ϵ-Lexicase selection: a probabilistic and multi-objective analysis of lexicase selection in continuous domains

March 16, 2017

Abstract

Lexicase selection is a parent selection method that considers test cases separately, rather than in aggregate, when performing parent selection. As opposed to previous work that has demonstrated the ability of lexicase selection to solve difficult problems, the goal of this paper is to develop the theoretical underpinnings that explain its performance. To this end, we derive an analytical formula that gives the expected probabilities of selection under lexicase selection, given a population and its behavior. In addition, we expand upon the relation of lexicase selection to many-objective optimization methods to show the effect of lexicase, which is to select individuals on the boundaries of Pareto fronts in high-dimensional space. We show analytically why lexicase selection performs more poorly for certain sizes of population and training cases, and why it has been shown to perform more poorly in continuous error spaces. To addres this last concern, we introduce ϵ-lexicase selection, which modifies the pass condition defined in lexicase selection to allow near-elite individuals to pass cases, thereby improving selection performance. We show that ϵ-lexicase outperforms several diversity-maintenance strategies for problems from three continuous-valued domains: regression, dynamical systems, and program synthesis.

1 Introduction

Evolutionary computation (EC) traditionally assigns scalar fitness values to candidate solutions to determine how to guide search. In the case of genetic programming (GP), this fitness value attempts to capture how closely, on average, the behavior of the candidate programs match the desired behavior. Take for example the task of symbolic regression, in which we attempt to find a model using a set of training examples, i.e. cases or tests. A typical fitness measure is the mean squared error (MSE), which averages the squared differences between the model's outputs and the target output. Given that individual comparisons of each program's output to y is reduced to a single value, the relationship of $\hat{y}$ to y can only be represented crudely by the fitness value. The fitness score thereby restricts the information conveyed to the search process about candidate programs relative to the description of their behavior available in the raw comparisons of the output to the target (i.e., the squared error), information which could help guide the search [9]. This observation has led to increased interest in the development of methods that can leverage the program outputs, i.e. semantics, directly to drive search more effecitvely [16].

In addition to reducing information, averaging test performance assumes all tests are equally informative, leading to the potential loss of individuals who perform poorly *on average* even if they are the best on a test case that is difficult for most of the population to solve. This is particularly relevant for problems that require different modes of behavior to produce an adequate solution to the problem [15]. The underlying assumption of GP in this regard is that selection pressure should be applied evenly with respect to test cases, ignoring the fact that cases that comprise the problem are unlikely to be uniformly difficult. As a result, the search is likely to benefit if it can take into account the difficulty of specific cases by recognizing individuals that perform well on harder parts of the problem. Underlying this last point is the assumption that GP solves problems by identifying, propagating and recombining partial solutions (i.e. building blocks) to the task at hand(cite schema Poli). As a result, a program that performs well on unique subsets of the problem may imply a partial solution to our task.

Several methods have been proposed to reward individuals with uniquely good test performance, such as implicit fitness sharing (IFS) [14], historically assessed hardness [7], and co-solvability [8], all of which assign greater weight to fitness cases that are judged to be more difficult in view of the population performance. Perhaps the most effective parent selection method designed to account for case hardnedss is lexicase selection [6, 15]. In particular, "global pool, uniform random sequence, elitist lexicase selection" [15], which we refer to simply as lexicase selection, has outperformed other similarly-motivated methods in recent studies [5, 13]. Despite these gains, it fails to produce such benefits when applied to continuous symbolic regression problems, due to its method of selecting individuals based on test case elitism. For this reason we recently proposed [11] modulating the case pass conditions in lexicase selection using an automatically defined ϵ threshold, by allows the benefits of lexicase selection to be achieved in continuous domains.

To date, the work to analyze lexicase selection and ϵ-lexicase selection has mostly been empirical studies, rather than algorithmic analysis. Previous work has not explicitly described the probabilities of selection under lexicase selection or how lexicase selection relates to multi-objective methods. The foremost purpose of this paper is to lay the groundwork for describing how lexicase selection and ϵ-lexicase selection pressure selection compared to traditional approaches. With this in mind, in §5 we derive an equation that describes the expected probability of selection for individuals in a given population based on their behavior on the training cases, for all variants of lexicase selection. Then in §6, we analyze lexicase and ϵ lexicase selection from a multi-objective viewpoint, in which we imagine each training case to be an objective. We prove that individuals selected by lexicase selection exist at the boundaries of the Pareto front in the high-dimensional space defined by the program error vectors. We show via an illustrative example population in §7 how the probabilities of selection differ under tournament, lexicase, and ϵ-lexicase selection.

We propose two new methods for defining pass conditions in ϵ-lexicase which are shown to improve the method compared to the original implementation. A final set of experiments compares variants of ϵ-lexicase selection to several existing selection techniques on a set of real world benchmark problems. The results show that ability of ϵ-lexicase selection to improve the predictive accuracy of models on these problems. We examine in detail the diversity of programs during these runs, as well as the number of cases used in selection events to validate our hypothesis that ϵ-lexicase selection allows for more cases to be used in selecting individuals compared to lexicase selection.

2 Preliminaries

In symbolic regression, we attempt to find a model $\hat{y}(\mathbf{x}) : \mathbb{R}^d \rightarrow \mathbb{R}$ using a set of training examples $\mathcal{T} = \{t_i = (y_i, \mathbf{x}_i)\}_{i=1}^{T}$, where $\mathbf{x}$ is a d-dimensional vector of variables i.e. features, and y is the desired output. A typical fitness measure is the mean squared error (MSE), which averages the squared differences between $\hat{y}$ and y.

We denote the population of N programs as $\mathcal{N} = \{n_i\}_{i=1}^{N}$ and the paired examples on which they learn as $\mathcal{T} = \{t_i = (y_i, \mathbf{x}_i)\}_{i=1}^{T}$. We refer to elements of $\mathcal{T}$ as "cases". A program $n \in \mathcal{N}$ produces an approximation $\hat{y}_t(n, \mathbf{x}_t) : \mathbb{R}^d \rightarrow \mathbb{R}$ when evaluated on case t. We refer to the squared error of this approximation as $e_t(n) = (y_t - \hat{y}_t(n))^2$, and denote the errors produced by a program on all of the training cases as the program's semantics. We use $\mathbf{e_t} \in \mathbb{R}^N$ to refer to the set of all errors in the population on training case t.

$\mathcal{T} = \{(y_t, \mathbf{x}_t)\}_{t=1}^{N}$, using e.g. the mean absolute error (MAE), which is quantified for individual program $i \in P$ as:

$$MSE(\mathcal{T}) = \frac{1}{N} \sum_{t \in \mathcal{T}} |y_t - \hat{y}_t(\mathbf{x}_t)| \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^D$ represents the variables or features, the target output is y and $\hat{y}(i, \mathbf{x})$ is the program's output. As a result of the aggregation of the absolute error vector $e(i) = |y - \hat{y}(i, \mathbf{x})|$ in Eq. (1), the relationship of $\hat{y}$ to y is represented crudely when choosing models to propagate.

3 Lexicase Selection

Lexicase selection is a parent selection technique based on lexicographic ordering of test (i.e. fitness) cases.

The lexicase selection algorithm for a single selection event is presented below:

Algorithm 2.1: Lexicase Selection

```
GetParent(𝒩, 𝒯) :
  T′ ← 𝒯                                  training cases
  S ← 𝒩                                   initial selection pool is the population
  while |T′| > 0 and |𝒮| > 1:              main loop
    case ← random choice from 𝒯′              consider a random case
    elite ← best fitness in 𝒮 on case         determine elite fitness
    𝒮 ← n ∈ 𝒮 if fitness(n) = elite          reduce selection pool to elites
    𝒯′ ← 𝒯′− case                             reduce remaining cases
  return random choice from 𝒮             return parent
```

Algorithm 2.1 is very simple to implement because it consists of just a few steps: 1) choosing a case, 2) filtering the selection pool based on that case, and 3) repeating until the cases are exhausted or the selection pool is reduced to one individual. If the selection pool is not reduced by the time each case has been considered, an individual is chosen randomly from the remaining pool, $\mathcal{S}$.

Under lexicase selection, cases in $\mathcal{T}$ can be thought of as filters that reduce the selection pool to the individuals in the pool that are best on that case. It's important to note that each parent selection event constructs a new path through these filters. The filtering strength of the case is affected by two main factors: 1) its difficulty as defined by the number of individuals that pass it, and 2) its order in the selection event, which varies from selection to selection. Regarding 1), consider two extreme examples: if a case is passed by the whole population, then it will perform no filtering, resulting in no selection pressure; if a case is passed by a single individual, that individual will be selected every time the case is considered for a selection pool containing the individual that passes it. This mechanism allows selective pressure to continually shift to individuals that are elite on cases that are not widely solved in $\mathcal{N}$. Because of 2), cases appear in various orderings during selection, resulting in selective pressure for individuals that solve difficult *subsets* of cases. Lexicase selection thereby accounts not only for the difficulty of individual cases but the difficulty of solving arbitrarily-sized subsets of cases. This selection pressure leads to the preservation of high behavioral diversity during evolution [6, 11].

The worst-case complexity of selecting N parents per generation is $O(TN^2)$, which occurs only if every parent passes or fails every case in $\mathcal{T}$. Normally, due to differential performance across the population and due to lexicase selection's tendency to promote diversity, the overall case depth per selection event is much less than T, and the selection poool typically winnows below N as well, reducing the runtime complexity[6].

We use an example population originally presented in [15] to illustrate some aspects of standard lexicase selection in the following sections. The population consists of five individuals and four test cases with discrete errors. An illustrative example of the filtering mechanism of selection is presented for this example in Figure 1.

An illustrative example of lexicase selection from the original paper is presented in Table 1 three iterations of lexicase selection is presented for an example population in Figure 1.

Table 1: Example population from original lexicase paper (Spector 2013).

Program	Test				$\mathcal{K}(\mathcal{T})$	MAE	P_{sel}	P_t
	t_1	t_2	t_3	t_4				
n_1	2	2	4	2	$\{t_2, t_4\}$	2.5	0.25	0.28
n_2	1	2	4	3	$\{t_2\}$	2.5	0.00	0.28
n_3	2	2	3	4	$\{t_2, t_3\}$	2.75	0.33	0.12
n_4	0	2	5	5	$\{t_1, t_2\}$	3.0	0.208	0.04
n_5	0	3	5	2	$\{t_1, t_4\}$	2.5	0.208	0.28

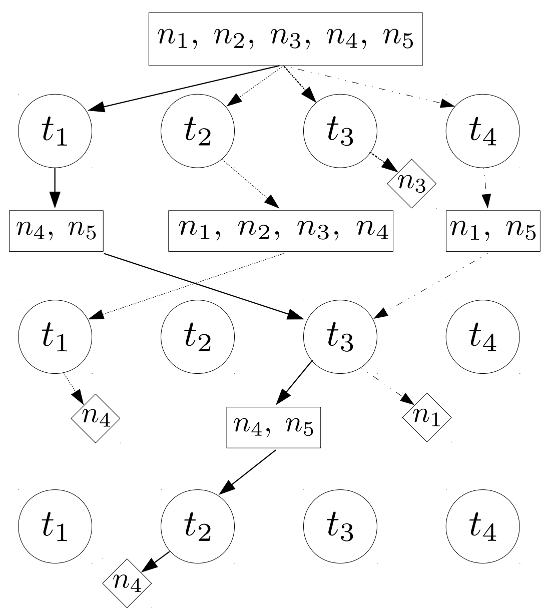

Figure 1: A graphical representation of four example parent selections using lexicase selection on the population in Table 1. The bold, dashed, bold-dashed and dot-dashed lines indicate different selection paths through the test cases in circles. The boxes indicate the selection pool at each step in the process. The diamonds show the individual selected by each selection event.

4 ϵ-Lexicase Selection

Lexicase selection has been shown to be effective in discrete error spaces, both for multi-modal problems [15] and for problems for which every case must be solved exactly to be considered a solution [6]. In continuous error spaces, however, the requirement for individuals to be *exactly* equal to the elite error on in the selection pool to pass a case during selection turns out to be overly stringent [11]. In continuous error spaces and especially for the symbolic regression task applied to noisy datasets, it is unlikely for two individuals to have exactly the same error on any training case unless they are equivalent models. As a result, lexicase selection in symbolic regression is prone to conducting selection based on single cases, where the selected individual is one satisfying $e_t \equiv e_t^*$, where e_t^* is the best error on t among $\mathcal{N}$. This limits lexicase's ability to leverage case information effecively, and can lead to poorer performance than traditional selection methods [11].

These observations led to the development of ϵ-lexicase selection [11], which modifies lexicase selection by modulating case filtering conditions using a ϵ threshold criteria.

$$\epsilon_t = \text{median}(\mathbf{e}_t - \text{median}(\mathbf{e}_t)) \tag{2}$$

We study three implementations of ϵ-lexicase selection in this paper: static, which is the version used in the conference paper [10], semi-dynamic, in which the error is defined relative to the pool, and dynamic, in which both the error and ϵ are defined relative to the current selection pool.

Static ϵ-lexicase selection can be viewed as a preprocessing step added to lexicase selection in which the program errors are convereted to pass/fail based on an ϵ threshold defined relative to e_t^*, the best error in $\mathcal{N}$ on t. It is defined below:

Algorithm 3.1: Static ϵ-Lexicase Selection

`GetParent`$(\mathcal{N}, \mathcal{T})$:	
$T' \leftarrow \mathcal{T}$	training cases
$S \leftarrow \mathcal{N}$	initial selection pool is the population
$\epsilon \leftarrow MAD(\text{fitness}(\mathcal{N}))$ for $t \in \mathcal{T}$	get ϵ for each case
$\text{fitness}(n) \leftarrow \mathbb{I}(e_t(n) \leq e_t^*(n) + \epsilon_t)$ for $t \in \mathcal{T}$ and $n \in \mathcal{N}$	convert fitness using within-ϵ pass condition
while $\lvert T' \rvert > 0$ and $\lvert \mathcal{S} \rvert > 1$:	main loop
`case` $\leftarrow$ random choice from $\mathcal{T}'$	consider a random case
`elite` $\leftarrow$ best fitness in $\mathcal{S}$ on `case`	determine elite fitness
$\mathcal{S} \leftarrow n \in \mathcal{S}$ if $\text{fitness}(n) \leq$ `elite`	reduce selection pool to elites
$\mathcal{T}' \leftarrow \mathcal{T}' -$ `case`	reduce remaining cases
return random choice from $\mathcal{S}$	return parent

Since calculating ϵ is $O(TN)$, static ϵ-lexicase shares a worst-case complexity with lexicase selection of $O(TN^2)$. Semi-dynamic ϵ-lexicase selection differs from static ϵ-lexicase selection in that the pass condition is defined relative to the best error *among the pool* rather than among $\mathcal{N}$. It is defined below:

Algorithm 3.2: Semi-dynamic ϵ-Lexicase Selection

`GetParent`$(\mathcal{N}, \mathcal{T})$:	
$T' \leftarrow \mathcal{T}$	training cases
$S \leftarrow \mathcal{N}$	initial selection pool is the population
$\epsilon \leftarrow MAD(\text{fitness}(\mathcal{N}))$ for $t \in \mathcal{T}$	get ϵ for each case
while $\lvert T' \rvert > 0$ and $\lvert \mathcal{S} \rvert > 1$:	main loop
`case` $\leftarrow$ random choice from $\mathcal{T}'$	consider a random case
`elite` $\leftarrow$ best fitness in $\mathcal{S}$ on `case`	determine elite fitness
$\mathcal{S} \leftarrow n \in \mathcal{S}$ if $\text{fitness}(n) \leq$ `elite`$+\epsilon_{\texttt{case}}$	reduce selection pool to elites
$\mathcal{T}' \leftarrow \mathcal{T}' -$ `case`	reduce remaining cases
return random choice from $\mathcal{S}$	return parent

The final variant of ϵ-lexicase selection is Dynamic ϵ-lexicase selection, in which both the error and ϵ are defined among the current selection pool. It is presented below:

Algorithm 3.3: Dynamic ϵ-Lexicase Selection

`GetParent`$(\mathcal{N}, \mathcal{T})$:	
$T' \leftarrow \mathcal{T}$	training cases
$S \leftarrow \mathcal{N}$	initial selection pool is the population
while $\lvert T' \rvert > 0$ and $\lvert \mathcal{S} \rvert > 1$:	main loop
`case` $\leftarrow$ random choice from $\mathcal{T}'$	consider a random case
`elite` $\leftarrow$ best fitness in $\mathcal{S}$ on `case`	determine elite fitness
$\epsilon \leftarrow MAD(\text{fitness}(\mathcal{S}))$ on `case`	determine ϵ for this case
$\mathcal{S} \leftarrow n \in \mathcal{S}$ if fitness$(n) \leq$ `elite`$+\epsilon_{\texttt{case}}$	reduce selection pool to elites
$\mathcal{T}' \leftarrow \mathcal{T}' -$ `case`	reduce remaining cases
return random choice from $\mathcal{S}$	return parent

5 Expected Probabilities of Selection

What is the probability of an individual being selected, given its performance in a given population on a set of training cases?

To put it into words, the probability of n being selected is the probability that a case n passes (a member of $\mathcal{K}_n$) is selected and:

1. no more cases remain and n is selected among the set of individuals that pass the selected case; or
2. n is the only individual that passes the case; or
3. n is selected via the selection of another case that n passes (repeating the process).

Formally, let $P_{sel}(n|\mathcal{N}, \mathcal{T})$ be the probability of n being selected in a population $\mathcal{N}$ with training cases $\mathcal{T}$. Let $\mathcal{K}_n(\mathcal{T}, \mathcal{N}) = \{k_i\}_{i=1}^{K} \subseteq \mathcal{T}$ be the training cases from $\mathcal{T}$ for which individual n is elite among $\mathcal{N}$. We will use $\mathcal{K}_n$ for brevity. Then the probability of selection under lexicase can be represented as a piece-wise recursive function:

$$P_{sel}(n|\mathcal{N}, \mathcal{T}) = \begin{cases} 1 & : \quad |\mathcal{T}| > 0, |\mathcal{N}| = 1; \\ 1/|\mathcal{N}| & : \quad |\mathcal{T}| = 0; \\ \frac{1}{|\mathcal{T}|} \sum_{k_s \in K_n} P_{sel}\left(n|\mathcal{N}(m|k_s \in K_m), \mathcal{T}(t|t \neq k_s)\right) & : \quad \text{otherwise} \end{cases} \tag{3}$$

The first two elements of P_{sel} follow from the lexicase algorithm: if there is one individual in $\mathcal{N}$, then it is selected; otherwise if there no more cases in in $\mathcal{T}$, then n has a probability of selection split among the individuals in $\mathcal{N}$, i.e., $1/|\mathcal{N}|$. If neither of these conditions are met, the remaining probability of selection is $1/|\mathcal{T}|$ times the summation of P_{sel} over n's elite cases. Each case for which n is elite among the current pool (represented by $k_s \in K_n$) has a probability of $1/|\mathcal{T}|$ of being selected. For each potential selection k_s, the probability of n being selected as a result of this case being chosen is dependent on the number of individuals that are also elite on these cases, represented by $\mathcal{N}(m|k_s \in K_m(\mathcal{T}))$, and the cases that are left to be traversed, represented by $\mathcal{T}(t|t \neq k_s)$.

Eqn. 3 also describes the probability of selection under ϵ-lexicase selection, with the condition that *elitism* on a case is defined as being within ϵ of the best error on that case, where the best error is defined among the whole population (statically) or among the current selection pool (semi-dynamic and dynamic). The definition of ϵ differs according to whether static, semi-dynamic, or dynamic lexicase are being used as well.

Intuitions according to edge cases According to Eqn. 3, when fitness values across the population are unique, selection probability is $\frac{1}{|\mathcal{T}|} \sum_{k_s \in \mathcal{K}_n(\mathcal{T})} 1 = \frac{|\mathcal{K}_n|}{|\mathcal{T}|} \ \forall \ n$, since filtering the population according to any case for which n is elite will result in n being selected. Conversely, if the population semantics are completely homogeneous such that every individual is elite on every case, the selection will be uniformly random, giving the selection probability $\frac{1}{N}$ $\forall$ n. In fact, this holds true for individual cases: a case t will have no impact on the probability of selection of n if $t \notin K_n$ $\forall$ n, t $\in \mathcal{N}, \mathcal{T}$. This is clear when viewed from the lexicase algorithm: any case that every individual passes provides no selective pressure.

Complexity Can we analytically calculate the probability of selection an individual with less complexity than executing the lexicase selection algorithm? It appears not. Eqn. 3 has a worst-case complexity of $O(T^N)$ when all individuals are elite on $\mathcal{T}$, which discourages its use as a selection method. The lexicase selection algorithm samples the expected probabilities of each individual by recursing on random orders of cases in $\mathcal{T}$, considering one at a time rather than branching to consider all combinations of cases that could result in selection for each individual in question. This gives lexicase selection a complexity of $O(TN)$ for selecting a single parent, and therefore a complexity of $O(TN^2)$ per generation.

5.1 Effect of N and T

What does the an analysis of the probability of selection for lexicase selection tell us about how lexicase behaves for cases in which $|\mathcal{N}| << |\mathcal{T}|$? $|\mathcal{T}| << |\mathcal{N}|$?

The sampling of P_{sel} done by lexicase is tied to the population size because lexicase selection conducts N depth-first searches of the case orderings to choose N parents. This implies that the value of N determines the fidelity with which P_{sel} is approximated via the sampling. Smaller populations will therefore produce poorer approximations of P_{sel}.

The effectiveness of lexicase selection has also been tied to the number of fitness cases, T. When T is small, there are very few ways in which individuals can be selected. For example, if $T = 2$, an individual must be elite on one of these two cases to be selected. For continuous errors in which few individuals are elite, this means that only two individuals are likely to produce all of the children for the subsequent generation.

5.2 Effect of different population structures

1. compare probability of lexicase selection to probability of selection with tournament / roulette
2. compare population structures: maintain correlation structure and vary population size/ number of test cases
3. see how many iterations of lexicase selection are required to converge on the probabilities of selection for the example problem in Table 1
4. population where there is one individual that sucks at everything but is good at other things - how does it compare to tournament selection probabilities?
5. do not assume that N rounds of lexicase selection are conducted!

Probabilities under tournament selection We compare the probability of selection under lexicase selection to that using tournament selection with an indentical population and fitness structure. To do so we must first formulate the probability of selection for an individual undergoing tournament selection with r-size tournaments. Consider that the mean absolute error is used to aggregate the fitness cases. Then the fitness ranks of $\mathcal{N}$ can be calculated, with lower rank indicating better fitness. Let S_i be the individuals in $\mathcal{N}$ with a fitness rank of i, and let Q be the number of unique fitness ranks. Then Xie et. al. [18] showed that the probability of selecting an individual with rank

j in a single tournament is

$$P_t = \frac{1}{|S_j|}\left(\left(\frac{\sum_{i=j}^{Q}|S_i|}{N}\right)^r - \left(\frac{\sum_{i=j+1}^{Q}|S_i|}{N}\right)^r\right) \tag{4}$$

In Table 1, the selection probabilities for the example population are shown according to tournament selection.

Example As an example of calculating absolute probabilities, we consider the illustrative problem from the original lexicase selection paper [15], shown in Table 1. Using Eqn. 3, the probabilities for each individual can be calculated as follows:

$$\begin{aligned}
P_{sel}(n_1) &= 1/4 * (1/3 * (1) + 1/3 * (1+1)) && = 0.25 \\
P_{sel}(n_2) &= 1/4 * (0) && = 0 \\
P_{sel}(n_3) &= 1/4 * (1/3 * (1 + 1/2 * (1)) + 1/3 * (1)) && = 0.20833 \\
P_{sel}(n_4) &= 1/4 * (1/3 * (1/2 * (1) + 1) + 1/3 * (1)) && = 0.20833
\end{aligned}$$

6 Multi-objective Interpretation of Lexicase Selection

Objectives and training cases are fundamentally different entities: objectives define the goals of the task being learned, whereas cases are the units by which progress towards those objectives is measured. By this criteria, lexicase selection and multi-objective optimization have historically been differentiated [4], although there is clearly a "multi-objective" interpretation of the behavior of lexicase selection with respect to the training cases. If we assume for a moment that we treat individual fitness cases as objectives to solve, we can consider most learning tasks to be high-dimensional many-objective optimization problems. At this scale, the most popular multi-objective methods (e.g. NSGA-II and SPEA-2) tend to perform poorly, a behavior that has been explained in literature [17, 3]. Farina and Amato [3] point out two short-comings of these multi-objective methods when many objectives are considered:

> the Pareto definition of optimality in a multi-criteria decision making problem can be unsatisfactory due to essentially two reasons: the number of improved or equal objective values is not taken into account, the (normalized) size of improvements is not taken into account

As we describe in §5, lexicase selection takes into account the number of improved or equal objectives (i.e. cases) by increasing the probability of selection for individuals who solve more cases (consider the summation in the third part of Eqn. 3). The increase per case is proportional to the difficulty of that case, as defined by the selection pool's performance. Regarding Farina's second point, the *size* of the improvements can be considered to be taken into account by ϵ-lexicase selection only. They are taken into account by the automated thresholding performed by ϵ which rewards individuals for being within an acceptable range of the best performance on the case. We develop the relationship between lexicase selection and Pareto optimization in the remainder of this section.

Lexicase selection guarantees the return of individuals that are on the Pareto front with respect to the fitness cases. This is a necessary but not sufficient condition. In fact, lexicase selection only selects those individuals in the "corners" of the Pareto front, meaning they are on the front *and* elite, globally, with respect to at least one fitness case. Put another way, no individual can be selected via lexicase selection unless it is elite with respect to at least one objective among the entire population, regardless of its performance on other objectives.

Interestingly, the worst-case complexity of NSGA-II is the best-case complexity for lexicase selection. Add notes from discourse

Here we define Pareto dominance relations with respect to the training cases.

Definition 6.1. *n_1 dominates n_2, i.e., $n_1 \prec n_2$, if $e_j(n_1) \leq e_j(n_2)$ $\forall j \in \{1, \dots, T\}$ and $\exists j \in \{1, \dots, T\}$ for which $e_j(n_1) < e_j(n_2)$.*

Definition 6.2. *The Pareto set of $\mathcal{N}$ is the subset of $\mathcal{N}$ that is non-dominated with respect to $\mathcal{N}$; i.e., $n \in \mathcal{N}$ is in the Pareto set if $n \not\prec m \; \forall \; m \in \mathcal{N}$.*

Definition 6.3. *$n \in \mathcal{N}$ is a Pareto set boundary if $n \in$ Pareto set of $\mathcal{N}$ and $\exists j \in \{1, \dots, T\}$ for which $e_j(n) \leq e_j(m) \; \forall \; m \in \mathcal{N}$.*

With these definitions in mind, we show that individuals selected by lexicase are on the Pareto set boundaries.

Theorem 6.4. *If individuals from a population $\mathcal{N}$ are selected by lexicase selection, those individuals are Pareto set boundaries of $\mathcal{N}$ with respect to $\mathcal{T}$.*

Proof: First, we prove (by supposing a contradiction) that individuals selected by lexicase are in the Pareto set. Second, we prove these individuals to be Pareto set boundaries.

First: Let $n_1, n_2 \in \mathcal{N}$ be individuals in a population selected by lexicase selection. Suppose $n_1 \prec n_2$. Then $e_j(n_1) \leq e_j(n_2)$ $\forall j \in \{1, \dots, N\}$ and $\exists j \in \{1, \dots, T\}$ for which $e_j(n_1) < e_j(n_2)$. Therefore n_1 is selected for every case that n_2 is selected, and $\exists t \in \mathcal{T}$ for which n_2 is removed from selection due to n_1. Hence, n_2 cannot be selected by lexicase selection and the supposition is false. Therefore n_1 and n_2 must be in the Pareto set.

Second: let n be an individual selected from population $\mathcal{N}$ by lexicase selection. Then by definition of the Algorithm 1, $\exists j \in \{1, \dots, T\}$ for which $e_j(n) \leq e_j(m) \; \forall \; m \in \mathcal{N}$. Therefore, since n is in the Pareto set of $\mathcal{N}$ and according to Definition 6.3, n is a Pareto set boundary of $\mathcal{N}$.

Extension to ϵ-lexicase selection We can extend our multi-objective analysis to ϵ-lexicase selection for conditions in which ϵ is pre-defined for each fitness case, i.e. in static and dynamic cases (Eqns. However when ϵ is recalculated for each selection pool, the theorem is not as easily extended due to the need to account for the dependency of ϵ on the current selection pool. We first define ϵ elitism in terms of a relaxed dominance relation and a relaxed Pareto set. We define the dominance relation with respect to ϵ as follows:

Definition 6.5. *n_1 ϵ-dominates n_2, i.e., $n_1 \prec_\epsilon n_2$, if $e_j(n_1) + \epsilon_j \leq e_j(n_2)$ $\forall j \in \{1, \dots, T\}$ and $\exists j \in \{1, \dots, T\}$ for which $e_j(n_1) + \epsilon_j < e_j(n_2)$, where $\epsilon_j > 0$ is defined according to Eqn. 2.*

It is important to note that this definition of ϵ-dominance differs from a previous ϵ-dominance definition used by Laumanns et. al. [12] (cf. Eqn. (6)) in which $n_1 \prec_\epsilon n_2$ if

$$e_j(n_1) + \epsilon_j \leq e_j(n_2) \; \forall \; j \in \{1, \dots, T\}$$

According to Definition 6.5, the ϵ-dominance criteria is more strict, requiring $e_j(n_1) + \epsilon_j < e_j(n_2)$ for at least one j. As a result, the non-ϵ-dominated set, defined as the ϵ-Pareto set below, is expected to be as large or larger than the ϵ-approximate Pareto set used in [12]. In order to extend Theorem 6.4, this definition must be more strict since a useful ϵ-dominance relation needs to capture the ability of an individual to force the removal of another from selection under ϵ-lexicase selection.

Definition 6.6. *The ϵ-Pareto set of $\mathcal{N}$ is the subset of $\mathcal{N}$ that is non-ϵ-dominated with respect to $\mathcal{N}$; i.e., $n \in \mathcal{N}$ is in the ϵ-Pareto set if $n \not\prec_\epsilon m \; \forall \; m \in \mathcal{N}$.*

Definition 6.7. *$n \in \mathcal{N}$ is an ϵ-Pareto set boundary if n is in the ϵ-Pareto set of $\mathcal{N}$ and $\exists j \in \{1, \dots, T\}$ for which $e_j(n_1) \leq e_j(m) + \epsilon_j \;\forall\; m \in \mathcal{N}$, where ϵ_j is defined according to Eqn 2.*

Theorem 6.8. *If ϵ is defined according to Eqn. 2, and if individuals are selected from a population $\mathcal{N}$ by ϵ-lexicase selection, then those individuals are elements are ϵ-Pareto set boundaries of $\mathcal{N}$.*

Proof. First: Let n_1 and n_2 be individuals selected from $\mathcal{N}$ by static or semi-dynamic ϵ-lexicase selection. Suppose $n_1 \prec_\epsilon n_2$. Then n_1 is selected for every case that n_2 is selected, and $\exists t \in \mathcal{T}$ in every selection event for which n_2 is removed from selection due to n_1. Hence, n_2 cannot be selected by lexicase selection and the supposition $n_1 \prec_\epsilon n_2$ is false. Therefore n_1 and n_2 must be in the ϵ-Pareto set of $\mathcal{N}$ to be selected.

Second: let n be an individual selected from population $\mathcal{N}$ by static or semi-dynamic ϵ-lexicase selection. Then by definition of Algorithm 2 or 3, $\exists j \in \{1, \dots, T\}$ for which $e_j(n) \leq e_j(m) \;\forall\; m \in \mathcal{N}$. Since n is in the ϵ-Pareto set of $\mathcal{N}$ and according to Definition 3, n must be a ϵ-Pareto set boundary of $\mathcal{N}$. ☐

The notion of the selection of Pareto set boundaries by lexicase selection is illustrated in Figure 2 for a scenario with two fitness cases. Each point in the plot represents an individual, and the squares are the Pareto set. Under a lexicase selection event with case sequence $\{t_1, t_2\}$, individuals would first be filtered to the two left-most individuals that are elite on e_1, and then to the individual among those two that is best on e_2, i.e. the selected square individual. Note that the selected individual is a Pareto set boundary. Given the opposite order of cases, i.e. $\{t_2, t_1\}$, the individual on the other end of the Pareto set shown as a black square would be selected.

Consider the analogous case for semi-dynamic ϵ-lexicase selection illustrated in Figure 3. In this case the Pareto set is indicated again by squares. Under a semi-dynamic ϵ-lexicase selection event with case order $\{t_1, t_2\}$, the population would first be filtered to the 4 left-most individuals that are within ϵ_1 of the elite error on case t_1, and then the indicated square would be selected by being the only individual within ϵ_2 of the elite error on t_2 among the current pool. It is important to note that that although the selected individual is an ϵ-Pareto set boundary by Definition 6.7, it is *not* a boundary of the Pareto set. Theorem 6.8 only guarantees that the selected program is within ϵ of the best error for at least one case, which in this scenario is t_1.

Comparing Figures 2 and 3 illustrates the effect of introducing ϵ to the filter conditions in lexicase: it reduces the selectivity of each case, ultimately resulting in the selection of individuals that are not as extremely positioned in objective space. Regarding the position of solutions in this space, it's worth noting the significance of boundary solutions (and near boundary solutions) in the context of multi-objective optimization. Boundary solutions are known to contribute significantly to hypervolume measures [2], where the hypervolume is a measure of how well-covered the objective space is by the current set of solutions. The boundary solutions have an infinite score according to NSGA-II's crowding measure [1], with higher being better, meaning they are the first non-dominated solutions to be preserved by selection when the population size is reduced. Nonetheless, multi-objective literature appears divided on how these boundary solutions drive search [17], when the goal of the algorithm is to approximate the optimal Pareto front. In contrast, the goal of GP is to preserve points in the search space that, when combined and varied, yield a single best solution. So while the descriptions above lend insight to the function of lexicase and ϵ-lexicase selection, it's important to remember the different goals of search and the high dimensionality of training cases when compared to traditional objectives.

Relation to Evolutionary Multi-objective Optimization There is an interesting relationship to be made regarding the complexity of lexicase selection in comparison to NSGA-II. Consdering test cases as objectives, we see that lexicase selection and NSGA-II have the same worst-case complexity

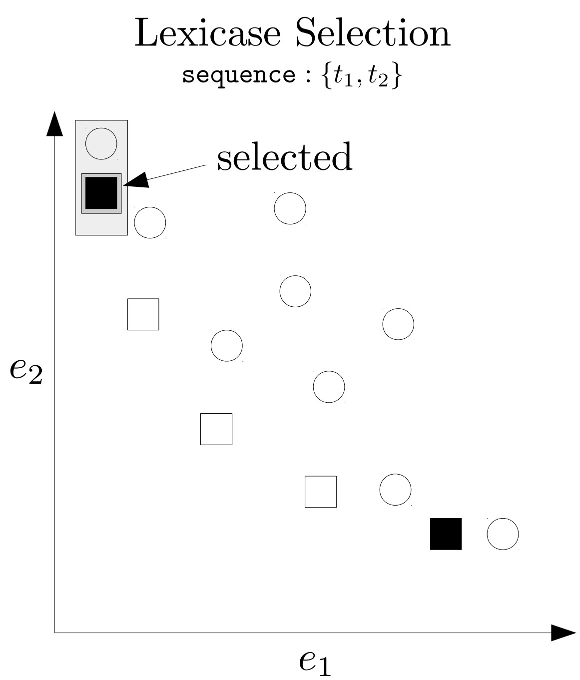

Figure 2: An illustration of the performance of lexicase selection in a scenario involving two cases. Each point represents and individual in the population. The squares are individuals in the Pareto set. A selection event for case sequence $\{t_1, t_2\}$ is shown by the gray rectangles. The black points are individuals that could be selected by any case ordering.

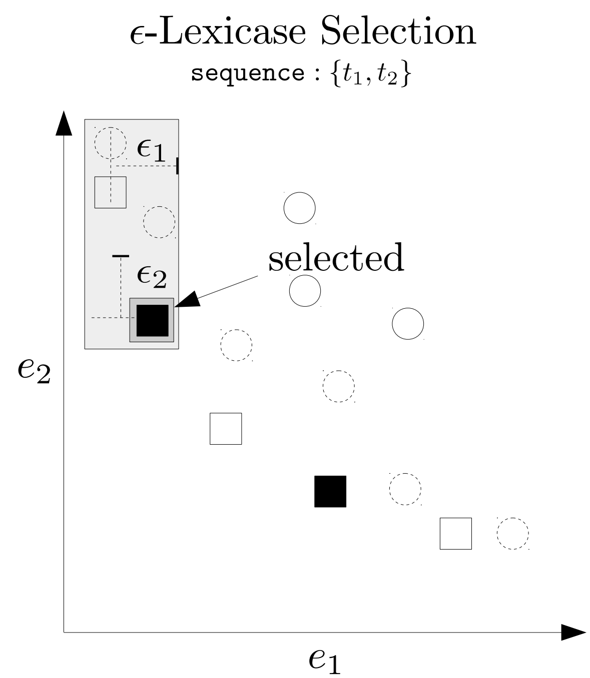

Figure 3: An illustration of the performance of semi-dynamic ϵ-lexicase selection in a scenario involving two cases. Each point represents and individual in the population. The squares are individuals in the Pareto set, and the circles with dotted lines are the subset of Pareto set individuals that are also in the ϵ-Pareto set. A selection event for case sequence $\{t_1, t_2\}$ is shown by the gray rectangles. The black points are individuals that could be selected by any case ordering.

Table 2: Example population with test case performances and selection probabilities according to the different algorithms.

$\mathcal{N}$	Cases					Mean	Probability of Selection				
	t_1	t_2	t_3	t_4	t_5		tourn	lex	ϵ lex static	ϵ lex semi	ϵ lex dyn
n_1	0.0	1.1	2.2	3.0	5.0	2.26	0.111	0.200	0.000	0.067	0.033
n_2	0.1	1.2	2.0	2.0	6.0	2.26	0.111	0.000	0.150	0.117	0.200
n_3	0.2	1.0	2.1	1.0	7.0	2.26	0.111	0.000	0.150	0.117	0.117
n_4	1.0	2.1	0.2	0.0	8.0	2.26	0.111	0.200	0.300	0.200	0.167
n_5	1.1	2.2	0.0	4.0	4.0	2.26	0.111	0.200	0.000	0.050	0.050
n_6	1.2	2.0	0.1	5.0	3.0	2.26	0.111	0.000	0.000	0.050	0.033
n_7	2.0	0.1	1.2	6.0	2.0	2.26	0.111	0.000	0.133	0.133	0.133
n_8	2.1	0.2	1.0	7.0	1.0	2.26	0.111	0.000	0.133	0.133	0.217
n_9	2.2	0.0	1.1	8.0	0.0	2.26	0.111	0.400	0.133	0.133	0.050
ϵ	0.9	0.9	0.9	2.0	2.0						

of $O(T^2N)$. However the algorithms differ with respect to the conditions under which worst case complexities arise.

NSGA-II has three main parts: sorting ($O(TN^2)$), crowding distance assignment $O(TN\log(N))$, and sorting with crowding comparison $O(2Nlog(2N))$, giving an overall complexity of $O(TN^2)$. The sorting complexity is $O(TN^2)$ to identify a single front; the crowding distance assignment complexity varies, however. Its worst case complexity arises when all individuals are non-dominated, which is excpected in high dimensions [3, 17].

Interestingly, *if the population is unique*, this is the minimum complexity scenario for lexicase selection: in other words, if the semantics of the population are unique and all are non-dominated in $\mathcal{T}$, only case depth 1 selections will occur, giving a run-time complexity of $O(N^2)$. Of course, the population could also be non-dominated by being semantically identical, in which case the complexity would be $O(TN^2)$. The important thing to note is that lexicase selection, which rewards individuals for being diverse, also runs more quickly when the population is more diverse.

7 Illustrative Example

A more complex population semantic structure is presented in Table 2 featuring floating point errors. In this case, the population semantics are completely unique, although they result in the same mean error across the training cases, as shown in the "Mean" column. As a result, tournament selection picks uniformly from among these individuals.

As mentioned earlier, with unique populations, lexicase selection is proportional to the number of cases for which an individual is elite. This leads lexicase selection to pick from among the 4 individuals that are elite on cases, i.e. n_1 (t_1), n_4 (t_4), n_5 ($t3$), and n_9 (t_2,t_5), with respective probabilities 0.2, 0.2, 0.2, and 0.4.

Due to its strict definition of elitism, lexicase selection does not account for the fact other individuals are very close to being elite on these cases as well. The ϵ-lexicase variants address this as noted by the smoother distribution of selection probabilities among this population. Focusing first on static ϵ-lexicase selection, it is worth illustrating the pass conditions for the population by applying the ϵ threshold, yielding the following errors:

	t_1	t_2	t_3	t_4	t_5
n_1	0	1	1	1	1
n_2	0	1	1	0	1
n_3	0	1	1	0	1
n_4	1	1	0	0	1
n_5	1	1	0	1	1
n_6	1	1	0	1	1
n_7	1	0	1	1	0
n_8	1	0	1	1	0
n_9	1	0	1	1	0

The selection probabilities for static ϵ lexicase selection are equivalent to the selection probabilities of lexicase selection on this converted semantic matrix. Despite elitism on case t_1, n_1 is not selected since n_2 and n_3 are ϵ-elite on this case as well as t_4. Consider n_4, which has a higher probability of selection under static ϵ-lex than lexicase selection. This is due to it being ϵ-elite on a unique combination of cases: t_3 and t_4. Lastly n_8 is selected in equal proportions to n_7 and n_6 since all are within ϵ of the elite error on the same cases.

Semi-dynamic ϵ lexicase selection allows for all 9 individuals to be selected with varying proportions that are similar to those derived for static ϵ lexicase selection. Selection probabilities for n_1 illustrate the differences in the static and semi-dynamic variants: n_1 has a chance for selection in the semi-dynamic case because when t_1 is selected as the first case, n_1 is within ϵ of the best case errors *among the pool*, i.e. $\{n_1, n_2, n_3\}$, for any subsequent order of cases. The probability of selection for n_5 and n_6 follow the same pattern.

Dynamic ϵ-lexicase selection produces the most differentiated selection pressure for this example. Consider individual n_8 which is the most likely to be selected for this example. It is selected more often than n_7 or n_9 due to the adaptations to ϵ as the selection pool is winnowed. For example, n_8 is selected by case sequence $\{t_2, t_1, t_3\}$, for which the selection pool takes the following form after each case: $\{n_7, n_8, n_9\}, \{n_7, n_8\}, \{n_8\}$. Under the other semi-dynamic ϵ lexicase selection, n_7 and n_8 would not be removed by these cases due to the fixed nature of ϵ for those cases. The probabilities of selection for the population in Table 2 is shown graphically in Figure 7.

8 Related Work

9 Experimental Analysis

The problems studied in this section are listed in Table 4. The boxplots in Figure 5 show the test set MSE for various implementations of selection on these problems, and in comparison to Lasso.

1. show the diversity of the populations over time for different methods

2. show the number of cases used per method for the lexicase implementations

3. compare the wilcoxon tests of signficance for the regression problems

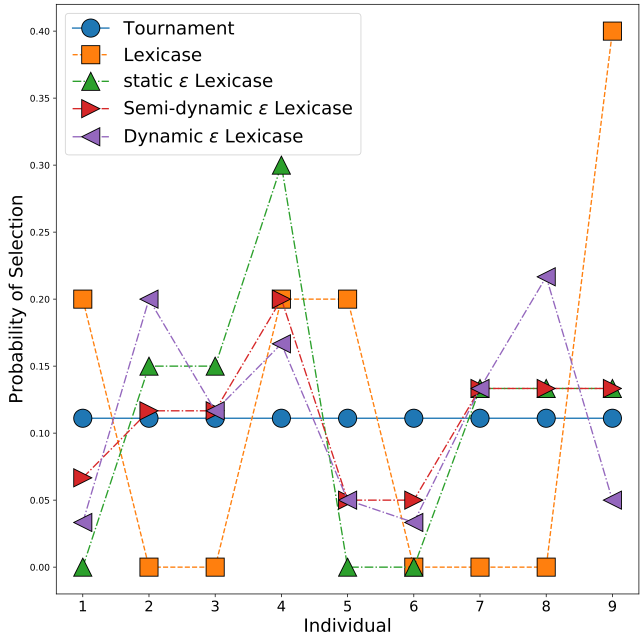

Figure 4: A graphical representation of the probabilities of selection for the population in Table 2 according to different selection methods.

Table 3: GP settings.

Setting	Value
Population size	1000
Crossover / mutation	60/40%
Program length limits	[3, 50]
ERC range	[-1,1]
Generation limit	1000
Trials	50
Terminal Set	$\{\mathbf{x}$, ERC, $+$, $-$, $*$, $/$, sin, cos, exp, log$\}$
Elitism	keep best

Table 4: Regression problems used for method comparisons.

Problem	Dimension	Samples
Airfoil	5	1503
Concrete	8	1030
ENC	8	768
ENH	8	768
Housing	14	506
Tower	25	3135
UBall5D	5	6024
Yacht	6	309

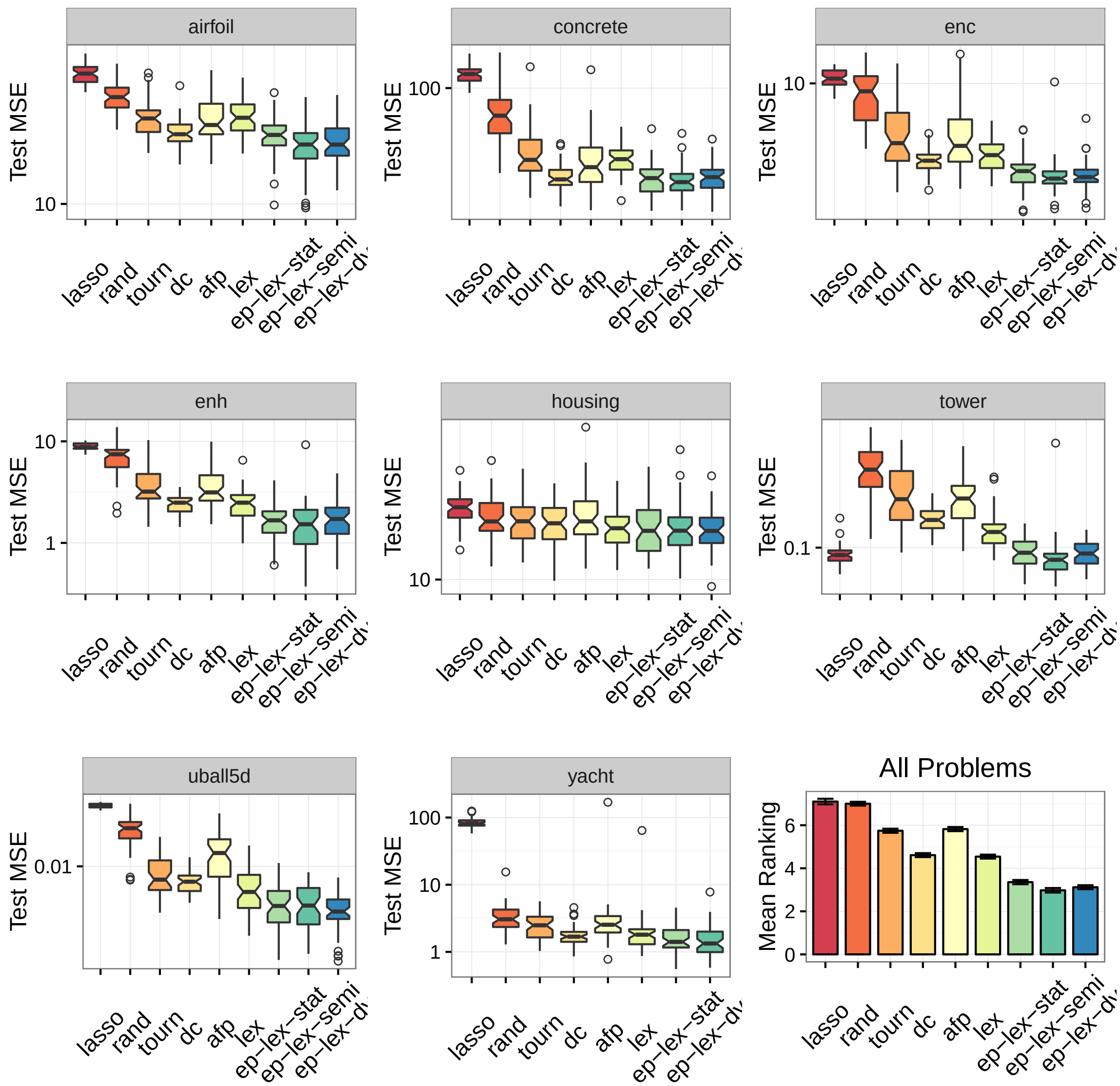

Figure 5: Boxplots of the mean squared error on the test set for 50 randomized trials of each algorithm on the regression benchmark datasets.

9.1 Results

10 Discussion

11 Conclusions

12 Acknowledgments

References

[1] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature PPSN VI*, volume 1917, pages 849–858. Springer Berlin Heidelberg.

[2] K. Deb, M. Mohan, and S. Mishra. Evaluating the -domination based multi-objective evolutionary algorithm for a quick computation of pareto-optimal solutions. 13(4):501–525.

[3] M. Farina and P. Amato. On the optimal solution definition for many-criteria optimization problems. In *Fuzzy Information Processing Society, 2002. Proceedings. NAFIPS. 2002 Annual Meeting of the North American*, pages 233–238. IEEE.

[4] T. Helmuth. General program synthesis from examples using genetic programming with parent selection based on random lexicographic orderings of test cases.

[5] T. Helmuth and L. Spector. General program synthesis benchmark suite. pages 1039–1046. ACM Press.

[6] T. Helmuth, L. Spector, and J. Matheson. Solving uncompromising problems with lexicase selection. PP(99):1–1.

[7] J. Klein and L. Spector. Genetic programming with historically assessed hardness. pages 61–75.

[8] K. Krawiec and P. Lichocki. Using co-solvability to model and exploit synergetic effects in evolution. In R. Schaefer, C. Cotta, J. Koodziej, and G. Rudolph, editors, *Parallel Problem Solving from Nature, PPSN XI*, pages 492–501. Springer Berlin Heidelberg.

[9] K. Krawiec and P. Liskowski. Automatic derivation of search objectives for test-based genetic programming. In *Genetic Programming*, pages 53–65. Springer.

[10] W. La Cava, K. Danai, L. Spector, P. Fleming, A. Wright, and M. Lackner. Automatic identification of wind turbine models using evolutionary multiobjective optimization. 87, Part 2:892–902.

[11] W. La Cava, L. Spector, and K. Danai. Epsilon-lexicase selection for regression. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, GECCO '16, pages 741–748. ACM.

[12] M. Laumanns, L. Thiele, E. Zitzler, and K. Deb. Archiving with guaranteed convergence and diversity in multi-objective optimization. In *Proceedings of the 4th Annual Conference on Genetic and Evolutionary Computation*, pages 439–447. Morgan Kaufmann Publishers Inc.

[13] P. Liskowski, K. Krawiec, T. Helmuth, and L. Spector. Comparison of semantic-aware selection methods in genetic programming. In *Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation*, GECCO Companion '15, pages 1301–1307. ACM.

[14] R. I. B. McKay. An investigation of fitness sharing in genetic programming. 7(1):43–51.

[15] L. Spector. Assessment of problem modality by differential performance of lexicase selection in genetic programming: a preliminary report. In *Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference companion*, pages 401–408.

[16] L. Vanneschi, M. Castelli, and S. Silva. A survey of semantic methods in genetic programming. 15(2):195–214.

[17] T. Wagner, N. Beume, and B. Naujoks. Pareto-, aggregation-, and indicator-based methods in many-objective optimization. In *Evolutionary Multi-Criterion Optimization*, pages 742–756. Springer, Berlin, Heidelberg. DOI: 10.1007/978-3-540-70928-2_56.

[18] H. Xie, M. Zhang, and P. Andreae. Another investigation on tournament selection: modelling and visualisation. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 1468–1475. ACM.