Parallel Models for Genetic Algorithms

The whole is more than the sum of the parts.

Aristotle (384 - 322 BC) - Greek philosopher

In the previous chapter we offered a brief introduction to metaheuristics. Now, this chapter is devoted to genetic algorithms (GA) and their parallel models. GAs [4, 20] are stochastic search methods designed for exploring complex problem spaces in order to find optimal solutions, possibly using information of the problem to guide the search. Unlike most other optimization (search, learning) techniques, a population of multiple structures is used by GAs to perform the search along many different areas of the problem space at the same time. The structures composing the population (individuals) encode tentative solutions, which are manipulated competitively by applying them some stochastic operators to find a satisfactory, if not globally, optimal solution.

Algorithm 2. Pseudocode of a canonical GA.

1: \( P \leftarrow \text{GenerateInitialPopulation()} \)
2: \( \text{Evaluate}(P) \)
3: \textbf{while not Termination\_Condition()} \textbf{do}
4: \( P' \leftarrow \text{SelectParents}(P) \)
5: \( P' \leftarrow \text{Recombination}(P') \)
6: \( P' \leftarrow \text{Mutation}(P') \)
7: \( \text{Evaluate}(P') \)
8: \( P \leftarrow \text{SelectNewPopulation}(P,P') \)
9: \textbf{end while}
10: \textbf{Return} The best solution found;

In Algorithm 2 the outline of a classical GA is described. A GA proceeds in an iterative way by successively generating a new population \( P(t) \) of individuals from \( P(t-1) \), the previous one \((t = 1,2,3,\ldots)\). The initial population \( P(0) \) is generated randomly. A fitness function associates a value to every individual, which is representing its suitability to the problem in hands. The
canonical algorithm applies stochastic operators such as selection, recombination, and mutation on a population in order to compute a whole generation of new individuals. In a general formulation, variation operators are applied to create a temporary population $P'(t)$, whose individuals are evaluated; then, a new population $P(t+1)$ is obtained by using $P'(t)$ and, optionally, $P(t)$. In GAs, these variation operators are typically recombination and mutation. The stop criterion is usually set as reaching a preprogrammed number of iterations of the algorithm, and/or to find an individual with a given error if the optimum, or an approximation to it, if known beforehand.

For nontrivial problems, the execution of the reproductive cycle of a simple GA may require high computational resources (e.g., large memory and long search times), and thus a variety of algorithmic issues has been studied to design efficient GAs. For this goal, numerous advances are continuously being achieved by designing new operators, hybrid algorithms, termination criteria, and so on [8]. In this chapter, we address one such improvement, consisting in adding parallelism to GAs. In the field of parallel GAs (pGAs) [9], there exists a large number of implementations and algorithms. The reasons of this success have to do first, with the fact that GAs are naturally prone to parallelism, since most variation operators can be easily undertaken in parallel, and second, that using a pGA often takes us not only to use a faster algorithm, but also to get a superior numerical performance [27, 30]. Parallel GAs are characterized by the use of a structured population (a spatial distribution of individuals), either in the form of a set of islands [31] or a diffusion grid [32], which is the responsible of such benefits. As a consequence, many authors do not use a parallel machine at all to run structured-population models, and still get better results than with serial traditional GAs [30].

The goal of this chapter is to give a modern classification of the different models and implementations concerning parallel GAs, a field of the evolutionary computation (EC) discipline. In addition, we will empirically test the behavior of some of the most important proposed models, providing to the community what we hope to be a useful baseline comparison for other researchers among the main models for parallelizing GAs existing in the literature.

This chapter is organized as follows. First, a description of the standard model of GA, in which the whole population is considered as a single pool of individuals, is given. In the next section, we address the structured models, in which the population is decentralized somehow. Later, some different implementations of parallel GAs are presented, and a pGA classification is given. In Section 2.4, we test and compare the behavior of several parallel models when solving an instance of the well-known MAXSAT problem. Finally, we summarize our most important conclusions.