IDEA GROUP PUBLISHING

701 E. Chocolate Avenue, Suite 200, Hershey PA 17033-1240, USA ITB10416
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com

Chapter V

Gene Expression Programming and the Evolution of Computer Programs

Cândida Ferreira, Gepsoft, UK

ABSTRACT

In this chapter an artificial problem solver inspired in natural genotype/phenotype systems — gene expression programming — is presented. As an introduction, the fundamental differences between gene expression programming and its predecessors, genetic algorithms and genetic programming, are briefly summarized so that the evolutionary advantages of gene expression programming are better understood. The work proceeds with a detailed description of the architecture of the main players of this new algorithm (chromosomes and expression trees), focusing mainly on the interactions between them and how the simple yet revolutionary structure of the chromosomes allows the efficient, unconstrained exploration of the search space. And finally, the chapter closes with an advanced application in which gene expression programming is used to evolve computer programs for diagnosing breast cancer.

EVOLUTIONARY ALGORITHMS IN PROBLEM SOLVING

The way nature solves problems and creates complexity has inspired scientists to create artificial systems that learn by themselves how to solve a particular problem. The first attempts were done in the 1950s by Friedberg (1958; Friedberg et al., 1959), but ever

since highly sophisticated systems have been developed that apply Darwin's ideas of natural evolution to the artificial world of computers and modeling. Of particular interest to this work are the Genetic Algorithms (GAs) and the Genetic Programming (GP) technique, as they are the predecessors of Gene Expression Programming (GEP), the most recent development in evolutionary computation and the theme of this chapter. A brief introduction to these three techniques is given below.

Genetic Algorithms

Genetic algorithms were invented by John Holland in the 1960s and they also apply biological evolution theory to computer systems (Holland, 1975). Like all evolutionary computer systems, GAs are an oversimplification of biological evolution. In this case, solutions to a problem are usually encoded in strings of 0s and 1s (chromosomes), and populations of such strings (individuals or candidate solutions) are used in order to evolve a good solution to a particular problem. From generation to generation candidate solutions are reproduced with modification and selected according to fitness. Modification in the original genetic algorithm was introduced by the genetic operators of mutation, crossover, and inversion.

It is worth pointing out that GAs' individuals consist of naked chromosomes or, in other words, GAs' individuals are simple replicators. And like all simple replicators, the chromosomes of genetic algorithms function simultaneously as genotype and phenotype: they are both the object of selection and the guardians of the genetic information that must be replicated and passed on with modification to the next generation. Consequently, the whole structure of the replicator determines the functionality and, therefore, the fitness of the individual. For instance, in such systems it would not be possible to use only a particular region of the replicator as a solution to a problem: The whole replicator is always the solution: nothing more, nothing less.

Genetic Programming

Genetic programming, invented by Cramer in 1985 (Cramer, 1985) and further developed by Koza (1992), solves the problem of fixed length solutions through the use of nonlinear structures (parse trees) with different sizes and shapes. The alphabet used to create these structures is also more varied, creating a richer, more versatile system of representation. Notwithstanding, the created individuals also lack a simple, autonomous genome. Like the linear chromosomes of genetic algorithms, the nonlinear structures of GP are also cursed with the dual role of genotype/phenotype.

The parse trees of genetic programming resemble protein molecules in their use of a richer alphabet and in their complex and unique hierarchical representation. Indeed, parse trees are capable of exhibiting a great variety of functionalities. The problem with these complex replicators is that their reproduction with modification is highly constrained in evolutionary terms because the modifications must take place on the parse tree itself and, consequently, only a limited range of modification is possible. Indeed, special kinds of genetic operators were developed that operate at the tree level, modifying or exchanging particular branches between trees.

Although at first sight this might appear advantageous, it greatly limits this technique (we all know the limits of grafting and pruning in nature). Consider for instance crossover, the most used and often the only search operator used in genetic program-