Specialization and Elitism in Lexicase and Tournament Selection

Edward Pantridge
MassMutual
epantridge@massmutual.com

Thomas Helmuth
Hamilton College
thelmuth@hamilton.edu

Nicholas Freitag McPhee
University of Minnesota Morris
mcphee@morris.umn.edu

Lee Spector
Hampshire College
lspector@hampshire.edu

ABSTRACT

Prior work has demonstrated that genetic programming systems often maintain higher levels of population diversity when using lexicase selection than when using other parent selection methods, and that the use of lexicase selection improves problem-solving performance in many circumstances. It has been suggested that it is not only the maintenance of diversity that is responsible for the performance of lexicase selection, but more specifically the production and maintenance of "specialists" that matters, where specialists are defined to be individuals with the lowest error, relative to the rest of the population, on a small number of training cases regardless of total error. Here we provide results of experiments that uphold this suggestion by tracking the numbers of specialists selected by lexicase selection and by tournament selection in a genetic programming system solving software synthesis problems. Our results also show that lexicase selection selects parents with poor total error far more frequently than tournament selection, even near the ends of successful runs, suggesting that such selections are integral to the improved problem-solving performance of lexicase selection.

CCS CONCEPTS

• **Software and its engineering → Automatic programming**; **Genetic programming**;

KEYWORDS

lexicase selection; tournament selection; genetic programming; program synthesis;

ACM Reference Format:
Edward Pantridge, Thomas Helmuth, Nicholas Freitag McPhee, and Lee Spector. 2018. Specialization and Elitism in Lexicase and Tournament Selection. In *Proceedings of the Genetic and Evolutionary Computation Conference 2018 (GECCO '18).* ACM, New York, NY, USA, Article 4, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Genetic programming proceeds through a cycle of selecting parent programs, producing child programs from those parents, and assessing those children with respect to their performance on a target problem [6]. In the present paper we focus on the parent selection step of the genetic programming algorithm, which determines which programs in the current population will be used as the source material out of which the next generation's programs will be constructed.

The most commonly employed parent selection methods select parents on the basis of the quality of potential parents, which is assumed to be represented by a single numerical value for each program, in combination with some form of randomization. One such parent selection method is tournament selection, in which, for each parent selection event, a small, constant number of potential parents is chosen from the population with uniform probability, and then the highest quality individual from that "tournament set" is selected as the parent.

By contrast, in lexicase selection [5, 8] program errors on individual training cases all contribute to the selection of parents without being aggregated. Prior research has demonstrated that lexicase selection can significantly improve problem-solving performance and the diversity of the populations on which it operates [4].

The specific hypothesis that we explore is that lexicase selection not only promotes diversity in general, but that it more specifically promotes the generation and maintenance of "specialists." While this might seem to follow naturally from the definition of lexicase selection (see below), it has never before been demonstrated empirically.

In the following section we describe the lexicase selection algorithm and some of the prior results on the interactions between lexicase selection and population diversity. We then describe the methods that we employed for our experiments, followed by our experimental results. We conclude with comments on the implications of these results and suggestions for future research.

2 LEXICASE SELECTION

Unlike the majority of contemporary parent selection methods, lexicase selection does not choose a parent individual based on an aggregated error value. Instead lexicase selection considers an individual's error on each particular training case separately. Lexicase selection considers training cases one at a time, in random order, iteratively filtering the population down to just individuals with the lowest error on the current training case. Once there is either a single individual, or all training cases have been used to filter the population, lexicase selection terminates by returning a random individual from the filtered population.

The lexicase selection algorithm is described in detail by [8] and [5]. Notice that after any number of training cases the collection

GECCO '18, July 15–19, 2018, Kyoto, Japan

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

of `candidates` might contain a single individual which becomes the selected parent. The error values of that individual on the remaining training cases could be arbitrarily high without disqualifying the individual for selection.

One important note about the implementation of lexicase selection is that the population generally undergoes a "pre-selection" phase which filters the population down to one randomly chosen individual for each distinct vector of errors in the population. This implementation produces the same behavior, but removes the worst case runtime situation and is why we do not see any lexicase selection events which utilize every training case in the results presented in section 4.

We define a "specialist" as an individual with elite error values on some relatively small subset of the training cases regardless of the individual's error values on the rest of the training set. Given the lexicase selection algorithm described above, it is clear that there is the possibility for lexicase selection to select specialists. This paper presents empirical evidence that lexicase selects specialists in practice, and it is suggested that these selections are beneficial to evolution.

2.1 Previous Results

Since its initial proposal, lexicase selection has proven to be useful in genetic programming systems designed for software synthesis tasks. The reason for lexicase selection's use in this domain is the higher success rate observed in many problems across multiple genetic programming techniques [1, 5]. Since the discovery of these results in favor of lexicase selection there has been an ongoing effort to better understand the behavior of lexicase selection in order to determine what makes it so effective at finding solutions.

One important finding about lexicase selection is that it maintains a higher diversity in the population throughout an evolutionary run compared to other selection methods [3]. Furthermore, lexicase selection has been shown to be much better at recovering diversity than tournament selection after a diversity crash [2].

3 METHODS

Data on lexicase selection was gathered on a suite of 5 problems, all of which were taken from the General Program Synthesis Benchmark Suite [4]. Lexicase selection has been shown to produce more solutions to these problems and maintain higher diversity than tournament selection, with or without implicit fitness sharing, as discussed in Section 2.1. All experimental runs were done using the Clojush genetic programming framework.[1] A summary of the hyperparameter configuration of the Clojush system used for every experimental run is given in [4].

The parent selection algorithm hyperparameter was varied across experimental runs for each problem. The empirical results presented in this paper were gathered from three genetic programming runs per selection method per problem. While this is clearly not enough runs to demonstrate the superiority of one setting over another, that is not our aim here; it is, however, sufficient to gather data that characterizes the behavior of the selection algorithms when applied to these problems. The two parent selection methods included in the comparison are lexicase selection and tournament selection. These runs produced a mean number of selection events per configuration of 699,278. The range of number of selection events for each experimental configuration is 124,087 to 1,529,694.

[1] https://github.com/lspector/Clojush

4 RESULTS

4.1 Cases Utilized for Selection

The first set of results, shown in Figure 1, presents the distribution of number of training cases utilized by lexicase selection. It is clear that in all problems the majority of selection events are concluded using less than 25% of the training cases, and very few selection events utilize more than 50% of the training cases.

The "mirror image" problem shows a slightly different trend than the other problems in that it has a peak centered around 12 training cases where other problems peak at 1 or 2 training cases. The "mirror image" problem is also the only problem that has more selection events that use 75+ training cases than 50 to 75 training cases. These unique qualities of the mirror image problem may indicate some interesting dynamics of lexicase selection, but they do not disagree with the hypothesis regarding specialists because the vast majority of the selection events still utilize fewer than 50% of the training cases.

Every time lexicase selection utilizes a small subset of training cases, it might be selecting an individual with a very high total error relative to the rest of the population. If only 50% of the training cases were considered when selecting an individual it is possible that some, or all, of the unseen training cases produced very high errors. These high errors could come from particularly inaccurate predictions or even penalties. Under other selection methods, such as tournament selection, the errors from these training cases would be aggregated into an exceptionally high total error and thus virtually disqualify the specialist from selection.

4.2 Total Error Rank of Selections

It is clear that lexicase selection provides the opportunity for selection of individuals with relatively high total error. However, it has only anecdotally been shown that lexicase selection selects individuals with high total error in practice [7].

To better illustrate the degree to which lexicase selection is elitist with respect to total error, we instrumented the total error rank of each individual selected during a selection event for both lexicase selection and tournament selection. As mentioned previously, the individual with the lowest total error in the population at the current generation will have a rank of 1, the second lowest total error will have a rank of 2, and so on.

Figure 2 shows a histogram depicting the distribution of total error rank for selected individuals. Both lexicase selection and tournament selection are strongly right tailed, suggesting that individuals with a low total error relative to the rest of the population have a higher chance of being selected under both algorithms. However, lexicase selection clearly selects individuals with a high total error rank more frequently.

Figure 3 shows the average rank of selected individuals at each generation of each run. Lexicase selection's tolerance of individuals with relatively high total error is clear in this chart, because at all generations of all runs on all problems lexicase selection appears to

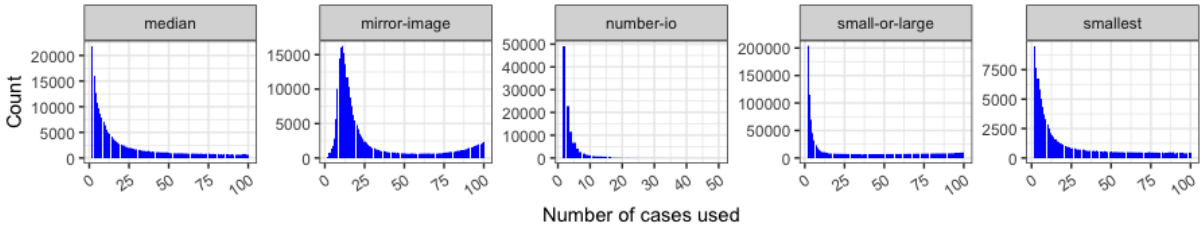

Figure 1: The distribution of number of test cases utilized by lexicase selection across all runs.

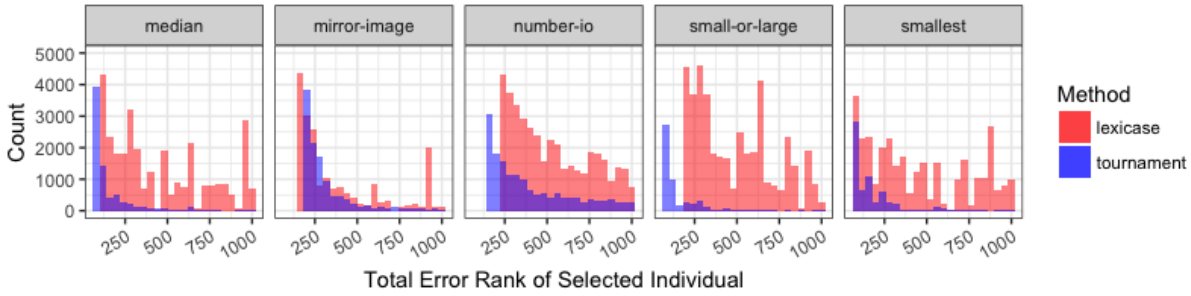

Figure 2: Histograms showing the number of selection events of individuals with each total error rank, for each of the studied problems under lexicase and tournament selection.

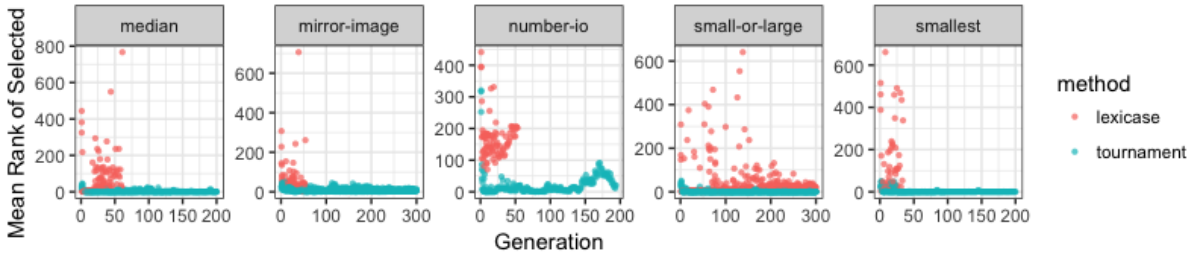

Figure 3: The average rank of selected individuals at each generation for each run, including linear trend lines. All lexicase selection runs, except for "small or large", found solutions before the maximum number of generations, which is why there is no data for later generations.

select individuals with a higher total error rank than tournament selection.

4.3 Selection of Specialists

To show that lexicase selection makes frequent selections of specialists Figure 4 plots the proportion of training cases utilized by lexicase selection versus the total error rank of the selected individuals. It is evident that the selection events which select individuals with high total error rank tend to also utilize fewer training cases. The individuals selected by lexicase selection at these events are specialists. The data presented in section 4.2 distinctly conveys that tournament selection will rarely select individuals with high total error ranks. Thus, it must be the case that tournament selection would almost never select the specialists seen in the data of lexicase selection events.

4.4 Do Specialists Produce Solutions?

Lexicase selection has been shown to have a higher solution rate than tournament selection for software synthesis problems. As seen in section 4.1 through 4.3, lexicase selection selects specialist individuals with relatively high total error, while tournament selection does not. Thus it can be shown that there is a correlation between selecting specialist individuals and higher solution rates.

But is there a causal connection here, beyond the correlation? To examine the link between selecting specialists with higher total error and improved solution rates, figure 5 plots data from the final 8 generations of runs which found solution programs in a similar fashion to figure 4. The "small or large" problem is the only problem which experiences almost no selections of specialists in the final generations leading to a solution. This finding merits future study.

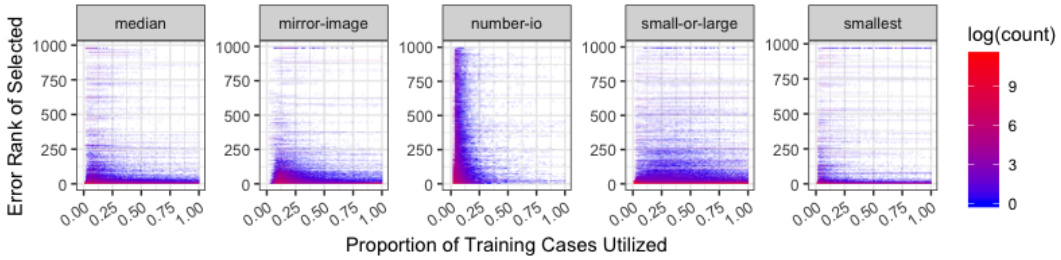

Figure 4: The proportion of training cases utilized by lexicase selection versus the total error rank of the selected individuals across all generations of all runs.

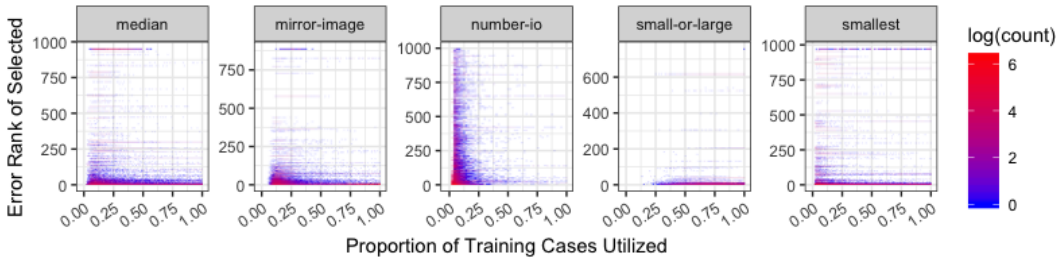

Figure 5: The proportion of training cases utilized by lexicase selection versus the total error rank of the selected individuals in the final 8 generations of runs which resulted in find a solution.

5 CONCLUSION

This paper presents empirical evidence that suggests lexicase selection selects "specialist" individuals. The results in this paper also show that lexicase selection is far less elitist with respect to total error than tournament selection, both throughout an evolutionary run and directly before finding solution programs. This may explain the higher levels of diversity seen throughout evolution when using lexicase selection, and it may also help to explain the strong problem-solving performance of lexicase selection.

In the future it would be informative to include more selection methods in the comparison. We feel that instrumenting implicit fitness sharing and novelty selection using the methods utilized in this paper could provide a more interesting comparison of selection methods. This would come at a greatly increased computational and storage cost, as all selection methods produce a large number of selection events per run.

It would also be informative to see if the behaviors demonstrated in this paper generalize to other problem domains.

ACKNOWLEDGEMENTS

Data storage and some compute infrastructure provided by MassMutual. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of MassMutual.

This material is based upon work supported by the National Science Foundation under Grant No. 1617087. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] Stefan Forstenlechner, David Fagan, Miguel Nicolau, and Michael O'Neill. 2017. A Grammar Design Pattern for Arbitrary Program Synthesis Problems in Genetic Programming. In *Genetic Programming*. Springer International Publishing, Cham, 262–277.

[2] Thomas Helmuth, Nicholas Freitag McPhee, and Lee Spector. 2016. Effects of Lexicase and Tournament Selection on Diversity Recovery and Maintenance. In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion (GECCO '16 Companion)*. ACM, New York, NY, USA, 983–990. http://doi.acm.org/10.1145/2908961.2931657

[3] Thomas Helmuth, Nicholas Freitag McPhee, and Lee Spector. 2016. *Lexicase Selection for Program Synthesis: A Diversity Analysis.* Springer International Publishing, Cham, 151–167. https://doi.org/10.1007/978-3-319-34223-8_9

[4] Thomas Helmuth and Lee Spector. 2015. General Program Synthesis Benchmark Suite. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation (GECCO '15)*. ACM, New York, NY, USA, 1039–1046. http://doi.acm.org/10.1145/2739480.2754769

[5] T. Helmuth, L. Spector, and J. Matheson. 2015. Solving Uncompromising Problems With Lexicase Selection. *IEEE Transactions on Evolutionary Computation* 19, 5 (Oct 2015), 630–643.

[6] John R. Koza. 1992. *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* MIT Press, Cambridge, MA, USA.

[7] Nicholas Freitag McPhee, David Donatucci, and Thomas Helmuth. 2016. *Using Graph Databases to Explore the Dynamics of Genetic Programming Runs.* Springer International Publishing, Cham, 185–201. https://doi.org/10.1007/978-3-319-34223-8_11

[8] Lee Spector. 2012. Assessment of Problem Modality by Differential Performance of Lexicase Selection in Genetic Programming: A Preliminary Report. In *Proceedings of the 14th Annual Conference Companion on Genetic and Evolutionary Computation (GECCO '12)*. ACM, New York, NY, USA, 401–408. http://doi.acm.org/10.1145/2330784.2330846