Using Metaheuristics as Soft Computing Techniques for Efficient Optimization

Sergio Nesmachnow
Universidad de la República, Uruguay

INTRODUCTION

Nowadays, many real-world problems are intrinsically complex. Optimizing a (set of) function(s) is a complex problem that underlies in many scientific, academic, and commercial applications. In practice, many optimization problems are NP-hard. They are difficult to solve, because of having large solution spaces, complex functions, hard constraints, and/or managing large volumes of data, as the dimension of the search space significantly grows when increasing the input data. Traditional exact techniques (enumeration/backtracking/dynamic programming), are often not useful in practice for solving these hard problems, as they demand very large execution times when solving realistic instances.

Metaheuristics are soft computing methods that allow computing good-quality solutions, for hard-to-solve problems in reasonable execution times. As soft computing techniques, metaheuristics are conceived to take advantage of tolerating imprecision/uncertainty/approximation in the problem data, and they can handle partial information and non-exact solutions in order to compute accurate solutions for optimization problems. These features make metaheuristics highly valuable techniques, as they allow meeting realistic resolution times in many application areas, from informatics to industrial and commercial.

Metaheuristics advanced steadily in the last twenty-five years, and nowadays the field is consolidated. Many metaheuristic proposals and advances regarding theoretical models and applications have been formulated. This article provides an insight into the main concepts, advances, and results in the field of metaheuristics as soft computing techniques for efficiently solving optimization problems. It presents a general view of well-known metaheuristics and the main applications in nowadays real-world problems from several domains. Finally, the main current and future research lines in the field are also summarized.

BACKGROUND

Metaheuristics are high-level soft computing strategies that define algorithmic frameworks and techniques to find approximate solutions for optimization problems (Blum & Roli, 2003; Talbi, 2009). Many efficient and accurate metaheuristics have been proposed, which can be applied to solve a variety of optimization problems underlying many applications in science/technology, industry, and commerce.

Metaheuristics Concepts: Definition and Classification

Metaheuristics were originally proposed as high-level problem-independent strategies to coordinate several heuristic search methods, which can be instantiated to solve hard problems. This definition has broadened to include a wide range of search and learning processes (shaking, construction/deconstruction, adaptation, swarming/collective behavior, hybridization, etc.) applied to improve the search.

Metaheuristics are conceived to apply the heuristic components in an intelligent way, providing accurate and balanced methods for diversification and intensification. These two concepts are crucial for any successful search method: diversification refers to achieving a good exploration pattern for the search space, providing a reasonable coverage and avoiding stagnation in local optima, while intensification means exploiting accurate solutions to increase their quality. By providing different trade-offs between diversification and intensification, metaheuristics have emerged as robust methods for optimization.

DOI: 10.4018/978-1-4666-5888-2.ch727

Regarding the number of solutions handled, metaheuristics are classified in trajectory and population based. Trajectory metaheuristics work with a single solution, which is replaced by another (often the best) solution found in its neighborhood. The search is characterized by a trajectory in the space of solutions. Trajectory metaheuristics allow quickly exploiting solutions, thus they are referred as intensification-oriented methods. Population metaheuristics work with a set of candidate solutions, which are modified and/or combined following some common guidelines. Some solutions in the population are replaced by newly generated solutions (often the best). These methods are characterized as diversification-oriented.

Other classification criteria for metaheuristics are related to the search strategy, memory vs. memory-less, and dynamic vs. static objective function.

Metaheuristic Techniques

Trajectory metaheuristics include well-known techniques from the early years of this area:

- **Simulated Annealing (SA):** Inspired on the annealing process of metals, probabilistically allows moving to a solution with worst objective function value, trying to escape from local optima (Kirkpatrick et al., 1983). SA was the first metaheuristic proposed, although the term "metaheuristic" was not known in those years.
- **Tabu Search (TS):** Enhances a local search strategy by using memory to store information about visited solutions. To promote diversification, returning to recently visited solutions is not allowed. TS was introduced by Glover (1986), who first used the term "metaheuristic."
- **Greedy Randomized Adaptive Search Procedure:** A constructive metaheuristic proposed by Feo and Resende (1989; 1995), which greedily selects components to construct a solution, and then applies a local search to improve it.
- **Variable Neighborhood Search:** Local search techniques based on using different neighborhood structures during the search (Mladenovic & Hansen, 1997).
- **Iterated Local Search:** Uses a hill-climbing to find local optima and a stochastic perturbation/restart strategy to prevent the search getting stuck in local optima (Lourenço et al., 2002).

Population metaheuristics include techniques that use cooperation in the search:

- **Evolutionary Computation (EC):** Emulates the evolution of species in nature. EC applies stochastic operators (recombination and random changes) on a population, guided by a selection-of-the-best technique to find high-quality solutions. EC includes Genetic Programming (Koza, 1992), Evolution Strategies, and Evolutionary Algorithms (EAs) (Goldberg, 1989).
- **Swarm Intelligence (SI):** Based on the collective behavior of self-organized systems, uses agents to perform explorations while they interact with neighbors and the environment. Although agents have limited search capabilities, an "intelligent" pattern emerges in large swarms. SI include Ant Colony Optimization (ACO), emulating the behavior of ants foraging for food (Dorigo, 1992), Particle Swarm Optimization (PSO), simulating flocks of birds (Kennedy & Eberhart, 1995), Artificial Immune System (DasGupta, 1998), and the recently proposed Bacterial Foraging, Fish/Glowworm Swarm, Firefly Algorithm, Cuckoo Search, and Bee Algorithms.
- **Evolutionary-Inspired Metaheuristics:** Apply an evolutionary search, with different features. It includes Estimation of Distribution Algorithms, which learn by building/sampling probabilistic models of promising solutions (Larrañaga & Lozano, 2002); Scatter Search/Path Relinking, applying an EC-based approach with a specific recombination and ideas from TS (Glover, 1999); Differential Evolution, specialized for optimizing real functions (Storn & Price, 1997); Harmony Search (HS), emulating the improvisation of musicians (Geem et al., 2001); and Memetic Algorithms, including