IMPROVED COMPARATIVE PARTNER SELECTION WITH BROOD RECOMBINATION FOR GENETIC PROGRAMMING

Muhammad Waqar Aslam [*] *Zhechen Zhu, Asoke Kumar Nandi* [†]

The University of Liverpool
Liverpool L69 3GJ, UK.
waqaraslam271@gmail.com

Brunel University
Uxbridge, UB8 3PH, UK.
{zhechen.zhu, asoke.nandi}@brunel.ac.uk

ABSTRACT

The aim of all evolutionary methods is to find the best solution from search space without testing every solution in search space. This study employs strengths and weaknesses of solutions for finding the best solution of any problem in genetic programming. The strengths and weaknesses are used to assist in finding the right partners (solutions) during crossover operation. The probability of crossover between two solutions is evaluated using relative strengths and weaknesses as well as overall strengths of solutions (Improved Comparative Partner Selection (ICPS)). The solutions qualifying for crossover through ICPS criteria are supposed to produce better solutions and are allowed to produce more children through brood recombination. The brood recombination helps to exploit the search space close to the optimum solution more efficiently. The proposed method is applied on different benchmarking problems and results demonstrate that the method is highly efficient in exploring the search space.

Index Terms— Diversity in genetic programming, Improved comparative partner selection, Brood recombination

1. INTRODUCTION

Genetic programming (GP) has received a lot of attention in recent years due to its ability to produce human competitive solutions [1–3]. The flexibility in choosing various parameters of the algorithm and the ability to produce human interpretable solutions have made it superior over other evolutionary algorithms. However, despite numerous advantages offered by GP, there are some inherent issues in GP that restrict its performance when applied to complex problems. One of these issues is the premature convergence towards local optimum [4–7]. The main reason for such convergence is believed to be a loss of diversity in fairly fit population of individuals as the population evolves.

Diversity is an important element in explaining the structural and behavioural variation within a population. Although, promoting diversity maintains a wider search space, promoting it blindly has considerable computational cost. Numerous methods have been proposed in the literature for preserving diversity and avoiding premature convergence [8–14]. The concept of fitness sharing was introduced in [8] where the individuals (solutions) sharing similar fitness values were penalised. A multi objective method for promoting diversity, reducing code growth and optimising fitness was presented by Edwin, Watson and Pollack [9]. Day and Nandi [10] evaluated strengths and weaknesses of individuals using response of individuals for each training case. The strengths and weaknesses were utilised to promote crossover between two diverse individuals (comparative partner selection (CPS)). Their experiments on benchmarking problems demonstrated that the method was successful in eliminating population-wide weakness.

The method proposed in this study is an extension of the CPS method presented in [10]. The CPS method promoted crossover between two individuals having strengths and weaknesses in different areas, ignoring the overall strength during crossover. In this study we demonstrate that ignoring overall strength in crossover might neglect potentially more suitable partners. A hybrid approach named as improved CPS, involving CPS method and overall strength of individuals is proposed for finding the right partners during crossover. In addition, since the individuals selected by improved CPS method for crossover are supposed to produce better children than a normal crossover, they are allowed to produce more children (brood recombination [15]). This way the search space near optimum solutions is exploited more efficiently.

2. MEASURING DIVERSITY FOR FINDING OPTIMUM SOLUTION

The diversity of any population can be defined in two ways, structural diversity (genotype diversity) and behavioural di-

[*] Muhammad Waqar Aslam would like to thank the University of Azad Jammu and Kashmir, Pakistan, for their financial Support.

[†] Zhechen Zhu would like to thank the School of Engineering and Design, Brunel University for the financial support. Asoke K. Nandi would like to thank TEKES for their award of the Finland Distinguished Professorship.

978-1-4799-1180-6/13/$31.00 ©2013 IEEE

versity (phenotype diversity). The genotype diversity explores the similarity between actual structures of individuals and has been evaluated in various ways in literature (absolute edit distance, weighted edit distance, etc). The phenotype diversity on the other hand analyses the behaviour of individuals. The most common method for evaluating this diversity is to find the distribution of fitness values in a population.

The phenotype diversity can be further divided into two subcategories; population-wide diversity and pairwise diversity. As the name suggests, the population-wide diversity measures the diversity across an entire population while the pairwise diversity measures the difference between two individuals. In this study, the pairwise phenotype diversity is used to explore the search space for finding the optimum solution.

2.1. Binary String Fitness Characterisation

The idea of binary string fitness characterisation (BSFC) was introduced by Day and Nandi [10] which explains the efficacy of an individual for each input training example. Generally, a number of training examples are provided to GP for solving a problem. In a standard GP algorithm (SGP), the performance of GP individual for solving all training examples is summed up in one parameter, the fitness function. A typical fitness function is a sum of errors for all training examples. Such a fitness function does not give insight into response of a GP individual for each training example and any two individuals solving different training examples may get the same overall fitness value. Although, the overall fitness value is important for evolving towards the optimum solution, the insight into efficacy of individuals for each training example can help to explore search space more efficiently. In BSFC, a binary string (b_i) is attached to each individual with as many bits as training examples. For logical problems where the output is either 1 or 0 and is generally already known, the evaluation of b_i is straightforward. If an individual finds the correct output for a training example, it gets a '1' in the corresponding bit of b_i, otherwise '0'. A '1' can be considered as strength of the individual and a '0' as weakness. If all the training examples are solved, the binary string will consist solely of ones. This binary string gives us insight into the abilities of GP individuals for solving each training example.

The assignment of binary string is not as straightforward for non binary problems. Day and Nandi [10] proposed a mean error based technique for regression problems. For classification problems, a technique based on the mean of output distribution of a class was proposed in [16]. An example of a b_i for a GP tree for 3 bit parity problem (explained in section 4.1) in a sub optimally converged run is shown in Figure 1. On the extreme left hand side are three inputs of 3 bit parity problem and on the extreme right is the target output. GP is expected to take these inputs and combine them with the help of some given functions to get the target output. One of the combinations of inputs and functions found during evolution

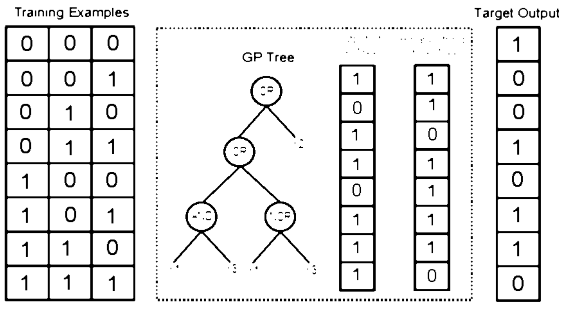

Fig. 1. An example of a GP Tree in a sub optimally converged evolution for 3 bit parity problem.

of GP is shown in the form of a tree in the middle of Figure 1, where X1, X2 and X3 are three inputs. Since it was a sub optimally converged run, GP was unable to find a perfect solution as demonstrated by the output and binary string of the GP tree.

2.2. Comparative Partner Selection

In SGP, the parents for crossover (during creation of new generation) are selected just using their fitness values. This might be limiting as two individuals solving different examples may get same fitness value. In CPS, an additional criteria for parent selection is introduced which is based on binary strings of two parents. This criteria uses simple logical operations to promote the crossover between two diverse (in terms of binary string) individuals. A crossover between two individuals is encouraged if one individual shows strength for an example for which other individual shows weakness (XOR) and a crossover is discouraged if two individuals share similar weaknesses (NOR). The binary string can be used to calculate the probability of crossover (P_{cps}) between two individuals.

$$P_{cps}(b_1, b_2) = \frac{\sum XOR(b_1, b_2)}{\sum XOR(b_1, b_2) + \sum NOR(b_1, b_2)} \qquad (1)$$

where P_{cps} is the probability of crossover and b_1 and b_2 are the binary strings of two individuals. The probability of crossover between any two similar individuals (in terms of binary string) will be lower compared to diverse individuals.

The process of CPS follows these steps: two parents (p_1 and p_2) are selected based on fitness value and their probability of crossover P_{cps} is calculated. A random number between 0 and 1 is generated and if P_{cps} is greater than this number, crossover takes place otherwise not (this is to include stochasticity in the method). If the crossover does not take place, a new second parent (p_2) is selected (without changing p_1) and the above procedure is repeated. If a suitable partner is not found after N/2 attempts (N is population size), p_2 is chosen randomly, ignoring CPS criteria. In order to penalize

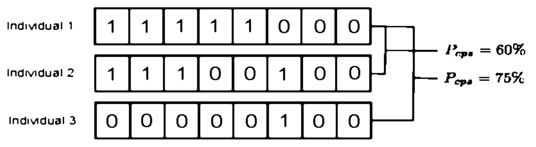

Fig. 2. An example showing CPS flaw.

the CPS for not finding a suitable partner, the probability of crossover is decreased by 1/N and the probability of mutation is increased by 1/N, in the current generation. In the next generation, these probabilities go back to their initial values.

3. THE PROPOSED METHOD

3.1. Improved Comparative Partner Selection

In the CPS process, the probability of crossover calculated using equation (1) favours the crossover between two individuals where strengths of one individual coincide with weaknesses of other individual. During this process the strengths of one individual coinciding with strengths of other individual are ignored. Authors believe it is an important aspect and can enhance the overall strength of new individuals being produced. An example is used to describe the importance of this aspect.

Figure 2 shows the binary strings of three individuals for 3 bit parity problem. The first individual solves five training examples, the second solves four examples while the last individual solves only one example. Imagine individual 1 has to chose one of the other two individuals for crossover. Looking at Figure 2, a crossover between individual 1 and 2 or between individual 1 and 3 can produce a child, potentially capable of solving first six examples. Since potentially both individuals (2 and 3) can produce an equally capable child (when crossed with 1), its just a matter of finding out which one is better out of the two. If we consider the two individuals alone (ignoring crossover), individual 2 looks like a better choice than individual 3 since it solves four examples while individual 3 solves only one.

CPS uses equation (1) for finding the right partner for crossover. According to equation (1), the probability of crossover (P_{cps}) between individual 1 and individual 2 is 60% while P_{cps} between individual 1 and 3 is 75%. This is because for calculating a probability of crossover, the CPS method considers only those examples where at least one of the individuals is weak and ignores the examples where both individuals are strong. The examples for which the two individuals share strengths are important in defining the overall strength of newly produced child and should not be ignored. A new parameter (β) has been introduced in this study which takes into account the shared strengths of two individuals. The parameter β is defined as

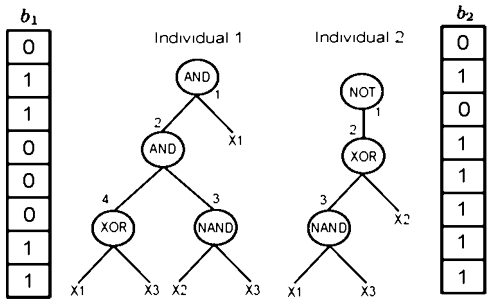

Fig. 3. Two individuals selected by CPS for crossover.

$$\beta(b_1, b_2) = \frac{\sum AND(b_1, b_2)}{\sum AND(b_1, b_2) + \sum NAND(b_1, b_2)} \tag{2}$$

The parameter β finds the percentage of shared strengths between two individuals. The probability of crossover in our proposed improved comparative partner selection (ICPS) method is an average of P_{cps} and β, and can be calculated as

$$P_{icps}(b_1, b_2) = \frac{P_{cps} + \beta}{2} \tag{3}$$

This new probability P_{icps} takes into account both, the examples where both individuals are strong and the examples where atleast one of the individual is weak. The new probability (P_{icps}) between individual 1 and 2 is 49% and between individual 1 and 3 is 37.5%.

3.2. Brood Recombination

The probability of crossover P_{icps} is used to decide whether two individuals qualify for a crossover or not. If they qualify for crossover, the following actions are performed. A node is randomly chosen on each individual and the sub branch downwards from that node is swapped with each other. Each individual may have many crossover points and choosing only one point for crossover may ignore some potentially good solutions. Since the individuals selected for crossover by ICPS process are supposed to produce better solutions, the search space close to these individuals should be explored more efficiently.

This fact is demonstrated by an example. Figure 3 shows two trees (with their respective binary strings) generated by GP for solving 3-bit parity problem and the inputs and target for these trees is same as given in Figure 1. Imagine these two trees are selected for crossover by ICPS process for 3 bit parity problem. Looking at the binary strings, there are four examples where strength of one individual coincides with weakness of other individual. According to CPS process, the children produced by crossover of these two individuals should have strengths for these four examples. Suppose a crossover takes place at node 3 of both individuals and two

Table 1. Parameters used for the experimental work

Parameter	Standard Value
Generations	100
Population Size	100
Function Pool	
3-bit parity	AND, OR, NAND, NOR
5-bit parity	AND, OR, NAND, NOR, XOR
Multiplexer	AND, OR, NOT, IF
Regression	+, -, x, sin, cos, log
Operators	Crossover, Mutation (60%, 40%)

children are produced. These children when evaluated for 3-bit parity training examples (given in Figure 1) will have these binary strings 01001011 and 01100011. These binary strings tell us that the assumption that during crossover of two parents, a strength coinciding with a weakness should result in a strength in the child may not necessarily be true. The reason is that it is difficult to establish which part of the individual is responsible for a certain strength. If a crossover takes place at node 4 (individual 1) and node 3 (individual 2), one of the new individual produced by this crossover is a perfect solution and has binary string consisting solely of ones.

The above discussion clearly shows that choosing one crossover point may ignore potentially good solutions. In this study the individuals selected by ICPS are allowed to produce ten children (instead of two), each time with a different crossover point. The best two children (in terms of fitness) out of ten are selected and the rest are discarded. This way the search space close to ICPS selected parents is exploited more efficiently. This idea of allowing certain parents to produce more children is called brood recombination (BR) [15].

4. EXPERIMENTS AND RESULTS

4.1. Problems Used and GP Parameter Settings

In this study, four benchmarking problems have been used to compare the proposed method with existing methods. These problems belong to two different domains, logical (3-bit and 5-bit even parity, 11-bit multiplexer) and regression (quartic polynomial). The details about these problems is given in [10]. The fitness function for all the problems is the sum of errors. Different GP parameters used for these problems are given in Table 1.

4.2. Results

Figure 4 shows a comparison between SGP, CPS and the proposed method (ICPS with BR) for all the problems. Experiments for each method and for each problem were run 100 times, and the average fitness of best individuals is shown in the Figure where a lower fitness means a better individual. It is clear from the Figure that the proposed method outperforms

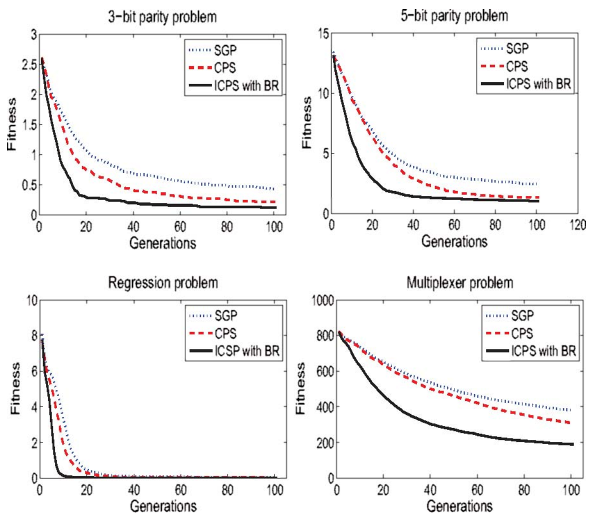

Fig. 4. Fitness comparison for 100 generations averaged over 100 runs.

Table 2. Time taken for 100 runs in hours

Problem/Method	SGP	CPS	ICPS with BR
3-bit parity	6	5	4
5-bit parity	17	9	12
Regression	13	8	13
11-bit multiplexer	8	5	19

the other two methods for all the problems and the difference in performance is more evident for more complex problems (5-bit parity and 11-bit multiplexer).

The number of optimum solutions found by SGP, CPS and ICPS with BR were {53, 74, 88} for 3-bit parity problem, {37, 54, 64} for 5-bit parity problem, and {88, 90, 97} for regression problem. For multiplexer problem, only ICPS with BR was able to find 9 optimum solutions, while the other two methods failed to find any optimum solution. It is clear from this discussion that ICPS with BR performs better than the other two methods.

The complexity comparison in terms of time is given in Table 2. For 3-bit parity problem the time taken by the proposed method is less than the other two methods. Since for this problem the proposed method finds the optimum solution for most of the runs very early in the run, it takes less time than other methods. The time taken by the proposed method for other problems is either equal to or less than the other methods except for multiplexer problem where it takes more time. The main reason for this increase in time is the choice of function pool. Since a simple function pool was chosen for multiplexer problem, the search for the optimum solution became difficult. If the function pool was richer, the probability of finding the optimum solution would increase, causing a reduction in the time taken. The CPS method takes less time than other methods for most of the problems due to the fact that it increases probability of mutation once it fails to find suitable partners and the complexity of mutation is almost half of crossover. The ICPS process on the other hand, finds suitable partners most of the time for crossover due to the β factor and does not allow mutation probability to increase that much.

In a nutshell, the proposed method performs better than other methods in terms of fitness values and the number of optimum solutions produced. The time taken by the proposed method is higher than the other methods but it can be reduced by using a richer function pool. Our future work will focus on adopting various techniques to reduce the training time of the proposed algorithm for complex problems.

5. CONCLUSIONS

This paper proposes a method for guiding the search towards the optimum solution in genetic programming. The relative and overall strengths of solutions are exploited for promoting more fruitful crossover operation. The partners found using the proposed method are allowed to produce more children since they are expected to be close to the optimum solution. The results demonstrate that the performance of the proposed method is much better than traditional methods.

6. REFERENCES

[1] J. R. Koza, "Human-competitive machine invention by means of genetic programming," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, vol. 22, no. 3, pp. 185–193, 2008.

[2] H. Iba, Y. Hasegawa, and T. K. Paul, *Applied Genetic Programming and Machine Learning*, CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 2009.

[3] R. Poli, W. B. Langdon, and N. F. McPhee, *A field guide to genetic programming*, `http://lulu.com`, 2008.

[4] R. I. Mckay, "Fitness sharing in genetic programming," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2000, pp. 435–442.

[5] T. F. Bersano-Begey, "Controlling exploration, diversity and escaping local optima in gp: Adapting weights of training sets to model resource consumption," in *Late Breaking Papers at the 1997 Genetic Programming Conference*, 1997, pp. 7–10.

[6] V. Ciesielski and D. Mawhinney, "Prevention of early convergence in genetic programming by replacement of similar programs," in *Proceedings of the 2002 Congress on Evolutionary Computation (CEC)*, 2002, vol. 1, pp. 67–72.

[7] D. Mawhinney, "Preventing early convergence in genetic programming by replacing similar programs," in *Proceedings of the Congress On Evolutionary Computation*, 2000, pp. 67–72.

[8] D. E. Goldberg and J. Richardson, "Genetic algorithms with sharing for multimodal function optimization," in *Proceedings of the Second International Conference on Genetic Algorithms and their applications*, 1987, pp. 41–49.

[9] D. Edwin, R. A. Watson, and J. B. Pollack, "Reducing bloat and promoting diversity using multi-objective methods," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 2001, pp. 11–18.

[10] P. Day and A. K. Nandi, "Binary string fitness characterization and comparative partner selection in genetic programming," *IEEE Transaction on Evolutionary Computation*, vol. 12, pp. 724–735, 2008.

[11] E. Burke, S. Gustafson, and G. Kendall, "Diversity in genetic programming: An analysis of measures and correlation with fitness," *IEEE Transaction on Evolutionary Computation*, vol. 8, pp. 47–62, 2004.

[12] E. Burke, S. Gustafson, G. Kendall, and N. Krasnogor, "Advanced population diversity measures in genetic programming," in *Parallel Problem Solving from Nature (PPSN VII)*, vol. 2439, pp. 341–350. 2002.

[13] W. B. Langdon, "How many good programs are there? how long are they?," in *Proceedings of Foundations of Genetic Algorithms*, 2003, pp. 183–202.

[14] J. M. Daida, D. J. Ward, A. M. Hilss, S. L. Long, M. R. Hodges, and J. T. Kriesel, "Visualizing the loss of diversity in genetic programming," in *IEEE Congress on Evolutionary Computation (CEC)*, 2004, vol. 2, pp. 1225–1232.

[15] W. A. Tackett, *Recombination, Selection and the Genetic Construction of Computer Programs.*, Ph.D. thesis, University of Souithern Califomia, Department of Electrical Engineering Systems, 1994.

[16] M. W. Aslam and A. K. Nandi, "Detection of diabetes using genetic programming," in *Proceedings of the 18th European Signal Processing Conference (EUSIPCO)*, Aug. 2010, pp. 1184–1188.