Chapter 1
Introduction

This chapter provides a basic introduction to optimization methods, defining their main characteristics. This chapter provides a basic introduction to optimization methods, defining their main characteristics. The main objective of this chapter is to present to metaheuristic methods as alternative approaches for solving optimization problems. The study of the optimization methods is conducted in such a way that it is clear the necessity of using metaheuristic methods for the solution of engineering problems.

1.1 Definition of an Optimization Problem

The vast majority of image processing and pattern recognition algorithms use some form of optimization, as they intend to find some solution which is "best" according to some criterion. From a general perspective, an optimization problem is a situation that requires to decide for a choice from a set of possible alternatives in order to reach a predefined/required benefit at minimal costs [1].

Consider a public transportation system of a city, for example. Here the system has to find the "best" route to a destination location. In order to rate alternative solutions and eventually find out which solution is "best," a suitable criterion has to be applied. A reasonable criterion could be the distance of the routes. We then would expect the optimization algorithm to select the route of shortest distance as a solution. Observe, however, that other criteria are possible, which might lead to different "optimal" solutions, e.g., number of transfers, ticket price or the time it takes to travel the route leading to the fastest route as a solution.

Mathematically speaking, optimization can be described as follows: Given a function $f : S \rightarrow \mathbb{R}$ which is called the objective function, find the argument which minimizes f:

E. Cuevas et al., *Advances in Metaheuristics Algorithms: Methods and Applications*, Studies in Computational Intelligence 775,
https://doi.org/10.1007/978-3-319-89309-9_1

$$x^* = \arg\min_{x \in S} f(x) \tag{1.1}$$

S defines the so-called solution set, which is the set of all possible solutions for the optimization problem. Sometimes, the unknown(s) x are referred to design variables. The function f describes the optimization criterion, i.e., enables us to calculate a quantity which indicates the "quality" of a particular x.

In our example, S is composed by the subway trajectories and bus lines, etc., stored in the database of the system, x is the route the system has to find, and the optimization criterion $f(x)$ (which measures the quality of a possible solution) could calculate the ticket price or distance to the destination (or a combination of both), depending on our preferences.

Sometimes there also exist one or more additional constraints which the solution x^* has to satisfy. In that case we talk about constrained optimization (opposed to unconstrained optimization if no such constraint exists). As a summary, an optimization problem has the following components:

- One or more design variables x for which a solution has to be found
- An objective function $f(x)$ describing the optimization criterion
- A solution set S specifying the set of possible solutions x
- (optional) one or more constraints on x.

In order to be of practical use, an optimization algorithm has to find a solution in a reasonable amount of time with reasonable accuracy. Apart from the performance of the algorithm employed, this also depends on the problem at hand itself. If we can hope for a numerical solution, we say that the problem is well-posed. For assessing whether an optimization problem is well-posed, the following conditions must be fulfilled:

1. A solution exists.
2. There is only one solution to the problem, i.e., the solution is unique.
3. The relationship between the solution and the initial conditions is such that small perturbations of the initial conditions result in only small variations of x^*.

1.2 Classical Optimization

Once a task has been transformed into an objective function minimization problem, the next step is to choose an appropriate optimizer. Optimization algorithms can be divided in two groups: derivative-based and derivative-free [2].

In general, $f(x)$ may have a nonlinear form respect to the adjustable parameter x. Due to the complexity of $f(\cdot)$, in classical methods, it is often used an iterative algorithm to explore the input space effectively. In iterative descent methods, the next point x_{k+1} is determined by a step down from the current point x_k in a direction vector $\mathbf{d}$:

$$x_{k+1} = x_k + \alpha \mathbf{d}, \tag{1.2}$$

where α is a positive step size regulating to what extent to proceed in that direction. When the direction d in Eq. 1.1 is determined on the basis of the gradient ($\mathbf{g}$) of the objective function $f(\cdot)$, such methods are known as gradient-based techniques.

The method of steepest descent is one of the oldest techniques for optimizing a given function. This technique represents the basis for many derivative-based methods. Under such a method, the Eq. 1.3 becomes the well-known gradient formula:

$$x_{k+1} = x_k - \alpha \mathbf{g}(f(x)), \tag{1.3}$$

However, classical derivative-based optimization can be effective as long the objective function fulfills two requirements:

- The objective function must be two-times differentiable.
- The objective function must be uni-modal, i.e., have a single minimum.

A simple example of a differentiable and uni-modal objective function is

$$f(x_1, x_2) = 10 - e^{-\left(x_1^2 + 3 \cdot x_2^2\right)} \tag{1.4}$$

Figure 1.1 shows the function defined in Eq. 1.4.

Unfortunately, under such circumstances, classical methodsare only applicable for a few types of optimization problems. For combinatorial optimization, there is no definition of differentiation.

Furthermore, there are many reasons why an objective function might not be differentiable. For example, the "floor" operation in Eq. 1.5 quantizes the function in Eq. 1.4, transforming Fig. 1.1 into the stepped shape seen in Fig. 1.2. At each step's edge, the objective function is non-differentiable:

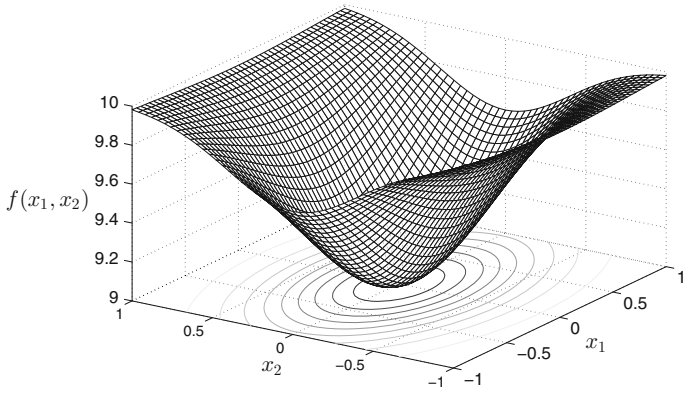

Fig. 1.1 Uni-modal objective function

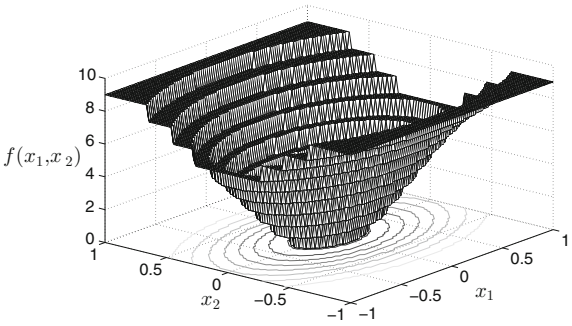

Fig. 1.2 A non-differentiable, quantized, uni-modal function

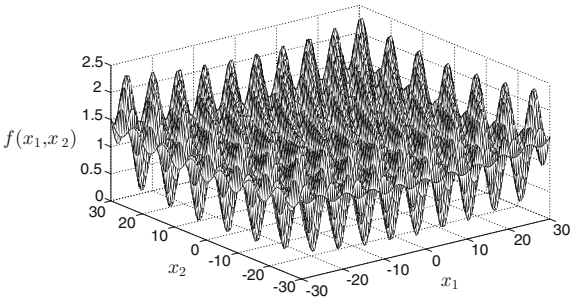

Fig. 1.3 The Griewank multi-modal function

$$f(x_1, x_2) = \text{floor}\left(10 - e^{-\left(x_1^2 + 3 \cdot x_2^2\right)}\right) \tag{1.5}$$

Even in differentiable objective functions, gradient-based methods might not work. Let us consider the minimization of the Griewank function as an example.

$$\begin{aligned} \text{minimize} \quad & f(x_1, x_2) = \frac{x_1^2 + x_2^2}{4000} - \cos(x_1)\cos\left(\frac{x_2}{\sqrt{2}}\right) + 1 \\ \text{subject to} \quad & -30 \leq x_1 \leq 30 \\ & -30 \leq x_2 \leq 30 \end{aligned} \tag{1.6}$$

From the optimization problem formulated in Eq. 1.6, it is quite easy to understand that the global optimal solution is $x_1 = x_2 = 0$. Figure 1.3 visualizes the function defined in Eq. 1.6. According to Fig. 1.3, the objective function has many local optimal solutions (multimodal) so that the gradient methods with a randomly generated initial solution will converge to one of them with a large probability.

Considering the limitations of gradient-based methods, image processing and pattern recognition problems make difficult their integration with classical optimization methods. Instead, some other techniques which do not make assumptions and which can be applied to wide range of problems are required [3].

1.3 Metaheuristic Computation Methods

Metaheuristic computation (EC) [4] methods are derivative-free procedures, which do not require that the objective function must be neither two-times differentiable nor uni-modal. Therefore, metaheuristic methods as global optimization algorithms can deal with non-convex, nonlinear, and multimodal problems subject to linear or nonlinear constraints with continuous or discrete decision variables.

The field of EC has a rich history. With the development of computational devices and demands of industrial processes, the necessity to solve some optimization problems arose despite the fact that there was not sufficient prior knowledge (hypotheses) on the optimization problem for the application of a classical method. In fact, in the majority of image processing and pattern recognition cases, the problems are highly nonlinear, or characterized by a noisy fitness, or without an explicit analytical expression as the objective function might be the result of an experimental or simulation process. In this context, the metaheuristic methods have been proposed as optimization alternatives.

An EC technique is a general method for solving optimization problems. It uses an objective function in an abstract and efficient manner, typically without utilizing deeper insights into its mathematical properties. metaheuristic methods do not require hypotheses on the optimization problem nor any kind of prior knowledge on the objective function. The treatment of objective functions as "black boxes" [5] is the most prominent and attractive feature of metaheuristic methods.

Metaheuristic methods obtain knowledge about the structure of an optimization problem by utilizing information obtained from the possible solutions (i.e., candidate solutions) evaluated in the past. This knowledge is used to construct new candidate solutions which are likely to have a better quality.

Recently, several metaheuristic methods have been proposed with interesting results. Such approaches uses as inspiration our scientific understanding of biological, natural or social systems, which at some level of abstraction can be represented as optimization processes [6]. These methods include the social behavior of bird flocking and fish schooling such as the Particle Swarm Optimization (PSO) algorithm [7], the cooperative behavior of bee colonies such as the Artificial Bee Colony (ABC) technique [8], the improvisation process that occurs when a musician searches for a better state of harmony such as the Harmony Search (HS) [9], the emulation of the bat behavior such as the Bat Algorithm (BA) method [10], the mating behavior of firefly insects such as the Firefly (FF) method [11], the social-spider behavior such as the Social Spider Optimization (SSO) [12], the simulation of the animal behavior in a group such as the Collective Animal Behavior [13], the emulation of immunological systems as the clonal selection algorithm (CSA) [14], the simulation of the electromagnetism phenomenon as the electromagnetism-Like algorithm [15], and the emulation of the differential and conventional evolution in species such as the Differential Evolution (DE) [16] and Genetic Algorithms (GA) [17], respectively.

1.3.1 Structure of a Metaheuristic Computation Algorithm

From a conventional point of view, an EC method is an algorithm that simulates at some level of abstraction a biological, natural or social system. To be more specific, a standard EC algorithm includes:

1. One or more populations of candidate solutions are considered.
2. These populations change dynamically due to the production of new solutions.
3. A fitness function reflects the ability of a solution to survive and reproduce.
4. Several operators are employed in order to explore an exploit appropriately the space of solutions.

The metaheuristic methodology suggest that, on average, candidate solutions improve their fitness over generations (i.e., their capability of solving the optimization problem). A simulation of the evolution process based on a set of candidate solutions whose fitness is properly correlated to the objective function to optimize will, on average, lead to an improvement of their fitness and thus steer the simulated population towards the global solution.

Most of the optimization methods have been designed to solve the problem of finding a global solution of a nonlinear optimization problem with box constraints in the following form:

$$\begin{array}{ll} \text{maximize} & f(\mathbf{x}), \quad \mathbf{x} = (x_1, \ldots, x_d) \in \mathbb{R}^d \\ \text{subject to} & \mathbf{x} \in \mathbf{X} \end{array} \tag{1.7}$$

where $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is a nonlinear function whereas $\mathbf{X} = \left\{\mathbf{x} \in \mathbb{R}^d | l_i \leq x_i \leq u_i, i = 1, \ldots, d.\right\}$ is a bounded feasible search space, constrained by the lower (l_i) and upper (u_i) limits.

In order to solve the problem formulated in Eq. 1.6, in an Metaheuristic computation method, a population $\mathbf{p}^k\left(\left\{\mathbf{p}_1^k, \mathbf{p}_2^k, \ldots, \mathbf{p}_N^k\right\}\right)$ of N candidate solutions (individuals) evolves from the initial point ($k = 0$) to a total gen number iterations ($k = gen$). In its initial point, the algorithm begins by initializing the set of N candidate solutions with values that are randomly and uniformly distributed between the pre-specified lower (l_i) and upper (u_i) limits. In each iteration, a set of metaheuristic operators are applied over the population $\mathbf{P}^k$ to build the new population $\mathbf{P}^{k+1}$. Each candidate solution $\mathbf{p}_i^k$ $(i \in [1, \ldots, N])$ represents a d-dimensional vector $\left\{p_{i,1}^k, p_{i,2}^k, \ldots, p_{i,d}^k\right\}$ where each dimension corresponds to a decision variable of the optimization problem at hand. The quality of each candidate solution $\mathbf{p}_i^k$ is evaluated by using an objective function $f\left(\mathbf{p}_i^k\right)$ whose final result represents the fitness value of $\mathbf{p}_i^k$. During the evolution process, the best candidate solution $\mathbf{g}$ $(g_1, g_2, \ldots, g_d)$ seen so-far is preserved considering that it represents the best available solution. Figure 1.4 presents a graphical representation of a basic cycle of a metaheuristic method.

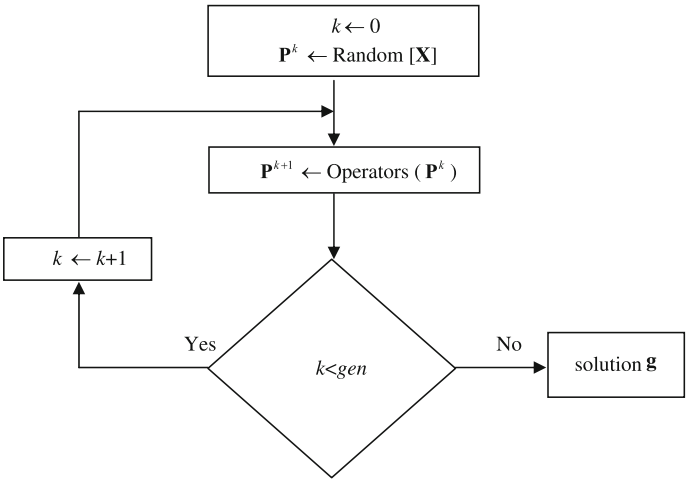

Fig. 1.4 The basic cycle of a metaheuristic method

References

1. Akay, B., Karaboga, D.: A survey on the applications of artificial bee colony in signal, image, and video processing. SIViP **9**(4), 967–990 (2015)
2. Yang, X.-S.: Engineering Optimization. Wiley, USA (2010)
3. Treiber, M.A.: Optimization for Computer Vision: An Introduction to Core Concepts and Methods. Springer, Berlin (2013)
4. Simon, D.: Evolutionary Optimization Algorithms. Wiley, USA (2013)
5. Blum, C., Roli, A.: Metaheuristics in combinatorial optimization: overview and conceptual comparison. ACM Comput. Surv. (CSUR) **35**(3), 268–308 (2003). https://doi.org/10.1145/937503.937505
6. Nanda, S.J., Panda, G.: A survey on nature inspired metaheuristic algorithms for partitional clustering. Swarm Evol. Comput. **16**, 1–18 (2014)
7. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of the 1995 IEEE International Conference on Neural Networks, vol. 4, pp. 1942–1948, December 1995
8. Karaboga, D.: An idea based on honey bee swarm for numerical optimization. Technical Report-TR06. Engineering Faculty, Computer Engineering Department, Erciyes University (2005)
9. Geem, Z.W., Kim, J.H., Loganathan, G.V.: A new heuristic optimization algorithm: harmony search. Simulations **76**, 60–68 (2001)
10. Yang, X.S.: A new metaheuristic bat-inspired algorithm. In: Cruz, C., González, J., Krasnogor, G.T.N., Pelta, D.A. (eds.) Nature Inspired Cooperative Strategies for Optimization (NISCO 2010), Studies in Computational Intelligence, vol. 284, pp. 65–74. Springer, Berlin (2010)
11. Yang, X.S.: Firefly algorithms for multimodal optimization. In: Stochastic Algorithms: Foundations and Applications, SAGA 2009. Lecture Notes in Computer Sciences, vol. 5792, pp. 169–178 (2009)
12. Cuevas, E., Cienfuegos, M., Zaldívar, D., Pérez-Cisneros, M.: A swarm optimization algorithm inspired in the behavior of the social-spider. Expert Syst. Appl. **40**(16), 6374–6384 (2013)

13. Cuevas, E., González, M., Zaldivar, D., Pérez-Cisneros, M., García, G.: An algorithm for global optimization inspired by collective animal behaviour. Discrete Dyn. Nat. Soc. art. no. 638275 (2012)
14. de Castro, L.N., von Zuben, F.J.: Learning and optimization using the clonal selection principle. IEEE Trans. Evol. Comput. **6**(3), 239–251 (2002)
15. Birbil, Ş.I., Fang, S.C.: An electromagnetism-like mechanism for global optimization. J. Glob. Optim. **25**(1), 263–282 (2003)
16. Storn, R., Price, K.: Differential evolution—a simple and efficient adaptive scheme for global optimisation over continuous spaces. Technical Report TR-95–012. ICSI, Berkeley, CA (1995)
17. Goldberg, D.E.: Genetic Algorithm in Search Optimization and Machine Learning. Addison-Wesley, Boston (1989)