_ing	**lseedverb**	ls_monotrans	ls_sc_ing
lseeditrverbsc_to	**lseedverb**	ls_intr	ls_sc_to
lseedmtrverbsc_to	**lseedverb**	ls_monotrans	ls_sc_to
lseeditrverbsc_extrapos	**lseedverb**	ls_intr	
lseedmtrverbsc_extrapos	**lseedverb**	ls_monotrans	
lseeditrverbsc_s	**lseedverb**	ls_intr	ls_sc_s
lseedmtrverbsc_s	**lseedverb**	ls_monotrans	ls_sc_s
lseedditrverbsc_s	**lseedverb**	ls_ditrans	ls_sc_s

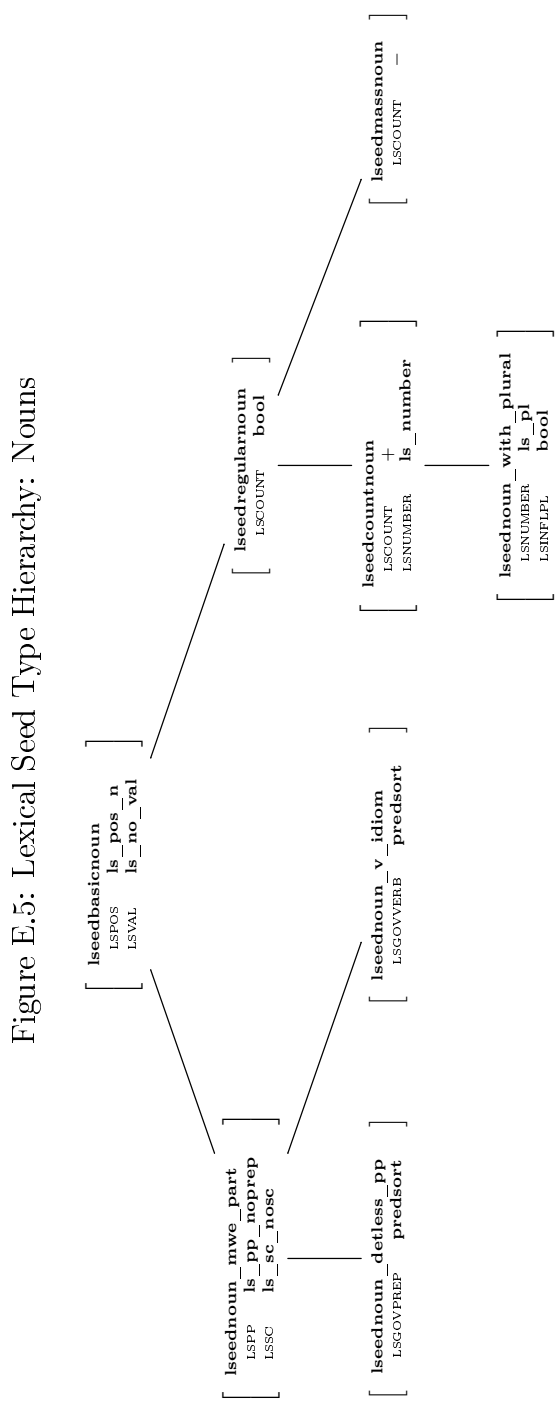
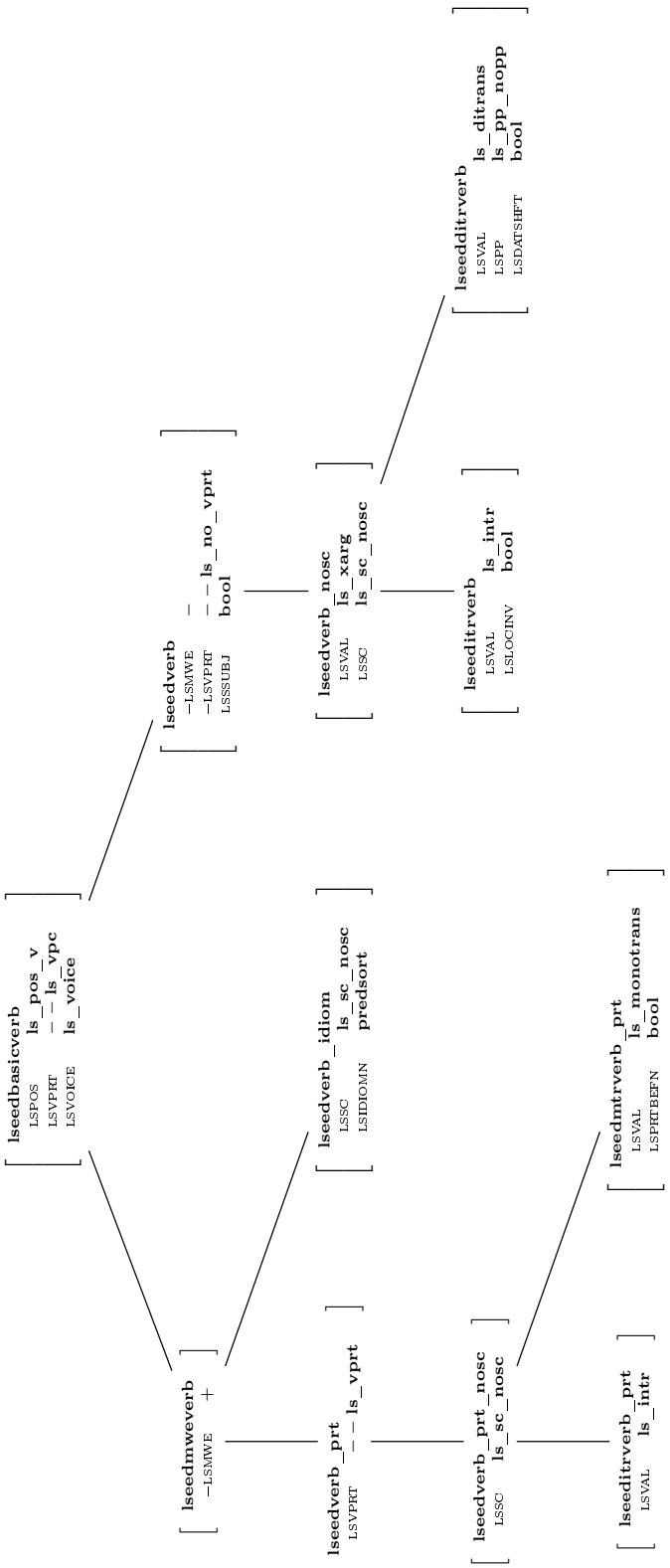


Figure E.5: Lexical Seed Type Hierarchy: Nouns

Figure E.6: Lexical Seed Type Hierarchy: Verbs (fragmentary)



E.4 Lexical Seed Type Hierarchy: Adjectives

There are no further features in addition to the top level hierarchy.

Figure E.7: Lexical Seed Type **ls_attrprd**

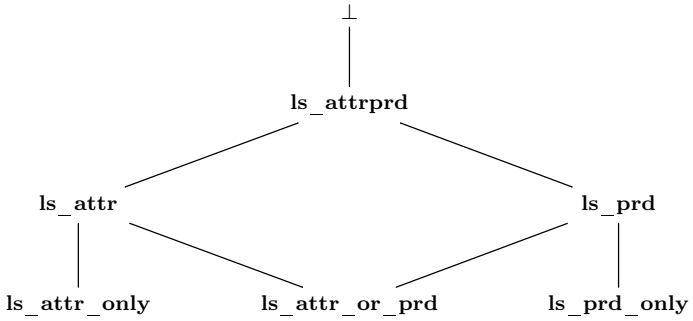
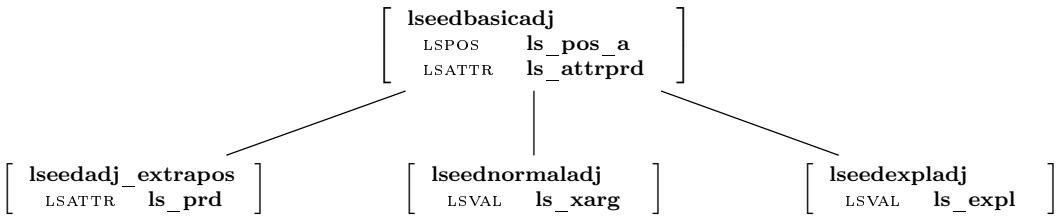


Figure E.8: Lexical Seed Type Hierarchy: Adjectives



Index

Glossary

u	utterance	55
E	experience set	56
C	challenge set	55
$\langle L, A\rangle$	feature signature	81
FS	feature system	81
$\mathrm{F}x\mathbf{v}$	feature and value in predicate logic	82
ϕ	first-order logical formula	83
σ, τ	types in typed feature logic	83
$\boldsymbol{\phi}^{\sigma}$	feature structure with root σ	83
Θ	theory (in the context of linguistic modelling)	83
$\mathrm{M}u\mu$	μ is u-model of utterance u	84
$\mathcal{L}$	lexeme	84
$\sigma \sqsubseteq \tau$	type σ subsumes type τ	86
Γ	grammar	90
Λ	lexicon	90
$\mathbf{M}$	(symbolic) language model	90
$\mathbf{M} : u \Rightarrow \mathcal{M}^u$	language model derives u-model set	90
ψ	realization context of a feature	91
$\alpha = \langle \sigma, \psi, F\rangle$	lexical feature of type σ and realization context ψ	91
$\alpha^{\mathcal{L}}_{\mathbf{v}}$	lexeme $\mathcal{L}$ admits value v of feature α	91
$\alpha^{\mathcal{L}}_{\mathbf{v}\in V}$	lexeme $\mathcal{L}$ admits all values v $\in$ V of feature α	91
$\Lambda \vdash \alpha^{\mathcal{L}}_{\mathbf{v}}$	lexicon entails feature admission	92
$\Lambda \vdash -\alpha^{\mathcal{L}}_{\mathbf{v}}$	lexicon entails feature constraint	92
$\mathbf{M}_0$	the initial model	95
$\mathbf{M}_{opt}$	the optimal model	96
$\Lambda \sqsubset \Lambda'$	extend a lexicon	97
$\mathcal{M}/w \vDash \phi$	the formula ϕ is true in world w of model system $\mathcal{M}$	99

$\langle[\Gamma \cup \Lambda_0], R_\Box ..., \mathcal{V}\rangle$	model system defined over $\mathbf{M}$	99
$\Box\phi$	ϕ holds in the optimal model	99
$B_s\phi$	system believes that ϕ	101
$\Box\alpha^{\mathcal{L}}_{\mathbf{v}}$	lexeme admits the feature value in the optimal model	101
r	reference	104
$E \mapsto \Box\alpha^{\mathcal{L}}_{\mathbf{v}}$	$\alpha^{\mathcal{L}}_{\mathbf{v}}$ is induced based on experience set E	110
$E^{\mathcal{L}}_{\alpha}$	the set of utterances in which the lexeme $\mathcal{L}$ realizes one of the values of the feature α	112
θ	unknown parameter of a distribution	112
$\hat{p}$	maximum-likelihood-estimated (MLE-) probability	112
x	observed number of feature value realizations	112
$X \sim D$	random variable is D-distributed	112
$f(x\|\theta)$	distribution with unknown parameter	112
Θ	parameter space (in the context of statistical inference)	112
δ	decision function	112
H_0	null hypothesis	113
$\pi(x)$	p-value of an observation x	113
α_s	level of significance (Fisher)	113
p^{fv}_{min}	minimum feature value realization probability	114
$X \sim \mathbf{B}(p, N)$	binomial distribution with probability p and sample size N	114
$F_{\mathbf{B}(p,N)}(x)$	cdf of binomial distribution with probability p and sample size N	115
H, K	alternative hypotheses	116
A, R	acceptance region / region of rejection	117
α_e	maximally allowed type-I error (Neyman-Pearson)	117
β_{FN}	maximally allowed type-II error	119
β_{FP}	maximally allowed type-I error in Learn$-\alpha$	190
$\hat{MI}$	approximated pointwise mutual information	122
MI_{min}	minimal MI value for lexeme detection	122
ϵ	noise level	124
$\epsilon^{\rho=n}_{eff}$	effective noise level	129

$\hat{\epsilon}_{eff}^{\rho=n}$	approximate effective noise level	129
ϵ_{max}	maximally assumed noise level of the system	130
SNR	signal-to-noise ratio	130
p_{max}^{fv}	maximum feature value realization property	131
RDoA_{max}	maximally allowed Relative Degree of Ambiguity	145
ρ	$=\mathrm{RDoA}_{max}$	178
$r \mapsto^{\rho=n} \Box\alpha_{\mathbf{v}}^{\mathcal{L}}$	reference under RDoA_{max}=n	182
$c_{ref}^{\rho=n}$	number of references (reference count)	181
$c_{LO}^{\rho=n}$	number of learning opportunities	183
$freq^{\rho=n}$	relative feature value realization frequency	183
$\underline{freq}$	lower bound of relative feature value realization frequency	184
$\overline{freq}$	upper bound of relative feature value realization frequency	184
$NF^{\rho=n}(\alpha_{\mathbf{v}}^{\mathcal{L}})$	neutralization factor	185
RF	recoverability factor	186
c_{ref}^{min}	minimum number of references needed for reliability	189
$p_{\psi}^{\rho=n}$	adjusted minimum feature value realization probability	192
f_{π}	strength-of-belief function	195
$Cn(T)$	Tarskian consequence operator	201
$T + \phi$	theory T expanded by ϕ	201
$T\dot{-}\phi$	theory T contracted by ϕ	201
$T\dot{+}\phi$	theory T revised by ϕ	201
$B_1 > B_2$	belief B_1 is epistemically more entrenched than B_2	203

Acronyms and Abbreviations

ACC	accusative	48
ACL	Association for Computational Linguistics	15
ALR	ANALYZE-LEARN-REDUCE	209–211, 213, 214, 224, 237, 246, 257
ANLE	automatic noise level estimation	28, 190, 222, 320, 324
ANLT	Alvey Natural Language Tools	12
AVM	attribute value matrix	156, 337, 338
BHT	binomial hypothesis testing	23, 24, 28, 30, 113, 220, 273, 320
BNC	British National Corpus	1, 34, 35, 124, 263, 264, 266–268, 272, 322, 323
CAUS	causative	48
cdf	cumulative distribution function	115, 329
CFG	context free grammar	33, 35, 47
CI	confidence interval	330
DAT	dative	48
DFKI	Deutsche Forschungsgesellschaft für künstliche Intelligenz	209, 243, 244
DLA	deep lexical acquisition	16, 52, 315
DLR	deep language resource	17

EM	expectation-maximization	34
ERG	English Resource Grammar	2, 8, 10, 55, 63, 69, 72, 92, 141, 160, 164, 177, 209, 210, 224–227, 230, 231, 233, 237–244, 248, 251, 263–270, 273, 280, 282–285, 312, 315, 321, 322, 346
FN	false negative	13, 23, 119, 288
FP	false positive	13, 22, 23, 293
FPE	false positive example	23
GB	Government and Binding Theory	84
GPSG	Generalized Phrase Structure Grammar	84
HoG	Heart of Gold	209, 237, 243–248, 325, 346
HPSG	Head-driven Phrase Structure Grammar	iii, 9, 10, 38, 40, 47, 81, 84, 152, 155, 166, 209, 210, 244, 321, 322, 336, 342
ISO	International Organization for Standardization	245
LE	language engineer	3, 7, 22, 53, 55–57, 93–97, 104, 111, 123–126, 128, 130, 150, 152, 190, 195, 218, 219, 221, 318, 319, 322

LFG	Lexical Functional Grammar	84, 105, 336, 341, 342
LKB	Linguistic Knowledge Building	63, 210, 244
LO	learning opportunity	182–184, 192, 193, 198, 271, 288, 319
LRT	likelihood ratio test	28, 113, 121, 194, 220, 280, 320
LSA	latent semantic analysis	173
LTH	lexical type hierarchy	87, 178, 180, 182, 187, 192, 196, 203, 205, 210, 215, 222, 224–227, 237, 238, 241, 251, 253, 284, 296, 297, 301, 302, 304, 305, 317, 319, 322, 325, 347, 351
MDL	minimum description length	4
MI	mutual information	122, 174, 194, 220, 280, 281
MLE	maximum likelihood estimate	28, 30, 112, 113, 124, 184, 185, 190, 220, 328, 331
MRS	minimal recursion semantics	227, 240
MTT	Meaning-Text-Theory	154, 155
MWE	multiword expression	10, 161–166, 169–171, 173–177, 239, 240, 242, 263, 301, 312

NER	named entity recognition	10, 279
NLP	natural language processing	1–10, 12, 13, 27, 31, 32, 36, 37, 43, 47, 50, 51, 55–57, 59, 60, 209, 237, 243, 244, 316, 321, 324, 325
NOM	nominative	48
OALD	Oxford Advanced Learner's Dictionary	12, 34, 35, 151, 264, 266, 277, 279–281, 284, 285, 287, 291, 293, 296–298, 300, 304–306, 310, 313, 315, 344
ODBC	Open Database Connectivity	251
PCFG	probabilistic context free grammar	33, 34, 36, 50, 51
PIC	PET XML input chart	245–247
PoS	part-of-speech	15, 17, 18, 25, 27, 28, 34, 49, 139, 224, 243–245, 264–266, 273, 277, 279, 282, 322, 325
RDoA	relative degree of ambiguity	61, 62, 105, 147, 148, 178, 182, 207, 221
RMRS	robust minimal recursion semantics	245, 247, 325

SCF	subcategorization frame	13, 19–21, 24–31, 36, 48, 52, 116, 132, 141, 236, 263
SID	sentence identifier	83
SNR	signal-to-noise ratio	130, 132, 220, 272
SQL	Structured Query Language	219
SVC	support verb construction	150
TDL	Type Description Language	224, 226, 233, 238, 251
TEI	Text Encoding Initiative	245, 246, 248, 251
TID	text identifier	83
TP	true positive	13, 23, 279, 297, 312
TPE	true positive example	23
UMP	uniformly most powerful test	118, 121, 324
VPC	verb particle combination	10, 63, 69, 70, 75, 141, 165–169, 175, 179, 239, 242, 274, 300–302, 313, 347
WSJ	Wall Street Journal Corpus	35
XML	Extensible Markup Language	243–248, 251, 264
XSLT	extended style sheet transformations	244, 245, 247

Bibliography

Abney, Steven. 1991. Parsing by Chunks. *In:* Bouchard, D. and K. Leffel (eds), *Views on Phrase Structure*. Kluwer Academic Publishers.

Adolphs, Peter, S. Oepen, U. Callmeier, B. Crysmann, D. Flickinger, and B. Kiefer. 2008. Some Fine Points of Hybrid Natural Language Parsing. *In:* (ELRA), European Language Resources Association (ed), *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*. Marrakech, Morocco.

Agresti, Alan and Brent A. Coull. 1998. Approximate Is Better than "Exact" for Interval Estimation of Binomial Proportions. *The American Statistician*, **52**(2), 119–126.

Aït-Kaci, Hassan. 1984. *A lattice theoretic approach to computation based on a calculus of partially ordered type structures*. Ph.D. thesis, University of Pennsylvania.

Alchourrón, Carlos E., Peter Gärdenfors, and David Makinson. 1985. On the logic of theory change: Partial meet contraction and revision functions. *Journal of Symbolic Logic*, **50**(2), 510–530.

Backofen, Rolf and Gert Smolka. 1993. A Complete and Recursive Feature Theory. *Pages 193–200 of: Proceedings of the 31st ACL*. Columbus, Ohio, USA. A full version has appeared as Research Report RR-92-30, DFKI.

Baldwin, Timothy. 2005a. Bootstrapping Deep Lexical Resources: Resources for Courses. *Pages 67–76 of: Proceedings of the ACL-SIGLEX Workshop on Deep Lexical Acquisition*. Ann Arbor, MI, USA.

Baldwin, Timothy. 2005b. The Deep Lexical Acquisition of English Verb-particle Constructions. *Computer Speech and Language, Special Issue on Multiword Expressions*, **19**, 398 – 414.

Baldwin, Timothy. 2005c. General-Purpose Lexical Acquisition: Procedures, Questions and Results. *Pages 23–32 of: Proceedings of the Pacific Association for Computational Linguistics 2005*. Tokyo, Japan.

Baldwin, Timothy. 2005d. Looking for Prepositional Verbs in Corpus Data. *Pages 115–126 of: Proceedings of the 2nd ACL-SIGSEM Workshop on the Linguistic Dimensions of Prepositions and their Use in Computational Linguistics Formalisms and Applications*. Colchester, UK.

Baldwin, Timothy. 2007. Scalable Deep Linguistic Processing: Mind the Lexical Gap. *Pages 3–12 of: Proceedings of the 21st Pacific Asia Conference on Language, Information and Computation (PACLIC21)*. Seoul, Korea.

Baldwin, Timothy and Francis Bond. 2003a. Learning the countability of English nouns from corpus data. *Pages 463–70 of: Proceedings of the 41st Annual Meeting of the ACL*. Sapporo, Japan.

Baldwin, Timothy and Francis Bond. 2003b. A plethora of methods for learning English countability. *Pages 73–80 of: Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP 2003)*. Sapporo, Japan.

Baldwin, Timothy and Aline Villavicencio. 2002. Extracting the unextractable: A case study on verb-particles. *Pages 98–104 of: Proceedings of the 6th Conference on Natural Language Learning (CoNLL-2002)*. Taipei, Taiwan.

Baldwin, Timothy, C. Bannard, T. Tanaka, and D. Widdows. 2003a. An empirical model of multiword expression decomposability. *Pages 89–96 of: Proceedings of the ACL-SIGLEX Workshop on Multiword Expressions: Analysis, Acquisition and Treatment*. Sapporo, Japan.

Baldwin, Timothy, J. Beavers, L. van der Beek, F. Bond, D. Flickinger, and I. A. Sag. 2003b. In Search of a Systematic Treatment of Determinerless PPs. *Pages 145 – 56 of: Proceedings of the ACL-SIGLEX Workshop on the Linguistic Dimensions of Prepositions and their Use in Computational Linguistics Formalisms and Applications*. Toulouse, France.

Baldwin, Timothy, E. M. Bender, D. Flickinger, A. Kim, and S. Oepen. 2004. Road-testing the English Resource Grammar over the British National Corpus. *Pages 2047–2050 of: Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC 2004)*. Lisbon, Portugal.

Barg, Petra. 1992. Automatic Acquisition of DATR Theories from Observations. *In: Theorie des Lexikons: Arbeiten des Sonderforschungsbereichs 282.* Düsseldorf: Heinrich-Heine-Universität Düsseldorf.

Barg, Petra. 1996a. *Automatischer Erwerb von linguistischem Wissen: ein Ansatz zur Inferenz von DATR-Theorien.* Tübingen: Niemeyer.

Barg, Petra. 1996b. Erstellung von Lexikoneinträgen aus Merkmalsstrukturen. *Pages 249–254 of:* Gibbon, D. (ed), *Natural Language Processing and Speech Technology. Results of the 3rd KONVENS Conference, Bielefeld, October 1996.* Berlin: Mouton de Gruyter.

Barg, Petra and James Kilbury. 2000. Incremental Identification of Inflectional Types. *In: Proceedings of COLING2000.* Saarbrücken, Germany.

Barg, Petra and Markus Walther. 1998. Processing Unknown Words in HPSG. *Pages 91–95 of: Proceedings of COLING-ACL'98.* Montréal, Canada.

Benczes, Réka. 2004. Idioms. *Linguistics*, **5**(1–2), 1–21.

Berger, Adam L., Stephen A. Della Pietra, and Vincent J. Della Pietra. 1996. A Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics*, **22**(1), 39–71.

Berger, James O. 2003. Could Fisher, Jeffreys, and Neyman Have Agreed on Testing? *Statistical Science*, **18**(1), 1–12.

Bisang, Walter. 2011. Word classes. *In:* Song, Jae Jung (ed), *The Oxford Handbook of Language Typology.* Oxford: Oxford University Press.

Blaheta, Don and Mark Johnson. 2001. Unsupervised learning of multi-word verbs. *Pages 54–60 of: Proceedings of the ACL/EACL 2001 Workshop on the Computational Extraction, Analysis and Exploitation of Collocations.* Toulouse, France.

Blunsom, Phil and Timothy Baldwin. 2006. Multilingual Deep Lexical Acquisition for HPSGs via Supertagging. *Pages 164–171 of: Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP 2006).* Sydney, Australia.

Bond, Francis and Caitlin Vatikiotis-Bateson. 2002. Using an ontology to determine English countability. *In: Proc. of the 19th International Conference on Computational Linguistics (COLING 2002).* Taipei, Taiwan.

Brants, Thorsten. 2000. TnT – A Statistical Part-of-Speech Tagger. *Pages 224–231 of: Proceedings of the Sixth Conference on Applied Natural Language Processing*. Seattle, WA, USA.

Brent, Michael R. 1991. Automatic acquisition of subcategorization frames from untagged text. *Pages 209–214 of: Proceedings of the 29th Meeting of the ACL*. Berkeley, CA, USA.

Brent, Michael R. 1993. From grammar to lexicon: unsupervised learning of lexical syntax. *Computational Linguistics*, **19**(2), 243–262.

Brent, Michael R. 1994. Acquisition of subcategorization frames using aggregated evidence from local syntactic cues. *Lingua*, **92**, 433–470. Reprinted in Acquisition of the Lexicon, L. Gleitman and B. Landau (Eds.), Cambridge, MA: MIT Press.

Brill, Eric. 1995. Transformation-based error-driven learning and natural language processing. *Computational Linguistics*, **21**(4), 543–565.

Brill, Eric and Raymond J. Mooney. 1997. An Overview of Empirical Natural Language Processing. *AI Magazine*, **18**(4), 13–24.

Brinton, Laural J. and Elizabeth Closs Traugott. 2005. *Lexicalization and Grammaticalization in Language Change*. Cambridge: Cambridge University Press.

Briscoe, Edward J. 2001. From Dictionary to Corpus to Self-Organizing Dictionary: Learning Valency Associations in the Face of Variation and Change. *In: Proceedings of Corpus Linguistics*. Lancaster University, UK.

Briscoe, Edward J. and John Carroll. 1997. Automatic Extraction of Subcategorization from Corpora. *Pages 356–363 of: Proceedings of the 5th ANLP Conference*. Washington, D.C., USA.

Brown, Lawrence D., Tony T. Cai, and Anirban Dasgupta. 2001. Interval Estimation for a Binomial Proportion. *Statistical Science*, **16**(2), 101–133.

Burger, Harald, Peter Kühn, and Neal R Norrick. 2007. *Phraseologie: ein internationales Handbuch zeitgenössischer Forschung*. Berlin, New York: Walter de Gruyter.

Callmeier, Ulrich. 2001. *Efficient Parsing with Large-Scale Unification Grammars*. Diplomarbeit, Universität des Saarlandes, Informatik, Saarbrücken, Germany.

Callmeier, Ulrich, A. Eisele, U. Schäfer, and M. Siegel. 2004. The DeepThought Core Architecture Framework. *Pages 1205–1208 of: Proceedings of the 4th International Conference on Language Resources and Evaluation. LREC-04.* Lisbon, Portugal: European Language Resources Association.

Caraballo, Sharon A. and Eugene Charniak. 1999. Determining the specificity of nouns from text. *In: Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora.* College Park, MD, USA.

Cardie, Claire. 1993. A Case-Based Approach to Knowledge Acquisition for Domain-Specific Sentence Analysis. *Pages 798–803 of: The Eleventh National Conference on Artificial Intelligence.* Washington, D.C., USA.

Cardie, Claire. 1996. Embedded Machine Learning Systems for Natural Language Processing: A General Framework. *In:* Wermter et al. (1996).

Cardie, Claire and Raymond J. Mooney. 1999. Guest Editor's Introduction: Machine Learning and Natural Language. *Machine Learning. Special Issue on Natural Language Learning*, **34**(1/2/3), 5–9.

Carpenter, Bob. 1991. Typed Feature Structures: an Extension of First-order Terms. *In: Proceedings of the International Symposium on Logic Programming.* San Diego, CA, USA.

Carpenter, Bob. 1992. *The Logic of Typed Feature Structures.* New York: Cambridge University Press.

Carpenter, Bob. 1993. Skeptical and Credulous Unification with Applications to Lexical Templates and Inheritance. *Pages 13–37 of:* Briscoe, Edward J., A. Copestake, and V. de Paiva (eds), *Inheritance, Defaults, and the Lexicon.* Cambridge: Cambridge University Press.

Carpenter, Bob and Gerald Penn. 2001. *ALE - The Attribute Logic Engine User's Guide Version 3.2.1.*

Carroll, Glenn and Mats Rooth. 1998. Valence induction with a head-lexicalized PCFG. *In: Proceedings of the 3rd Conference on Empirical Methods in Natural Language Processing (EMNLP 3).* Granada, Spain.

Carroll, John A. and Alex C. Fang. 2004. The Automatic Acquisition of Verb Subcategorizations and their Impact on the Performance of an HPSG parser. *In: The First International Conference on Natural Language Processing.* Sanya City, Hainan Island, China.

Chafaï, Djalil and Didier Concordet. 2009. Confidence regions for the multinomial parameter with small sample size. *Journal of the American Statistical Association*, **104**(487), 1071–1079.

Chierchia, Gennaro. 1998. Reference to Kinds across Languages. *Natural Language Semantics*, **6**, 339–405.

Chomsky, Noam. 1957. *Syntactic Structures.* The Hague: Mouton.

Chomsky, Noam. 1965. *Aspects of the Theory of Syntax.* Cambridge, MA: MIT Press.

Chomsky, Noam. 1981. *Lectures on Government and Binding.* Foris Publications: Foris Publications.

Chomsky, Noam. 1995. *The Minimalist Program.* Cambridge, MA: MIT Press.

Chomsky, Noam and Morris Halle. 1968. *The Sound Patterns of English.* New York: Harper and Row.

Church, Kenneth W. and Patrick Hanks. 1990. Word association norms, mutual information and lexicography. *Computational Linguistics*, **16**(1), 22–29.

Church, Kenneth W., W. Gale, P. Hanks, and D. Hindle. 1991. Using statistics in lexical analysis. *Pages 115–164 of:* Zernik, Uri (ed), *Lexical Acquisition: Using On-line Resources to Build a Lexicon.* Hillsdale: Lawrence Erlbaum.

Ciaramita, Massimiliano. 2002. Boosting automatic lexical acquisition with morphological information. *Pages 17–25 of: Proceedings of the ACL-02 workshop on Unsupervised lexical acquisition.* Philadelphia, PA, USA.

Cimiano, Philipp and Johanna Wenderoth. 2005. Automatically Learning Qualia Structures from the Web. *Pages 28–37 of: Proceedings of the ACL-SIGLEX Workshop on Deep Lexical Acquisition.* Ann Arbor, MI, USA.

Clark, James. 1999. *XSL Transformations (XSLT) Version 1.0.* World Wide Web Consortium. http://www.w3.org/TR/xslt.

Claveau, Vincent, P. Sébillot, C. Fabre, and P. Bouillon. 2003. Learning Semantic Lexicons from a Part-of-Speech and Semantically Tagged Corpus Using Inductive Logic Programming. *Journal of Machine Learning Research*, **4**, 493–525.

Copestake, Ann. 2001. The semi-generative lexicon: Limits on lexical productivity. *Pages 41–49 of: Proceedings of the First International Workshop on Generative Approaches to the Lexicon.* Geneva, Switzerland.

Copestake, Ann. 2007. Semantic Composition with (Robust) Minimal Recursion Semantics. *Pages 73–80 of: ACL 2007 Workshop on Deep Linguistic Processing.* Prague, Czech Republic.

Copestake, Ann and Dan Flickinger. 2000. An open-source grammar development environment and broad-coverage English grammar using HPSG. *In: Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC 2000).* Athens, Greece.

Copestake, Ann and Alex Lascarides. 1997. Integrating Symbolic and Statistical Representations: The Lexicon Pragmatics Interface. *Pages 136–143 of: Proceedings of the 35th Annual Meeting of the ACL and 8th Conference of the EACL (ACL-EACLŠ97).* Madrid, Spain.

Copestake, Ann, D. Flickinger, I.A. Sag, and C. Pollard. 1999a. *Minimal Recursion Semantics: an Introduction.* Draft.

Copestake, Ann, J. Carroll, R. Malouf, and S. Oepen. 1999b. *The (New) LKB System.* CSLI, Stanford University.

Copestake, Ann, F. Lambeau, A. Villavicencio, F. Bond, T. Baldwin, I.A. Sag, and D. Flickinger. 2002. Multiword Expressions: Linguistic Precision and Reusability. *Pages 1941–7 of: Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC 2002).* Las Palmas, Canary Islands.

Copestake, Ann, D. Flickinger, C. Pollard, and I.A. Sag. 2005. Minimal Recursion Semantics: an Introduction. *Research on Language and Computation*, **3.4**, 281–332.

Croft, William. 1991. *Syntactic categories and grammatical relations : the cognitive organization of information.* Chicago: The Univ. of Chicago Press.

Cucerzan, Silviu and David Yarowsky. 2003. Minimally supervised induction of grammatical gender. *Pages 40–47 of: NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology.* Edmonton, Canada.

Daelemans, Walter, Antal Van Den Bosch, and Jakub Zavrel. 1999. Forgetting Exceptions is Harmful in Language Learning. *Machine Learning, Special issue on Natural Language Learning*, **34**(1–3), 11–41.

De Marcken, Carl G. 1996. *Unsupervised language acquisition*. Ph.D. thesis, Massachusetts Institute of Technology.

Dorr, Bonnie. 1992. The Use of Lexical Semantics in Interlingual Machine Translation. *Journal of Machine Translation*, **7**(3), 135–193.

Dorr, Bonnie J. and Doug Jones. 1996. Role of word sense disambiguation in lexical acquisition: predicting semantics from syntactic cues. *Pages 322–327 of: Proceedings of the 16th conference on Computational linguistics*. Copenhagen, Denmark.

Dunning, Ted. 1993. Accurate methods for the statistics of surprise and coincidence. Computational Linguistics. *Computational Linguistics*, **19**, 61–74.

Earley, Jay. 1970. An efficient context-free parsing algorithm. *Communications of the ACM*, **13**(2), 94–102.

Eckle-Kohler, Judith. 1998. Methods for quality assurance in semi-automatic lexicon acquisition from corpora. *In: Proceedings of EURALEX 1998*. Liège, Belgium.

Erbach, Gregor. 1990. Syntactic processing of Unknown words. *Pages 371–382 of:* Jorrand, P. and V. Sgurev (eds), *Artificial Intelligence IV: Methodology, Systems, Applications (AIMSA '90)*. Amsterdam, The Netherlands: Elsevier Science Ltd.

Erbach, Gregor. 1992. Head-Driven Lexical Representation of Idioms in HPSG. *Pages 11–24 of: International Conference on Idioms*. Tilburg, The Netherlands.

Erbach, Gregor and Brigitte Krenn. 1993. *Idioms and Support Verb Constructions in HPSG*. CLAUS-Report. Universität des Saarlandes, Saarbrücken.

Erjavec, Tomaž and Sasčo Džeroski. 2004. Machine Learning of Morphosyntactic Structure: Lemmatizing Unknown Slovene Words. *Applied Artificial Intelligence*, **18**, 17–41.

Ersan, Murat and Eugene Charniak. 1996. A Statistical Syntactic Disambiguation Program and What It Learns. *Pages 146–159 of:* Wermter, S., E. Riloff, and G. Scheler (eds), *Connectionist, Statistical and Symbolic Approaches in Learning for Natural Language Processing.* Lecture Notes in Artifical Intelligence, vol. 1040. Berlin: Springer-Verlag.

Evans, Roger and Gerald Gazdar. 1989. The Semantics of DATR. *Pages 79–87 of:* Cohn, A. (ed), *Proceedings of the Seventh Conference of the Society for the Study of Artificial Intelligence and Simulation of Behaviour.* London: Pitman.

Evert, Stefan. 2004. The Statistical Analysis of Morphosyntactic Distributions. *Pages 1539–1542 of: Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC 2004).* Lisbon, Portugal.

Evert, Stefan. 2005. *The Statistics of Word Cooccurrences: Word Pairs and Collocations.* Ph.D. thesis, Institut für maschinelle Sprachverarbeitung, University of Stuttgart.

Fabre, Cecile and Didier Bourigault. 2001. Linguistic clues for corpus-based acquisition of lexical dependencies. *Pages 176–184 of: Proceedings of the Corpus Linguistics 2001 Conference, UCREL Technical Papers*, vol. 3. Lancaster University.

Farrar, Scott and William D. Lewis. 2005. The GOLD Community of Practice - An Infrastructure for Linguistic Data on the Web. *In: Proceedings of the EMELD 2005 Workshop on Digital Language Documentation: Linguistic Ontologies and Data Categories for Language Resources.*

Fermé, Eduardo L. and Sven Ove Hansson. 2001. Shielded Contraction. *Pages 85–107 of:* Rott, H. and M.-A. Williams (eds), *Frontiers of Belief Revision.* Kluwer Academic Publishers.

Firth, John Rupert. 1957. A synopsis of linguistic theory 1930-55. *Pages 1–32 of: Studies in Linguistic Analysis (special volume of the Philological Society).* Oxford: The Philological Society.

Fisher, Ronald A. 1925. *Statistical methods for research workers.* Edinburgh: Oliver & Boyd.

Fitzpatrick, Simon and Alastair Scott. 1987. Quick Simultaneous Confidence Intervals for Multinomial Proportions. *Journal of the American Statistical Association*, **82**(399), 875–878.

Flickinger, Dan. 2000. On building a more efficient grammar by exploiting types. *Natural Language Engineering*, **6**(1), 15–28.

Flickinger, Dan, Jan T. Lønning, H. Dyvik, S. Oepen, and F. Bond. 2005. SEM-I rational MT. Enriching deep grammars with a semantic interface for scalable machine translation. *Pages 165–172 of: Proceedings of the 10th Machine Translation Summit.* Phuket, Thailand.

Fouvry, Frederik. 2003. Lexicon acquisition with a large-coverage unification-based grammar. *In: Proceedings of the 10th Conference of the EACL (EACL 2003).* Budapest, Hungary.

Frank, Anette, K. Spreyer, W. Drozdzynski, H.-U. Krieger, and U. Schäfer. 2004. Constraint-Based RMRS Construction from Shallow Grammars. *In:* Müller, Stefan (ed), *Proceedings of the HPSG04 Conference Workshop on Semantics in Grammar Engineering.* Leuven, Belgium.

Fraser, Bruce. 1974. *The Verb-Particle Combination in English.* Tokyo: Taishukan Publishing Company.

Fritz, Peter. 2009. *Belief Revision in Dynamic Epistemic Logic and Ranking Theory.* Tech. rept. Bachelor thesis, Universität Konstanz.

Gärdenfors, Peter and David Makinson. 1988. Revisions of Knowledge Systems Using Epistemic Entrenchment. *Pages 83–95 of: Proceedings of the 2nd Conference on Theoretical Aspects of Reasoning about Knowledge.* Pacific Grove, CA, USA.

Gazdar, Gerald, Ewan Klein, Geoffrey Pullum, and Ivan A. Sag. 1985. *Generalized Phrase Structure Grammar.* Oxford: Basil Blackwell.

Gerdemann, Dale and Erhard W. Hinrichs. 1988. Unicorn: A unification parser for atribute-value grammars. *Studies in Linguistic Sciences*, **18**(2), 41–86.

Girju, Roxana, Adriana Badulescu, and Dan Moldovan. 2003. Learning semantic constraints for the automatic discovery of part-whole relations. *Pages 80–87 of: Proceedings of HLT/NAACL-03.* Edmonton, Canada.

Goldsmith, John. 2001. Unsupervised learning of the morphology of a natural language. *Computational Linguistics*, **27**(2), 153–198.

Goodman, Leo A. 1965. On simultaneous confidence intervals for multinomial proportions. *Technometrics*, **7**(2), 247–254.

Grishman, Ralph, Catherine Macleod, and Adam Meyers. 1994. Comlex syntax: Building a computational lexicon. *Pages 268–272 of: Proceedings of the 15th International Conference on Computational Linguistics*. Kyoto, Japan.

Grove, Adam. 1988. Two Modellings for Theory Change. *Journal of Philosophical Logic*, **17**(2), 157–170.

Haas, Mary R. 1942. The Use of Numeral Classifiers in Thai. *Language*, **18**, 201–206.

Heid, Ulrich and Hannah Kermes. 2002. Providing Lexicographers with Corpus Evidence for fine-grained syntactic description: Adjectives taking subject and complement clauses. *In: Proceedings of the Xth EURALEX International Congress*. Copenhagen, Denmark.

Hermjakob, Ulf. 1997. *Learning parse and translation decisions from examples with rich context*. Ph.D. thesis, The University of Texas at Austin.

Heylen, Dirk, Kerry G. Maxwell, and Marc Verhagen. 1994. Lexical functions and machine translation. *Pages 1240–1244 of: Proceedings of the 15th conference on Computational linguistics*. Kyoto, Japan.

Hindle, Donald and Mats Rooth. 1993. Structural Ambiguity and Lexical Relations. *Computational Linguistics*, **19**(1), 103–120.

Hintikka, Jaakko. 1962. *Knowledge and Belief: An Introduction to the Logic of the Two Notions*. Ithaca, N. Y.: Cornell University Press.

Horiguchi, Keiko, Kentaro Torisawa, and Jun'ichi Tsujii. 1995. Automatic acquisition of content words using an HPSG-based parser. *Pages 320–325 of: Proceedings of the Natural Language Processing Pacific Rim Symposium*. Seoul, Korea.

Hornby, A. S. 2005. *Advanced Learner's Dictionary of Current English*. 7th edn. Oxford: Oxford University Press.

Hou, Chia-Ding, Jengtung Chiang, and John J. Tai. 2003. A family of simultaneous confidence intervals for multinomial proportions. *Computational Statistics and Data Analysis*, **43**(1), 29–45.

Hubbard, Raymond and M. J. Bayarri. 2003. Confusion Over Measures of Evidence (p's) Versus Errors (α's) in Classical Statistical Testing. *American Statistical Association*, **57**(3), 171–182.

Jakobson, Roman, Gunnar Fant, and Morris Halle. 1952. *Preliminaries to Speech Analysis.* Cambridge, MA: The MIT Press.

Jeffreys, Harold. 1961. *Theory of Probability, 3rd Edition.* Oxford: Oxford University Press.

Joanis, Eric and Suzanne Stevenson. 2003. A General Feature Space for Automatic Verb Classification. *In: Proceedings of the 10th Conference of the EACL (EACL 2003.* Budapest, Hungary.

Johnson, Mark. 1988. Attribute-Value Logic and the Theory of Grammar. *In: CSLI Lecture Notes*, vol. 14. Stanford: Center for the Study of Language and Information.

Kanger, Stig. 1970. New Foundations for Ethical Theory. *Pages 36–58 of:* Hilpinen, Risto (ed), *Deontic Logic: Introductory and Systematic Readings.* Dordrecht: Reidel.

Kaplan, Ronald M. and Joan Bresnan. 1982. Lexical Functional Grammar: A Formal System for Grammatical Representation. *In:* Bresnan, J. (ed), *The Mental Representation of Grammatical Relations.* Cambridge, MA: MIT Press.

Kasper, Robert T. and William C. Rounds. 1986. A logical semantics for feature structures. *Pages 257–266 of: Proceedings of 24th Meeting of the Association for Computational Linguistics.* New York, NJ, USA: Association for Computational Linguistics.

Katz, Jerrold J. and Paul M. Postal. 1964. *An Integrated Theory of Linguistic Description.* Cambridge, MA: The MIT Press.

Kay, Martin. 1979. Functional Grammar. *Pages 142–158 of: Proceedings of the 5th Annual Meeting of the Berkeley Linguistic Society.* Berkeley, CA, USA.

Kilbury, James, Petra Naerger [Barg], and Ingrid Renz. 1992. New Lexical Entries for Unknown Words. *In: Theorie des Lexikons: Arbeiten des Sonderforschungsbereichs 282.* Düsseldorf: Heinrich-Heine-Universität Düsseldorf.

Kilbury, James, Petra Barg, and Ingrid Renz. 1994. Simulation Lexikalischen Erwerbs. *Pages 251–271 of:* Felix, S.W., Chr. Habel, and G. Rickheit (eds), *Kognitive Linguistik: Repräsentation und Prozesse.* Westdeutscher Verlag.

Kim, Su Nam and Timothy Baldwin. 2005. Automatic interpretation of noun compounds using WordNet similarity. *Pages 945–956 of: Proceedings of the 2nd International Joint Conference on Natural Language Processing, Jeju Island, South Korea.* Jeju, South Korea.

Kim, Su Nam and Timothy Baldwin. 2006. Automatic Identification of English Verb Particle Constructions using Linguistic Features. *Pages 65–72 of: Proceedings of the Third ACL-SIGSEM Workshop on Prepositions.* Trento, Italy.

Kim, Su Nam and Timothy Baldwin. 2007. Detecting Compositionality of English Verb-Particle Constructions using Semantic Similarity. *Pages 40–48 of: Proceedings of PACLING 2007 - 10th Conference of the Pacific Association for Computational Linguistics.* Melbourne, Australia.

Kim, Su Nam and Timothy Baldwin. 2008. Standardised Evaluation of English Noun Compound Interpretation. *Pages 39–42 of: Proceedings of the LREC Workshop Towards a Shared Task for Multiword Expressions (MWE 2008).* Marrakech, Morocco.

Korhonen, Anna. 2002. *Subcategorization Acquisition.* Ph.D. thesis, University of Cambridge.

Korhonen, Anna and Edward J. Briscoe. 2004. Extended Lexical-Semantic Classification of English Verbs. *Pages 38–45 of: Proceedings of the HLT-NAACL 2004 Workshop on Computational Lexical Semantics.* Boston, MA, USA.

Krieger, Hans-Ulrich and Ulrich Schäfer. 1994. *TDL - A Type Description Language for HPSG. Part 1: Overview.* Research Report RR-94-37. DFKI, Saarbrücken.

Kuhn, Jonas. 1997. *Die Behandlung von Funktionsverbgefügen in einem HPSG-basierten Übersetzungsansatz.* Verbmobil-Report Nr. 66. Universität Stuttgart.

Kuhn, Jonas, Judith Eckle-Kohler, and Christian Rohrer. 1998. Lexicon Acquisition with and for Symbolic NLP-Systems – a Bootstrapping Approach. *Pages 89–95 of: Proceedings of LREC-98, 1st International Conference on Language Resources and Evaluation.* Grenada, Spain.

Lapata, Maria. 1999. Acquiring lexical generalizations from corpora: A case study for diathesis alternations. *Pages 397–404 of: Proceedings of 37th Meeting of ACL.* College Park, Maryland, USA.

Lapata, Maria. 2002. The Disambiguation of Nominalizations. *Computational Linguistics*, **28**, 357–388.

Lee, Kiyong, Lou Burnard, Laurent Romary, Eric De La Clergerie, Thierry Declerck, Syd Bauman, Harry Bunt, Lionel Clement, Tomaz Erjavec, Azim Roussanaly, and Claude Roux. 2004. Towards an international standard on feature structures representation. *Pages 373–376 of: 4th International Conference on Language Resources and Evaluation - LREC'04*. Lisbonne, Portugal.

Leech, Geoffrey. 1992. 100 million words of English: the British National Corpus. *Language Research*, **28**(1), 1–13.

Leech, Geoffrey, Paul Rayson, and Andrew Wilson. 2001. *Word Frequencies in Written and Spoken English: Based on the British National Corpus*. London: Longman.

Lehmann, Erich L. 2006. On likelihood ratio tests. *IMS LECTURE NOTES-MONOGRAPH SERIES*, **49**, 1.

Lehmann, Erich L. and Joseph P. Romano. 2005. *Testing Statistical Hypotheses (Springer Texts in Statistics)*. third edn. Springer.

Lehrer, Keith. 1998. *Theory of Knowledge*. Cambridge MA: MIT Press.

Light, Marc. 1996. Morphological Cues for Lexical Semantics. *Pages 25–31 of: Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*. Santa Cruz, CA, USA.

Manning, Christopher D. 1991. *LFG within King's descriptive formalism*. ms. Stanford University, Stanford CA.

Manning, Christopher D. 1993. Automatic Acquisition of a Large Subcategorization Dictionary from Corpora. *Pages 235–242 of: Proceedings of the 31st Meeting of the ACL*. Columbus, Ohio.

Manning, Christopher D. and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. Cambridge, MA: The MIT Press.

Matsumoto, Yuji and Takehito Utsuro. 2000. Lexical Knowledge Acquisition. *Pages 563–610 of:* Dale, Robert, Hermann Moisl, and H. L. Somers (eds), *Handbook of Natural Language Processing*. New York, NY: Marcel Dekker, Inc.

McCarthy, Diana. 2001. *Lexical Acquisition at the Syntax-Semantics Interface: Diathesis Alternations, Subcategorization Frames and Selectional Preferences.* Ph.D. thesis, University of Sussex, Sussex, UK.

McCarthy, Diana, John Carroll, and Judita Preiss. 2001. Disambiguating noun and verb senses using automatically acquired selectional preferences. *In: Proceedings of the SENSEVAL-2 Workshop at ACL/EACL'01.* Toulouse, France.

McCarthy, Diana, Bill Keller, and John Carroll. 2003. Detecting a continuum of compositionality in phrasal verbs. *Pages 73–80 of: Proceedings of the ACL 2003 workshop on Multiword expressions.* Sapporo, Japan.

McKeown, Kathleen R. and Dragomir R. Radev. 2000. Collocations. *Pages 507–523 of: A Handbook of Natural Language Processing.* New York, NY: Marcel Dekker.

Meurers, Walt Detmar. 2001. On expressing lexical generalizations in HPSG. *Nordic Journal of Linguistics*, **24**(2), 161–217. Special issue on "The Lexicon in Linguistic Theory".

Moldovan, Dan, A. Badulescu, M. Tatu, D. Antohe, and R. Girju. 2004. Models for the Semantic Classification of Noun Phrases. *Pages 60–67 of: HLT-NAACL 2004: Workshop on Computational Lexical Semantics.* Boston, MA, USA.

Moon, Rosamund. 1998. *Fixed Expressions and Idioms in English: A Corpus-Based Approach.* Oxford Studies in Lexicography and Lexicology. New York: Oxford University Press.

Nakov, Preslav and Marti A. Hearst. 2008. Solving Relational Similarity Problems Using the Web as a Corpus. *Pages 452–460 of: Proceedings of ACL-08: HLT.* Columbus, OH, USA.

Neyman, Jerzy and Egon S. Pearson. 1928. On the use and interpretation of certain test criteria for purposes of statistical inference I, II. *Biometrica*, **20A**, 175–240, 263–294.

Neyman, Jerzy and Egon S. Pearson. 1933. On the Problem of the Most Efficient Tests of Statistical Hypotheses. *Royal Society of London Philosophical Transactions Series A*, **231**, 289–337.

Nicholson, Jeremy, Timothy Baldwin, and Phil Blunsom. 2006. Die Morphologie (f): Targeted Lexical Acquisition for Languages Other than English. *Pages 67–74 of: Proceedings of the Australasian Language Technology Workshop 2006*. Sydney, Australia.

Nunberg, Geoffrey, Thomas Wasow, and Ivan A. Sag. 1994. Idioms. *Language*, **70**(3), 491–539.

Ó Séaghdha, Diarmuid. 2008. *Learning compound noun semantics*. Ph.D. thesis, Computer Laboratory, University of Cambridge. Published as University of Cambridge Computer Laboratory Technical Report 735.

Pearce, Darren. 2001. Synonymy in Collocation Extraction. *Pages 41–46 of: Proceedings of the NAACL 2001 Workshop on WordNet and Other Lexical Resources: Applications, Extensions and Customizations*. Pittsburgh, PA, USA.

Pecina, Pavel. 2005. An Extensive Empirical Study of Collocation Extraction Methods. *In: ACLstudent '05 Proceedings of the ACL Student Research Workshop*. Ann Arbor, MI, USA.

Pollard, Carl and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar*. Chicago, London: University of Chicago Press and CSLI Publications.

Poznański, Victor and Antonio Sanfilippo. 1996. Detecting dependencies between semantic verb subclasses and subcategorization frames in text corpora. *Pages 175–190 of: Corpus processing for lexical acquisition*. Cambridge, MA, USA: MIT Press.

Pustejovsky, James. 1995. *The Generative Lexicon*. Cambridge, MA: MIT Press.

Quine, Willard v. O. 1953. *From a Logical Point of View*. New York: Harper and Row.

Quirk, Randolph, S. Greenbaum, G. Leech, and J. Svartvik. 1994. *A Comprehensive Grammar of the English Language*. Twelfth impression edn. London and New York: Longman.

Resnik, Philip Stuart. 1993. *Selection and Information: A Class-based Approach to Lexical Relationships*. Ph.D. thesis, University of Pennsylvania.

Ribas, Francesc. 1995. *On Acquiring Appropriate Selectional Restrictions from Corpora Using a Semantic Taxonomy.* Ph.D. thesis, University of Catalonia.

Roland, Douglas and Daniel Jurafsky. 1998. How Verb Subcategorization Frequencies Are Affected By Corpus Choice. *Pages 1122–1128 of: COLING '98 Proceedings of the 17th international conference on Computational linguistics*, vol. 2. Morristown, NJ, USA.

Romary, M. Laurent (chair) and TC 37/SC 4/WG 2. 2006. *Language resource management – Feature structures – Part 1: Feature structure representation.* International Organization for Standardization (ISO), ISO 24610–1.

Rooth, Mats, S. Riezler, D. Prescher, S. Schulte im Walde, G. Carroll, and F. Beil. 1998. Inducing Lexicons with the EM Algorithm. *In: AIMS - Arbeitspapiere des Instituts für Maschinelle Sprachverarbeitung, Vol.4, No. 3.* Stuttgart, Germany.

Rounds, William C. 1997. Feature Logics. *In:* Benthem, J. Van and A. ter Meulen (eds), *Handbook of Logic and Language.* Amsterdam: Elsevier Science.

Russel, Dale W. 1993. *Language Acquisition in a Unification-Based Grammar Processing System Using a Real-World Knowledge Base.* Ph.D. thesis, University of Illinois.

Sag, Ivan A., T. Baldwin, F. Bond, A. Copestake, and D. Flickinger. 2002. Multiword Expressions: a Pain in the Neck for NLP. *In: Proceedings of CICLING 2002.* Mexico City, Mexico.

Samuelsson, Christer and Manny Rayner. 1991. Quantitative Evaluation of Explanation-Based Learning as an Optimization Tool for a Large-Scale Natural Language System. *Pages 609–615 of: Proceedings of the Twelfth International Joint Conference on Artificial Intelligence.* Sydney, Australia.

Sarkar, Anoop and Woottiporn Tripasai. 2002. Learning verb argument structure from minimally annotated corpora. *In: COLING '02 Proceedings of the 19th international conference on Computational linguistics.* Taipei, Taiwan.

Sarkar, Anoop and Daniel Zeman. 2000. Automatic Extraction of Subcategorization Frames for Czech. *In: Proceedings of COLING 2000.* Saarbrücken, Germany.

Schäfer, Ulrich. 2007. *Integrating Deep and Shallow Natural Language Processing Components – Representations and Hybrid Architectures.* Ph.D. thesis, Faculty of Mathematics and Computer Science, Saarland University, Saarbrücken, Germany. Doctoral Dissertation; also available as Vol. 22 of the Saarbrücken Dissertations in Computational Linguistics and Language Technology series (http://www.dfki.de/lt/diss), ISBN 978-3-933218-21-6.

Schulte im Walde, Sabine. 2002. Evaluating Verb Subcategorisation Frames learned by a German Statistical Grammar against Manual Definitions in the *Duden* Dictionary. *Pages 187–197 of: Proceedings of the 10th EURALEX International Congress.* Copenhagen, Denmark.

Schulte im Walde, Sabine. 2003. *Experiments on the Automatic Induction of German Semantic Verb Classes.* Ph.D. thesis, Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart. Published as AIMS Report 9(2).

Schulte im Walde, Sabine. 2009. The Induction of Verb Frames and Verb Classes from Corpora. *Chap. 44, pages 952–971 of:* Lüdeling, Anke and Merja Kytö (eds), *Corpus Linguistics. An International Handbook.* Handbooks of Linguistics and Communication Science, vol. 2. Berlin: Mouton de Gruyter.

Schulte im Walde, Sabine, H. Schmid, M. Rooth, S. Riezler, and D. Prescher. 2001. Statistical Grammar Models and Lexicon Acquisition. *Pages 389–440 of: Linguistic Form and its Computation.* Stanford, CA: CSLI Publications.

Schütze, Hinrich. 1998. Automatic Word Sense Discrimination. *Computational Linguistics*, **24**(1), 97–123.

Segerberg, Krister. 1995. Belief Revision From the Point of View of Doxastic Logic. *Bulletin of the IGPL*, **3**, 535–553.

Shannon, Claude E. 1948. A mathematical theory of communication. *Bell system technical journal*, **27**, 379–423,623–656.

Shieber, Stuart M. 1985. Using restriction to extend parsing algorithms for complex feature-based formalisms. *Pages 145–152 of: ACL Proceedings of the 23rd Annual Meeting.* Chicago, IL, USA.

Shieber, Stuart M., H. Uszkoreit, F. Pereira, J. Robinson, and M. Tyson. 1983. The Formalism and Implementation of PATR-II. *Techreport 4, pages 39–79 of:* Grosz, Barbara J. and Mark Stickel (eds), *Research on*

Interactive Acquisition and Use of Knowledge. Menlo Park, CA, USA: SRI International. Final report for SRI Project 1894.

Siegel, Eric V. and Kathleen R. McKeown. 2000. Learning Methods to Combine Linguistic Indicators: Improving Aspectual Classification and Revealing Linguistic Insights. *Computational Linguistics*, **26**(4), 595–628.

Sison, Cristina P. and Joseph Glaz. 1995. Simultaneous Confidence Intervals and Sample Size Determination for Multinomial Proportions. *Journal of the American Statistical Association*, **90**(429), 366–369.

Soehn, Jan-Philipp. 2005. Selectional Restrictions in HPSG: I'll eat my hat! *Pages 343–353 of:* Müller, Stefan (ed), *The Proceedings of the 12th International Conference on Head-Driven Phrase Structure Grammar*. Department of Informatics, University of Lisbon.

Soehn, Jan-Philipp. 2006. *Über Bärendienste und erstaunte Bauklötze – Idiome ohne freie Lesart in der HPSG*. Phil. Dissertation (2005), Friedrich-Schiller-Universität Jena.

Soehn, Jan-Philipp and Manfred Sailer. 2003. At first blush on tenterhooks. About selectional restrictions imposed by nonheads. *Pages 149–161 of: Proceedings of Formal Grammar 2003*. Vienna, Austria.

Spohn, Wolfgang. 1988. Ordinal conditional functions: A dynamic theory of epistemic states. *Pages 105–134 of:* Harper, W. L. and B. Skyrms (eds), *Causation in Decision, Belief Change and Statistics*. Springer.

Stevenson, Suzanne and Paola Merlo. 1999. Automatic verb classification using distributions of grammatical features. *Pages 45–52 of: Proceedings of EACL '99*. Bergen, Norway.

Stowell, Tim. 1992. The Role of the Lexicon in Syntactic Theory. *Pages 9–21 of:* Stowell, Tim and Eric Wehrli (eds), *Syntax and the Lexicon*. San Diego, California: Academic Press.

Stvan, Laurel Smith. 1998. *The Semantics and Pragmatics of Bare Singular Noun Phrases*. Ph.D. thesis, Northwestern University.

Thompson, Cynthia A. and Raymond J. Mooney. 1998. *Semantic Lexicon Acquisition for Learning Natural Language Interfaces*. Tech. rept. Department of Computer Sciences, University of Texas.

Trawiński, Beata, M. Sailer, J.P. Soehn, L. Lemnitzer, and F. Richter. 2008. Cranberry Expressions in English and in German. *Pages 35–38 of: Proceedings of the LREC Workshop Towards a Shared Task for Multiword Expressions (MWE 2008).* Marrakech, Morocco.

Ushioda, Akira, David A. Evans, Ted Gibson, and Alex Waibel. 1993. The Automatic Acquisition of Frequencies of Verb Subcategorization Frames from Tagged Corpora. *Pages 95–106 of:* Boguraev, B. and J. Pustejovsky (eds), *Proceedings of the Workshop on Acquisition of Lexical Knowledge from Text.* Columbus, OH, USA.

Uszkoreit, Hans. 1986. Categorial Unification Grammar. *Pages 187–194 of: Proceedings of the 11th International Conference on Computational Linguistics.* Bonn, Germany.

Viegas, Evelyne, B. Onyshkevych, V. Raskin, and S. Nirenburg. 1996. From Submit to Submitted via Submission: On Lexical Rules in Large-scale Lexicon Acquisition. *Pages 32–39 of: Proceedings of the 34st Annual Meeting of the Association for Computational Linguistics.* Santa Cruz, CA, USA.

Villavicencio, Aline. 2003a. Verb-Particle Constructions and Lexical Resources. *Pages 57–64 of: Proceedings of the ACL-SIGLEX Workshop on Multiword Expressions: Analysis, Acquisition and Treatment.* Sapporo, Japan.

Villavicencio, Aline. 2003b. Verb Particle Constructions in the WWW. *Pages 101–111 of: Proceedings of the ACL-SIGSEM Workshop on the Linguistic Dimensions of Prepositions and their Use in Computational Linguistics Formalisms and Applications.* Toulouse, France.

Villavicencio, Aline and Ann Copestake. 2002. Verb-particle constructions in a computational grammar of English. *Pages 357–371 of: Proceedings of the 9th International Conference on Head-Driven Phrase Structure Grammar.* Seoul, Korea.

Villavicencio, Aline, V. Kordoni, Y. Zhang, M. Idiart, and C. Ramisch. 2007. Validation and Evaluation of Automatically Acquired Multiword Expressions for Grammar Engineering. *Pages 1034–1043 of: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL).* Prague, Czech Republic.

Waldron, Ben, A. Copestake, U. Schäfer, and B. Kiefer. 2006. Preprocessing and Tokenisation Standards in DELPH-IN Tools. *In: Proceedings of the 5th*

International Conference on Language Resources and Evaluation LREC-2006. Genoa, Italy.

Wauschkuhn, Oliver. 1999. *Automatische Extraktion von Verbvalenzen aus deutschen Textcorpora*. Aachen: Shaker Verlag.

Weerahandi, Sam. 1995. *Exact statistical methods for data analysis*. New York: Springer.

Wermter, Stefan, Ellen Riloff, and Gabriele Scheler. 1996. Learning approaches for natural language processing. *Pages 1–16 of:* Wermter, Stefan, Ellen Riloff, and Gabriele Scheler (eds), *Connectionist, Statistical and Symbolic Approaches in Learning for Natural Language Processing*. Lecture Notes in Artifical Intelligence, vol. 1040. Berlin: Springer-Verlag.

Widdows, Dominic. 2004. *Geometry and Meaning*. Stanford, CA: CLSI Publications.

Wilcock, Graham. 2007. An OWL Ontology for HPSG. *Pages 169–172 of: Proceedings of the ACL-2007 Demo and Poster Sessions. 45th Annual Meeting of the Association for Computational Linguistics*. Prague, Czech Republic.

Wilson, Edwin B. 1927. Probable Inference, the Law of Succession, and Statistical Inference. *Journal of the American Statistical Association*, **22**(158), 209–212.

Wunderlich, Dieter and Ray Fabri. 1995. Minimalist morphology: an approach to inflection. *Zeitschrift für Sprachwissenschaft*, **14**(2), 236–294.

Yallop, Jeremy, Anna Korhonen, and Edward J. Briscoe. 2005. Automatic Acquisition of Adjectival Subcategorization from Corpora. *In: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*. Michigan, USA.

Yamada, Ichiro and Timothy Baldwin. 2004. Automatic Discovery of Telic and Agentive Roles from Corpus Data. *Pages 115–126 of: Proceedings of the 18th Pacific Asia Conference on Language, Information and Computation (PACLIC 18)*. Tokyo, Japan.

Yamada, Ichiro, T. Baldwin, H. Sumiyoshi, M. Shibata, and N. Yagi. 2007. Automatic Acquisition of Qualia Structure from Corpus Data. *IEICE Transactions on Information and Systems*, **E90-D**(10), 1534–1541.

Yarowsky, David. 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. *Pages 189–196 of: Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics.* Cambridge, MA, USA.

Yarowsky, David and Richard Wicentowski. 2000. Minimally supervised morphological analysis by multimodal alignment. *Pages 207–216 of: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics.* ACL '00. Hong Kong.

Zelle, John M. and Raymond J. Mooney. 1997. *An Inductive Logic Programming Method for Corpus-based Parser Construction.* Unpublished Technical Note. University of Texas, Austin.

Zhang, Yi and Valia Kordoni. 2005. A Statistical Approach towards Unknown Word Type Prediction for Deep Grammars. *Pages 24–31 of: Proceedings of the Australasian Language Technology Workshop 2005.* Sydney, Australia.

Zhang, Yi and Valia Kordoni. 2006. Automated deep lexical acquisition for robust open texts processing. *Pages 275–280 of: Proceedings of the 5th international conference on language resources and evaluation (LREC 2006).* Genoa, Italy.

Zhang, Yi, Timothy Baldwin, and Valia Kordoni. 2007. The Corpus and the Lexicon: Standardising Deep Lexical Acquisition Evaluation. *Pages 152–159 of: Proceedings of ACL 2007 Workshop on Deep Linguistic Processing.* Prague, Czech Republic.

Zinsmeister, Heike and Ulrich Heid. 2004. Collocations of complex nouns: Evidence for lexicalisation. *In: Proceedings of KONVENS 2004.* Vienna, Austria.

Zipf, George Kingsley. 1949. *Human Behaviour and the Principle of Least Effort.* Cambridge, MA: Addison-Wesley.

Ich versichere, dass ich die vorliegende Arbeit selbständig angefertigt und keine als die angegebenen Hilfsmittel benutzt habe.

Peter Jäger
Heidesheim, den 14.03.2011