A STUDY OF CROSSOVER OPERATORS IN GENETIC PROGRAMMING

William M. Spears
Navy Center for Applied Research in AI
Naval Research Laboratory, Code 5510
Washington, D.C. 20375-5000
and
Vic Anand
Massachusetts Institute of Technology
Cambridge, MA 02139

Abstract

Holland's analysis of the sources of power of genetic algorithms has served as guidance for the applications of genetic algorithms for more than 15 years. The technique of applying a recombination operator (crossover) to a population of individuals is a key to that power. Nevertheless, there have been a number of contradictory results concerning crossover operators with respect to overall performance. Recently, for example, genetic algorithms were used to design neural network modules and their control circuits. In these studies, a genetic algorithm without crossover outperformed a genetic algorithm with crossover. This report re-examines these studies, and concludes that the results were caused by a small population size. New results are presented that illustrate the effectiveness of crossover when the population size is larger. From a performance view, the results indicate that better neural networks can be evolved in a shorter time if the genetic algorithm uses crossover.

1. Introduction

Recently, two heuristic search techniques have generated interest in the artificial intelligence community: genetic algorithms (GAs) and neural networks (NNs). Both GAs and NNs are based on models from nature. A genetic algorithm is modeled on genetics and Darwinian evolution, whereas a neural network is based on models of human cognition. One common application of a genetic algorithm is as a function optimizer. Another common application of a genetic algorithm is to evolve organisms that perform well in a given environment. In either application, the GA is based on the survival-of-the-fittest (natural selection) tenet of
Darwinian evolution. Neural networks, on the other hand, appear to be useful as control mechanisms for organisms themselves (e.g., an organism should avoid danger and seek food). These two methods naturally reflect a difference in scale. While a neural network can be used to control a particular organism, a genetic algorithm can be used to evolve a population of organisms (e.g., NNs) that perform well in a given environment. If a neural network is used to encapsulate a particular behaviour, then genetic algorithms can be used to evolve that behaviour, by evolving a population of neural networks.

One particular approach to the evolution of behaviour is described by de Garis [1]. In this approach, a GA is used to evolve a population of neural networks. Each NN has a set of adjustable weights and is used to encapsulate some desired behaviour (e.g., walking). In other words, once good weights have been found, the NN can be used by itself to perform the desired behaviour. However, since a set of good weights is not known in advance, they must be learned. Instead of the more traditional NN learning algorithms (e.g., backpropagation) de Garis uses a genetic algorithm to learn a set of good weights. No learning is being done by the neural network itself. This approach is called genetic programming [1].

As mentioned above, GAs evolve a population of individuals according to the process of natural selection. During this process, genetic operators create new (child) individuals from highly fit old (parent) individuals. Recombination (also referred to as crossover in this report) is one of the genetic operators and is a key to the power of the genetic algorithm [6]. In his studies of genetic programming, though, de Garis reports that a genetic algorithm without recombination outperforms a genetic algorithm with recombination. These results motivated us to re-examine genetic programming for two reasons. First, from a theoretical standpoint, we sought to explain what appear to be anomalous findings. Second, from a practical standpoint, we wished to use recombination (as theory suggests) to improve de Garis’s results, allowing better behaviour to be learned in less time.

2. Genetic Algorithms

The book "Adaptation in Natural and Artificial Systems" [6], lays the groundwork for GAs. A genetic algorithm consists of a population of individuals that reproduce (over many generations) according to their fitness in an environment. Those individuals that are most fit are most likely to survive, mate, and bear children. Children are created by the stochastic application of genetic operators to the (parent) individuals. Individuals of the population, coupled with the genetic operators, combine to perform an efficient domain-independent search strategy that makes few assumptions about the search space.