Towards a Dynamic Benchmark for Genetic Programming

Clíodhna Tuite, Michael O'Neill, and Anthony Brabazon
Complex and Adaptive Systems Laboratory
University College Dublin, Ireland
cliodhna.tuite@gmail.com, m.oneill@ucd.ie, anthony.brabazon@ucd.ie

ABSTRACT

Following a recent call for a suite of benchmarks for genetic programming [4], we investigate the criteria for a meaningful dynamic benchmark for GP. We explore the design of a dynamic benchmark for symbolic regression, based on semantic distance between evaluated functions, where larger distances serve as a proxy for greater environmental change. We do not find convincing evidence that lower semantic distance is a good proxy for greater ease in adapting to a change. We conclude that due to fundamental characteristics of GP, it is difficult to come up with a single dynamic benchmark problem which is generally applicable.

Categories and Subject Descriptors: I.2.8 [Problem Solving, Control Methods, and Search]: Heuristic methods

Keywords: Dynamical optimization, genetic programming

1. INTRODUCTION

Following a recent call to create a suite of benchmarks for GP [4], this paper investigates the criteria for a meaningful GP dynamic benchmark. We are not aware of a large body of research on GP benchmarks for dynamic environments. Greater effort has been spent on GP techniques to cope with dynamism in the environment [6, 2]; in the broader field of EC, Dempsey et al. [3] organise these techniques into five categories including: memory-based approaches, diversity-based approaches and use of multiple populations.

Branke's popular moving peaks dynamic benchmark [1] (MPB) for EAs consists of a multi-dimensional, multi-modal real-valued parameter space - located in the space are multiple peaks of different heights and widths. The fitness at each point in the landscape is given by the maximum over the peak functions at that point. The task for the EA is to locate the optimum fitness point within the landscape. None of the research reviewed in a recent review [5] of all algorithms known to have been tested on the MPB uses GP. Instead, many approaches used fixed-length EAs, or PSO algorithms, whose representation is more suited to searching for points in a real-valued multidimensional space.

Our first approach was to formulate a dynamic benchmark for GP which, like the MPB, was tunable in the manner of being able to tune the height, width and position of peaks in a fitness landscape. We decided against pursuing this approach, for two reasons. The first was the inability to enumerate all points in the fitness landscape for trees of any significant depth. The second problem was that the location of fitness peaks in the landscape would be difficult to control, due to the presence of trees in the landscape, which, while structurally different, evaluated to the same solution point.

GECCO'13 Companion, July 6-10, 2013, Amsterdam, The Netherlands.
ACM 978-1-4503-1964-5/13/07.

Table 1: List of Target Equations

Function	Range	Num. Pts.
(1) $X^8 + 3 \times \cos(X) + X^5 + X$	[-1, 1]	20
(2) $X^5 + X^4 + X^3 + \frac{X^2}{4} + X$ **	[-1, 1]	20
(3) $\cos(1) \times \sin(X^3) - \ln(1) + e^{(X+1)} + (X^4)$	[-1, 1]	20
(4) $\frac{X}{X+1} - X^3 + \pi$	[0, 2]	20
(5) $X^6 + X^5 + X^4 + X^3 + X^2 + X$ *	[-1, 1]	20
(6) $\ln(X + 1) + \ln(X^2 + 1)$ *	[0, 2]	20
(7) $\sqrt{X}$ *	[0, 4]	20
(8) $\sin(X^2) \times \cos(X) - 1$ *	[-1, 1]	20
(9) $X^8 + X^7 + X^6 + X^5 + X^4$	[-1, 1]	20
(10) $X^3 e^{(-X)} \cos(X) \sin(X)(\sin^2(X) \cos(X) - 1)$ *	[0, 10]	200
(11) $\frac{\ln(2)}{X} + \cos(X) \times 7$	[2, 4]	20
(12) $\frac{\sin(X)+\sin(X+X^2)}{(X-2)}$ **	[-1, 1]	20
(13) $0.3X \sin(2\pi X) + X^{10}$ **	[-1, 1]	20

We then turned our focus to an alternate benchmark specification. Instead of directly tuning a landscape of fitness peaks, we allowed the landscape take shape in whatever way was determined by the specification of a single optimum, with the fitness of other points in the landscape determined by their semantic distance to this single optimum. This required us to have a defined idea of semantic distance, and thus necessitated that the benchmark be made for a particular application area: we chose symbolic regression. Our benchmark measured the semantic distance between two functions using mean squared distance. The key issue we wished to investigate was: whether or not it is sensible to control for the degree of change using mean squared distance as a metric for the difference between two search targets.

2. EXPERIMENTAL PROCESS

There were two periods of evolution in our experiments - both were 20 generations long. We used an implementation of standard tree-based GP, which allowed for the target to change after 20 generations. The rate of crossover was 80%, with mutation of 5% and 15% elitism. The population size was 100, and the maximum tree depth possible was ten. The function set comprised the four standard arithmetic operators, as well as the sine, cosine, natural logarithm, and exponential functions. The terminal set consisted of the variable X, and the constant value 1. "Base" target functions refer to the target function in the first period of evolution. In each run, we started with one of 13 base functions. The base target functions used are given in Table 1. Sampled points were evenly spaced in the range. Most of the base functions were either taken directly from (*) or modeled on (**) the functions proposed for (static) symbolic regression in [4].

"Other" functions refer to functions which served as the target function for the second period of evolution. The other functions were randomly generated. We divided the other functions into three buckets, based on mean squared distance from the base function. We ran 100 other target functions from each bucket in the second period. We were not interested in measuring the effects of huge changes in the target - bounds on buckets were chosen so that only functions that weren't very far from the original target were considered (with the degree of closeness varying between the three buckets of functions). Each bucket was labeled with its lower bound (0.25, 0.75 and 1.25), and had a width of 0.1.

There may have been other factors at play - aside from distance from the base function - that impacted how difficult it was for GP to find "other" target functions. These included: the absolute difficulty of finding the function from scratch, neglecting dynamics, and, separately, the convergence present in the population at the end of the first (base) period. The former effect was mitigated by populating each of the three buckets of other functions with a large number (100) of functions. To control for the latter effect, the base function was only run once. The same base population from the final generation of the initial period, was used to seed the populations for each of the 300 target functions in the second period.

Table 2: Experimental Results

Base Target	Better Performance: Continuous or Restart?	Continuous: Closer functions better performance?
1	Continuous	Yes
2	0.25: Continuous 0.75: Restart 1.25: Restart.	Yes
3	Continuous	Yes
4	Continuous	0.25 better than 1.25: Yes 0.25 better than 0.75: V. little diff. 0.75 better than 1.25: Yes
5	Continuous	0.25 better than 1.25: V. little diff. 0.25 better than 0.75: Yes 0.75 better than 1.25: No
6	0.25: Continuous 0.75: V. little difference 1.25: V. little difference	Yes
7	0.25: Continuous 0.75: V. little difference 1.25: Restart	Yes
8	0.25: V. little difference 0.75: Restart 1.25: Restart	Yes
9	0.25: Continuous 0.75: V. little difference 1.25: Restart	0.25 better than 1.25: Yes 0.25 better than 0.75: Yes 0.75 better than 1.25: No
10	Restart	Yes
11	Continuous	No
12	0.25: V. little difference 0.75: Restart 1.25: V. little difference	0.25 better than 1.25: Yes 0.25 better than 0.75: Yes 0.75 better than 1.25: No
13	Restart	Yes

For comparison purposes, we also re-initialized the population when the target function changed. We did this to compare continuous evolution - where the population which had evolved towards the old target was carried forward in the second period - against evolution where the old population was discarded, and evolution was re-started from scratch.

Results are summarized in Table 2. Results are interpreted in terms of mean fitness for the second 20 generations of evolution, averaged over 100 runs for each bucket. We record "very little difference" between two sets of results, where performances are within 10% of each other. We find limited preliminary evidence that semantic distance is a good proxy for ease of adaptation for GP in the dynamic environments described. For nine of the base target functions, functions in closer buckets have better or similar performance in the second period, than functions in the further away buckets. However, in three of these cases (functions 8, 10 and 13), for all buckets, the performance of the restart populations is better than or similar to the performance of the continuous populations. Semantic closeness appears to confer an advantage on closer functions in the second period of continuous evolution in these cases, but, significantly, restarting evolution at generation 20 outperforms continuous evolution. This raises questions as to whether the (generally) superior average performance of functions in the closer buckets, for continuous evolution, is the result of chance.

3. CONCLUSIONS AND FUTURE WORK

Existing EA dynamic benchmarks, such as the moving peaks benchmark, are unsuitable for GP due to the representation-bias towards EAs which use a fixed-length real-valued encoding. As a first step towards creating a GP dynamic benchmark, we investigated whether there was a predictable relationship between the mean squared distance between a first- and second-period target function, and GP performance in the second-period. If this relationship existed, specifying a benchmark using mean squared distance as a measure of environmental change, would be straightforward.

However, we do not find convincing evidence of a relationship between the mean squared distance between first- and second-period functions, and GP performance on the second-period function. The size of the search space seen by GP is huge. This is coupled with the fact that, while search is guided by fitness, it traverses an operator-based fitness landscape. This landscape will not necessarily have nodes which are semantically-close as neighbours (each node in the landscape represents a possible individual). This makes GP search difficult to predict and to analyse. Future work includes examining a wider range of functions. We will experiment with time periods other than 20 generations, and intend on examining a greater variety and range of distances.

4. REFERENCES

[1] J. Branke. Memory enhanced evolutionary algorithms for changing optimization problems. In *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, volume 3. IEEE, 1999.

[2] I. Dempsey, M. O'Neill, and A. Brabazon. Adaptive trading with grammatical evolution. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 2587–2592. IEEE, 2006.

[3] I. Dempsey, M. O'Neill, and A. Brabazon. *Foundations in Grammatical Evolution for Dynamic Environments.* Springer Verlag, 2009.

[4] J. McDermott et al. Genetic programming needs better benchmarks. In *Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference*, pages 791–798. ACM, 2012.

[5] I. Moser and R. Chiong. Dynamic function optimization: The moving peaks benchmark. *Metaheuristics for Dynamic Optimization*, pages 35–59, 2013.

[6] N. Wagner, Z. Michalewicz, M. Khouja, and R. McGregor. Time series forecasting for dynamic environments: the dyfor genetic program model. *Evolutionary Computation, IEEE Transactions on*, 11(4):433–452, 2007.

We gratefully acknowledge the financial support of Science Foundation Ireland under Grant Number 08/SRC/FM1389.