Chapter IV

Evolutionary Algorithms

Pierre Collet

Université du Littoral Côte d’Opale, France

ABSTRACT

Evolutionary computation is an old field of computer science that started in the end of the 1960s nearly simultaneously in different parts of the world. Each paradigm has evolved separately, apparently without knowledge of what was happening elsewhere, until people finally got together and shared their experience. This resulted in strong trends that still survive, even though it is now possible to outline a generic structure for an evolutionary algorithm that is described in this chapter.

INTRODUCTION AND HISTORY

The development of evolutionary algorithms almost dates back to the dark ages of computers. To put back everything in perspective, Computer Science really started when John von Neumann designed the EDVAC (electronic discrete variable automatic computer) in 1945, but the first prototype was actually implemented in 1949 with Wilkes’ EDSAC (electronic delay storage automatic calculator). Then, for a while, the only commercially available machines used valves and were therefore not that reliable (IBM 650 in 1953). A quantum leap was made when transistors became available around the 1960s, and finally, Integrated Circuits in 1964.

By that time, evolutionary computation had about ten independent beginnings in Australia, the United States, and Europe, starting in 1953, traced by David Fogel’s excellent *Fossil Record* (Fogel, 1998): Alex Fraser had evolved binary strings using crossovers (Fraser, 1957), Friedberg had already thought of self-programming computers through mutations (Friedberg, 1958; Friedberg, Dunham, & North, 1958), and Friedman of how evolution could be digitally simulated (Friedman, 1959). However, the main evolutionary trends that survived are as follows:

- **Evolutionary Strategies:** By Rechenberg and Schwefel, best described in Rechenberg (1973) and Schwefel (1995).
Evolutionary Algorithms

- **Genetic Algorithms:** By Holland, later popularized by Goldberg on the U.S. East Coast (Michigan) (Holland, 1975; Goldberg, 1989).
- **Evolutionary Programming:** By Lawrence Fogel and later David Fogel on the U.S. West Coast (Fogel, Owens, & Walsh, 1966; Fogel, 1992).
- **Genetic Programming:** By Cramer (1985) and later developed by Koza (1992) (see Chapter V).

Evolutionary computation cannot, therefore, be seen as a recent development of computer science, or even classified as artificial intelligence, which is a different concept that also started back in the mid-1950s with John McCarthy and many others.

However, until the principles of evolutionary computation were clearly understood, these techniques necessitated a larger amount of computer power than was available until the beginning of the 1990s.

Thus although evolutionary computation really started in the late 1960s, it only came of age when computers had enough power to make them a technique competitive with other (posterior) stochastic optimization paradigms such as simulated annealing (Kirkpatrick, Gellat, & Vecchi, 1983) or tabu search (Glover, 1977, 1989, 1990) (see Chapter III).

### A Unified Evolutionary Algorithm

Kenneth DeJong has been giving a GECCO tutorial on the unification of evolutionary algorithms for several years now and has come up with a recent book on the subject (DeJong, 2005). Indeed, the previously quoted currents (evolutionary strategies, genetic algorithms, evolutionary programming, genetic programming) all share the same principles copied from natural selection.

Rather than describing each algorithm, this chapter will describe a generic and complete version that can emulate virtually any paradigm, depending on chosen parameters.

### Representation of Individuals

Due to the similarities between artificial evolution and natural evolution that was the source of its inspiration, a good part of the vocabulary was borrowed from biologists. In artificial evolution, a potential solution to a problem is called an *individual*.

Using a correct representation to implement individuals is a very essential step that is trivial for some kinds of problems and much less trivial for others. The American trend (genetic algorithms) advocates using a representation that is as generic as possible—for example, a bit string...