Natural Computing
https://doi.org/10.1007/s11047-020-09828-w

On sampling error in genetic programming

Dirk Schweim[1] · David Wittenberg[1] · Franz Rothlauf[1]

Accepted: 25 November 2020

Abstract
The initial population in genetic programming (GP) should form a representative sample of all possible solutions (the search space). While large populations accurately approximate the distribution of possible solutions, small populations tend to incorporate a sampling error. This paper analyzes how the size of a GP population affects the sampling error and contributes to answering the question of how to size initial GP populations. First, we present a probabilistic model of the expected number of subtrees for GP populations initialized with full, grow, or ramped half-and-half. Second, based on our frequency model, we present a model that estimates the sampling error for a given GP population size. We validate our models empirically and show that, compared to smaller population sizes, our recommended population sizes largely reduce the sampling error of measured fitness values. Increasing the population sizes even more, however, does not considerably reduce the sampling error of fitness values. Last, we recommend population sizes for some widely used benchmark problem instances that result in a low sampling error. A low sampling error at initialization is necessary (but not sufficient) for a reliable search since lowering the sampling error means that the overall random variations in a random sample are reduced. Our results indicate that sampling error is a severe problem for GP, making large initial population sizes necessary to obtain a low sampling error. Our model allows practitioners of GP to determine a minimum initial population size so that the sampling error is lower than a threshold, given a confidence level.

Keywords Sampling error · Initial supply · Genetic programming · Building blocks · Initial population · Ramped half-and-half · Full · Grow · *n*-Grams

1 Introduction

In optimization, evaluating all solutions for a problem instance (complete enumeration) is often too difficult, expensive, or time-consuming (Rothlauf 2011). Therefore, population-based heuristic search methods like genetic programming (GP; Koza 1992) start with a small sample taken from the set of all solutions and improve these solutions through the application of variation operators and selection.[1]

When using a sample, there are usually differences between the properties of the statistical population and the information obtained from the sample. These differences are called *errors* (Lee et al. 2013). Non-systematic errors, describing random variations caused by observing only a subset of the statistical population are called *sampling error* (Lee et al. 2013; Cochran 1977; Särndal et al. 1992). The expected amount of sampling error can be reduced by using larger samples (Lee et al. 2013; Cochran 1977; Särndal et al. 1992).

Sampling error is a problem in evolutionary algorithms (EAs), leading to unreliable search results due to random

✉ Dirk Schweim
schweim@uni-mainz.de

David Wittenberg
wittenberg@uni-mainz.de

Franz Rothlauf
rothlauf@uni-mainz.de

[1] Johannes Gutenberg University, Jakob-Welder-Weg 9, 55128 Mainz, Germany

[1] In statistical analysis, a (statistical) population is defined as a set of objects of interest. Any subset of such a statistical population is a *sample* (Lee et al. 2013; Cochran 1977). In GP, the (initial) sample of solutions is usually also called a population. Thus, we use the terms statistical population and GP population in this article to distinguish between the two types of populations. The initial GP population is a sample of the statistical population of possible solutions to a problem instance.

Published online: 31 January 2021

variations. The problem has been discussed in the genetic algorithm (GA) literature. For example, Goldberg and Segrest (1987) note that small initial populations in a genetic algorithm (GA) can be problematic when relevant *building blocks* (BBs) are not represented by the sample.[2] Goldberg et al. (2001) and Reeves (1993) argue that an initial supply of BBs is necessary for the search to allow for the possibility that high-quality BBs will take over the population in later generations (*BB growth*). Recent work of Burlacu et al. (2015, 2018a, 2018b) evaluates this hypothesis for GP. The authors show that building blocks have a large influence on the evolutionary process. Thus, if the sampling error in an initial GP population is large, random variations can have a negative effect on the search performance. In addition, having a low sampling error is also relevant for estimation of distribution genetic programming (EDA-GP), where standard GP variation operators such as crossover and mutation are replaced by model building and sampling from the learned model (Kim et al. 2014; Shan et al. 2006). In EDA-GP algorithms, early sampling errors are learned by the model and, as a consequence, finding favorable solutions can be more difficult. Therefore, we argue that an initial (EDA-)GP population should form a representative sample of the statistical population of possible solutions and that the sampling error in the initial GP population should be low.

This article studies how the initial GP population size affects the sampling error of subtree frequencies. We present a model that estimates the minimum size of a GP population that is required for a sampling error to be below a certain value that can be specified a priori by a GP user. Therefore, we first present a probabilistic model of the expected frequencies of subtrees in GP populations initialized with full, grow, or ramped half-and-half (Koza 1992). We use n-grams of ancestors (Hemberg et al. 2012) as a possible measure to describe subtrees in GP. An n-gram of ancestors in a GP parse tree is the sequence of the values represented by a node i and its $n-1$ ancestor nodes on the same branch (parent, grandparent, greatgrandparent, etc.; Hemberg et al. 2012). The difference between the expected and the observed frequencies of n-grams in a sample is the sampling error. Thus, our model allows us to measure and investigate sampling error in initial GP populations.

Based on the model of the expected frequencies of n-grams of ancestors, we present a model to estimate the size of an initial GP population, given the desired degree of sampling error and a confidence level. Our model allows GP practitioners to estimate a minimum initial GP population size in such a way that the sampling error is lower than a threshold.

Furthermore, we empirically validate our model. First, we measure the frequencies of subtrees in very large GP populations and compare these with the expected frequencies calculated with our model. As expected, there are no differences between the expected and measured frequencies for all BBs and thus, the reliability of our model is good. Second, we use our model to estimate the GP population sizes for different desired degrees of sampling error. Then, we sample (random) GP populations of the respective estimated population sizes and compare the resulting empirical sampling errors with our predictions. We find that our model accurately estimates the sampling error. Our results indicate that the *estimated* sampling error calculated with expected frequencies of n-grams is a good proxy for the *empirical* variance of fitness values in our experiments. Last, we recommend minimum population sizes for benchmark problems that are often used in the literature to avoid problems with sampling error. We make our code publicly available in the form of a GP population size calculator, so that users of GP can calculate population sizes for other problem instances as well.[3]

In summary, our results indicate that sampling error is a severe problem for GP, making large initial population sizes necessary to obtain a low sampling error. Our model allows to estimate a minimum population size that is necessary to reduce sampling error to a given amount. Lowering the sampling error means that the overall random variations in a random sample are reduced. As a consequence, the reliability of the results of a GP run increases with larger population sizes.

In Sect. 2, we present a short summary of related work on population sizing and BB supply in GAs and GP. In Sect. 3 we describe initialization in GP with full, grow, and ramped half-and-half and introduce our model of the expected frequencies of n-grams of ancestors. Section 4 presents the model to estimate a GP population size, given a threshold of sampling error and a confidence level. We validate the proposed models in Sect. 5 with experimental results and recommend minimum population sizes for benchmark problems that are often used in the literature. The article ends with concluding remarks.

[2] BBs are defined by Goldberg (1989a, p. 41) as "short, low-order, and highly fit schemata". A schema is defined as a similarity template describing a subset of solutions within a population with similarities at certain positions of the genotype (Goldberg 1989a; Holland 1975; Goldberg 2002).

[3] The calculator can be found at https://gitlab.rlp.net/schweim/sampling-error-in-GP/.

2 Related work

We summarize previous work on population sizing and BB supply in GAs and GP. Some researchers (Goldberg et al. 1992, 2001; Goldberg and Segrest 1987) argue that a successful GA needs to ensure a sufficient supply of relevant BBs. An adequate supply of BBs can be ensured by

1. a high diversity in the BBs of the initial population (*spatial approach*) and/or by
2. generating BB diversity during runtime, e.g., by applying a suitable mutation operator (*temporal approach*; Goldberg et al. 2001).

A spatial approach for GAs was first proposed by Holland (1975). He discussed the issue of initial BB supply and proposed a model to estimate the expected number of observations of BBs in a population, given the size of the population. Later, Goldberg (1989b) improved Holland's model. Reeves (1993) developed a model to estimate the minimum population size needed to ensure the presence of at least one instance of every BB. However, Reeves' model only considered BBs with a length of one. Therefore, Goldberg et al. (2001) proposed a more general model that also considers larger BBs of a fixed size.

Other work analyzed the probability that high-quality BBs will take over the population in later generations (*BB growth*). E.g., Holland proposed to use the two-armed bandit problem to model the decision between competing BBs (Holland 1973, 1975). Later models also considered decision errors due to genetic drift (De Jong 1975) and variance of BB fitness (*collateral noise*; Goldberg and Rudnick 1991). The population sizing model presented in (Goldberg et al. 1992) permits the inclusion of other sources of decision errors to estimate population sizes that minimize these errors. Harik et al. (1999) proposed a model to predict the convergence quality of GAs based on the size of the population. They considered the initial supply of BBs as well as the selection of the best BB(s) over competing BBs.

In summary, previous research on GAs found that the initial supply of relevant BBs leads to improved BB growth (Goldberg et al. 2001; Reeves 1993). However, at the beginning of a search run, it is not known if a BB is relevant or not. Therefore, the authors argue that the initial GA population should be large enough to ensure that at least one copy of each BB is present in the initial population.

Following the GA literature, papers about population sizing in GP focus on BB supply. In the context of GP, BBs describe relationships between nodes in GP parse trees. BBs in GP were usually defined as subtrees of a GP parse tree (Poli and Langdon 1998; Poli 2001; Walsh and Ryan 1996; Koza 1992; O'Reilly and Oppacher 1994; Whigham 1995; Sastry et al. 2003, 2005; Hemberg et al. 2012). GP subtrees can be described by using *n-grams of ancestors* (Sastry et al. 2003, 2005; Hemberg et al. 2012). An *n*-gram of ancestors in a GP parse tree is the sequence of the values represented by a node i and its $n-1$ ancestor nodes on the same branch (parent, grandparent, greatgrandparent, etc.). Hemberg et al. (2012) found that *n*-grams of ancestors represent relevant relationships between nodes of a GP parse tree.

Sastry et al. (2003) proposed a model of the initial supply of BBs in GP based on *n*-grams of ancestors. The authors estimate the population size required to ensure—with a given error—the initial presence of at least one copy of all possible BBs in the initial GP population. Sastry et al. (2005) improved the population sizing model by incorporating decision-making errors among competing BBs in the population sizing model. However, both models only hold for simple test problems and assume full trees, binary functions, and knowledge about the size of the trees (i.e., a given tree size distribution). Therefore, the models are only generalizable to a limited extent (Sastry et al. 2003, 2005; Hemberg et al. 2012).

In the GP community, there is still a debate on the role of building blocks. Following GA literature, O'Reilly and Oppacher (1994) formulated a GP schema theorem and a GP building block hypothesis. In their analysis they focused on dynamic aspects of the search (i.e., *BB growth*). They concluded that due to many reasons the probability of the recombination of building blocks is difficult to predict. This is identified as a major obstacle in formulating a model to verify the GP building block hypothesis. Based on their results they question whether building blocks are relevant for GP.

The early GA as well as GP work on building blocks has been criticized due to the strong assumptions that it requires to work and the large simplifications that—at least initially—were made. As a consequence, more recent work (e.g., by Poli and McPhee 2003) proposed exact schema theorems for GP that improve on previous definitions of a schema. Since Poli and McPhee (2003) use a *temporal approach*, we do not discuss the details but refer the interested reader to the original article.

Recently, Burlacu et al. (2015, 2018a, 2018b) evaluated the BB hypothesis for GP empirically. They performed schema analyses on GP populations and identified schemata with an above-average quality as well as an increasing frequency in the populations over multiple generations. They found that GP is able to effectively evolve BBs at least for some problem instances.

3 Expected frequencies of *n*-grams of ancestors in initial GP populations

We present a model to calculate the expected frequencies of *n*-grams of ancestors in GP populations initialized by the GP initialization methods full, grow, and ramped half-and-half. First, we introduce the relevant notation and describe tree initialization in GP. Then, we develop a model of the expected number of nodes representing functions and terminals in a GP parse tree. Based on this, we model the expected frequencies of *n*-grams of ancestors.

3.1 Initialization of GP populations

We describe the GP initialization methods full, grow, and ramped half-and-half, using the following notation: The leaf nodes of GP parse trees are terminals t from a terminal set T ($t \in T$) and all remaining nodes ("inner nodes") are functions f from a function set F ($f \in F$) (Koza 1992; Poli et al. 2008). Let $a(f)$ be the arity (number of parameters) of $f\ \forall f \in F$. Then all nodes in a GP parse tree that represent a function f have $a(f)$ child nodes. Nodes representing a terminal do not have any child nodes.

In full, grow, and ramped half-and-half, the user specifies two hyperparameters that determine the depth of the generated trees, where the tree depth is defined as the length of the longest non-backtracking path from the root of the tree to any tree node (Koza 1992). The user specifies a minimum allowed tree depth $d_{\min} \geq 0$ and a set of allowed (maximum) tree depths D. When sampling a tree with full, grow, or ramped half-and-half, we first randomly sample a maximum tree depth $d_{\max}$ from D with uniform probability (Koza 1992; Poli et al. 2008; Fortin et al. 2012). Note that $d_{\max}$ defines a maximum tree depth for *one* particular tree that is sampled.

After $d_{\max}$ has been determined, a GP parse tree is sampled. In full, grow, and ramped half-and-half, sampling starts with the root node of the GP parse tree at depth 0. We first decide whether the root node represents a function $f \in F$ or a terminal $t \in T$. If the root node represents a function, the respective number of child nodes are created, depending on the arity of the selected function. Sampling continues by deciding for each child node if it represents a function $f \in F$ or a terminal $t \in T$. Afterwards, the appropriate number of child nodes are created. This process is repeated until no more decisions have to be made (all leaf nodes of the GP parse tree represent terminals).

The probability for a node to represent either a function or a terminal depends on the initialization method and the depth d of the node in the GP parse tree. Similar to the tree depth, the depth of a node is the length of the longest non-backtracking path from the root to the respective node.

The full method creates GP parse trees where all nodes with a depth $d < d_{\max}$ are only allowed to represent functions f, randomly chosen with uniform probability from F. Thus, all leaf nodes are sampled at depth $d_{\max}$ and represent terminals (Koza 1992).[4]

Grow creates GP parse trees where the leaf nodes can be sampled at different depths in the GP parse tree (Koza 1992). For each node with a depth $d < d_{\min}$, a function f is randomly chosen from F. After that, the nodes at depth $d_{\min} \leq d < d_{\max}$ are sampled and each of these nodes can represent either a function or a terminal. If a terminal is chosen, the sampling process stops for the respective branch. For each node at depth $d = d_{\max}$, a terminal t is randomly chosen from T. Therefore, a parse tree created with grow can have a depth that is less than or equal to $d_{\max}$ and greater than or equal to $d_{\min}$.

Ramped half-and-half is a combination of full and grow where half of the population is initialized with trees created using full and half is initialized with trees constructed using grow (Koza 1992).

3.2 Expected number of functions and terminals

We develop a model of the expected number of nodes representing a specific type of function or terminal in a GP parse tree. We begin by determining the probability that, during initialization, a node will be selected to represent a function. We assume the condition that either the parent node represents a function or that we sample the root node.

Let $d_{\min}$ and $d_{\max} \in D$ be the minimum and maximum tree depth for a tree to be sampled. Furthermore, let d be the depth of a node in the parse tree ($0 \leq d \leq d_{\max}$). Then, a function is always sampled for all nodes with a depth $d < d_{\max}$ in full and for all nodes with a depth $d < d_{\min}$ in grow. Thus, the probability to sample a function is 1. In grow, for all nodes where $d_{\min} \leq d < d_{\max}$, we sample a function or a terminal from $F \cup T$ with uniform probability. Therefore, the probability to sample a function in these cases is the number of functions $|F|$ divided by the overall number of functions and terminals $|F \cup T|$. In full and grow, we always sample a terminal when $d = d_{\max}$ with uniform probability from T. Thus, at depth $d_{\max}$, the probability to sample a function is 0. The probability to sample a function in F with the full method is

$$P^{\text{full}}(F) = \begin{cases} 1 & \text{if } 0 \leq d < d_{\max}, \\ 0 & \text{if } d = d_{\max} \end{cases} \tag{1}$$

and with the grow method it is

[4] Note that trees created by the full method should be described as perfect trees with mixed arities. However, in line with the GP literature, we will call trees created by the full method full trees.

$$P^{\text{grow}}(F) = \begin{cases} 1 & \text{if } 0 \le d < d_{\min}, \\ \dfrac{|F|}{|F \cup T|} & \text{if } d_{\min} \le d < d, \text{ and} \\ 0 & \text{if } d = d_{\max}. \end{cases} \tag{2}$$

In the following equations, we do not differentiate between $P^{\text{full}}(F)$ and $P^{\text{grow}}(F)$ since the equations are the same for both initialization methods. Therefore, we will use $P(F)$ to represent both possibilities.

Since all functions of the function set are sampled with uniform probability, the expected number of child nodes over all functions in F is the average function arity $\overline{a}$, which is defined as

$$\overline{a} = \frac{1}{|F|} \sum_{f \in F} a(f). \tag{3}$$

Luke (2000) showed that $E_{\text{nodes}}(d)$, the expected number of nodes at depth d in a parse tree, is

$$E_{\text{nodes}}(d) = \begin{cases} 1 & \text{if } d = 0, \\ E_{\text{nodes}}(d-1)P(F)\overline{a} & \text{if } 0 < d \le d_{\max}. \end{cases} \tag{4}$$

Thus, the number of nodes at a given depth $d > 0$ in the GP parse tree depends on

- $E_{\text{nodes}}(d-1)$, the number of nodes at depth $d-1$,
- $P(F)$, the conditional probability that these nodes represent functions, and
- $\overline{a}$, the expected number of child nodes of a node that represents a function in F.

Let $E_{\text{tree}}(d_{\max})$ be the expected size of a parse tree, given a maximum tree depth $d_{\max}$. Then, $E_{\text{tree}}(d_{\max})$ is the sum of the expected number of nodes $E_{\text{nodes}}(d)$ over all depths $0 \le d \le d_{\max}$ (Luke 2000):

$$E_{\text{tree}}(d_{\max}) = \sum_{d=0}^{d_{\max}} E_{\text{nodes}}(d). \tag{5}$$

Equation (5) calculates the expected tree size for one particular tree depth $d_{\max} \in D$ and therefore requires that $d_{\max}$ has already been determined. $d_{\max}$ is uniformly sampled from D and, therefore, the expected tree size over all depths in D is the average expected tree size over the possible (maximum) tree depths $d_{\max} \in D$

$$E_{\text{tree}}(D) = \frac{\sum_{d_{\max} \in D} E_{\text{tree}}(d_{\max})}{|D|}. \tag{6}$$

For all $s \in F$, let $E_s(d)$ be the expected number of nodes representing a function in a parse tree at depth d. Then, analogously to the expected tree size, $E_s(d)$ can be determined by first multiplying $E_{\text{nodes}}(d)$, the expected number of nodes at depth d, by the probability $P(F)$ that these nodes represent functions. Then, we divide by $|F|$ since all functions in F are sampled with uniform probabilities. Therefore,

$$E_s(d) = E_{\text{nodes}}(d)\frac{P(F)}{|F|} \quad \forall s \in F. \tag{7}$$

Since a node represents either a function or a terminal, the probability to sample a terminal from T for a node at a given depth d is

$$P(T) = 1 - P(F). \tag{8}$$

Terminals are sampled with uniform probability from T. Thus—analogously to Eq. (7)—given a maximum tree depth $d_{\max} \in D$, the expected number of nodes at depth d in a parse tree, representing a terminal $s \in T$, is

$$E_s(d) = E_{\text{nodes}}(d)\frac{P(T)}{|T|} \quad \forall\, s \in T. \tag{9}$$

We are interested in the expected number of nodes that represent s in a particular depth interval of a parse tree. Let $E_s(d_{\max}, l, u)$ be the expected number of nodes representing s in depths d in the parse tree where $l \le d \le u$. Then, $E_s(d_{\max}, l, u)$ is the sum of $E_s(d)$, the expected number of nodes representing s, over all depths $l \le d \le u$

$$E_s(d_{\max}, l, u) = \sum_{d=l}^{u} E_s(d) \quad \forall\, s \in F \cup T. \tag{10}$$

Then, given a set of possible (maximum) depths D, the expected number of nodes representing $s \in F \cup T$ in a parse tree is

$$E_s(D, l, u) = \frac{\sum_{d_{\max} \in D} E_s(d_{\max}, l, u)}{|D|} \quad \forall\, s \in F \cup T \tag{11}$$

because depths are uniformly sampled from D.

Let $E_s^{\text{node}}(D, l, u)$ be the expected frequency of a node to represent $s \in F \cup T$. Then $E_s^{\text{node}}(D, l, u)$ is the expected number of nodes representing s in a parse tree, $E_s(D, l, u)$, divided by the expected tree size $E_{\text{tree}}(D)$:

$$E_s^{\text{node}}(D, l, u) = \frac{E_s(D, l, u)}{E_{\text{tree}}(D)}. \tag{12}$$

In ramped half-and-half, 50% of the trees are created by the full method and 50% by the grow method (Koza 1992). Thus, the expected number of nodes representing functions and terminals in parse trees sampled with ramped half-and-half can be calculated by averaging the respective equations for full and grow. For example, let $E_s^{\text{full}}(D, l, u)$ and $E_s^{\text{grow}}(D, l, u)$ be the expected number of nodes representing $s \in F \cup T$, calculated by using $P^{\text{full}}(F)$ and $P^{\text{grow}}(F)$, respectively. Then, the expected number of nodes representing s in a parse tree created with the ramped half-and-half method is

$$E_s^{\mathrm{rhh}}(D,l,u) = \frac{1}{2}E_s^{\mathrm{full}}(D,l,u) + \frac{1}{2}E_s^{\mathrm{grow}}(D,l,u). \tag{13}$$

3.3 Expected frequency of *n*-grams of ancestors

n-grams of ancestors can be described as follows. Each node i of the GP parse tree represents a function f from the function set F ($f \in F$) or a terminal t from the terminal set T ($t \in T$). Let s_i be the function or terminal that is represented by an arbitrary node i in a parse tree. Furthermore, let the nodes $i-1, i-2, \ldots, i-(n-1)$ be the ancestors of the node i on the same branch in the parse tree and let $s_{i-1}, s_{i-2}, \ldots, s_{i-(n-1)}$ be the function values represented by the respective nodes. Then an n-gram of ancestors in a parse tree is the sequence of the node values s_i and the values of its $n-1$ ancestor nodes on the same branch (parent, grandparent, great-grandparent, etc.; Hemberg et al. 2012). Therefore, one specific n-gram of ancestors can be expressed using an ordered list such as $[s_i, s_{i-1}, \ldots, s_{i-(n-1)}]$. Root nodes do not have ancestor nodes and for these cases the values of (non-existent) ancestor nodes are defined as $s = \varnothing$. This is done to also represent root nodes as child nodes in some of the n-grams of ancestors since root nodes are usually very important for the semantics of a GP parse tree. The definition of n-grams of ancestors in a GP parse tree implies that $s_i \in F \cup T$ and $s_{i-1}, \ldots, s_{i-(n-1)} \in F \cup \{\varnothing\}$. Therefore an n-gram in a GP population can be expressed with an ordered list of the form

$$[s_1 \in F \cup T, s_2 \in F \cup \{\varnothing\}, \ldots, s_n \in F \cup \{\varnothing\}]. \tag{14}$$

All n-grams that are observed in a GP population together form a multiset.[5]

Figure 1a shows an example of a GP parse tree sampled by using the function set $F = \{+\}$ and the terminal set $T = \{x\}$. The respective n-grams of ancestors are visualized for $n = 1$ (Fig. 1b, c), for $n = 2$ (Fig. 1d–f), and for $n = 3$ (Fig. 1g–j). The n-grams of ancestors shown in Fig. 1b–j can also be expressed by using ordered lists. For example, the 3-grams of ancestors are $[+, \varnothing, \varnothing]$ (Fig. 1g), $[+, +, \varnothing]$ (Fig. 1h), $[x, +, \varnothing]$ (Fig. 1i), and $[x, +, +]$ (Fig. 1j). An n-gram of ancestors can occur several times and at different positions within one individual. For example, the 2-gram $[x, +]$ has a frequency of 3 in the exemplary parse tree.

The expected frequency over *all* n-grams of ancestors where the child node value is s_1 and the ancestor node values represent arbitrary functions in F is given by the expected frequency of s_1. The functions $s_2, \ldots, s_n$ are picked with uniform probabilities, but can have different arities. Nodes representing functions with higher arities have more child nodes and therefore the expected frequencies for n-grams where these functions are ancestor nodes is also higher. Thus, to calculate the expected frequency of an n-gram of ancestors $[s_1, s_2, \ldots, s_n]$ in a GP parse tree, we first determine the expected frequency of s_1 and weight this frequency by the arities of $s_2, \ldots, s_n$.

We will explain this idea on the example of 1-grams, 2-grams, and 3-grams. For the n-gram of ancestors $[s_1, s_2, \ldots, s_n]$, let $K_{[s_1,s_2,\ldots,s_n]}$ be the expected frequency of s_1 depending on the ancestor nodes $s_2, \ldots, s_n$. Furthermore, let $W_{[s_1,s_2,\ldots,s_n]}$ be a weighting factor, depending on the arities of $s_2, s_3, \ldots, s_n$. Then, the expected frequency of an n-gram $[s_1, s_2, \ldots, s_n]$ in a parse tree is

$$E_{[s_1,s_2,\ldots,s_n]} = K_{[s_1,s_2,\ldots,s_n]} W_{[s_1,s_2,\ldots,s_n]}. \tag{15}$$

For $n = 1$, $K_{[s_1]}$ is independent from any ancestor nodes and therefore can be calculated using Eq. (12):

$$K_{[s_1]} = E_{s_1}^{node}(D, 0, d_{max}). \tag{16}$$

1-grams do not take ancestor nodes into account and therefore, $W_{[s_1]} = 1$. Thus, we can calculate the expected frequency for any 1-gram in a GP tree by

$$E_{[s_1]} = K_{[s_1]} = E_{s_1}^{node}(D, 0, d_{max}). \tag{17}$$

A 2-gram of ancestors is the combination of the values represented by a node and its ancestor node. Since the root node of a GP parse tree has no ancestor nodes, we define the value of a (non-existent) ancestor node as $s_{i-1} = \varnothing$. The value of $K_{[s_1,s_2]}$ strongly depends on s_2. If $s_2 = \varnothing$, $K_{[s_1,s_2]}$ is the expected frequency of s_1 in the root node ($d = 0$). Otherwise s_1 has an ancestor node that represents a function $f \in F$ and so s_1 needs to be at a depth greater than 0 in the tree ($d \geq 1$). Thus, using Eq. (12), we define

$$K_{[s_1,s_2]} = \begin{cases} E_{s_1}^{node}(D, 1, d_{max}) & \text{if } s_2 \in F, \\ E_{s_1}^{node}(D, 0, 0) & \text{if } s_2 = \varnothing. \end{cases} \tag{18}$$

If s_2 is a function $f \in F$, we need to weight $K_{[s_1,s_2]}$, depending on the arity of s_2 compared to the arities of other functions in F. Functions with higher arities have more child nodes and therefore the expected frequencies for 2-grams where these functions are parent nodes is also higher. Let a_{sum} be the sum of all function arities of the functions in the function set

$$a_{\mathrm{sum}} = \sum_{f \in F} a(f). \tag{19}$$

Then $W_{[s_1,s_2]}$ is

[5] A multiset is a set that allows multiple instances for each of its elements.

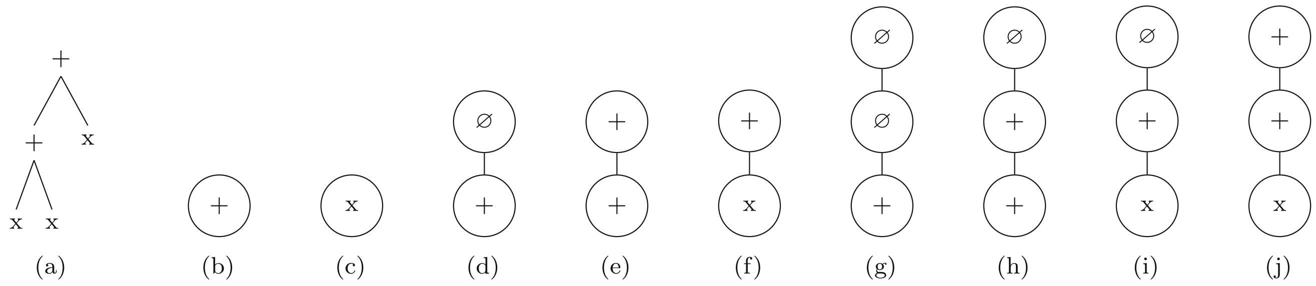


Fig. 1 Exemplary GP parse tree (**a**) with the corresponding n-grams of ancestors for $n = 1$ (**b**, **c**), $n = 2$ (**d–f**), and $n = 3$ (**g–j**)

$$W_{[s_1,s_2]} = \begin{cases} \dfrac{a(s_2)}{a_{\text{sum}}} & \text{if } s_2 \in F, \\ 1 & \text{if } s_2 = \emptyset. \end{cases} \tag{20}$$

For the frequencies of 3-grams of ancestors we use

$$K_{[s_1,s_2,s_3]} = \begin{cases} E^{node}_{s_1}(D, 2, d_{max}) & \text{if } s_2, s_3 \in F, \\ E^{node}_{s_1}(D, 1, 1) & \text{if } s_2 \in F, s_3 = \emptyset, \\ E^{node}_{s_1}(D, 0, 0) & \text{if } s_2 = \emptyset, s_3 = \emptyset, \end{cases} \tag{21}$$

and

$$W_{[s_1,s_2,s_3]} = \begin{cases} \dfrac{a(s_2)a(s_3)}{a_{\text{sum}}} & \text{if } s_2, s_3 \in F, \\ \dfrac{a(s_2)}{a_{\text{sum}}} & \text{if } s_2 \in F, s_3 = \emptyset, \\ 1 & \text{if } s_2 = \emptyset, s_3 = \emptyset. \end{cases} \tag{22}$$

Expected frequencies of n-grams of ancestors with $n > 3$ can be calculated analogously.

4 Estimating sampling error of *n*-grams of ancestors in GP populations

The Cochran formula is a standard method in statistics to estimate a minimum sample size N for a large statistical population. The Cochran formula needs an estimate of the relative frequency p of the property that is evaluated (e.g., the relative frequency of an n-gram of ancestors; Cochran 1977). In general, it is a problem to estimate p. In our case, we already know the expected relative frequency of n-grams of ancestors (see Sect. 3). However, we need to assume that p is normally distributed (Cochran 1977).

Furthermore, we need to choose an acceptable confidence level. For this, the Cochran formula uses z-scores of a normal distribution. For example, if a confidence level of 95% is chosen, the corresponding z-score is 1.96.

Last, we define a desired margin r of the relative statistical error e, so that $e \leq r$, where e is the absolute difference between the expected frequency p and the measured frequency p' relative to p

$$e = \frac{|p' - p|}{p}. \tag{23}$$

The Cochran formula (Cochran 1977) is

$$N = \frac{z^2(1-p)}{r^2 p}, \tag{24}$$

where p is the expected frequency, r is a margin of the relative error, and the confidence level is determined by a z-score.

Thus, if we take a sample of size N, the value of p will be in the interval

$$[p(1-r), p(1+r)] \tag{25}$$

with a probability equal to the confidence level. For example, we decide to use a confidence level of 95% ($z = 1.96$) and it is known that $p = 7\%$ of a statistical population have the respective property; the desired level of precision is $r = 10\%$. Then, using Eq. (24), we estimate $N = 5103.84$. As a result, if we take a random sample of size N, with a probability of 0.95 we measure p with $0.063 \leq p \leq 0.077$ ($P(0.063 \leq p \leq 0.077) = 0.95$).

The decision for a confidence level and a relative error is, to some extent, arbitrary (Cochran 1977). Values widely used in the literature and also recommended by Cochran (1977) are a confidence level of at least 95% ($z \geq 1.96$) and a relative error of not more than 5%. Estimated sample sizes calculated by using these values have a high precision and a high confidence. Given the expected frequency of an n-gram (Sect. 3.3) as the value for p, we can estimate the size of a GP population.

So far, we are only able to estimate the necessary GP population size for one n-gram. However, in a GP population, we typically expect a large number of different n-grams. Therefore, we have more than one statistical item, for which we need to estimate a proper sample size. For such a case, Cochran recommends to first identify the most important items and afterwards estimate the sample size separately for each of these items. Then, Cochran's pragmatic recommendation is to simply select the largest estimate for a sample size of any of the items (Cochran 1977).

5 Experiments

We empirically validate our model of expected n-gram frequencies and the model of the estimated sampling error in a GP population. Furthermore, we recommend population sizes for some widely used benchmark problem instances.

5.1 Frequencies of *n*-grams of ancestors

First, we validated the model of the expected frequencies of n-grams of ancestors in GP parse trees. We initialized five different large GP populations, each with a size of 100,000,000 individuals, measured the resulting frequencies for all n-grams of ancestors, and compared them with the expected frequencies calculated with our model.

The GP populations were initialized with ramped-half-and-half because it includes both, trees initialized with full and grow. The minimum tree depth was set to different values ($d_{\min} \in \{0, 1, 2, 3, 4\}$) for each of the five populations to take into account different scenarios. The set of allowed maximum tree depths used is $D = \{d_{\max} \in \mathbb{N}_0 | d_{\min} \leq d_{\max} \leq 4\}$. We used three different terminals and four different functions to be able to create a large variety of different trees. The function set included two functions with an arity of one, a binary function, and a ternary function. Since we do not evolve the initial population, it is not necessary to define a fitness function, variation operators, or a selection method.

We measured the error for n-gram frequencies with $n \in \{1, 2, 3\}$ in each of the five GP populations. The results are presented in Table 1. The table shows the mean and maximum relative error by $d_{\min}$ and n. The mean relative error is the mean over the relative errors for each n-gram frequency, measured separately for each value of n in the respective populations. Analogously, the maximum relative error was measured.

The values shown in Table 1 indicate that both, the mean and maximum relative error, are very small in all settings as expected. The error is larger for larger values of n. This is expected, since the population size is constant in all experiments, but there are much more different 3-grams than there are 1-grams. We used Pearson's chi-squared test to investigate the null hypothesis that the expected and measured frequencies of n-grams of ancestors are statistically *different* in their distributions. The p-values are very high ($p \gg 0.05$), strongly indicating that the expected and measured n-gram frequencies are **not** statistically different in their distributions. Therefore, our model is able to reliably estimate the expected frequencies for n-grams of ancestors in large GP populations.

Table 1 Mean and maximum relative error for different values of $d_{\min}$ and n; high p-values indicate that expected and measured n-gram frequencies are of the same statistical distribution

$d_{\min}$	n	Mean relative error	Maximum relative error	p value
0	1	6.0×10^{-5}	1.1×10^{-4}	0.9599
1	1	4.0×10^{-5}	9.0×10^{-5}	0.9087
2	1	4.0×10^{-5}	7.0×10^{-5}	0.9051
3	1	4.0×10^{-5}	7.0×10^{-5}	0.9594
4	1	6.0×10^{-5}	1.0×10^{-4}	0.8329
0	2	1.2×10^{-4}	5.0×10^{-4}	0.9999
1	2	1.0×10^{-4}	3.0×10^{-4}	0.9984
2	2	1.3×10^{-4}	5.0×10^{-4}	0.8941
3	2	1.5×10^{-4}	5.2×10^{-4}	0.9542
4	2	2.3×10^{-4}	7.6×10^{-4}	0.998
0	3	2.4×10^{-4}	1.4×10^{-3}	1
1	3	2.5×10^{-4}	1.4×10^{-3}	1
2	3	3.4×10^{-4}	1.4×10^{-3}	0.9844
3	3	4.5×10^{-4}	2.6×10^{-3}	0.9996
4	3	5.4×10^{-4}	2.3×10^{-3}	1

5.2 Sampling error

We estimate the GP population size for ramped half-and-half with $d_{min} = 2$, $D = \{2, 3, 4, 5, 6\}$, desired margins of sampling error $r \in \{0.01, 0.05, 0.1\}$, and a confidence level of 95% ($z = 1.96$) based on the expected frequencies of 1-grams, 2-grams, and 3-grams, respectively. We used the same function and terminal sets as in the first experiment (three different terminals as well as two unary, one binary, and a ternary function).

We calculated the expected n-gram frequencies of all n-grams and used these to estimate the respective expected population sizes. Note that we estimate the GP population size on the expected frequency of an n-gram per node (not per parse tree) using Eq. (12). The result is an estimate of the number of nodes. Since we want to obtain an estimate of the population size, we divide the estimate number of nodes by the expected tree size (which is the expected number of nodes in one tree). From the resulting list of estimated population sizes, we get three different values that take the most important n-grams into account with different degrees:

- The minimum estimated population size which corresponds to the estimate based on the highest expected frequency of all n-grams (*max*),
- the mean of the five lowest estimated population sizes which is based on the five most frequent n-grams (*top 5*),

- and the median over the population size estimates (*median*).

Overall, we estimated 27 different population sizes using the above settings. The used settings and the resulting population size estimates are presented in Table 2. We can see that the estimated population sizes for larger values of n are much higher compared to smaller values of n (with otherwise unchanged variables). This is expected since the number of n-grams grows exponentially and thus, the expected frequencies of these n-grams is lower, which then leads to higher estimated population sizes. The desired margin of error also has a high influence on the population size estimates: high values of r lead to lower population size estimates and vice versa.

Table 2 Estimated GP population sizes for 1-grams, 2-grams, and 3-grams with different margins of error $r \in \{0.01, 0.05, 0.1\}$

n	Method	r	Estimated population size
1	Max	0.1	107
1	Max	0.05	428
1	Max	0.01	10,691
1	Top 5	0.1	114
1	Top 5	0.05	453
1	Top 5	0.01	11,311
1	Median	0.1	123
1	Median	0.05	490
1	Median	0.01	12,241
2	Max	0.1	276
2	Max	0.05	1101
2	Max	0.01	27,508
2	Top 5	0.1	303
2	Top 5	0.05	1212
2	Top 5	0.01	30,297
2	Median	0.1	864
2	Median	0.05	3455
2	Median	0.01	86,368
3	Max	0.1	668
3	Max	0.05	2670
3	Max	0.01	66,748
3	Top 5	0.1	805
3	Top 5	0.05	3220
3	Top 5	0.01	80,482
3	Median	0.1	3080
3	Median	0.05	12,318
3	Median	0.01	307,940

max, top 5, and *median* indicate how many n-grams are taken into account in the population size estimate. Furthermore, we use $z = 1.96$, ramped half-and-half, $d_{\min} = 2$, and $D = \{2, 3, 4, 5, 6\}$

If we take only the most important n-gram frequencies into account (settings *max* and *top 5*), the estimated population sizes are lower compared to the setting *median*. This is because in *median* many n-grams with low frequencies are taken into account, resulting in large population size estimates. The difference between the settings can be large, e.g., in the case of 3-grams where the population size estimates with *median* are about 5 times as high as with *max*.

Next, we empirically analyzed the resulting error using the estimated population sizes from Table 2. For each estimate, we initialized 100 GP populations with the respective population size and measured the resulting relative sampling error by comparing the measured with the expected n-gram frequencies. In total, we initialized 2700 GP populations. The results are presented in Fig. 2.

Each of the 27 box plots visualizes the relative sampling errors that were measured in 100 GP populations. In Fig. 2, we differentiate between n-grams (columns), margin of error r (rows), and the number of n-gram frequencies taken into account when estimating the population size (each horizontal axis). The vertical axes show the corresponding

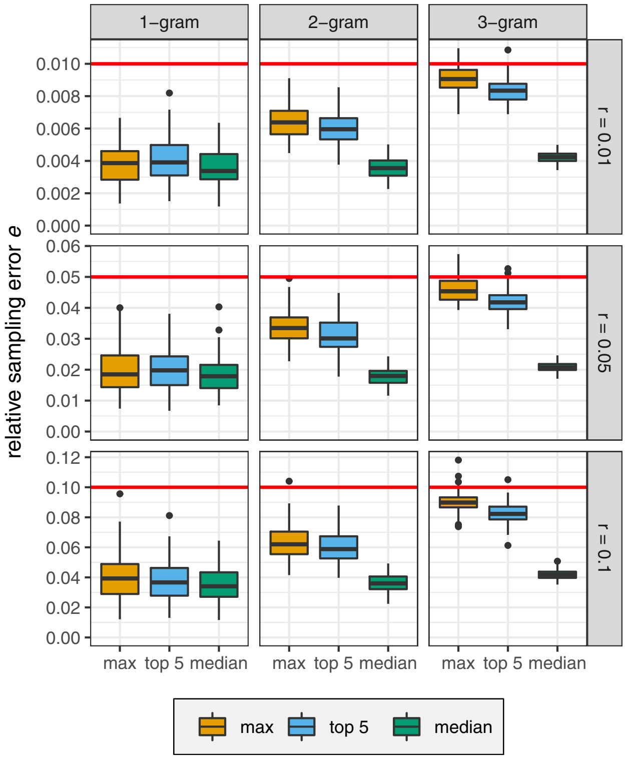


Fig. 2 Relative sampling error measured for 1-grams, 2-grams, and 3-grams with different margins of error ($r \in \{0.01, 0.05, 0.1\}$) indicated by a red horizontal line. *max, top 5*, and *median* indicate how many n-grams are taken into account in the population size estimate. Furthermore, we use $z = 1.96$, ramped half-and-half, $d_{\min} = 2$, and $D = \{2, 3, 4, 5, 6\}$

values of sampling error. The desired margin of error is also depicted by a red line in each plot.

As expected, we can see that the majority of values is below the margin of error. Interestingly, this is also the case if we only take into account the most important n-grams (settings *max* and *top 5*). For both of these cases, the estimated and measured sampling error are close to each other. When we estimate the GP population size using the median, we take more n-grams with a lower frequency into account. Therefore, we overestimate the GP population size. This leads to a sampling error that is well below the margin of error. In other words, it is only necessary to take into account the most important n-gram frequencies, which is in line with Cochran's general recommendation (Cochran 1977).

Our results show that the Cochran formula together with the results of the model of expected n-gram frequencies reliably estimate the GP population size for a desired margin of error.

5.3 Variance of fitness values

We analyze the variance of fitness values in generation 0 for different population sizes of $N \in \{10, 20, \ldots, 100, 200, \ldots, 1000, 2000, \ldots, 10{,}000, 15{,}000, \ldots, 30{,}000\}$ for four benchmark problem instances (McDermott et al. 2012)—6-Multiplexer, 11-Multiplexer, Koza-1, and Pagie-1. These four problem instances were chosen because their primitive sets are widely known in the community and interesting differences between the four primitive sets exist (e.g., different number of functions, different arities of the functions, different number of terminals). We use ramped half-and-half with $d_{\min} = 2$ and $D = \{2, 3, 4, 5, 6\}$. The results are presented in Figs. 3 and 4. For the 6-Multiplexer (Fig. 3a) and the 11-Multiplexer (Fig. 3b) we plot the median, 25-, and 75-quartile of the *mean* fitness in generation 0 over population sizes. Since there are infinite fitness values in the symbolic regression problem instances, we plot the median, 25-, and 75-quartile of the *median* fitness for Koza-1 (Fig. 4a) and Pagie-1 (Fig. 4b). The x-axes are log-scaled for better visibility of the results of small population sizes. We can see that the variance of mean and median fitness values is very high with small population sizes and asymptotically gets lower with higher population sizes.

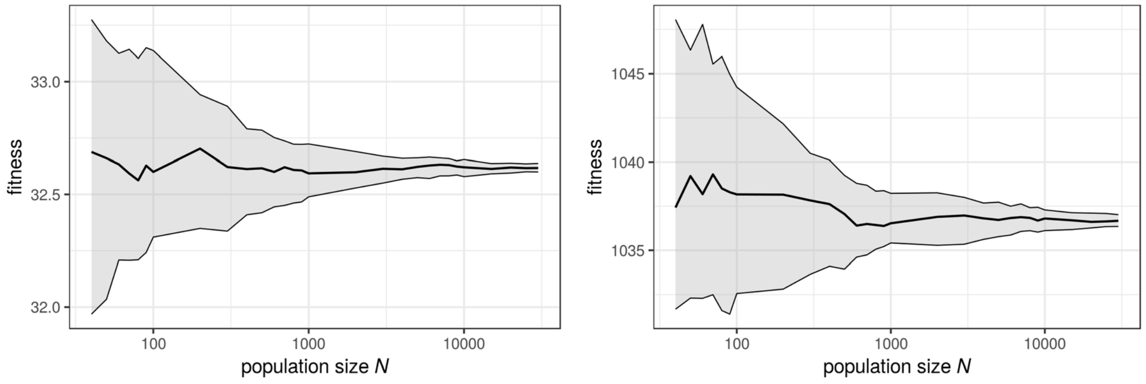
To further analyze the variance of fitness values over population sizes, we plot the quartile coefficient of dispersion (QCD) (Figs. 5, 6). The QCD is calculated using the first (Q_1) and third (Q_3) quartiles of the data set:

$$QCD = \frac{Q_3 - Q_1}{Q_3 + Q_1}. \tag{26}$$

High values of the QCD indicate that the data has large variance. Similar to the results in Figs. 3 and 4, we calculate the QCD of the mean fitness (Fig.5) and median fitness (Fig. 6). For comparison, we plot the estimated sampling error calculated by using the five highest expected frequencies of 1-, 2-, and 3-grams for each of the population sizes. To estimate the error we use the Cochran formula (Eq. 24) and transform it to

$$r = \sqrt{\frac{z^2(1-p)}{Np}}. \tag{27}$$

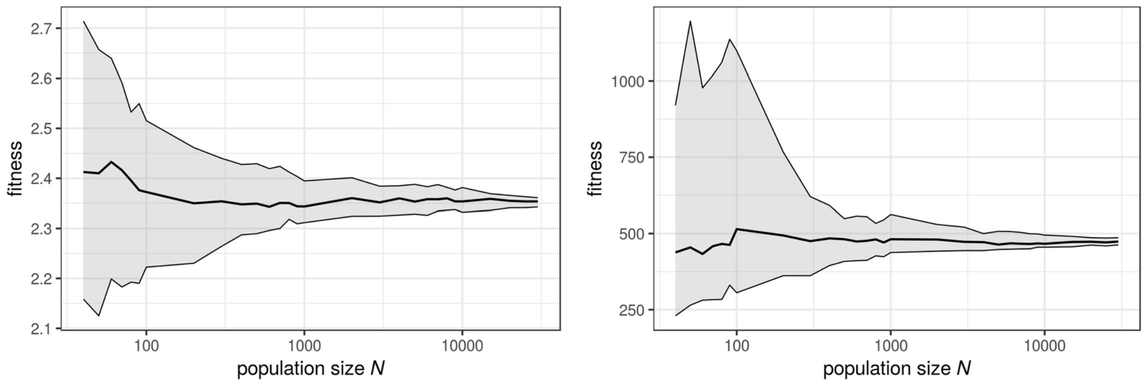
For a better comparison, we chose the scales in such a way that the QCD starts at about the same point as the estimated sampling error using expected 2-gram frequencies. In Figs. 5 and 6 we can see that the decrease of the QCD is analogous to the decrease of the estimated sampling error. Using Pearson's correlation coefficient, we measure the correlation between estimated sampling errors and QCD for estimates with 1-, 2-, and 3-grams on all four problem instances. The correlation coefficients are between 0.992 and 0.999, indicating a strong correlation. This means that the *estimated* sampling error calculated with expected



(a) 6-Multiplexer

(b) 11-Multiplexer

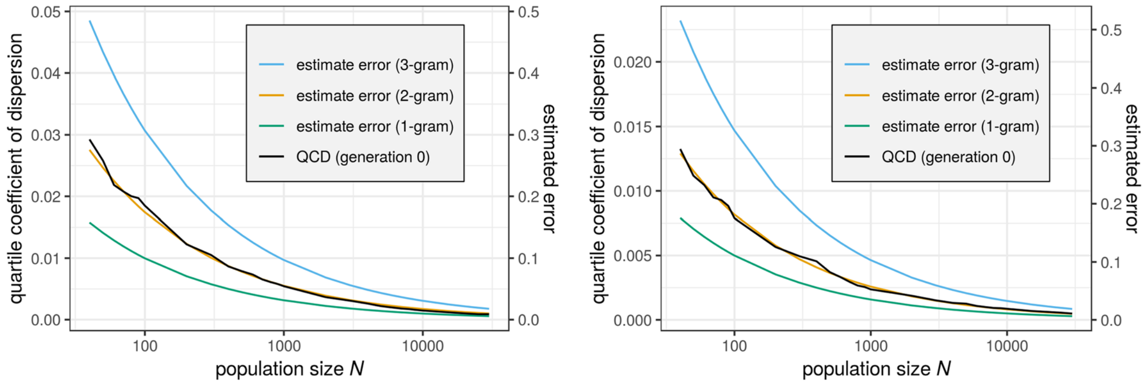
Fig. 3 Median, 25-, and 75-quartile of the mean fitness in generation 0 over population sizes



(a) Koza-1

(b) Pagie-1

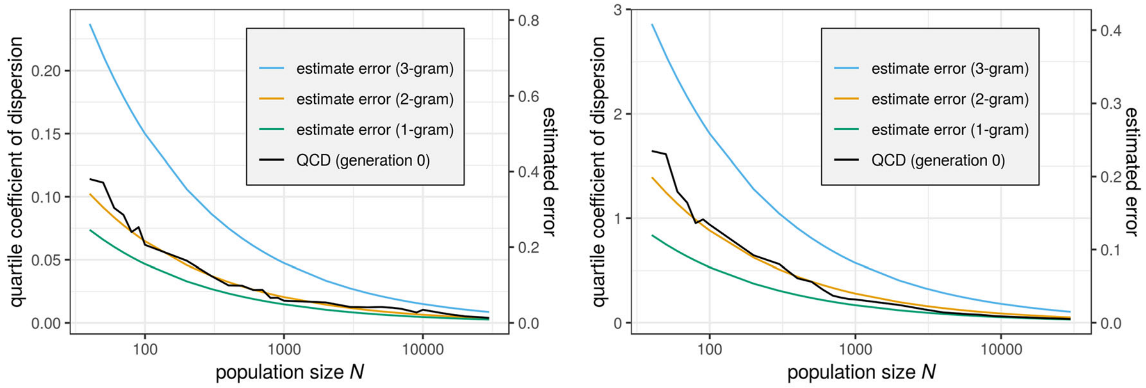
Fig. 4 Median, 25-, and 75-quartile of the median fitness in generation 0 over population sizes



(a) 6-Multiplexer

(b) 11-Multiplexer

Fig. 5 QCD of the mean fitness in generation 0 and estimated error of 1-, 2-, and 3-grams over population sizes



(a) Koza-1

(b) Pagie-1

Fig. 6 QCD of the median fitness in generation 0 and estimated error of 1-, 2-, and 3-grams over population sizes

frequencies of n-grams is a good proxy for the *empirical* variance of fitness values in our experiments. Thus, the variance of mean/median fitness values is a proper indicator of sampling error.

5.4 Estimated GP population sizes for common benchmark problem instances

We use our models to recommend reasonable population sizes for eight widely used benchmark problem instances (McDermott et al. 2012). In general, we used the function and terminal sets proposed by the authors of the benchmarks. However, in our analysis we ignored ephemeral random constants, e.g., used in (Pagie and Hogeweg 1997). Furthermore, in the model of expected frequencies, we used $d_{\min} = 2$ and $D = \{2, 3, 4, 5, 6\}$. In the sampling error model, we used $r = 0.05$, $z = 1.96$ (confidence level of 95%), and the estimates are based on the five highest expected 2- and 3-gram frequencies (*top 5*).

The results are presented in Table 3. The estimated population sizes using expected 2-gram frequencies are between 337 and 3440. Using expected 3-gram frequencies, the estimated population sizes are between 839 and 12,180.

Practitioners are usually faced with strict CPU time constraints. As a result, there is a trade-off between either choosing a larger population size or running the search for more generations. The population sizes indicated by our model help to make an informed decision of the population size. Increasing the population size beyond the indicated size would not help much. Instead, it would be better to increase the number of generations.

6 Conclusions

We developed a model of the expected frequencies of n-grams of ancestors in GP. We used the model of expected n-gram frequencies and Cochrans formula to determine a minimum size of an initial GP population, given a desired degree of sampling error and a confidence level. Then, we used our models to estimate initial GP population sizes for common benchmark problems, giving a recommendation to avoid sampling error. Furthermore, we find that the *estimated* sampling error calculated with expected frequencies of n-grams is a good proxy for the *empirical* variance of fitness values in our experiments. Last, we find for selected benchmark problems that the initial population sizes should be between 800 and 12,200, depending on the problem instance to reduce the amount of sampling error below 5%.

Our results show that GP and variants like EDA-GP benefit from high population sizes to avoid problems with sampling error. However, our model does not consider that—in addition to BBs being present—these BBs have an effect on the fitness (i.e., by definition introns do not influence fitness; Sastry et al. 2003).

Furthermore, our analysis focuses on subtree frequencies, where subtrees are represented by n-grams of ancestors. Other forms of n-grams could be interesting as well, e.g., n-grams that use sibling nodes (Hemberg et al. 2012). Also, it could be relevant to analyze other population statistics to evaluate whether our initial population is sufficiently representative (i.e., has a low sampling error). Examples are the distribution of tree depths or the distribution of tree shapes.

Of course, GP search is not only influenced by the initial population but also by other factors. Therefore, exploring a combination of our initialization model and an adaptive population size approach, e.g., the one presented by Hu and Banzhaf (2009), is promising.

We cannot guarantee a certain solution quality with our model as competing BBs or expressions are not considered. Thus, future studies need to extend our models, taking variation and selection into account (temporal models).

Table 3 Estimated GP population sizes for common benchmark problem instances ($r = 0.05$, $z = 1.96$, ramped half-and-half, estimates are based on the five highest expected 2- and 3-gram frequencies)

Benchmark name	Number of functions	Number of terminals	Estimated population size ($n = 2$)	Estimated population size ($n = 3$)
Koza-1 (Koza 1992)	4 unary, 4 binary	1	1911	9980
Nguyen-1–Nguyen-10 (Uy et al. 2011)	4 unary, 4 binary	2	3440	12,180
Pagie-1 (Pagie and Hogeweg 1997)	4 binary	2	636	2672
Keijzer-1–Keijzer-15 (Keijzer 2003)	3 unary, 2 binary	2	3129	7536
6-Multiplexer (Koza 1992)	1 unary, 2 binary, 1 ternary	6	1219	3771
11-Multiplexer (Koza 1992)	1 unary, 2 binary, 1 ternary	11	1327	4289
Intertwined Spirals (Koza 1992)	1 unary, 4 binary, 1 quaternary	2	527	1779
Artificial Ant (Koza 1992)	2 binary, 1 ternary	3	337	839

Acknowledgements Parts of this research were conducted using the supercomputer Mogon offered by Johannes Gutenberg University Mainz (https://hpc.uni-mainz.de). The authors gratefully acknowledge the computing time granted on Mogon. Also, we thank the anonymous reviewers whose suggestions helped us to improve and clarify this manuscript.

Funding Open Access funding enabled and organized by Projekt DEAL.

References

Burlacu B, Kommenda M, Affenzeller M (2015) Building blocks identification based on subtree sample counts for genetic programming. In: Proceedings of the 2015 Asia-Pacific conference on computer aided system engineering, IEEE Computer Society, APCASE '15, pp 152–157

Burlacu B, Affenzeller M, Kommenda M, Kronberger G, Winkler S (2018a) Analysis of schema frequencies in genetic programming. In: Moreno-Díaz R, Pichler F, Quesada-Arencibia A (eds) Computer aided systems theory—EUROCAST 2017. Springer, Cham, pp 432–438

Burlacu B, Affenzeller M, Kommenda M, Kronberger G, Winkler S (2018b) Schema analysis in tree-based genetic programming. In: Banzhaf W, Olson RS, Tozier W, Riolo R (eds) Genetic programming theory and practice XV. Springer, Cham, pp 17–37

Cochran WG (1977) Sampling techniques, 3rd edn. Wiley, New York

De Jong KA (1975) An analysis of the behavior of a class of genetic adaptive systems. Doctoral dissertation, University of Michigan, Ann Arbor, MI

Fortin FA, De Rainville FM, Gardner MA, Parizeau M, Gagné C (2012) DEAP: evolutionary algorithms made easy. J Mach Learn Res 13:2171–2175

Goldberg DE (1989a) Genetic algorithms in search, optimization, and machine learning. Addison-Wesley Publishing Company Inc, Boston

Goldberg DE (1989b) Sizing populations for serial and parallel genetic algorithms. In: Schaffer J (ed) Proceedings of the 3rd international conference on genetic algorithms. Morgan Kaufmann Publishers Inc., San Francisco, pp 70–79

Goldberg DE (2002) The design of innovation: lessons from and for competent genetic algorithms, genetic algorithms and evolutionary computation, vol 7. Springer, Boston. https://doi.org/10.1007/978-1-4757-3643-4

Goldberg DE, Rudnick M (1991) Genetic algorithms and the variance of fitness. Complex Syst 5(3):265–278

Goldberg DE, Segrest P (1987) Finite Markov chain analysis of genetic algorithms. In: Proceedings of the second international conference on genetic algorithms and their application. L. Erlbaum Associates Inc., Hillsdale, pp 1–8. http://dl.acm.org/citation.cfm?id=42512.42513

Goldberg DE, Deb K, Clark JH (1992) Genetic algorithms, noise, and the sizing of populations. Complex Syst 6(4):333–362

Goldberg DE, Sastry K, Latoza T (2001) On the supply of building blocks. In: Spector L, Goodman ED, Wu A, Langdon WB, Voigt HM, Gen M, Sen S, Dorigo M, Pezeshk S, Garzon MH, Burke E (eds) Proceedings of the genetic and evolutionary computation conference 2001. Morgan Kaufmann Publishers, San Francisco, pp 336–342

Harik G, Cantú-Paz E, Goldberg DE, Miller BL (1999) The Gambler's ruin problem, genetic algorithms, and the sizing of populations. Evol Comput 7(3):231–253

Hemberg E, Veeramachaneni K, McDermott J, Berzan C, O'Reilly UM (2012) An investigation of local patterns for estimation of distribution genetic programming. In: Proceedings of the 14th annual conference on genetic and evolutionary computation (GECCO '12). ACM, New York, pp 767–774

Holland JH (1973) Genetic algorithms and the optimal allocation of trials. SIAM J Comput 2(2):88–105

Holland JH (1975) Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor

Hu T, Banzhaf W (2009) The role of population size in rate of evolution in genetic programming. In: Vanneschi L, Gustafson S, Moraglio A, De Falco I, Ebner M (eds) Proceedings of the 12th European conference on genetic programming (EuroGP 2009), LNCS, vol 5481. Springer, Berlin, pp 85–96

Keijzer M (2003) Improving symbolic regression with interval arithmetic and linear scaling. In: European conference on genetic programming. Springer, Berlin, pp 70–82

Kim K, Shan Y, Nguyen XH, McKay RIB (2014) Probabilistic model building in genetic programming: a critical review. Genet Program Evol Mach 15(2):115–167

Koza JR (1992) Genetic programming: on the programming of computers by means of natural selection. MIT Press, Cambridge

Lee CF, Lee JC, Lee AC (2013) Statistics for business and financial economics, 3rd edn. Springer, New York

Luke S (2000) Two fast tree-creation algorithms for genetic programming. IEEE Trans Evol Comput 4(3):274–283

McDermott J, White D, Luke S, Manzoni L, Castelli M, Vanneschi L, Jaśkowski W, Krawiec K, Harper R, De Jong K, O'Reilly UM (2012) Genetic programming needs better benchmarks. In: GECCO'12—proceedings of the 14th international conference on genetic and evolutionary computation, pp 791–798

O'Reilly UM, Oppacher F (1994) The troubling aspects of a building block hypothesis for genetic programming. In: Whitley LD (ed) Foundations of genetic algorithms, vol 3. Morgan Kaufmann, Estes Park, pp 73–88

Pagie L, Hogeweg P (1997) Evolutionary consequences of coevolving targets. Evol Comput 5(4):401–418

Poli R (2001) Exact schema theory for genetic programming and variable-length genetic algorithms with one-point crossover. Genet Program Evol Mach 2(2):123–163

Poli R, Langdon WB (1998) Schema theory for genetic programming with one-point crossover and point mutation. Evol Comput 6(3):231–252

Poli R, McPhee NF (2003) General schema theory for genetic programming with subtree-swapping crossover: part II. Evol Comput 11(2):169–206

Poli R, Langdon WB, McPhee NF (2008) A field guide to genetic programming. Lulu Enterprises, http://www.gp-field-guide.org.uk

Reeves CR (1993) Using genetic algorithms with small populations. In: Proceedings of the 5th international conference on genetic algorithms. Morgan Kaufmann Publishers Inc., San Francisco, pp 92–99

Rothlauf F (2011) Design of modern heuristics: principles and application. Natural computing series. Springer, Heidelberg

Särndal CE, Swensson B, Wretman J (1992) Model assisted survey sampling. Springer series in statistics. Springer, New York

Sastry K, O'Reilly UM, Goldberg DE, Hill D (2003) Building-block supply in genetic programming. Genetic programming theory and practice. Springer, Boston, pp 137–154

Sastry K, O'Reilly UM, Goldberg DE (2005) Population sizing for genetic programming based on decision-making. In: Genetic programming theory and practice II. Springer, New York, pp 49–65. https://doi.org/10.1007/0-387-23254-0_4

Shan Y, McKay RIB, Essam D, Abbass H (2006) A survey of probabilistic model building genetic programming. Scal Optim Probab Model 160:121–160. https://doi.org/10.1007/978-3-540-34954-9_6

Uy NQ, Hoai NX, O'Neill M, McKay RI, Galván-López E (2011) Semantically-based crossover in genetic programming: application to real-valued symbolic regression. Genet Program Evol Mach 12(2):91–119. https://doi.org/10.1007/s10710-010-9121-2

Walsh P, Ryan C (1996) Paragen: a novel technique for the autoparallelisation of sequential programs using GP. In: Proceedings of the 1st annual conference on genetic programming. MIT Press, Cambridge, pp 406–409

Whigham PA (1995) A schema theorem for context-free grammars. In: IEEE conference on evolutionary computation, vol 1. IEEE Press, Perth, pp 178–181

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.