Chapter 2
Multi-objective Optimization

Abstract In this chapter, we introduce multi-objective optimization, and recall some of the most relevant research articles that have appeared in the international literature related to these topics. The presented state-of-the-art does not have the purpose of being exhaustive; it aims to drive the reader to the main problems and the approaches to solve them.

2.1 Multi-objective Management

The choice of a route at a planning level can be done taking into account time, length, but also parking or maintenance facilities. As far as advisory or, more in general, automation procedures to support this choice are concerned, the available tools are basically based on the “shortest-path problem”. Indeed, the problem to find the single-objective shortest path from an origin to a destination in a network is one of the most classical optimization problems in transportation and logistic, and has deserved a great deal of attention from researchers worldwide. However, the need to face real applications renders the hypothesis of a single-objective function to be optimized subject to a set of constraints no longer suitable, and the introduction of a multi-objective optimization framework allows one to manage more information. Indeed, if for instance we consider the problem to route hazardous materials in a road network (see, e.g., Erkut et al., 2007), defining a single-objective function problem will involve, separately, the distance, the risk for the population, and the transportation costs. If we regard the problem from different points of view, i.e., in terms of social needs for a safe transshipment, or in terms of economic issues or pol-
olution reduction, it is clear that a model that considers simultaneously two or more such objectives could produce solutions with a higher level of equity. In the following, we will discuss multi-objective optimization and related solution techniques.

2.2 Multi-objective Optimization and Pareto-optimal Solutions

A basic single-objective optimization problem can be formulated as follows:

$$\min f(x)$$
$$x \in S,$$

where $f$ is a scalar function and $S$ is the (implicit) set of constraints that can be defined as

$$S = \{x \in \mathbb{R}^m : h(x) = 0, g(x) \geq 0\}.$$

Multi-objective optimization can be described in mathematical terms as follows:

$$\min [f_1(x), f_2(x), \ldots, f_n(x)]$$
$$x \in S,$$

where $n > 1$ and $S$ is the set of constraints defined above. The space in which the objective vector belongs is called the objective space, and the image of the feasible set under $F$ is called the attained set. Such a set will be denoted in the following with

$$C = \{y \in \mathbb{R}^n : y = f(x), x \in S\}.$$

The scalar concept of “optimality” does not apply directly in the multi-objective setting. Here the notion of Pareto optimality has to be introduced. Essentially, a vector $x^* \in S$ is said to be Pareto optimal for a multi-objective problem if all other vectors $x \in S$ have a higher value for at least one of the objective functions $f_i$, with $i = 1, \ldots, n$, or have the same value for all the objective functions. Formally speaking, we have the following definitions:

- A point $x^*$ is said to be a weak Pareto optimum or a weak efficient solution for the multi-objective problem if and only if there is no $x \in S$ such that $f_i(x) < f_i(x^*)$ for all $i \in \{1, \ldots, n\}$. 
A point $x^*$ is said to be a \textit{strict} Pareto optimum or a \textit{strict} efficient solution for the multi-objective problem if and only if there is no $x \in S$ such that $f_i(x) \leq f_i(x^*)$ for all $i \in \{1, \ldots, n\}$, with at least one strict inequality.

We can also speak of locally Pareto-optimal points, for which the definition is the same as above, except that we restrict attention to a feasible neighborhood of $x^*$. In other words, if $B(x^*, \epsilon)$ is a ball of radius $\epsilon > 0$ around point $x^*$, we require that for some $\epsilon > 0$, there is no $x \in S \cap B(x^*, \epsilon)$ such that $f_i(x) \leq f_i(x^*)$ for all $i \in \{1, \ldots, n\}$, with at least one strict inequality.

The image of the efficient set, i.e., the image of all the efficient solutions, is called Pareto front or Pareto curve or surface. The shape of the Pareto surface indicates the nature of the trade-off between the different objective functions. An example of a Pareto curve is reported in Fig. 2.1, where all the points between $(f_2(\hat{x}), f_1(\hat{x}))$ and $(f_2(\tilde{x}), f_1(\tilde{x}))$ define the Pareto front. These points are called non-inferior or non-dominated points.

![Fig. 2.1 Example of a Pareto curve](image)

An example of weak and strict Pareto optima is shown in Fig. 2.2: points $p_1$ and $p_5$ are weak Pareto optima; points $p_2$, $p_3$ and $p_4$ are strict Pareto optima.
2.3 Techniques to Solve Multi-objective Optimization Problems

Pareto curves cannot be computed efficiently in many cases. Even if it is theoretically possible to find all these points exactly, they are often of exponential size; a straightforward reduction from the knapsack problem shows that they are NP-hard to compute. Thus, approximation methods for them are frequently used. However, approximation does not represent a secondary choice for the decision maker. Indeed, there are many real-life problems for which it is quite hard for the decision maker to have all the information to correctly and/or completely formulate them; the decision maker tends to learn more as soon as some preliminary solutions are available. Therefore, in such situations, having some approximated solutions can help, on the one hand, to see if an exact method is really required, and, on the other hand, to exploit such a solution to improve the problem formulation (Ruzica and Wieccek, 2005).

Approximating methods can have different goals: representing the solution set when the latter is numerically available (for convex multi-objective problems); approximating the solution set when some but not all the Pareto curve is numerically available (see non-linear multi-objective problems); approximating the solution set
2.3 Techniques to Solve Multi-objective Optimization Problems

when the whole efficient set is not numerically available (for discrete multi-objective problems).

A comprehensive survey of the methods presented in the literature in the last 33 years, from 1975, is that of Ruzica and Wiecek (2005). The survey analyzes separately the cases of two objective functions, and the case with a number of objective functions strictly greater than two. More than 50 references on the topic have been reported. Another interesting survey on these techniques related to multiple objective integer programming can be found in the book of Ehrigott (2005) and the paper of Erghott (2006), where he discusses different scalarization techniques. We will give details of the latter survey later in this chapter, when we move to integer linear programming formulations. Also, T’Kindt and Billaut (2005) in their book on “Multicriteria scheduling”, dedicated a part of their manuscript (Chap. 3) to multi-objective optimization approaches.

In the following, we will start revising, following the same lines of Erghott (2006), these scalarization techniques for general continuous multi-objective optimization problems.

### 2.3.1 The Scalarization Technique

A multi-objective problem is often solved by combining its multiple objectives into one single-objective scalar function. This approach is in general known as the weighted-sum or scalarization method. In more detail, the weighted-sum method minimizes a positively weighted convex sum of the objectives, that is,

\[
\min_{\gamma} \sum_{i=1}^{n} \gamma_i \cdot f_i(x)
\]

\[
\sum_{i=1}^{n} \gamma_i = 1
\]

\[
\gamma_i > 0, \ i = 1, \ldots, n
\]

\[
x \in S
\]

that represents a new optimization problem with a unique objective function. We denote the above minimization problem with \( P_\gamma \).

It can be proved that the minimizer of this single-objective function \( P(\gamma) \) is an efficient solution for the original multi-objective problem, i.e., its image belongs to
the Pareto curve. In particular, we can say that if the $\gamma$ weight vector is strictly greater than zero (as reported in $P(\gamma)$), then the minimizer is a strict Pareto optimum, while in the case of at least one $\gamma_i = 0$, i.e.,

$$
\min \sum_{i=1}^{n} \gamma_i \cdot f_i(x)
$$

$$
\sum_{i=1}^{n} \gamma_i = 1
$$

$$
\gamma_i \geq 0, \ i = 1, \ldots, n
$$

$$
x \in S,
$$

it is a weak Pareto optimum. Let us denote the latter problem with $P_w(\gamma)$.

There is not an a-priori correspondence between a weight vector and a solution vector; it is up to the decision maker to choose appropriate weights, noting that weighting coefficients do not necessarily correspond directly to the relative importance of the objective functions. Furthermore, as we noted before, besides the fact that the decision maker cannot be aware of which weights are the most appropriate to retrieve a satisfactorily solution, he/she does not know in general how to change weights to consistently change the solution. This means also that it is not easy to develop heuristic algorithms that, starting from certain weights, are able to define iteratively weight vectors to reach a certain portion of the Pareto curve.

Since setting a weight vector conducts to only one point on the Pareto curve, performing several optimizations with different weight values can produce a considerable computational burden; therefore, the decision maker needs to choose which different weight combinations have to be considered to reproduce a representative part of the Pareto front.

Besides this possibly huge computation time, the scalarization method has two technical shortcomings, as explained in the following.

- The relationship between the objective function weights and the Pareto curve is such that a uniform spread of weight parameters, in general, does not produce a uniform spread of points on the Pareto curve. What can be observed about this fact is that all the points are grouped in certain parts of the Pareto front, while some (possibly significative) portions of the trade-off curve have not been produced.
Non-convex parts of the Pareto set cannot be reached by minimizing convex combinations of the objective functions. An example can be made showing a geometrical interpretation of the weighted-sum method in two dimensions, i.e., when $n = 2$. In the two-dimensional space the objective function is a line

$$y = \gamma_1 \cdot f_1(x) + \gamma_2 \cdot f_2(x),$$

where

$$f_2(x) = -\frac{\gamma_1 \cdot f_1(x)}{\gamma_2} + \frac{y}{\gamma_2}.$$ 

The minimization of $\gamma \cdot f(x)$ in the weight-sum approach can be interpreted as the attempt to find the $y$ value for which, starting from the origin point, the line with slope $-\frac{\gamma_1}{\gamma_2}$ is tangent to the region $C$. Obviously, changing the weight parameters leads to possibly different touching points of the line to the feasible region. If the Pareto curve is convex then there is room to calculate such points for different $\gamma$ vectors (see Fig. 2.3).

On the contrary, when the curve is non-convex, there is a set of points that cannot be reached for any combinations of the $\gamma$ weight vector (see Fig. 2.4).
The following result by Geoffrion (1968) states a necessary and sufficient condition in the case of convexity as follows:

*If the solution set $S$ is convex and the $n$ objectives $f_i$ are convex on $S$, $x^*$ is a strict Pareto optimum if and only if it exists $\gamma \in \mathbb{R}^n$, such that $x^*$ is an optimal solution of problem $P_s(\gamma)$. Similarly: If the solution set $S$ is convex and the $n$ objectives $f_i$ are convex on $S$, $x^*$ is a weak Pareto optimum if and only if it exists $\gamma \in \mathbb{R}^n$, such that $x^*$ is an optimal solution of problem $P_w(\gamma)$.*

If the convexity hypothesis does not hold, then only the necessary condition remains valid, i.e., the optimal solutions of $P_s(\gamma)$ and $P_w(\gamma)$ are strict and weak Pareto optima, respectively.

### 2.3.2 $\epsilon$-constraints Method

Besides the scalarization approach, another solution technique to multi-objective optimization is the $\epsilon$-constraints method proposed by Chankong and Haimes in 1983. Here, the decision maker chooses one objective out of $n$ to be minimized; the remaining objectives are constrained to be less than or equal to given target val-
ues. In mathematical terms, if we let $f_2(x)$ be the objective function chosen to be minimized, we have the following problem $P(\varepsilon_2)$:

$$\min f_2(x)$$

$$f_i(x) \leq \varepsilon_i, \forall i \in \{1, \ldots, n\} \setminus \{2\}$$

$$x \in S.$$  

We note that this formulation of the $\varepsilon$-constraints method can be derived by a more general result by Miettinen, that in 1994 proved that:

If an objective $j$ and a vector $\varepsilon = (\varepsilon_1, \ldots, \varepsilon_{j-1}, \varepsilon_{j+1}, \ldots, \varepsilon_n) \in \mathbb{R}^{n-1}$ exist, such that $x^*$ is an optimal solution to the following problem $P(\varepsilon)$:

$$\min f_j(x)$$

$$f_i(x) \leq \varepsilon_i, \forall i \in \{1, \ldots, n\} \setminus \{j\}$$

$$x \in S,$$

then $x^*$ is a weak Pareto optimum.

In turn, the Miettinen theorem derives from a more general theorem by Yu (1974) stating that:

$x^*$ is a strict Pareto optimum if and only if for each objective $j$, with $j = 1, \ldots, n$, there exists a vector $\varepsilon = (\varepsilon_1, \ldots, \varepsilon_{j-1}, \varepsilon_{j+1}, \ldots, \varepsilon_n) \in \mathbb{R}^{n-1}$ such that $f(x^*)$ is the unique objective vector corresponding to the optimal solution to problem $P(\varepsilon)$.

Note that the Miettinen theorem is an easy implementable version of the result by Yu (1974). Indeed, one of the difficulties of the result by Yu, stems from the uniqueness constraint. The weaker result by Miettinen allows one to use a necessary condition to calculate weak Pareto optima independently from the uniqueness of the optimal solutions. However, if the set $S$ and the objectives are convex this result becomes a necessary and sufficient condition for weak Pareto optima. When, as in problem $P(\varepsilon_2)$, the objective is fixed, on the one hand, we have a more simplified version, and therefore a version that can be more easily implemented in automated decision-support systems; on the other hand, however, we cannot say that in the presence of $S$ convex and $f_i$ convex, $\forall i = 1, \ldots, n$, all the set of weak Pareto optima can be calculated by varying the $\varepsilon$ vector.

One advantage of the $\varepsilon$-constraints method is that it is able to achieve efficient points in a non-convex Pareto curve. For instance, assume we have two objective
functions where objective function $f_1(x)$ is chosen to be minimized, i.e., the problem is

$$\begin{align*}
\min & \quad f_1(x) \\
\text{s.t.} & \quad f_2(x) \leq \epsilon_2 \\
& \quad x \in S,
\end{align*}$$

we can be in the situation depicted in Fig. 2.5 where, when $f_2(x) = \epsilon_2$, $f_1(x)$ is an efficient point of the non-convex Pareto curve.

![Diagram](image)

**Fig. 2.5** Geometrical representation of the $\epsilon$-constraints approach in the non-convex Pareto curve case

Therefore, as proposed in Steurer (1986) the decision maker can vary the upper bounds $\epsilon_i$ to obtain weak Pareto optima. Clearly, this is also a drawback of this method, i.e., the decision maker has to choose appropriate upper bounds for the constraints, i.e., the $\epsilon_i$ values. Moreover, the method is not particularly efficient if the number of the objective functions is greater than two.

For these reasons, Erghott and Rusika in 2005, proposed two modifications to improve this method, with particular attention to the computational difficulties that the method generates.
2.3 Techniques to Solve Multi-objective Optimization Problems

2.3.3 Goal Programming

Goal Programming dates back to Charnes et al. (1955) and Charnes and Cooper (1961). It does not pose the question of maximizing multiple objectives, but rather it attempts to find specific goal values of these objectives. An example can be given by the following program:

\[
\begin{align*}
    f_1(x) & \geq v_1 \\
    f_2(x) & = v_2 \\
    f_3(x) & \leq v_3 \\
    x & \in S.
\end{align*}
\]

Clearly we have to distinguish two cases, i.e., if the intersection between the image set \( C \) and the utopian set, i.e., the image of the admissible solutions for the objectives, is empty or not. In the former case, the problem transforms into one in which we have to find a solution whose value is as close as possible to the utopian set. To do this, additional variables and constraints are introduced. In particular, for each constraint of the type

\[ f_1(x) \geq v_1 \]

we introduce a variable \( s_1^- \) such that the above constraint becomes

\[ f_1(x) + s_1^- \geq v_1. \]

For each constraint of the type

\[ f_2(x) = v_2 \]

we introduce a surplus two variables \( s_2^+ \) and \( s_2^- \) such that the above constraint becomes

\[ f_2(x) + s_2^- - s_2^+ = v_2. \]

For each constraint of the type

\[ f_3(x) \leq v_3 \]

we introduce a variable \( s_3^+ \) such that the above constraint becomes

\[ f_3(x) - s_3^+ \leq v_3. \]
Let us denote with \( s \) the vector of the additional variables. A solution \((x, s)\) to the above problem is called a strict Pareto-slack optimum if and only if a solution \((x', s')\), for every \( x' \in S \), such that \( s'_i \leq s_i \) with at least one strict inequality does not exist.

There are different ways of optimizing the slack/surplus variables. An example is given by the Archimedean goal programming, where the problem becomes that of minimizing a linear combination of the surplus and slack variables each one weighted by a positive coefficient \( \alpha \) as follows:

\[
\begin{align*}
\min & \quad \alpha_{s_1^-} s_1^- + \alpha_{s_2^+} s_2^+ + \alpha_{s_2^-} s_2^- + \alpha_{s_3^+} s_3^+ \\
& \quad f_1(x) + s_1^- \geq v_1 \\
& \quad f_2(x) + s_2^- - s_2^+ = v_2 \\
& \quad f_3(x) - s_3^+ \leq v_3 \\
& \quad s_1^- \geq 0 \\
& \quad s_2^+ \geq 0 \\
& \quad s_2^- \geq 0 \\
& \quad s_3^+ \geq 0 \\
& \quad x \in S.
\end{align*}
\]

For the above problem, the Geoffrion theorem says that the resolution of this problem offers strict or weak Pareto-slack optimum.

Besides Archimedean goal programming, other approaches are the lexicographical goal programming, the interactive goal programming, the reference goal programming and the multi-criteria goal programming (see, e.g., T’kindt and Billaut, 2005).

### 2.3.4 Multi-level Programming

Multi-level programming is another approach to multi-objective optimization and aims to find one optimal point in the entire Pareto surface. Multi-level programming orders the \( n \) objectives according to a hierarchy. Firstly, the minimizers of the first objective function are found; secondly, the minimizers of the second most important
objective are searched for, and so forth until all the objective function have been optimized on successively smaller sets.

Multi-level programming is a useful approach if the hierarchical order among the objectives is meaningful and the user is not interested in the continuous trade-off among the functions. One drawback is that optimization problems that are solved near the end of the hierarchy can be largely constrained and could become infeasible, meaning that the less important objective functions tend to have no influence on the overall optimal solution.

Bi-level programming (see, e.g., Bialas and Karwan, 1984) is the scenario in which \( n = 2 \) and has received several attention, also for the numerous applications in which it is involved. An example is given by hazmat transportation in which it has been mainly used to model the network design problem considering the government and the carriers points of view: see, e.g., the papers of Kara and Verter (2004), and of Erkut and Gzara (2008) for two applications (see also Chap. 4 of this book).

In a bi-level mathematical program one is concerned with two optimization problems where the feasible region of the first problem, called the upper-level (or leader) problem, is determined by the knowledge of the other optimization problem, called the lower-level (or follower) problem. Problems that naturally can be modelled by means of bi-level programming are those for which variables of the first problem are constrained to be the optimal solution of the lower-level problem.

In general, bi-level optimization is issued to cope with problems with two decision makers in which the optimal decision of one of them (the leader) is constrained by the decision of the second decision maker (the follower). The second-level decision maker optimizes his/her objective function under a feasible region that is defined by the first-level decision maker. The latter, with this setting, is in charge to define all the possible reactions of the second-level decision maker and selects those values for the variable controlled by the follower that produce the best outcome for his/her objective function.

A bi-level program can be formulated as follows:

\[
\begin{align*}
\min f(x_1, x_2) \\
x_1 \in X_1 \\
x_2 & \in \arg \min g(x_1, x_2) \\
x_2 & \in X_2.
\end{align*}
\]
The analyst should pay particular attention when using bi-level optimization (or multi-level optimization in general) in studying the uniqueness of the solutions of the follower problem. Assume, for instance, one has to calculate an optimal solution $x_1^*$ to the leader model. Let $x_2^*$ be an optimal solution of the follower problem associated with $x_1^*$. If $x_2^*$ is not unique, i.e., $|\arg\min g(x_1^*, x_2)| > 1$, we can have a situation in which the follower decision maker can be free, without violating the leader constraints, to adopt for his problem another optimal solution different from $x_2^*$, i.e., $\hat{x}_2 \in \arg\min g(x_1^*, x_2)$ with $\hat{x}_2 \neq x_2^*$, possibly inducing a $f(x_1^*, \hat{x}_2) > f(x_1^*, x_2^*)$ on the leader, forcing the latter to carry out a sensitivity analysis on the values attained by his objective function in correspondence to all the optimal solutions in $\arg\min g(x_1^*, x_2)$.

Bi-level programs are very closely related to the van Stackelberg equilibrium problem (van Stackelberg, 1952), and the mathematical programs with equilibrium constraints (see, e.g., Luo et al. 1996). The most studied instances of bi-level programming problems have been for a long time the linear bi-level programs, and therefore this subclass is the subject of several dedicated surveys, such as that by Wen and Hsu (1991).

Over the years, more complex bi-level programs were studied and even those including discrete variables received some attention, see, e.g., Vicente et al. (1996). Hence, more general surveys appeared, such as those by Vicente and Calamai (1994) and Falk and Liu (1995) on non-linear bi-level programming. The combinatorial nature of bi-level programming has been reviewed in Marcotte and Savard (2005).

Bi-level programs are hard to solve. In particular, linear bi-level programming has been proved to be strongly NP-hard (see, Hansen et al., 1992); Vicente et al. (1996) strengthened this result by showing that finding a certificate of local optimality is also strongly NP-hard.

Existing methods for bi-level programs can be distinguished into two classes. On the one hand, we have convergent algorithms for general bi-level programs with theoretical properties guaranteeing suitable stationary conditions; see, e.g., the implicit function approach by Outrata et al. (1998), the quadratic one-level reformulation by Scholtes and Stohr (1999), and the smoothing approaches by Fukushima and Pang (1999) and Dussault et al. (2004).

With respect to the optimization problems with complementarity constraints, which represent a special way of solving bi-level programs, we can mention the papers of Kocvara and Outrata (2004), Bouza and Still (2007), and Lin and Fukushima
The first work presents a new theoretical framework with the implicit programming approach. The second one studies convergence properties of a smoothing method that allows the characterization of local minimizers where all the functions defining the model are twice differentiable. Finally, Lin and Fukushima (2003, 2005) present two relaxation methods.

Exact algorithms have been proposed for special classes of bi-level programs, e.g., see the vertex enumeration methods by Candler and Townsley (1982), Bialas and Karwan (1984), and Tuy et al. (1993) applied when the property of an extremal solution in bi-level linear program holds. Complementary pivoting approaches (see, e.g., Bialas et al., 1980, and Júdice and Faustino, 1992) have been proposed on the single-level optimization problem obtained by replacing the second-level optimization problem by its optimality conditions. Exploiting the complementarity structure of this single-level reformulation, Bard and Moore (1990) and Hansen et al. (1992), have proposed branch-and-bound algorithms that appear to be among the most efficient. Typically, branch-and-bound is used when the lower-level problem is convex and regular, since the latter can be replaced by its Karush–Kuhn–Tucker (KKT) conditions, yielding a single-level reformulation. When one deals with linear bi-level programs, the complementarity conditions are intrinsically combinatorial, and in such cases branch-and-bound is the best approach to solve this problem (see, e.g., Colson et al., 2005). A cutting-plane approach is not frequently used to solve bi-level linear programs. Cutting-plane methods found in the literature are essentially based on Tuy’s concavity cuts (Tuy, 1964). White and Anandalingam (1993) use these cuts in a penalty function approach for solving bi-level linear programs. Marcotte et al. (1993) propose a cutting-plane algorithm for solving bi-level linear programs with a guarantee of finite termination. Recently, Audet et al. (2007), exploiting the equivalence of the latter problem with a mixed integer linear programming one, proposed a new branch-and-bound algorithm embedding Gomory cuts for bi-level linear programming.

2.4 Multi-objective Optimization Integer Problems

In the previous section, we gave general results for continuous multi-objective problems. In this section, we focus our attention on what happens if the optimization problem being solved has integrality constraints on the variables. In particular, all
the techniques presented can be applied in these situations as well, with some limitations on the capabilities of these methods to construct the Pareto front entirely. Indeed, these methods are, in general, very hard to solve in real applications, or are unable to find all efficient solutions. When integrality constraints arise, one of the main limits of these techniques is in the inability of obtaining some Pareto optima; therefore, we will have supported and unsupported Pareto optima.

![Fig. 2.6 Supported and unsupported Pareto optima](image)

Fig. 2.6 gives an example of these situations: points $p_6$ and $p_7$ are unsupported Pareto optima, while $p_1$ and $p_5$ are supported weak Pareto optima, and $p_2$, $p_3$, and $p_4$ are supported strict Pareto optima.

Given a multi-objective optimization integer problem (MOIP), the scalarization in a single objective problem with additional variables and/or parameters to find a subset of efficient solutions to the original MOIP, has the same computational complexity issues of a continuous scalarized problem.

In the 2006 paper of Ehrgott “A discussion of scalarization techniques for multiple objective integer programming” the author, besides the scalarization techniques also presented in the previous section (e.g., the weighted-sum method, the $\epsilon$-constraint method), satisfying the linear requirement imposed by the MOIP formulation (where variables are integers, but constraints and objectives are linear),
presented more methods like the Lagrangian relaxation and the elastic-constraints method.

By the author’s analysis, it emerges that the attempt to solve the scalarized problem by means of Lagrangian relaxation would not lead to results that go beyond the performance of the weighted-sum technique. It is also shown that the general linear scalarization formulation is NP-hard. Then, the author presents the elastic-constraints method, a new scalarization technique able to overcome the drawback of the previously mentioned techniques related to finding all efficient solutions, combining the advantages of the weighted-sum and the \( \varepsilon \)-constraint methods. Furthermore, it is shown that a proper application of this method can also give reasonable computing times in practical applications; indeed, the results obtained by the author on the elastic-constraints method are applied to an airline-crew scheduling problem, whose size oscillates from 500 to 2000 constraints, showing the effectiveness of the proposed technique.

2.4.1 Multi-objective Shortest Paths

Given a directed graph \( G = (V, A) \), an origin \( s \in V \) and a destination \( t \in V \), the shortest-path problem (SPP) aims to find the minimum distance path in \( G \) from \( o \) to \( d \). This problem has been studied for more than 50 years, and several polynomial algorithms have been produced (see, for instance, Cormen et al., 2001).

From the freight distribution point of view the term shortest may have quite different meanings from faster, to quickest, to safest, and so on, focusing the attention on what the labels of the arc set \( A \) represent to the decision maker. For this reason, in some cases we will find it simpler to define for each arc more labels so as to represent the different arc features (e.g., length, travel time, estimated risk).

The problem to find multi-objective shortest paths (MOSPP) is known to be NP-hard (see, e.g., Serafini, 1986), and the algorithms proposed in the literature faced the difficulty to manage the large number of non-dominated paths that results in a considerable computational time, even in the case of small instances. Note that the number of non-dominated paths may increase exponentially with the number of nodes in the graph (Hansen, 1979).

In the multi-objective scenario, each arc \((i, j)\) in the graph has a vector of costs \( c_{ij} \in \mathbb{R}^n \) with \( c_{ij} = (c_{ij}^1, \ldots, c_{ij}^n) \) components, where \( n \) is the number of criteria.
A path $P_{si}$ from the origin $s$ to node $i$ is a sequence of nodes (and arcs) $P_{si} = (s \equiv n_1, \ldots, n_h \equiv i)$ of length $h \geq 2$, where each arc $(n_l, n_{l+1}) \in A$ for $l = 1, \ldots, h - 1$. Such a path is evaluated by means of a performance vector $c(P_{si}) = (c^1(P_{si}), \ldots, c^n(P_{si}))$ where

$$c^l(P_{si}) = \sum_{(p,q) \in P_{si}} c^l_{pq}$$

for $l = 1, \ldots, n$.

Let

$$x_{ij} = \begin{cases} 
1 & \text{if } (i, j) \text{ is in a path;} \\
0 & \text{otherwise.}
\end{cases}$$

The problem to find the multi-objective shortest paths from the origin $s$ to destination $t$ could be stated as follows:

$$\min f^l(x) = \sum_{(i,j) \in A} c^l_{ij} x_{ij} \forall l \in \{1, \ldots, n\}$$

$$\sum_{\{j: (i,j) \in A\}} x_{ij} - \sum_{\{j: (j,i) \in A\}} x_{ji} = \begin{cases} 
1 & i = s \\
0 & \forall i \in V \setminus \{s,t\} \\
-1 & i = t
\end{cases}$$

$$x_{ij} \geq 0, \forall (i,j) \in A.$$

Let the solution space of the above problem be denoted with $S_{st}$. Let $P_{st}^1$ and $P_{st}^2$ be two feasible paths and $c(P_{st}^1)$ and $c(P_{st}^2)$ be the relative performance vectors.

Given a path $P_{st}^1 \in S_{st}$, the vector $c(P_{st}^1)$ is non-dominated if there does not exist another vector $c(P_{st}^2)$, with $P_{st}^2 \in S_{st}$, such that $c^l(P_{st}^2) \leq c^l(P_{st}^1)$, $l = 1, \ldots, n$ and $c^l(P_{st}^2) \neq c^l(P_{st}^1)$ for some $l$; otherwise, $c(P_{st}^2)$ dominates $c(P_{st}^1)$.

Given a path $P_{st}^1 \in S_{st}$, the vector $c(P_{st}^1)$ is weakly non-dominated if and only if there does not exist another vector $c(P_{st}^2)$, with $P_{st}^2 \in S_{st}$, such that $c^l(P_{st}^2) < c^l(P_{st}^1)$, $l = 1, \ldots, n$; otherwise, $c(P_{st}^2)$ strictly dominates $c(P_{st}^1)$.

Correspondingly, efficient solutions are defined as follows: a path $P_{st}^1$ is strict efficient (or strict Pareto optimal) if only if $P_{st}^1 \in S_{st}$ and there does not exist another path $P_{st}^2 \in S_{st}$ such that $c(P_{st}^2)$ dominates $c(P_{st}^1)$.

A path $P_{st}^1$ is weakly efficient if and only if $P_{st}^1 \in S_{st}$ and there does not exist another path $P_{st}^2 \in S_{st}$ such that $c(P_{st}^2)$ strictly dominates $c(P_{st}^1)$.

Hansen’s (1980) work was one of the first studies on the multi-objective shortest-path problem; he analyzed some bi-objective problems and demonstrated that, in the
worst case, the non-dominated paths number grows exponentially with the network size. So it is clear that, in the worst case, the generation of the efficient frontier may require a huge computational effort. A way to find a subset of non-dominated solutions can be to use a linear combination of the objectives in question, changing the used weights. On this subject, Henig (1986) proposed a review of exact methods for finding all non-dominated paths, and presented an approximated method whose average computational complexity is polynomial in the size of the network nodes.

Even if there are some dynamic programming approaches (e.g., in Henig, 1986, and Kostreva and Wiecek, 1993), the study on algorithms for the multi-objective shortest-path problem was mainly devoted to two classes of methods: one based on label-setting algorithms and the other based on label-correcting algorithms. In the former class, most of the work done was devoted to problems where all the objective functions are of the sum type, as in Hansen (1979), and Martins (1984).

Martins (1999) proposed a multi-objective algorithm based on vertex labelling techniques that generalizes the Bellman optimality principle. In his paper he assumed the attributes associated on arcs to be non-negative, deterministic and additive along the route. Martins’ algorithm uses a multiple labelling approach. Each node \( i \in V \) is associated with several labels, and the \( l \)-th label contains the \( n \) objective values and two pointers. The label can be represented as

\[
[c^1(P_{si}), \ldots, c^n(P_{si}), j, l_1],
\]

where \( c^h(P_{si}) \) is the length of the path \( P_{si} \) from origin \( s \) to node \( i \), for \( h = 1, \ldots, n \), \( j \neq i \) is some node of \( G \), and \( l_1 \) indicates a certain label of node \( j \) for which

\[
c_{[h,l]}(P_{si}) = c_{[h,l_1]}(P_{sj}) + c_{ij}^h,
\]

where \( c_{[h,l]}^h \) is the \( h \)-th component of the \( l \)-th label of node \( i \).

At each iteration, there are two kinds of labels, permanent ones and temporary ones. The algorithm selects a temporary label in a node \( i \), converts it to permanent, and updates all the labels of the successors \( j \) of \( i \), for each \((i, j) \in A\); then, it deletes all the labels that represent a dominated path \( P_{sj} \). The algorithm stops when it runs out of temporary labels, and each permanent label represents a unique efficient path.

The node-selection step is made considering, among all the labels in each node, the lexicographically smallest one. We recall that for some node \( i \), a label
is said to be lexicographically smaller than a label
\[ [c_1(P_{si}), \ldots, c_n(P_{si}), -, -]_\xi \]
if
\[ c_{[1, \xi]}(P_{si}) = c_{[1, \delta]}(P'_{si}), \ldots, c_{[k-1, \xi]}(P_{si}) = c_{[k-1, \delta]}(P'_{si}), c_{[k, \xi]}(P_{si}) < c_{[k, \delta]}(P'_{si}) \]
holds for some \( k \in \{1, \ldots, n\} \).

In Table 2.1, Martins’ algorithm is sketched.

### Table 2.1 Martins’ algorithm

| Step 1 | Assign the temporary label \([0,0,\ldots,0],-,-\]_1 to node s. |
| Step 2 | If the set of temporary labels is empty go to step 5. Otherwise, among all the temporary labels determine the lexicographically smallest one. Let the \( l \)th label be associated with node \( i \). Set this label as a permanent one. |
| Step 3 | While some node \( j \in V \) exists, such that \((i,j) \in A\), execute |
| Step 3.1 | \( c^k(P_{sj}) = c_{[k,l]}(P_{si}) + c^k_{ij} \) for every \( k = 1, \ldots, n \) and let \([c^1(P_{sj}), \ldots, c^n(P_{sj})], i, l]\_\xi\) be a new temporary label of node \( j \). |
| Step 3.2 | Among all the temporary labels of node \( j \), delete all labels representing a dominated path from \( s \) to \( j \). |
| Step 4 | Return to step 2. |
| Step 5 | Find the non-dominated paths from \( s \) to \( t \). For that, the two pointers of each label must be used. |
| Step 6 | Stop. |

Recently, Gandibleux et al. in 2006, proposed a problem with a max-min objective and two min-sum objectives that was solved using a revisited version of Martins’ algorithm. Referring to the class of label-setting algorithms, where node labels are marked as permanent only at the last iteration of the run, we refer the reader to, e.g., Brumbaugh-Smith and Shier (1989), Mote et al. (1991), and Skriver and Andersen (2000).

For the bi-objective case, there are algorithms for the path-ranking problem, as in Climaco and Martins (1982) and Martins and Climaco (1981), and for the max-ordering problem, as in Ehrgott and Skriver (2003). In Paixão et al. (2003) ranking is obtained for the general multi-objective shortest-path problem using a utility function based on the norm value associated with each path.
As described in Tarapata (2007), one of the most popular methods of solving multi-objective shortest-path problems is the construction of \((1 + \varepsilon)\)-Pareto curves (see, e.g., Papadimitriou and Yannakakis, 2000, and Vassilvitskii and Yannakakis, 2004). Informally, a \((1 + \varepsilon)\)-Pareto curve \(P_\varepsilon\) is a subset of feasible solutions such that for any Pareto-optimal solution there exists a solution in \(P_\varepsilon\) that is no more than \((1 - \varepsilon)\) away in the objectives (Tarapata, 2007). Papadimitriou and Yannakakis (2000) show that for any multi-objective optimization problem there exists a \((1 + \varepsilon)\)-Pareto curve \(P_\varepsilon\) of (polynomial) size. Extensions to this method to produce a constant approximation to the smallest possible \((1 + \varepsilon)\)-Pareto curve for the cases of 2 and 3 objectives are presented in Vassilvitskii and Yannakakis (2004), while for a number of objectives greater than three, inapproximability results are shown for such a constant approximation.

For the case of the multi-objective shortest-path problem (and some other problems with linear objectives), Papadimitriou and Yannakakis (2000) show how a gap routine can be constructed (based on a pseudopolynomial algorithm for computing exact paths) and, consequently, provide a fully polynomial time approximation scheme for this problem. Note that fully polynomial time approximation schemes for the multi-objective shortest-path problem were already known in the case of two objectives (Hansen, 1979), as well as in the case of multiple objectives in directed acyclic graphs (Warburton, 1987). In particular, the bi-objective case was extensively studied (see, e.g., Ehrgott and Gandibleux, 2002), while for a number of objectives greater than two, very little has been achieved. The results in Warburton (1987), Papadimitriou and Yannakakis (2000) and Tsaggouris and Zaroliagis (2005) are the best fully polynomial time approximation schemes known.

Warburton (1987) has proposed polynomial methods for generating all “quasi” Pareto-optimal paths. Another approach can be to treat each objective as a constraint: Sancho (1988) has considered the problem with three objectives, two of which have been transformed into constraints. Each of the generated solutions cannot be Pareto optimal with respect to all three considered objectives.

An approach not using deterministic attributes was proposed by List et al. (1991). In particular, they developed a method that finds, given an origin and a destination, the non-dominated paths replacing to each stochastic attribute (supposed normally distributed) its mean and variance; then, the non-dominated paths number is reduced by comparing all pairs of paths, using the “stochastic dominance” criterion.
2.4.2 Multi-objective Travelling Salesman Problem

The *travelling salesman problem* (TSP) is different from the shortest-path problem, since here the node set $V$ of the graph $G = (V, A)$ must be visited once and exactly once. If arcs are labelled with arc lengths we seek the tour of minimum total length. We remark again, as for the SPP, that labels on the arcs can have different meanings. For a recent survey on TSP and its variations, the reader is referred, e.g., to the book “The Travelling Salesman Problem and its Variations” by Gutin and Punnel (2006).

Manthey and Ram proposed in 2006 an excellent state-of-the-art analysis on the multi-objective travelling salesman problem. Also, Ehrgott in 2000 developed a relevant survey on the travelling salesman problem with multiple criteria in his paper on “Approximation algorithms for multi-criteria combinatorial optimization problems”. Beyond the two mentioned papers, important results on the multi-criteria TSP have been achieved by Angel et al. (2004, 2005). In this section, following the latter papers, we recall the main results in this area.

Ehrgott (2000) analyzed a generalization of Christofides’ algorithm for the TSP with triangle inequalities and the symmetry property, i.e., $\Delta$-STSP. Instead of considering Pareto curves, he measured the quality of a solution as a norm of the objective function vector, i.e., scalarizing the objectives in a single one. The approximation ratio achieved is between $\frac{3}{2}$ and 2, depending on the norm used to combine the different criteria.

Angel et al. (2004) considered the two-criteria symmetric TSP with weights one and two. They presented a $\frac{3}{2}$-approximation algorithm for this problem by using a local search heuristic.

Successively, Angel et al. (2005) generalized these results to the $k$-criteria symmetric TSP with weights one and two by presenting a $2 - \frac{2}{k+1}$-approximation, with $k \geq 3$.

More recently, in Manthey and Ram (2006), the authors presented a new approximation results for the multi-objective symmetric TSP of $(1 - \gamma - \epsilon)$-approximation with $\gamma \in [\frac{1}{2}, 1]$. 
2.4.3 Other Work in Multi-objective Combinatorial Optimization Problems

In the 2006 paper of Ehrgott et al. “A level set method for multi-objective combinatorial optimization: application to the quadratic assignment problem”, the authors studied the assignment problem in the multi-objective case.

The basic version of the well-known assignment problem (AP) can be defined on a bipartite graph. One may be interested in finding the maximum number of edges that do not have a common vertex: when AP is used, e.g., to model the problem of assigning loads to trucks or people (drivers) to vehicles this corresponds to finding the maximum number of loads assigned to trucks (where the edge between a load and a truck means compatibility). In other cases edges can be labelled, for instance, with an estimation of the payoff obtained by a certain assignment, and then the goal is to search for the set of edges with no vertex in common that maximizes the total estimated payoff. Similarly, if labels on edges represent a certain cost one may also seek for an assignment that minimizes the total cost. The reader can find algorithms and variations on this problem, e.g., in “Network Flows: Theory, Algorithms, and Applications”, by Ahuja et al. (1993).

Studying a quadratic version of the assignment problem (QAP) with multiple objectives, Erghott et al. (2006) pose a twofold goal: on the one hand, developing a procedure for the determination of Pareto-optimal solutions in a multi-criteria combinatorial optimization problem; on the other hand, demonstrating the effectiveness of the previous method on QAP. The method proposed to accomplish the former goal is based on a result due to Ehrgott et al. (1997). The routine, based on the notion of level sets and level curves, works with an arbitrary number of objectives, and uses an algorithm that solves the problem of finding a $K$ best solution in a (single-objective) combinatorial optimization problem. Regarding the second goal, i.e., solving the multi-objective quadratic assignment problem (MQAP), the proposed approach represents the first method able to handle the MQAP. Furthermore, the authors present two algorithms for ranking QAP solutions and report an experimental comparison between them on randomly generated examples with a number of objective functions ranging from 2 to 6.
2.5 Multi-objective Combinatorial Optimization by Metaheuristics

In this section, we present the state-of-the-art on metaheuristics to solve multi-objective combinatorial optimization problems. A comprehensive study on multi-objective optimization by metaheuristics can be found in the 2001 book “Multiple Objective Metaheuristic Algorithms for Combinatorial Optimization” of Andrzej Jaszkiewicz.

The first multiple objective metaheuristic proposed in the literature was due to Schaffer in 1985 (vector evaluated genetic algorithm, VEGA). It is an adaptation of a single-objective genetic algorithm with a modified selection mechanism. It was observed by Coello (1999) that the solutions generated by VEGA are in general not uniformly spread in the Pareto front; indeed, they tend to poorly represent the middle regions of the non-dominated points set. Fonseca and Fleming (1995) and Tamaki et al. (1996) proposed a modification of this method in order to circumvent such a shortcoming. More recently, evolutionary algorithms have also been tailored to solve optimization problems with multiple conflicting objectives, by approximating the Pareto set in such problems. A complete tutorial on evolutionary multi-objective optimization can be found in the papers of Zitzler et al. (2004) and Deb (2001, 2005).

A genetic/evolutionary algorithm operates on a set of candidate solutions that is subsequently modified by two basic operators: selection and variation. Selection is in charge to model the reproduction mechanism among living beings, while variation mimics the natural capability of creating new living beings by means of recombination and mutation. Two goals have to be taken into account when designing a multi-objective genetic/evolutionary algorithm: guiding the search towards the Pareto set and keeping a diverse set of non-dominated solutions. The first goal is mainly related to mating selection, in particular to the problem of assigning scalar fitness values in the presence of multiple objectives. The second goal concerns selection in general because we want to avoid the situation that the population contains mostly identical solutions (with respect to the objective space and the decision space). Finally, there is a third issue that addresses both of the above goals, i.e., elitism that prevents non-dominated solutions from being lost.

In contrast to single-objective optimization, where objective function and fitness function are directly and easily related, in multi-objective optimization fitness as-
2.5 Multi-objective Combinatorial Optimization by Metaheuristics

Assignment and selection have to take into account all the different objectives. Among the different fitness assignment strategies, the most commonly used are those based on aggregation, single-objective, and Pareto dominance.

The first approach, which mimics the weighted-sum method, aggregates the objectives into a single parameterized objective function. The parameters of this function are systematically varied during the optimization run in order to find a set of non-dominated solutions instead of a single trade-off solution. In order to spread the search over all regions of the non-dominated set various objective weights should be used. Hajela and Lin (1992) proposed to encode the weights in the solutions description. Thus, the weights evolve in results of recombination and mutation operators. Fitness sharing is used in order to achieve diversification of weights. The weights are used in linear scalarizing functions. Murata et al. (1996) proposed to draw at random a weight vector for use in each iteration composed of a single recombination.

Single-objective-based methods switch between the objectives during the selection phase. Each time an individual is chosen for reproduction, potentially a different objective will decide which member of the population will be copied into the mating pool.

The Pareto dominance (or ranking) strategy is based on exploiting the partial order on the population. Some approaches use the dominance rank, i.e., the number of individuals by which an individual is dominated, to determine the fitness values; others make use of the dominance depth, where the population is divided into several fronts and the depth reflects to which front an individual belongs. Alternatively, the dominance count, i.e., the number of individuals dominated by a certain individual, can also be taken into account. Independent of the technique used, the fitness is related to the whole population, in contrast to aggregation-based methods that calculate an individual raw fitness value independently of other individuals. Pareto ranking alone, however, does not guarantee that the population will spread uniformly over the non-dominated points set. It is known that in the case of Pareto ranking-based selection schemes finite populations converge to a single optimum, a phenomenon known as genetic drift (Goldberg and Segrest, 1987), implying a convergence to small regions of the Pareto-optimal set. Some authors worked in the direction of generating implementations able to reduce this effect (see, e.g., Fonseca and Fleming, 1995, and Srinivas and Deb, 1994).

Also simulated annealing has been used to cope with multiple objective combinatorial optimization problems. Serafini, in 1992, was the first author to propose
such an approach. The method works in a similar way with respect to the standard (single-objective) simulated annealing implementation, and produces a set of (potentially Pareto-optimal) solutions not dominated by any other solution generated by the algorithm so far. In single-objective simulated annealing a new solution whose value is not worse than the current solution is accepted with probability equal to one, while it is accepted with a probability lower than one if it has a worse value than the current solution. In the multi-objective scenario, three situations can occur when comparing the new solution $s'$ with the current solution $s$: $s'$ dominates $s$, $s'$ dominates $s$, and $s$ and $s'$ are mutually non-dominated. In the first case, $s'$ is accepted with probability equal to one; in the second case the acceptance probability is lower than one; in the third case Serafini proposed different rules corresponding to some weighted scalarizing functions, showing that these rules guarantee that the algorithm achieves one of the Pareto-optimal solutions if the temperature is decreased sufficiently slowly; while they do not guarantee appropriate dispersion over the whole set of non-dominated points. To achieve dispersion the author proposed to modify the weights randomly over time. Also, Ulungu et al. (1999) proposed multi-objective acceptance rules by using predefined weight vectors, where each of the vectors is associated with an independent annealing process. As in the case of the algorithm of Serafini (1992), the algorithm of Ulungu et al. (1999) produces the set of potentially Pareto-optimal solutions containing all the solutions not dominated by any other solution generated by any of the annealing processes. The reader can also refer to the paper of Suppapitnarm and Parks (1999) to see another implementation of simulated annealing for multi-objective problems.

Gandibleux et al. (1997) proposed a multi-objective version of tabu search. The method is based on a periodical modification of the weights to be assigned to the scalarized objective function, and makes use of two tabu lists, one that prevents already visited solutions from being reconsidered, and the other related to weight vectors. Also, Hansen in 1998 developed a multi-objective tabu search based on assigning weights to the scalarized objectives; the method borrows some ideas from Pareto-simulated annealing since it uses the same solutions dispersion mechanism achieved by a proper modification of the weight vectors.
Multi-objective Management in Freight Logistics
Increasing Capacity, Service Level and Safety with Optimization Algorithms
Caramia, M.; Dell'Olmo, P.
2008, XVI, 187 p. 32 illus., Hardcover
ISBN: 978-1-84800-381-1