Controlling the Genetic Programming Search

Emin Erkan Korkmaz
Department of Computer Engineering
Middle East Technical University
Ankara-Turkey
korkmaz@ceng.metu.edu.tr

Göktürk Üçoluk
Department of Computer Engineering
Middle East Technical University
Ankara-Turkey
ucoluk@ceng.metu.edu.tr

Traditional GP randomly combines subtrees by applying crossover. In this study a new approach is presented for guiding the recombination process. Our method is based on extracting the global information of the promising solutions that appear during the genetic search. The aim is to use this information to control the crossover operation afterwards. [1] proposes a method based on calculating the performance values for subtrees of a GP tree during evolution and then applying recombination so that the subtrees with high performance are not disturbed. The aim is to control recombination by determining the building blocks. However for the deceptive class of problems focusing on the building blocks is questionable. The interaction between the partial solutions is high for these problems. Hence it is difficult to determine isolated building blocks.

The frequency information of the elements and their distribution in the tree have been used to determine the global information of a GP tree. The information is represented as a vector. In order to transform a GP tree to a vector, the elements are mapped to base vectors first and a bottom up construction is used to obtain a single vector for the whole tree. A leaf node is only mapped to its base vector while the vector for an internal node is obtained by adding the vectors of its children plus the base vector corresponding to it. Note that the dimension of the vector is the total number of elements used for the problem at hand. However this formalization would enable us to hold only the frequency information. To represent the distribution information too, the depth knowledge is used. This new information is represented as a fractional value to make a distinction with the frequency information. For example if $X(X_1, X_2)$ is a subpart of a three, then the vector corresponding to X is determined as:

$$V_X = V_{X_1} + V_{X_2} + V_{X_{base}} + V_{X_{base}} * 0.01 * depth(X)$$

A *Control Module* is designed to process this global information. The genetic search is started and for each chromosome the corresponding vector is formed. The control module collects the vectors and fitnesses for a certain period of generations, which we call the learning period. Then *"C4.5, Decision Tree Generator"* is used to generate an abstraction over the data collected. Then for each crossover, the genetic engine sends to the control module three different alternative crossover points. The control module predicts if the alternative offsprings will be in the positive or the negative class. Certainly the best alternative is chosen and the learning process is repeated at the end of each learning period. CFG induction and the N-Parity Problem have been selected as the testbeds. Both of them can be considered as highly deceptive. Figure 1 presents the progress obtained for the N-Parity problem. The results denote the average best fitness change for basic GP and for our method. Eight different runs are used to calculate the averages. An improvement has been achieved for the CFG induction problem too.

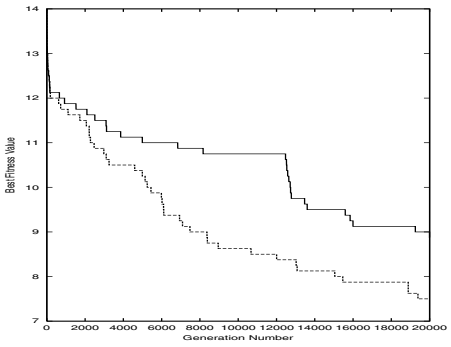

Figure 1: The dashed lines denote the performance of controlled search. Learning period is 500.

References

[1] Hitoshi Iba and Hugo de Garis, *Extending Genetic Programming with Recombinative Guidance*, In P. Angeline and K. E. Kinnear, Jr., editors, Advances in Genetic Programming 2, chapter 4. MIT Press, Cambridge, MA, USA, 1996.