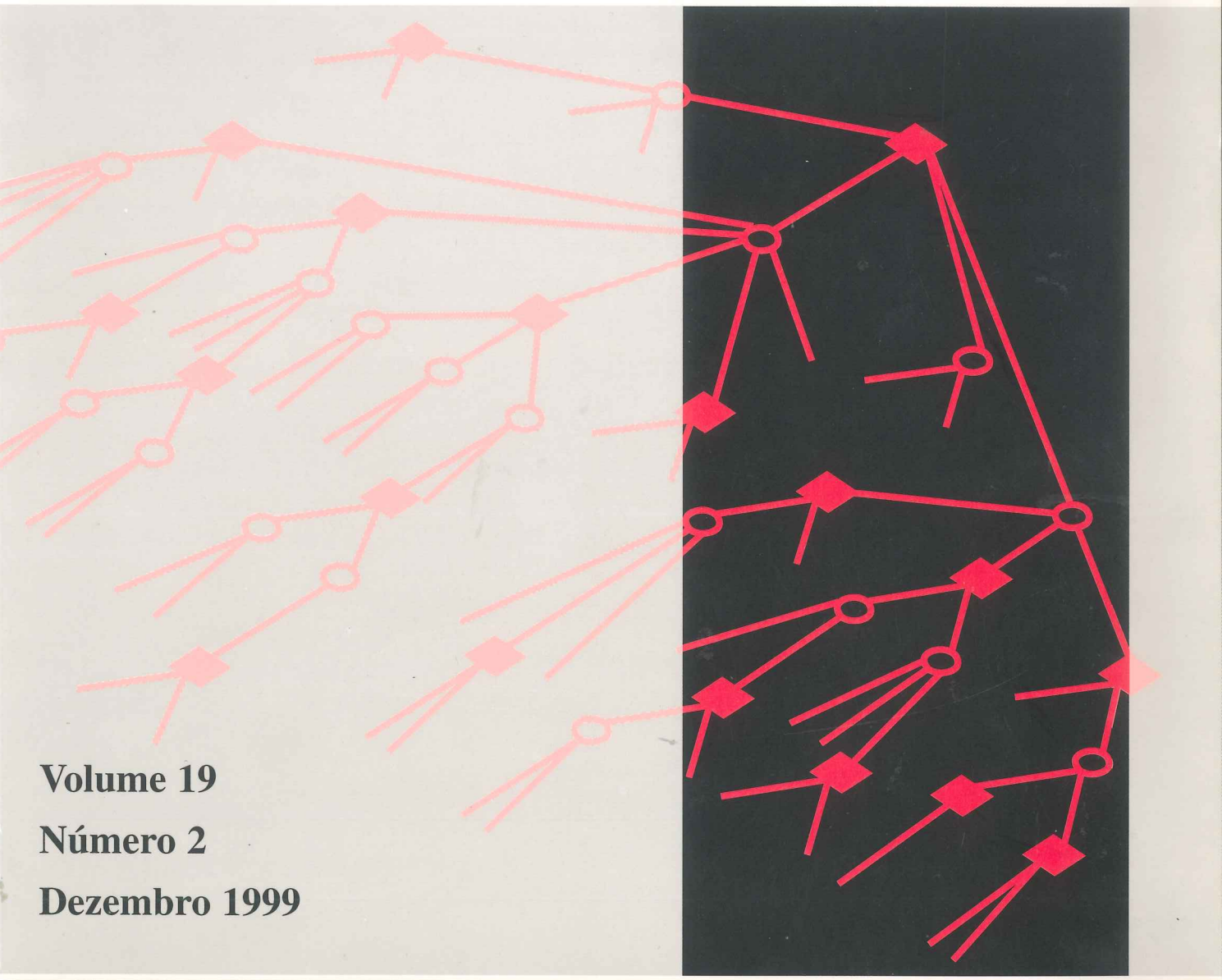
ISSN: 0874-5161

Investigação Operacional



Volume 19
Número 2
Dezembro 1999

INVESTIGAÇÃO OPERACIONAL

Propriedade:
APDIO — Associação Portuguesa de Investigação Operacional

ESTATUTO EDITORIAL

<<Investigação Operacional>>, órgão oficial da APDIO cobre uma larga gama de assuntos reflectindo assim a grande diversidade de profissões e interesses dos sócios da Associação, bem como as muitas áreas de aplicação da I. O. O seu objectivo primordial é promover a aplicação do método e técnicas da I.O. aos problemas da Sociedade Portuguesa.

A publicação acolhe contribuições nos campos da metodologia, técnicas, e áreas de aplicação e software de I. O. sendo no entanto dada prioridade a bons casos de estudo de carácter eminentemente prático.

Patrocinadores

BP PORTUGUESA

FCT
Fundação para a Ciência e a Tecnologia
MINISTÉRIO DA CIÊNCIA E DA TECNOLOGIA

Fundação Calouste Gulbenkian

ISSN nº 0874-5161
Dep. Legal nº 130 761 / 98
Execução Gráfica: J. F. Macedo - Astrofoto
700 Ex.
99/12

SOLUTION CONCEPTS FOR MULTIPLE OBJECTIVE N-PERSON GAMES*

Puerto J.

Fernández F.R.

Facultad de Matemáticas
Universidad de Sevilla

Hinojosa M.A.

Mármol A.M.

Monroy L.

Facultad de Económicas
Universidad de Sevilla

Abstract

Multiple criteria decision problems with one decision-maker haven been recognised and discussed in the literature in optimisation theory, operations research and management science. Nevertheless a multiple criteria problem can naturally arise in decision situations involving conflict among n-persons and conflict among the criteria of each person. The corresponding concept with n-decision makers, namely multiple objective n-person games, has not been extensively explored.

In this paper we consider several approaches for solving normal form multiple objective n-person games. One of them is to find optimal response strategies. In non-cooperative games, characterised by strategic behaviour and individual rationality, given all other player's strategies, each player may choose a best response as an efficient solution of a vector maximisation problem. Also a best response can be established as a maximin solution. Another approach to solve these games is based on security levels. We present the concept of Pareto optimal security strategies for multiple objective n-person games and explore the relations with the optimal response strategies.

Keywords

Game theory, multicriteria games, Nash equilibrium, maximin strategy, Pareto-optimal security strategy.

1. Introduction

Multicriteria decision making and game theory have contributed important insights to several areas of social science. Multicriteria decision making has gained broad interest and has been extensively described in the literature on operations research and decision theory over the last two decades.

* The research of the authors is partially supported by the Spanish Ministerio de Educación y Cultura grant n. PB97-07-07.

On the other hand, game theory, born in 1944 with the publication of the book "Theory of Games and Economics Behavior" by John von Neumann and Oscar Morgenstern [14], studies the behavior of decision-makers whose decisions affect the others. In the last ten years recent advances in game theory have ensured its recognition as a key subject across a wide range of disciplines in the social science.

Both theories, multicriteria decision making and game theory, jointly applied yield to important and novel results in various areas of applications. However, multiple objective games that emerge whenever players in a game have multiple objectives or a vector payoff to optimize, have attracted limited attention in the game theory literature and have not yet been extensively explored.

The first publication on multiple objective games dates back to 1956 [1] and some important papers are [17, 6, 22, 10, 4, 2, 5, 9, 23].

Multicriteria games can naturally arise in decision situations involving conflict among n persons. In practical problems, it is typical that a player deals not with one, but with several criteria which he would like to satisfy, and there is not an explicitly given utility function.

As the methodology to solve multicriteria is based in solving multiple criteria problems we do not need to scalarize all the objectives in order to get a single value function. Also, it is interesting to analyze the possible extension of the results in classical game theory, because due to the additional difficulty of dealing with multiple criteria, many of the elegant and intuitively appealing theoretical results in scalar criterion games could not hold.

Multicriteria games can be both, cooperative and non-cooperative. In this paper we study the non-cooperative case, although we also consider some cooperation and partial cooperation situations.

Our main effort in this paper, has been devoted to show that, in multicriteria games, a single solution concept in terms of equilibrium points is not sufficient. For this reason, we propose and discuss different solution concepts associated with multicriteria games.

The paper is organized as follows. In section 3 we give the preliminary terminology used throughout the paper. In section 3 we analyze four different solution concepts and some examples are included to illustrate them. Finally a section devoted to conclusions and a list of references is offered.

2. Preliminaries

In this paper we consider a multiobjective n-person game in normal form defined as $\Gamma = \{N, X^i, u^i\}$ where $N = \{1, 2,\ldots, n\}$ is the set of players. For each $i \in N$, X^i is player i's strategy set, which is assumed to be a non-empty subset in some finite-dimensional euclidean space ($X^i \subset \mathbb{R}^{L_i}$), $u^i : X = \prod_{i=1}^n X^i \to \mathbb{R}^{m_i}$ is player i's vector payoff which is a real m_i-dimensional vector function. A joint strategy is $x \in X = \prod_{i=1}^n X^i$, $x = \{x^1, x^2,\ldots, x^n\}$ where $x^i \in X^i$, and the joint vector payoff is $u \in \mathbb{R}(N) = \prod_{i=1}^n \mathbb{R}^{m_i}$, $u = \{u^1, u^2,\ldots, u^n\}$.

Notice that each player values a different number m_i of criteria. However, in order to simplify the notation, in some sections we will consider that $m_1 = m_2 = \ldots = m_n = m$.

Let $\mathfrak{N}$ denote the set of all non-empty subsets of N, then each element of $\mathfrak{N}$ represents a coalition of players. For each coalition $S \in \mathfrak{N}$, let $|S|$ denote the number of elements in S. For each joint strategy $x = \{x^1, x^2, \ldots, x^n\} \in X$ let x_S denote the strategy of coalition S, x_{-S} the strategy of the players in the complementary coalition N-S. For the payoff vector associated, $u = \{u^1, u^2, \ldots, u^n\} \in \mathbb{R}(N)$, we denote by u_S and u_{-S} the projection of u on $\mathbb{R}(S)$ and $\mathbb{R}(-S)$ respectively:

$$x_S = \{x^i / i \in S\} \in X_S = \prod_{i \in S} X^i$$

$$x_{-S} = \{x^i / i \notin S\} \in X_{-S} = \prod_{i \notin S} X^i$$

$$u_S = \{u^i / i \in S\} \in \mathbb{R}(S) = \prod_{i \in S} \mathbb{R}^{m_i}$$

$$u_{-S} = \{u^i / i \notin S\} \in \mathbb{R}(-S) = \prod_{i \notin S} \mathbb{R}^{m_i}.$$

By $\mathbb{R}(S)_+$ we denote the positive orthant of $\mathbb{R}(S)$.

Recall that a partition of players $\{1, 2, \ldots, n\}$ in the game Γ is a collection of coalitions $\Delta = \{S_1, S_2, \ldots, S_k\}$ such that

$$\bigcup_{i=1}^{k} S_i = N$$

$$S_i \cap S_j = \varnothing \quad \forall i \neq j.$$

Example:

Consider the bicriteria three-person game represented in Table 1. Each player can play two different strategies, A and B, and the results of the game are evaluated in two different scenarios or with respect to two different criteria.

$$N = \{I, II, III\}$$

$$X^1 = \{IA, IB\} \quad X^2 = \{IIA, IIB\} \quad X^3 = \{IIIA, IIIB\}.$$

$$X = \prod_{i=1}^{3} X^i$$

In this example with finite joint strategy set the payoff functions are discrete and:

$$\forall x \in X \quad u^i(x) \in \mathbb{R}^2, \; u = \{u^1, u^2, u^3\} \text{ and so } u(x) \in \mathbb{R}^6$$

For instance, when player I plays IA, player II plays IIB and player III plays IIIA, that is to say, x = (IA, IIB, IIIA) the payoffs are:

$$u^1(x) = \begin{pmatrix} -50 \\ 50 \end{pmatrix} \quad u^2(x) = \begin{pmatrix} 0 \\ -100 \end{pmatrix} \quad u^3(x) = \begin{pmatrix} 50 \\ 50 \end{pmatrix}$$

$$u(x) = (-50 \;\; 50 \;\; 0 \;\; -100 \;\; 50 \;\; 50)^t$$

This means that player I obtains -50 in his first criterion and 50 in his second criterion. Player II obtains 0 in his first criterion and -100 in his second criterion. Player III obtains 50 in his first criterion and 50 in his second criterion.

	IIIA		IIIB	
I \ II	IIA	IIB	IIA	IIB
IA	(-50,-50,50) (0,0,0)	(-50,0,50) (50,-100,50)	(-50,-50,0) (50,50,-100)	(100,0,0) (100,-50,-50)
IB	(0,-50,50) (0,50,50)	(0,0,100) (0,-50,-90)	(0,100,0) (50,100,-50)	(0,0,0) (0,0,0)

Table 1 - Game payoffs

3. Solution Concepts

3.1 Nash equilibria

The concept of Nash equilibrium in conventional game theory can be extended to vector game theory. Each player i takes other's strategies as given and chooses a "best response" as an efficient solution of a vector maximization utility problem. Then there are a variery of solutions that can be chosen as the "best response": properly efficient solutions, [23], efficient solutions, [17,5], weakly efficient solutions, [12]. In this paper weakly efficient solutions are chosen as the best response. In what follows we will refer to weak efficiency as efficiency.

Definition 3.1 A joint strategy $\bar{x} = \{\bar{x}^1, \bar{x}^2, \ldots, \bar{x}^n\} \in X$ is a Nash equilibrium of the multicriteria game $\Gamma = \{N, X^i, u^i\}$ if $\forall i \in N$, $\bar{x}^i$ is an efficient solution of the vector maximization problem:

$$\max_{x^i \in X^i} u^i(x^i, \bar{x}^{-i}) \text{ where } \bar{x}^{-i} = \{\bar{x}^1, \bar{x}^2, \ldots, \bar{x}^{i-1}, \bar{x}^{i+1}, \ldots, \bar{x}^n\} \quad (1)$$

This solution concept is a noncooperative solution characterized by the strategic behavior and the individual rationality. It requires that each player chooses a best response given all other players' strategies, and at the equilibrium no player has any incentive to deviate alone. However Nash equilibrium is a local solution. If there are several equilibria, the players have no compelling reason to choose among them.

Example (continued):

The Nash equilibria of the game whose payoffs are in Table 1 are the joint strategies:

x_1 = (IA, IIA, IIIA), x_2 = (IA, IIB, IIIA), x_3 = (IB, IIB, IIIA),

x_4 = (IB, IIA, IIIB), x_5 = (IB, IIA, IIIA)

For instance x_3 = (IB, IIB, IIIA)$\in$ X is a Nash equilibrium:

Player I obtains $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$. If the strategies of player II and player III are fixed, he can obtain $\begin{pmatrix} -50 \\ 50 \end{pmatrix}$ also, but this last payoff is not strictly better than the other.

Player II obtains $\begin{pmatrix} 0 \\ -50 \end{pmatrix}$. Assuming now I and III's strategies as given, he can also obtain $\begin{pmatrix} -50 \\ 50 \end{pmatrix}$, which is not strictly better than the other payoff.

Finally, III obtains $\begin{pmatrix} 100 \\ -90 \end{pmatrix}$. If I and II's strategies are fixed, he can obtain too $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$, but again, it is not strictly better than the other.

$x = (IB, IIB, IIIB) \in X$ is a Nash equilibrium because if I and III's strategies are fixed, II obtains $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$ but can also obtain $\begin{pmatrix} -100 \\ 100 \end{pmatrix}$. This payoff is strictly better than $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$.

Consider that there is a partition of players, then we have coexistence of competition across coalition and cooperation within each coalition. In the Nash equilibrium each player takes all other's strategies as given, because no cooperation with other players is allowed.

This strategic behavior can be generalized to a group of players: each coalition takes all other coalitions' strategies as given, because no cooperation with other coalition is allowed. Thus given coalition $S \in \mathfrak{N}$ and given the complementary strategies $\bar{x}_{-S}$, the coalition S has to solve the vector maximization utility problem:

$$\max_{x_S \in X_S} \; u_S(x_S, \bar{x}_{-S}) \tag{2}$$

where the complementary strategies x_{-S} are fixed parameters and there are $\sum_{i \in S} m_i$ objectives to be maximized. This leads to the following concept of S-efficiency:

Definition 3.2 For any coalition $S \in \mathfrak{N}$, a joint strategy $\bar{x} = \{\bar{x}^1, \bar{x}^2, \ldots, \bar{x}^n\} \in X$ is a S-efficient solution of the game $\Gamma = \{N, X^i, u^i\}$ if $\bar{x}_S$ is an efficient solution of the vector maximization problem (2).

Notice that a Nash equilibrium of the game Γ is a S-efficient solution of the game Γ for any coalition with only a player.

Suppose there is a partition $\Delta = \{S_1, S_2, \ldots, S_k\}$. If we consider a S-efficient solution for each $S \in \Delta$, we will have a hybrid solution concept between cooperative and non-cooperative solution concepts that we call a Nash equilibrium for the partition Δ:

Definition 3.3 For each partition of players $\Delta = \{S_1, S_2, \ldots, S_k\}$ in the game $\Gamma = \{N, X^i, u^i\}$ a joint strategy $\bar{x} = \{\bar{x}^1, \bar{x}^2, \ldots, \bar{x}^n\} \in X$ is a Nash equilibrium corresponding to Δ if $\forall S \in \Delta$, $\bar{x}_S$ is an efficient solution of the vector maximization problem (2) where $\bar{x}_S$ are the given strategies of the other players.

Notice that this kind of solutions generalizes the concept of Nash equilibrium because we obtain a Nash equilibrium when each coalition in the partition of the game Γ has only one player.

An existence theorem for these solutions is in [23].

Theorem 3.1 Given a partition of players $\Delta = \{S_1, S_2, \ldots, S_k\}$ in the game $\Gamma = \{N, X^i, u^i\}$ the set of the corresponding Nash equilibrium to Δ is non-empty if Γ satisfies:

1. For each player i, X^i is a closed bounded convex subset in $\mathbb{R}^{L_i}$.

2. For each coalition $S \in \Delta$,: $j \in S$ are all continuous in $x = (x_S, x_{-S})$ and are quasiconcave in x_S.

Notice that this theorem generalizes the earlier existence theorems in [13], [17] and [5].

It is difficult to solve the multicriteria problems involed in the computation of Nash equilibria strategies. When more restricting conditions hold, such as concavity instead of quasiconcavity, it could be computed some of the solutions based in weighted criteria. Notice that a player has to know the other players' strategies to make his choice because of the local character of this kind of solutions.

3.2 Maximin Solution

In this section we analyze how a player or a coalition can choose the strategy without knowing the other player's choice. We will work from the point of view of a coalition $S \in \mathfrak{N}$.

Players in S will try to maximize their payoffs. Being pessimist, for each strategy they can play, they consider the worst possible payoff. With this information they will choose the strategy with the better worst payoff. That is to say the maximin values of their utility function. Hence, for any $x_S \in X_S$, we will consider the efficient solutions of the vector minimization problem:

$$\min_{x_{-S} \in X_{-S}} u_S(x_S, x_{-S}) \qquad (3)$$

The maximin values set for the function u_S is:

$$\max \bigcup_{x_S \in X_S} \min_{x_{-S} \in X_{-S}} u_S(x_S, x_{-S})$$

We call a maximin strategy of coalition S to a strategy attaining a maximin value.

As it is shown in [15], if the strategy set is a compact set, the maximin values set is non-empty.

Example (continued):

In order to find the maximin strategies for player I, we first consider the worst possible payoffs when he plays strategy IA (efficient for the vector minimization problem):

$$\min\left\{\begin{pmatrix}-50\\0\end{pmatrix}\begin{pmatrix}-50\\50\end{pmatrix}\begin{pmatrix}-50\\50\end{pmatrix}\begin{pmatrix}100\\100\end{pmatrix}\right\} = \left\{\begin{pmatrix}-50\\0\end{pmatrix}\begin{pmatrix}-50\\50\end{pmatrix}\right\}.$$

The worst possible payoffs when he plays strategy IB:

$$\min\left\{\begin{pmatrix}0\\0\end{pmatrix}\begin{pmatrix}0\\0\end{pmatrix}\begin{pmatrix}0\\50\end{pmatrix}\begin{pmatrix}0\\0\end{pmatrix}\right\} = \left\{\begin{pmatrix}0\\0\end{pmatrix}\begin{pmatrix}0\\50\end{pmatrix}\right\}.$$

Next we seek for the maximum values, that is to say:

$$\max\left\{\begin{pmatrix}-50\\0\end{pmatrix}\begin{pmatrix}-50\\50\end{pmatrix}\begin{pmatrix}0\\0\end{pmatrix}\begin{pmatrix}0\\50\end{pmatrix}\right\} = \left\{\begin{pmatrix}-50\\50\end{pmatrix}\begin{pmatrix}0\\0\end{pmatrix}\begin{pmatrix}0\\50\end{pmatrix}\right\}.$$

Hence, maximin values can be attained either with strategy IA or with strategy IB and both are maximin strategies.

Notice that player I does not know which of the minimal values he is going to get when playing a maximin strategy. Notice also that the valuation of each strategy, in general, is not a vector but a set of vectors, what makes difficult the task of comparing them.

Now, we are going to obtain the maximin strategies for the fixed coalition S = {1,3}. The joint strategies for coalition S are:

$$x_S^1 = (IA, IIIA),\ x_S^2 = (IA, IIIB),\ x_S^3 = (IB, IIIA),\ x_S^4 = (IB, IIIB)$$

The worst possible joint payoff for coalition S are:

$$\min_{x_{-S}\in X_{-S}} u_S(x_S^1, x_{-S}) = \min\left\{\begin{pmatrix}-50\\0\\50\\0\end{pmatrix}\begin{pmatrix}-50\\50\\50\\50\end{pmatrix}\right\} = \left\{\begin{pmatrix}-50\\0\\50\\0\end{pmatrix}\begin{pmatrix}-50\\50\\50\\50\end{pmatrix}\right\}$$

$$\min_{x_{-S}\in X_{-S}} u_S(x_S^2, x_{-S}) = \min\left\{\begin{pmatrix}-50\\50\\0\\-100\end{pmatrix}\begin{pmatrix}100\\100\\0\\-50\end{pmatrix}\right\} = \left\{\begin{pmatrix}-50\\50\\0\\-100\end{pmatrix}\begin{pmatrix}100\\100\\0\\-50\end{pmatrix}\right\}$$

$$\min_{x_{-S}\in X_{-S}} u_S(x_S^3, x_{-S}) = \min\left\{\begin{pmatrix}0\\0\\50\\50\end{pmatrix}\begin{pmatrix}0\\0\\100\\-90\end{pmatrix}\right\} = \left\{\begin{pmatrix}0\\0\\50\\50\end{pmatrix}\begin{pmatrix}0\\0\\100\\-90\end{pmatrix}\right\}$$

$$\min_{x_{-S}\in X_{-S}} u_S(x_S^4, x_{-S}) = \min\left\{\begin{pmatrix}0\\50\\0\\-50\end{pmatrix}\begin{pmatrix}0\\0\\0\\0\end{pmatrix}\right\} = \left\{\begin{pmatrix}0\\50\\0\\-50\end{pmatrix}\begin{pmatrix}0\\0\\0\\0\end{pmatrix}\right\}$$

The maximin values are obtained from the comparison of these sets of vectors.

Strictly competitive games are an interesting particular case. In these conditions coalition S plays against the other players that form the coalition -S, maximizing his utility function $u_S(x_S, x_{-S})$. The goal of -S will be to minimize the coalition S utility function. To this end, S will look for a maximin strategy whereas -S will look for a minimax strategy.

We say that $\bar{x}_S \in X_S$ is a "optimal response" strategy for coalition S against the given strategy $\bar{x}_{-S} \in X_{-S}$ of the other players if $\bar{x}_S$ is an efficient solution for the vector maximization problem:

$$\max_{x_S\in X_S} u_S(x_S, \bar{x}_{-S}) \tag{4}$$

Similarly, we say that $\bar{x}_{-S} \in X_{-S}$ is a "optimal response" strategy for coalition -S against the given strategy $\bar{x}_S \in X_S$ of the other players if $\bar{x}_{-S}$ is an efficient solution for the vector minimization problem:

$$\max_{x_{-S}\in X_{-S}} u_S(\bar{x}_S, x_{-S}) \tag{5}$$

The set of all optimal response strategies of a coalition S against the opponent's given strategy $\bar{x}_{-S} \in X_{-S}$ is defined by $R_S(\bar{x}_{-S})$ and the set of all optimal response strategies of a coalition -S against the opponent's given strategy $\bar{x}_S \in X_S$ is defined by $R_{-S}(\bar{x}_S)$. For convenience, we denote each optimal response set as follows:

$$D_1 = \{(x_S, x_{-S}) \in X / x_{-S} \in R_{-S}(x_S),\ x_S \in X_S\}$$

$$D_2 = \{(x_S, x_{-S}) \in X / x_S \in R_S(x_{-S}),\ x_{-S} \in X_{-S}\}.$$

As we defined in definition 3.3, a joint strategy $\bar{x} = (\bar{x}_S, \bar{x}_{-S}) \in X$ is said to be a Nash equilibrium of the game for the partition $\Delta = \{S, -S\}$ if:

$$\bar{x}_S \in R_S(\bar{x}_{-S}) \text{ and } \bar{x}_{-S} \in R_S(\bar{x}_{-S}) \Leftrightarrow \bar{x} \in D_1 \cap D_2 \Leftrightarrow$$

$$u_S(\bar{x}_S, \bar{x}_{-S}) \in \min_{x_{-S} \in X_{-S}} u_S(\bar{x}_S, x_{-S}) \cap \max_{x_S \in X_S} u_S(x_S, \bar{x}_{-S})$$

We call an equilibrium value or saddle value to $u_S(\bar{x}_S, x_{-S})$ where $(\bar{x}_S, \bar{x}_{-S}) \in X$ is an equilibrium strategy or generalized saddle point. For convenience we will denote the set of all saddle values of u_S by $SV(u_S)$.

The set of all maximin values of u_S and the set of all minimax values of u_S respectively are:

$$\max \bigcup_{x_S \in X_S} \min_{x_{-S} \in X_{-S}} u_S(x_S, x_{-S}) \equiv \max u_S(D_1)$$

$$\min \bigcup_{x_{-S} \in X_{-S}} \max_{x_S \in X_S} u_S(x_S, x_{-S}) \equiv \min u_S(D_2).$$

Also we call a strategy $\bar{x}_S$ (respectively $\bar{x}_{-S}$) attaining a maximin value (respectively a minimax value) a maximin (respectively minmax) strategy.

The following theorems shows that under certain conditions, there exists at least an optimal response strategy for each coalition against and opponent's given strategy, and that there exist a maximin strategy and a minimax strategy. This results is an extension of Lema 5.5. given in [21].

Theorem 3.2 Given a coalition $S \in \mathfrak{N}$ let X_S and X_{-S} be nonempty compact subsets of $\mathbb{R}(S)$ and $\mathbb{R}(-S)$ respectively. If the vector-valued function u_S is continuous then for each $\overset{*}{x}_S \in X_S$ and $\overset{*}{x}_{-S} \in X_{-S}$:

$$\varnothing \neq \min_{x_{-S} \in X_{-S}} u_S(\overset{*}{x}_S, x_{-S}) \subset \max \bigcup_{x_S \in X_S} \min_{x_{-S} \in X_{-S}} u_S(x_S, x_{-S}) - \mathbb{R}(S)_+$$

$$\varnothing \neq \max_{x_{-S} \in X_{-S}} u_S(x_S, \overset{*}{x}_{-S}) \subset \min \bigcup_{x_{-S} \in X_{-S}} \max_{x_S \in X_S} u_S(x_S, x_{-S}) + \mathbb{R}(S)_+.$$

Unfortunately, maximin strategies do not always provide equilibrium values, but may give equilibrium values in the sense of security levels when maximin strategies satisfy certain conditions. Thus, we note that the maximin and minimax values are upper and lower bounds of such equilibrium values, respectively.

Corollary 3.1 For a coalition $S \in \mathfrak{N}$, if X_S and X_{-S} are non-empty compact subsets of $\mathbb{R}(S)$ and $\mathbb{R}(-S)$ and u_S is continuous then

$$SV(u_S) \subset \max \bigcup_{x_S \in X_S} \min_{x_{-S} \in X_{-S}} u_S(x_S, x_{-S}) - \mathbb{R}(S)_+$$

$$SV(u_S) \subset \min \bigcup_{x_{-S} \in X_S} \max_{x_{-S} \in X_{-S}} u_S(x_S, x_{-S}) + \mathbb{R}(S)_+.$$

Hence, if the game S against -S has a joint equilibrium strategy $\bar{x} = (\bar{x}_S, \bar{x}_{-S}) \in X$, then there exist:

$$z_2 \in \max \bigcup_{x_S \in X_S} \min_{x_{-S} \in X_{-S}} u_S(x_S, x_{-S}) - \mathbb{R}^{S_+}$$

$$z_1 \in \min \bigcup_{x_S \in X_S} \max_{x_{-S} \in X_{-S}} u_S(x_S, x_{-S}) + \mathbb{R}^{S_+}$$

such that $u_S(\bar{x}_S, \bar{x}_{-S}) \leq z_2$ and $u_S(\bar{x}_S, \bar{x}_{-S}) \geq z_1$. We will say that the maximin inequality holds if $z_1 \leq z_2$. In [19], [20] and [21] we can find conditions under which this maximin inequality holds.

Example (continued): In the case that player II is interested only in minimizing the coalition S = {1,3} joint payoff, he will seek for a minimax strategy.

$$\max_{x_S \in X_S} u_S(x_S, IIA) = \max \left\{ \begin{pmatrix} -50 \\ 0 \\ 50 \\ 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 50 \\ 50 \end{pmatrix} \begin{pmatrix} -50 \\ 50 \\ 0 \\ -100 \end{pmatrix} \begin{pmatrix} 0 \\ 50 \\ 0 \\ -50 \end{pmatrix} \right\}$$

$$\max_{x_S \in X_S} u_S(x_S, IIB) = \max \left\{ \begin{pmatrix} -50 \\ 50 \\ 50 \\ 50 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 100 \\ -90 \end{pmatrix} \begin{pmatrix} 100 \\ 100 \\ 0 \\ -50 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \right\}$$

Among these values player II will choose the minima.

3.3 Pareto-optimal security strategies (POSS)

We have seen in section 3.2 that the multicriteria extension of the concepts of maximin strategies is difficult and looses many of its interesting properties. Although, in some cases it is possible to establish the existence of such strategies, it is usually difficult to obtain them and in most cases the values provided are not unique. Thus when it is not possible to obtain maximin solutions, the concept of POSS becomes important in order to solve multicriteria games from a conservative point of view. This concept is independent of the notion of equilibrium, so that the opponents are only taken into account to establish the security levels for one's own payoff.

In this section we look for solutions to the n-person multiobjective game under the point of view of a coalition $S \in \mathfrak{N}$. We assume that all the players value the same m criteria. Players in S will use the security level vector defined as follows:

Definition 3.4 The security level vector or guaranteed payoff vector to coalition $S \in \mathfrak{N}$ is:

$$V_S : X_S \to \mathbb{R}(S)$$
$$V_S(x_S) = \left\{ \min_{x_{-S} \in X_{-S}} u_j^i(x_S, x_{-S}) : i \in S, \ j = 1, 2, \ldots, m \right\}.$$

For each $i \in S$

$$V_S(x_S)_i = \left\{ \min_{x_{-S} \in X_{-S}} u_j^i(x_S, x_{-S}) : j = 1, 2, \ldots, m \right\} \in \mathbb{R}^m$$

is player i's guaranteed vector payoff, given coalition's choice x_S. Then

$$V_j^i(x_S) = \min_{x_{-S} \in X_{-S}} u_j^i(x_S, x_{-S}) : j = 1, 2, \ldots, m.$$

Coalition $S \in \mathfrak{N}$ will try to choose a strategy among all the possible such that the associated security vector be as best as possible:

Definition 3.5 A strategy $\bar{x}_S \in X_S$ is a POSS for coalition S if it is an efficient solution of the vector maximization problem:

$$\max_{x_S \in X_S} V_S(x_S). \tag{6}$$

Problem (6) is equivalent to:

$$\max \left\{ V_1^i(x_S), V_2^i(x_S), \ldots, V_m^i(x_S), i \in S \right\}$$

$$\text{s.t.: } x_S \in X_S$$

By scalarization we will characterize efficient solutions in the multiple objective problem. When payoff functions, u_i, are concave, the problem is convex. In this case we obtain (see [16]) the efficient solutions of the problem by solving the associated nonlinear scalar problem $P(\lambda)$:

$$\max \sum_{\substack{j=1 \\ i \in S}}^{m} \lambda_j^i V_j^i$$
$$\text{s.t. } x_S \in X_S$$

where

$$\lambda \in \Lambda^0 = \left\{ \lambda \in \mathbb{R}(S) / \sum_{\substack{j=1 \\ i \in S}}^{m} \lambda_j^i = 1, \ \lambda_j^i \geq 0, \ j = 1, \ldots, m, \ i \in S \right\}$$

Each component λ_j^i of parameter λ can be seen as the relative importance that the coalition S assigns to the scalar game with scalar payoff function u_j^i.

For strictly competitive games, the security level vector for the opponent coalition -S is:

$$V_{-S} : X_{-S} \to \mathbb{R}(-S)$$
$$V_{-S}(x_{-S}) = \left\{ \max_{x_S \in X_S} u_j^i(x_S, x_{-S}) \ i \in S, \ j = 1, 2, \ldots, m \right\}$$

The set of POSS for coalition -S is the set of efficient solutions of the vector minimization

problem:

$$\min_{x_{-S}\in X_{-S}} \quad V_{-S}(x_{-S}). \tag{7}$$

Notice that if $(\bar{x}_S, \bar{x}_{-S})\in X$ is an equilibrium, then:

$$V_S(\bar{x}_S) \le u_S(\bar{x}_S, \bar{x}_{-S}) \le V_{-S}(\bar{x}_{-S})$$

Conversely if ($\bar{x}_S$ and $\bar{x}_{-S}$) are strategies such that $V_S(\bar{x}_S) = V_{-S}(\bar{x}_{-S})$, then $(\bar{x}_S, \bar{x}_{-S})$ is an equilibrium of the game.

In effect $\bar{x}_{-S}$ is a solution of problem (5) because if $\exists x^*_{-S}\in X_{-S}/u_S(\bar{x}_S, x^*_{-S}) \le u_S(\bar{x}_S, \bar{x}_{-S})$ then:

$$V_S(\bar{x}_S) \le u_S(\bar{x}_S, x^*_{-S}) \le u_S(\bar{x}_S, \bar{x}_{-S}) \le V_{-S}(\bar{x}_{-S}).$$

The inequalities above are really equalities; similarly we can see that $\bar{x}_S$ is a solution of problem (4).

A necessary and sufficient condition for $V_S(\bar{x}_S) = V_{-S}(\bar{x}_{-S})$ is that $\bar{x}_S$ and $\bar{x}_{-S}$ be ideal strategies for coalition S and coalition -S respectively; that is, $\bar{x}_S$ maximizes $V_S(x_S)$ simultaneously in all its components and $\bar{x}_{-S}$ minimizes $V_{-S}(x_{-S})$ simultaneously all its components. This result generalizes theorem 4.1 in [8] because $\forall j = 1, 2,\ldots, m$ each ideal strategy is an equilibrium for the scalar game with payoff function $u^i_j(x)$.

Example (continued):

To obtain the POSS for player I we compute the security level vectors:

$$V^1_1(IA) = \min\{-50,-50,-50,100\} = -50$$

$$V^1_2(IA) = \min\{0,50,50,100\} = 0$$

Then

$$V_1(IA) = \begin{pmatrix} -50 \\ 0 \end{pmatrix}$$

Analogously:

$$V_1(IB) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Now, in order to make the choice, player I has to compare two vectors.

The security level vectors for player III are:

$$V_3(IIIA) = \begin{pmatrix} 50 \\ -90 \end{pmatrix}$$

$$V_3(IIIB) = \begin{pmatrix} 0 \\ -100 \end{pmatrix}$$

We can also compute the POSS for any coalition. For coalition S = {1,3}, the security level vectors are:

$$V^1_{\{1,3\}_1}(IA,IIIA) = \min\{-50,-50\} = -50$$

$$V^1_{\{1,3\}_2}(IA,IIIA) = \min\{0,50\} = 0$$

Then

$$V^1_{\{1,3\}}(IA,IIIA) = \begin{pmatrix} -50 \\ 0 \end{pmatrix}.$$

Analogously $V^3_{\{1,3\}}(IA,IIIA) = \begin{pmatrix} 50 \\ 0 \end{pmatrix}$ and then $V_{\{1,3\}}(IA,IIIA) = \begin{pmatrix} -50 \\ 0 \\ 50 \\ 0 \end{pmatrix}$.

Similarly:

$$V_{\{1,3\}}(IA,IIIB) = \begin{pmatrix} -50 \\ 50 \\ 0 \\ -100 \end{pmatrix}, V_{\{1,3\}}(IB,IIIA) = \begin{pmatrix} 0 \\ 0 \\ 50 \\ -90 \end{pmatrix}, V_{\{1,3\}}(IB,IIIB) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ -50 \end{pmatrix}.$$

Obtaining the optimal strategies in n-person games with multiple objectives is very difficult with any of the proposed solution concepts. Nevertheless, the set of POSS can be obtained imposing additional properties in the payoff functions of the game. This is the case of multiple objective n-person games with multidimensional-matrix payoff.

Let $P^i = \{e^i_1, e^i_2, \ldots, e^i_{L_i}\}$ denote the pure strategies of player i.

Let $A = [A^1, A^2, \ldots, A^n]$ be the payoff function where $A^i = [A^i(1), A^i(2), \ldots, A^i(m)]$ and $A^i(j)$ is a matrix with L_i rows (pure strategies of player i) and $\prod_{\substack{j=1 \\ j\neq i}}^{n} L_j$ columns (pure joint strategies of the players in $N\backslash\{i\}$).

Example (continued):

In this example there are two objective functions. Therefore, the payoff matrices for player 1 are:

$$A^1 = [A^1(1)\ A^1(2)] = \left[\begin{bmatrix} -50 & -50 & -50 & 100 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 50 & 50 & 100 \\ 0 & 0 & 50 & 0 \end{bmatrix}\right].$$

Let $S = \{i_1, i_2, \ldots, i_S\}$ be a coalition of players. The set of pure strategies for the coalition S is given by $P^S = \prod_{i\in S} P^i$ and its payoff function is a vector of matrices $A^S = [A^S(1), A^S(2), \ldots, A^S(m)]$ where each $A^S(j)$ is a matrix with $|P^S|$ rows and $|P^{-S}|$ columns whose elements are the payoffs received in the j-th objective by the players which belong to S:

$$A^S(j) = \left(a^{i1}_{rt}(j), a^{i2}_{rt}(j), \ldots, a^{i|S|}_{rt}(j)\right)_{r\in P^S, t\in P^{-S}}$$

The following theorem reduces the computation of POSS, as defined in definition 3.5, to obtain the efficient solutions of a multiobjective linear problem.

Theorem 3.3 The set of POSS strategies and the corresponding security level vector for a coalition $S = \{i_1, i_2, \ldots, i_{|S|}\}$ in a multiple objective n-person game with multidimensional matrix payoff A, is given by the efficient solutions of the problem:

$$\max \quad \left(v_1^{i1}, v_2^{i1}, \ldots, v_m^{i1}, \ldots, v_1^{i|S|}, v_2^{i|S|}, \ldots, v_m^{i|S|}\right)$$

$$\text{s.t.:} \quad x_S(A^S(1), A^S(2), \ldots, A^S(m)) \geq \begin{bmatrix} v_1^{i1} \\ v_1^{i2} \\ \ldots \\ v_1^{i|S|} \\ \ldots \\ v_m^{i1} \\ v_m^{i2} \\ \ldots \\ v_m^{i|S|} \end{bmatrix}.$$

$$x_S \in X_S$$

The proof runs analogously to theorem 3.1 in [7] applied to the amalgation of the game induced for the player in N\S (see [3] for the definition of amalgation of games).

Example (continued):

The security level vector and the set of POSS for the coalition S = {1} is obtained solving the problem:

$$\begin{aligned} \max \quad & (v_1, v_2) \\ \text{s.t.:} \quad & -50x_1 \geq v_1 \\ & 100x_1 \geq v_1 \\ & 0 \geq v_2 \\ & 50x_1 \geq v_2 \\ & 50x_1 + 50x_2 \geq v_2 \\ & 100x_1 \geq v_2 \\ & x_1 + x_2 = 1 \\ & x_i \geq 0, \; i = 1, 2 \\ & v_i, \quad i = 1, 2. \end{aligned}$$

Using the software package ADBASE (see [18]), the POSS of this problem is:

$\bar{x}_{\{1\}} = (0, 1)$, $V_{\{1\}}(\bar{x}_{\{1\}}) = (0, 0)$.

Therefore, in this example there is a unique POSS of player I which is IB and the security levels are 0 in both payoff functions.

If we consider coalition S = {2} the set of POSS is given by the convex hull of the strategies $\bar{x}_{\{2\}} = (1, 0)$ and $\bar{y}_{\{2\}} = (0, 1)$ whose associated security level vectors are $V_{\{2\}}(\bar{x}_{\{2\}}) = (0, -100)$ and $V_{\{2\}}(\bar{y}_{\{2\}}) = (-50, 0)$ respectively. Analogously, for the coalition S = {3} we obtain the set of POSS as the convex hull of $\bar{x}_{\{3\}} = (13/25, 12/25)$ and $\bar{y}_{\{3\}} = (1, 0)$, and the corresponding security level vectors are $V_{\{3\}}(\bar{x}_{\{3\}}) = (658/25, -1684/25)$ and $V_{\{3\}}(\bar{y}_{\{3\}}) = (-50, -90)$.

Example: Consider a bicriteria three-person game in strategic form. The pure strategy sets for player I, player II and player III are:

$$P^1 = \{IA, IB, IC\},\ P^2 = \{IIA, IIB, IIC\},\ P^3 = \{IIIA, IIIB\},$$

respectively. The payoff matrices for this game are given in Table 2.

For the coalition $S = \{1, 3\}$ the joint pure strategy set is:

$$P^S = \{(IA, IIIA), (IB, IIIA), (IC, IIIA), (IA, IIIB), (IB, IIIB), (IC, IIIB)\}$$

and $P^{-S} = \{IIA, IIB, IIC\}$.

The payoff matrix for the coalition $S = \{1, 3\}$ is:

$$A^S = [A^S(1), A^S(2)]$$

with

$$A^S(1) = \begin{bmatrix} (1,-1) & (1,-1) & (1,-1) \\ (0,0) & (1,0) & (1,0) \\ (0,0) & (1,0) & (1,0) \\ (-2,1) & (0,1) & (0,1) \\ (0,2) & (-2,2) & (-2,2) \\ (1,-1) & (0,-1) & (0,-1) \end{bmatrix}$$

and

$$A^S(2) = \begin{bmatrix} (0,2) & (2,1) & (0,2) \\ (2,0) & (0,1) & (2,0) \\ (1,-1) & (0,0) & (1,-1) \\ (1,0) & (1,0) & (1,0) \\ (0,2) & (2,-1) & (0,2) \\ (0,0) & (-1,0) & (0,0) \end{bmatrix}$$

	IIIA			IIIB		
	IIA	IIB	IIC	IIA	IIB	IIC
IA	(1,2,-1) (0,1,2)	(1,0,-1) (2,0,1)	(1,1,-1) (0,-1,2)	(-2,0,1) (1,1,0)	(0,0,1) (1,0,0)	(0,-1,1) (1,-1,0)
IB	(0,1,0) (2,0,0)	(1,1,0) (0,1,1)	(1,-1,0) (2,-1,0)	(0,0,2) (0,1,2)	(-2,1,2) (2,0,-1)	(-2,-2,2) (0,0,2)
IC	(0,2,0) (1,0,-1)	(1,0,0) (0,1,0)	(1,2,0) (1,-1,-1)	(1,1,-1) (0,2,0)	(0,-1,-1) (-1,1,0)	(0,2,-1) (0,-1,0)

Table 2 - Payoffs matrices

In order to compute the POSS for the coalition S = {1, 3} and the security level vector associated, we solve the following multiobjective linear problem:

$$\begin{aligned}
\max\ & (v_1^1, v_2^1, v_1^3, v_2^3)\\
\text{s.t.:}\ & x_1 - 2x_4 + x_6 \geq v_1^1\\
& x_1 + x_2 + x_3 - 2x_5 \geq v_1^1\\
& 2x_2 + x_3 + x_4 \geq v_2^1\\
& 2x_1 + x_4 + 2x_5 - x_6 \geq v_2^1\\
& -x_1 + x_4 + 2x_5 - x_6 \geq v_1^3\\
& 2x_1 - x_3 + 2x_5 \geq v_2^3\\
& x_1 + x_2 - x_5 \geq v_2^3\\
& x_1 + x_2 + x_3 + x_4 + x_5 + x_6 = 1\\
& x_i \geq 0,\ i = 1, 2\\
& v_j^i\ \ i = 1, 3\ \ j = 1, 2
\end{aligned}$$

The extreme efficient solutions of this problem lead to the following POSS strategies and security level vectors:

Strategies $\bar{x}_S$	Security Levels $V_S(\bar{x}_S)$
(2/5,2/5,0,0,1/5,0)	(2/5,4/5,0,3/5)
(1/4,1/2,0,0,1/4,0)	(1/4,1,1/4,1/2)
(1/3,1/2,0,0,4/25,0)	(1/3,1,0,2/3)
(3/20,23/50,0,2/25,31/100,0)	(0,1,27/50,31/100)
(0,2/3,0,0,1/3,0)	(0,2/3,2/3,1/3)
(1/4,5/8,0,0,1/8,0)	(1/4,3/4,0,3/4)
(0,3/4,0,0,1/4,0)	(0,1/2,1/2,1/2)
(0,0,0,0,1,0)	(-2,0,2,-1)
(0,0,0,1/2,1/2,0)	(-1,1/2,3/2,-1/2)
(0,2/5,0,1/5,2/5,0)	(-2/5,1,1,0)
(1/2,1/2,0,0,0,0)	(1/2,1,-1/2,1)
(1,0,0,0,0,0)	(1,0,-1,1)

3.4 Core Solution

Security level vectors are used to define the core solution concept. A joint strategy is a core solution if it is stable in the sense that no coalition can guarantee a better result than the payoff obtained by the joint strategy.

Definition 3.6 A joint strategy $\bar{x} = (\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n) \in X$ is a core solution of the n-person multiobjective game $\Gamma = \{N, X^i, u^i\}$ if:

$$\forall S \in \mathfrak{N} : \Omega_S(\bar{x}) = \{x_S \in X_S / V_S(x_S) >> u_S(\bar{x})\} = \varnothing$$

Similarly, a vector $y \in \mathbb{R}(N)$ is a "core vector" (or in the core) of the game Γ if it is feasible and if:

$$\forall S \in \mathfrak{N} : \Phi_S(y) = \{x_S \in X_S / V_S(x_S) >> y_S\} = \varnothing$$

By feasible we mean $\exists x \in X / u(x) \geq y$

With >> we note strictly greater than componentwise.

In the following theorem we shoe a relation between core solution and POSS.

Theorem 3.4 For the game $\Gamma = \{N, X^i, u^i\}$, if the joint strategy $\bar{x}$ is such that $\bar{x}_S$ is POSS $\forall S \subset N$, then $\bar{x}$ is a core solution.

Proof: $\bar{x}_S$ is POSS for $S \subset N$, then there exists no $x_S \in X_S$ such that $v_S(x_S) >> v_S(\bar{x}_S) = \min_{x_{-S} \in X_{-S}} u_S(\bar{x}_S, x_{-S})$. But

$$u_S(\bar{x}) \geq \min_{x_{-S} \in X_{-S}} u_S(\bar{x}_S, x_{-S})$$

then there exists no $x_S \in X_S$ such that $v_S(x_S) >> u_S(\bar{x}_S)$. It follows that $\bar{x}$ is a core solution.

If there exists a fixed partition of the players $\Delta = \{S_1, S_2, \ldots, S_k\}$ then there will be partial cooperation among the members of each coalition and competence between the different coalitions. Then, the partition Δ will induce k parametric multiple objective games:

$$\Gamma_S(\bar{x}_{-S}) = \{S, X^i, u^i(x_S, \bar{x}_{-S})\} \quad \forall S = S_1, S_2, \ldots, S_k. \tag{8}$$

For the fixed parameter $\bar{x}_{-S}$, each $\Gamma_S(\bar{x}_{-S})$ has $|S|$ players with payoff functions $u^i(x_S, \bar{x}_{-S})$ and is simply a new multiple objective game. The concept of hybrid solution can now be given as:

Definition 3.7 For each partition of players $\Delta = \{S_1, S_2, \ldots, S_k\}$ in the n-person multiple objective game $\Gamma = \{N, X^i, u^i\}$, a joint strategy $\bar{x} = \{\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n\} \in X$ is a core solution corresponding to the partition Δ if $\forall S \in \Delta$, $\bar{x}_S$ is a core solution of the corresponding parametric game: $\Gamma_S(\bar{x}_{-S}) = \{S, X^i, u^i(x_S, \bar{x}_{-S})\}$.

Notice that this definition generalizes the core solution concept given in definition 3.6, because a core solution is a core solution for the trivial partition $\Delta = \{N\}$.

In [23] an existence theorem of core solutions is established for any partition. The proof of this result suggests an algorithm to find a core solution to the multiple objective game based on Kakutani's fixed point theorem in [11].

Theorem 3.5 Given a partition of players $\Delta = \{S_1, S_2, \ldots, S_k\}$ in the multiple objective game $\Gamma = \{N, X^i, u^i\}$, the corresponding core solutions set is non-empty if Γ satisfies:

1. For each player i, X^i is a closed bounded convex subset in $\mathbb{R}^{L_i}$.
2. For each coalition $S \in \Delta$, $u^j(x)$, $j \in S$, are all continuous in $x = (x_S, x_{-S})$ and are quasiconcave in x_S.

4. Conclusions

Nash equilibrium has been the most frequently used concepts in the analysis of n-person non-cooperative games. However, the exhibits the same difficulties observed in bimatrix games both in the computation and use. These inconveniences may be avoided by using different solution concepts taken from the theory of matrix games. In this paper, we reviewed the concepts of Nash equilibria, maximin strategies and Pareto-optimal security strategies in the framework of n-person non-cooperative multiple objective games. New definitions were given and properties showing relationships were stated. Examples were included to show so the difficulty of the calculation of those strategies as their use. Finally, we proved the relationship between POSS and core solutions in a game with a given coalitional structure.

References

[1] Blackwell, O., An Analog of the Minimax Theorem for Vector Payoff, Pacific Journal of Mathematics 6 (1956) 1-8.
[2] Borm, P.E.M, Tijs, S.H. and Aarssen van Den, J.C.M., Pareto Equilibria in Multiobjective Games, in B. Fuchssteiner, T. Lengauer, H.J. Skaka (ed.) Methods of Operations Research 60 (1988) 313-323.
[3] Borm, P.E.M., Garcia-Jurado, I., Potters, J. and Tijs, S.H., An Amalgation of Games, European Journal of Operations Research 89 (1996) 570-580.
[4] Charnes, A., Cooper, W.W., Huang, Z.M. and Wei, Q.T., Multiple-Payoff Constrained n-Person Games over Cones, ccs Research Report 583, Center for Cybernetic Studies, University of Texas at Austin (1987).
[5] Charnes, A., Huang, Z.M., Rousseau, J.J. and Wei, Q.L., Cone Extremal Solutions of Multi-Payoff Games with Cross-Constrained Strategy Set, Optimization 21 (1990) 51-69.
[6] Contini, M.B., Olivetti, I.C. and Milano, C.S., A Decision Model under Uncertainty with Multiple Payoff, In A. Mensch (ed.) Theory of Games (1966) 50-63, New York, American Elsevier Pub. Co.
[7] Fernández, F.R. and Puerto, J., Vector Linear Programming in Zero-Sum Multicriteria Matrix Games, Journal Optimization and Applications 63 (1996) 167-189.
[8] Goshe, D. and Prassard, R., Solution Concept in Two-Person Multicriteria Games, Journal Optimization Theory and Applications 63 (1989) 167-189.
[9] Goshe, D., A Necessary and Sufficient Condition for Pareto-Optimal Security Strategies in Multicriteria Matrix Games, Journal Optimization Theory and Applications 68 (1991) 463-480.
[10] Hannan, E.L., On Games with Multiple Payoff, International Journal of Game Theory 11 (1982) 13-15.
[11] Kakutani, S., A Generalization of Brouwer's Fixed Point, Duke Mathematical Journal 8 (1941).
[12] Li, Z.F. and Wang, Y., Lagrange Multipliers and Saddle Points in Multiobjective Programming, Journal Application Mathematical Optimization 83 (1994) 63-81.
[13] Nash, J.F. Jr., Equilibrium in n-Person Games, Academy of Science 36 (1950) 48-49.
[14] Newumann von J. and Morgenstern, O., Theory of Games and Economics Behavior, University Press, Princeton, New Jersey (1944).
[15] Nieuwenhuis, J.W., Some Minimax Theorems in Vector-Valued Functions, Journal Optimization Theory and Applications 40 (1983) 463-475.
[16] Puerto, J. and Fernández, F.E., Solution Concepts Based on Security Levels in Constraint Multicriteria Concave-Convex Games, Opsearch 32 (1994) 16-30.
[17] Shapley, L.S., Equilibrium Points in Games with Vector Payoff, Naval Research Logistics Quarterly 6 (1959) 57-61.
[18] Steuer, R.E., Manual for the ADBASE, Multiple Objective Linear Programming Package, University of Georgia, Athens, Georgia (1995).
[19] Tanaka, T., Two Types of Minimax Theorems for Vector-Valued Functions, Journal Optimization Theory and Applications 68 (1991) 321-334.
[20] Tanaka, T., Minimax Theorems in Vector Optimization, Ph. D. Thesis, Niigata University, Niigata (1992).
[21] Tanaka, T., Generalized Quasiconvexities, Cone Saddle Points, and Minimax Theorem for Vector-V alued Functions, Journal Optimization Theory and Applications 81 (1994) 355-377.
[22] Zeleny, M., Games with Multiple Payoff, International Journal of Game Theory 4 (1975) 179-191.
[23] Zhao, J., The Equilibria of a Multiple Objective Game, International Journal of Game Theory 20 (1991) 171-182.

UMA ABORDAGEM AO PROBLEMA DO TRAJECTO ÓPTIMO MULTIOBJECTIVO

José Luis Esteves dos Santos

Departamento de Matemática
Universidade de Coimbra
Portugal

Abstract

The multiobjective optimal path problem is viewed as the optimization of a function defined over a subset of paths between a pair of nodes and with values in $\mathbb{R}^k$ ($k > 1$); an order relation is considered in $\mathbb{R}^k$ which allows to determine a set of solutions - the nondominated paths.

The generalization for $\mathbb{R}^k$ of the optimal path problems is considered as well as some of its properties. The multiobjective shortest path problem is then presented and particularized for the biobjective case ($k = 2$), being emphasised the properties of the non dominance relation for this case. Some new algorithms for the resolution of the multiobjective shortest path problem are also presented, as well as computational results for $k = 2$.

Resumo

O problema do trajecto óptimo multiobjectivo é abordado em termos da optimização de uma função definida num subconjunto de trajectos entre um dado par de vértices e que assume valores em $\mathbb{R}^k$, ($k > 1$); tem-se em conta uma relação de ordem definida em $\mathbb{R}^k$ que permite a determinação de um conjunto de soluções ditas não dominadas.

É feita a generalização para $\mathbb{R}^k$ do trajecto óptimo assim como de algumas das suas propriedades. Aborda-se então o problema do trajecto mais curto multiobjectivo que se particulariza para o caso biobjectivo ($k = 2$), salientando-se as propriedades da relação de dominância para este caso. Apresentam-se ainda algoritmos novos para a resolução do problema do trajecto mais curto e resultados computacionais para $k = 2$.

Keywords

Path, multiobjective, dominance relation.

1. Introdução

Neste trabalho utilizam-se as notações e definições usadas em [4][1]. Para alertar o leitor, dir-se-á apenas que se designa por caminho um qualquer trajecto sem vértices repetidos (excepto, eventualmente, o inicial que pode coincidir com o terminal - caso em que o caminho será designado por ciclo).

No problema do trajecto óptimo com valor em $\mathbb{R}$, a cada arco da rede estão associados $\Upsilon(\Upsilon \geq 1)$ custos reais pretendendo-se optimizar uma função real, denotada genericamente por f. Designa-se pot $\mathfrak{T}_{i,j}$ o conjunto dos trajectos de i para j, com $i,j \in \mathfrak{N}$, e dados dois vértices s e

[1] acessível na internet no endereço:
"http://www.mat.uc.pt/~zeluis/INVESTIGACAO/mestrado_revisto.ps.gz"

$t(s \neq t)$, designa-se por $\mathfrak{T}$ o conjunto $T_{s,t}$; pretende-se determinar um trajecto $p^* \in \mathfrak{T}^* \subseteq \mathfrak{T}$ que optimize $f_{|\mathfrak{T}^*}$, ou seja, que optimize a restrição de f ao conjunto $\mathfrak{T}^*$; isto é, pretende-se determinar um trajecto $p^* \in \mathfrak{T}^*$ tal que

$$f(p^*) = \text{opt}\{f(q) : q \in \mathfrak{T}^*\}.$$

O trajecto p^* diz-se uma solução óptima do problema.

A generalização para $\mathbb{R}^k$ ($k > 1$) é óbvia. Associam-se ainda Υ parâmetros reais a cada arco, mas agora a função objectivo assume valores em $\mathbb{R}^k$, pelo que se pretende determinar $p^* \in \mathfrak{T}^*$ tal que $f(p^*) = \text{opt}\{f(q) : q \in \mathfrak{T}^*\}$. No entanto, o conceito de óptimo exige um tratamento mais cuidado pois normalmente existe conflito entre as várias componentes de f; isto é, o trajecto que optimiza uma componente de f não optimiza necessariamente alguma das restantes. Sendo assim, ter-se-á de formalizar este conceito.

Inicialmente, será necessário saber decidir quando é que um trajecto é melhor do que um outro, o que passa por definir uma relação de ordem. Para facilitar a exposição, assume-se a minimização de todas as componentes de f. Assim, define-se a seguinte relação em $\mathbb{R}^k$:

Definição 1: Sejam $\mathbf{a} = (a_1, \ldots, a_k)$ e $\mathbf{b} = (b_1, \ldots, b_k)$ dois elementos de $\mathbb{R}^k$.

$$\mathbf{a} \text{ "}\preceq\text{" } \mathbf{b}^2 \Leftrightarrow a_i \leq b_i, \text{ qualquer que seja } i = 1, \ldots, k.$$

Em [4] prova-se o seguinte teorema:

Teorema 1: A relação "$\preceq$" é de ordem parcial.

Esta relação tem a desvantagem de ser parcial pelo que não permite relacionar o custo de dois quaisquer trajectos, obrigando a rever o conceito de mínimo (isto é, óptimo). Assim, o mínimo de um conjunto segundo uma qualquer relação de ordem R, será definido da seguinte forma:

Definição 2: Seja A um subconjunto não vazio de $\mathbb{R}^k$ e R uma relação de ordem definida em A. Diz-se que $x \in A$ é mínimo de A relativamente a R, se e só se não existe $y \in A$ tal que $y \neq x$ e yRx.

Uma vez que "$\preceq$" é uma relação de ordem parcial, da definição 2 conclui-se que podem existir vários mínimos quando usada esta definição.

A partir desta relação de ordem em $\mathbb{R}^k$, pode-se definir uma relação em $\mathfrak{T}$ que permite concluir quando é que um trajecto é menor, dado que se assumiu a minimização, do que outro.

Definição 3: Seja $\mathcal{G}$ uma rede e, quaisquer que sejam $i,j \in \mathcal{N}$, sejam p,q dois trajectos de $\mathfrak{T}_{i,j}$; isto é, p e q são dois trajectos com o mesmo nó inicial i e com o mesmo nó terminal j. Define-se a relação "$\preceq$" em $\mathfrak{T}_{\mathcal{G}} = \bigcup_{i,j \in \mathcal{N}} \mathfrak{T}_{i,j}$ da seguinte forma:

$$p \text{ "}\preceq\text{" } q \text{ se e só se } f(p) \text{ "}\preceq\text{" } f(q).$$

---

2 Quando for mais vantajoso, também se usará $\mathbf{b} \succeq \mathbf{a}$ para denotar $\mathbf{a} \preceq \mathbf{b}$.

Esta não é uma relação de ordem, pois podem existir trajectos diferentes com o mesmo custo (mesmo para a restrição "$\preceq$"$|_{\mathcal{T}_{i,j}}$). No entanto, prova-se que é uma relação de pré-ordem parcial, [4]. Usando a relação da definição 3, obtém-se a relação de dominância em $\mathcal{T}$ da seguinte forma:

Definição 4: Seja $\mathcal{G}$ uma rede e, quaisquer que sejam $i,j \in \mathcal{N}$, sejam p,q dois trajectos de $\mathcal{T}_{i,j}$; isto é, p e q são dois trajectos com o mesmo nó inicial e com o mesmo nó terminal. Define-se a relação de dominância em $\mathcal{T}_{\mathcal{G}} = \bigcup_{i,j \in \mathcal{N}} \mathcal{T}_{i,j}$ e denota-se por "$\prec$", da seguinte forma:

$$p \text{ "}\prec\text{" } q \Leftrightarrow f(p) \text{ "}\preceq\text{" } f(q) \text{ e } f(p) \neq f(q);$$

ou seja,

$$p \text{ "}\overset{\bullet}{\prec}\text{" } q \Leftrightarrow f_\ell(p) \leq f_\ell(q),\ \forall \ell = 1,\ldots,k, \text{ e } \exists \upsilon \in \{1,\ldots,k\} : f_\upsilon(p) < f_\upsilon(q).$$

Neste caso diz-se que p domina q ou que q é dominado por p. Note-se que só é possível relacionar trajectos com o mesmo vértice inicial e o mesmo vértice terminal. Desta forma, dado $p \in \mathcal{T}_{i,j}$, pode-se definir o conjunto $D_p = \{q \in \mathcal{T}_{i,j} : p \text{ "}\prec\text{" } q\}$, isto é, o conjunto dos trajectos $\mathcal{T}_{i,j}$ dominados por p. Assim, o conjunto dos trajectos dominados em $\mathcal{T}_{i,j}$ é definido por $D_{i,j} = \bigcup_{p \in \mathcal{T}_{i,j}} D_p$, sendo os restantes trajectos de $\mathcal{T}_{i,j}$ o conjunto dos trajectos não dominados em $\mathcal{T}_{i,j}$, isto é, $\bar{D}_{i,j} = \mathcal{T}_{i,j} - D_{i,j}$. Para facilitar a notação, escreve-se D e $\bar{D}$ para denotar $D_{s,t}$ e $\bar{D}_{s,t}$, respectivamente, onde s e t são os nós inicial e terminal da rede.

Finalmente, o teorema 2 permite caracterizar as soluções óptimas do problema multiobjectivo.

Teorema 2: Seja $\mathcal{T}$ o conjunto dos trajectos definidos em $\mathcal{G}$ de s para t e seja $p^* \in \mathcal{T}$. $f(p^*)$ é mínimo de $f(\mathcal{T})$ relativamente à relação "$\preceq$" se e só se $p^* \in \bar{D}$.

Demonstração: Ver [4]. ♦

Assim, o problema do trajecto óptimo multiobjectivo consiste na determinação de todos os trajectos não dominados.

Dois conceitos muito importantes em qualquer problema geral do trajecto óptimo e, consequentemente, no problema do trajecto óptimo multiobjectivo é a *finitude* e o conceito que estabelece se o problema é ou não *limitado*. Estes conceitos definem-se de seguida:

Definição 5: Um problema do trajecto óptimo multiobjectivo diz-se finito se e só se para todo $v \in f(\bar{D})$, existe um trajecto óptimo finito p, isto é, um trajecto constituído por uma sequência finita de nós e de arcos, tal que $f(p) = v$. Caso contrário o problema diz-se não finito.

Definição 6: Um problema do trajecto óptimo multiobjectivo diz-se limitado se e só se todas as componentes de qualquer elemento de $f(\bar{D})$ são finitas. Caso contrário o problema diz-se ilimitado.

A abordagem computacional do problema do trajecto óptimo exige não só que este seja finito mas também limitado. Para isso impõem-se condições que dependem do problema em causa. Por tal motivo, estes dois conceitos devem ser estudados para cada problema do trajecto óptimo e, em particular, sê-lo-ão para o problema do trajecto mais curto multiobjectivo.

Até ao momento, no problema do trajecto óptimo multiobjectivo só foram definidas relações parciais. No entanto, é possível definir relações de ordem total em $\mathbb{R}^k$.

Definição 7: Sejam $\mathbf{a} = (a_1,\ldots, a_k)$ e $\mathbf{b} = (b_1,\ldots, b_k)$ dois elementos de $\mathbb{R}^k$. Diz-se que $\mathbf{a}$ é lexicograficamente maior (menor) do que $\mathbf{b}$ e denota-se por $\mathbf{a} \trianglerighteq \mathbf{b}$ ($\mathbf{a} \trianglelefteq \mathbf{b}$) se e só se $\mathbf{a} = \mathbf{b}$ ou existe $i \in \{1,\ldots, k\}$ tal que $a_j = b_j$ para todo $j = 1,\ldots, i-1$, e $a_i > b_i$ ($a_i < b_i$).

De forma idêntica pode definir-se uma relação lexicográfica em $\mathcal{T}$.

Definição 8: Dados $i,j \in \mathbb{N}$, sejam p e q dois trajectos de $\mathcal{T}_{i,j}$. Diz-se que p é lexicograficamente maior (menor) do que q e denota-se por $p \trianglerighteq q$ ($p \trianglelefteq q$) se e só se $f(p) \trianglerighteq f(q)$ ($f(p) \trianglelefteq f(q)$).

Em [4] prova-se que a relação da definição 7 é de ordem total. Porém, a ordem lexicográfica não traduz correctamente o problema do trajecto óptimo multiobjectivo, uma vez que impõe uma ordem às componentes da função objectivo, isto é, o trajecto óptimo depende da ordenação das componentes da função objectivo. Contudo, existem fortes ligações entre as duas relações definidas, ligações estas descritas nos resultados seguintes e que permitem utilizar a ordem lexicográfica para determinar as soluções óptimas do problema multiobjectivo.

Teorema 3: Sejam $\mathbf{a}$ e $\mathbf{b}$ dois elementos de $\mathbb{R}^k$, tais que $\mathbf{a}$ "$\preceq$" $\mathbf{b}$. Então $\mathbf{a} \trianglelefteq \mathbf{b}$.

Demonstração: Ver [4]. ♦

Deste teorema, resulta trivialmente o seguinte corolário:

Corolário 1: Seja A um subconjunto de $\mathbb{R}^k$. O mínimo de A relativamente à relação $\trianglelefteq$ é um mínimo de A relativamente à relação "$\preceq$".

É pois possível determinar uma solução não dominada do problema multiobjectivo resolvendo o problema do trajecto lexicograficamente menor - o que geralmente é mais fácil.

Uma vez que $\trianglelefteq$ é uma relação de ordem total, pode ser utilizada para realizar um "ranking", isto é, uma enumeração ordenada de trajectos. Desta forma, utilizando o teorema 3 que garante que os únicos trajectos que podem dominar um trajecto p são trajectos lexicograficamente menores do que p, ao realizar o "ranking" segundo a ordem lexicográfica, quando se determina o ℓ-ésimo trajecto lexicograficamente menor, pode-se garantir, nesse instante, se esse trajecto é ou não dominado.

Note-se a importância da ordenação das componentes na ordem lexicográfica. Se se pretendesse realizar o "ranking" da ω-ésima componente da função objectivo, bastaria trocar a ω-ésima componente com a primeira no custo de todos os arcos, podendo ainda realizar-se uma permutação das restantes componentes.

O facto de se poder realizar o "ranking" segundo a ordem lexicográfica (por ser uma relação total), permite sugerir novos algoritmos para resolver o problema do trajecto óptimo multiobjectivo (suportados pela enumeração de trajectos), algoritmos esses que irão ser descritos posteriormente para o caso particular do problema do trajecto mais curto multiobjectivo.

Finalmente, será dada alguma ênfase ao caso biobjectivo onde a relação de dominância tem propriedades especiais que facilitam a resolução do problema; para este caso serão apresentados alguns resultados computacionais.

2 - Princípio de Optimalidade e Algoritmos de Rotulação

Uma vez que as soluções óptimas já foram identificadas como sendo trajectos não dominados, então os Princípios de Optimalidade podem ser descritos da seguinte forma:

Algoritmo 1 (Algoritmo geral de rotulação em $\mathbb{R}^k$)

{X_j: conjunto de rótulos, ainda não testados, associados ao nó j}

{X: conjunto de rótulos ainda não testados, i.e., $X = \cup_{j\in \mathbb{N}} X_j$}

{Y_i: subconjunto de X_i}

{Π_j: conjunto de rótulos, não dominados, associados ao nó j}

{$p_{s,i}^{x_i}$ o trajecto de s para i com rótulo $x_i \in \Pi_i$}

$X_s \leftarrow \{f(\langle s\rangle)\}$

$\Pi_s \leftarrow \{f(\langle s\rangle)\}$

$X \leftarrow \cup_{\ell\in \mathbb{N}} X_\ell$

enquanto $X \neq \phi$ **faz**

escolher $Y_i \subseteq X_i$

para todo o arco $(i,j)\in A$ **faz**

$Z \leftarrow \{f(p_{s,i}^{y_i} \blacklozenge \langle i,(i,j),j\rangle) : y_i \in Y_i\}$

$\Pi_j \leftarrow$ rótulos não dominados de $\Pi_j \cup Z$

$X_j \leftarrow \Pi_j \cap (X_j \cup Z)$

$X \leftarrow \cup_{\ell\in \mathbb{N}} X_\ell$

fim_para

fim_enquanto

Princípio de Optimalidade Forte em $\mathbb{R}^k$: Todo o trajecto não dominado (isto é, trajecto óptimo) de i para j em $\mathcal{G}$, quaisquer que sejam $i,j \in \mathcal{N}$, é constituído por subtrajectos não dominados.

Princípio de Optimalidade Fraca em $\mathbb{R}^k$: quaisquer que sejam $i,j \in \mathcal{N}$, e para cada $v \in f(\bar{D}_{i,j})$, existe pelo menos um trajecto não dominado (isto é, trajecto óptimo) com custo v, de i para j em $\mathcal{G}$, que é constituído por subtrajectos não dominados.

Conclui-se assim que, quando um problema do trajecto óptimo multiobjectivo verifica o Princípio de Optimalidade, não há vantagem em gerar trajectos a partir de subtrajectos dominados, pelo que ao tentar gerar a árvore de todos os trajectos não dominados com raiz em s, muitos ramos ficam por desenvolver, evitando-se a pesquisa exaustiva no conjunto $\mathcal{T}$.

O algoritmo geral de rotulação apresenta-se no algoritmo 1, onde a operação $\blacklozenge$ representa a concatenação de trajectos.

Se o problema verificar o Princípio de Optimalidade Forte, este algoritmo garante a determinação do conjunto $\bar{D}$; no entanto, se o problema em causa só verificar o Princípio de Optimalidade Fraca, então o algoritmo 1 garante unicamente a determinação de $f(\bar{D})$.

O Princípio de Optimalidade é de extrema importância, pois quando o problema não verifica o Princípio de Optimalidade Fraca (e portanto, também não verifica o Princípio de Optimalidade Forte), pode ser difícil desenvolver um algoritmo para resolver esse problema; em muitos casos só é possível recorrer à pesquisa exaustiva para obter a solução do problema. Outro processo consiste na utilização de heurísticas que, embora não garantam a solução óptima, podem ser bastante eficientes. Considere-se, por exemplo, o problema do trajecto mais curto por unidade de tempo, onde se pretende minimizar o quociente de duas funções f_1 e f_2, descritas como a soma da distância ou do tempo, respectivamente, dos arcos do trajecto. Este problema é de difícil resolução por não verificar, em geral, o Princípio de Optimalidade. No entanto, em [4] refere-se que, sob algumas condições, o problema biobjectivo (min f_1, max f_2) verifica o Princípio de Optimalidade (Forte) pelo que o problema biobjectivo pode ser resolvido utilizando um algoritmo de rotulação. Em [4], prova-se que toda a solução óptima do problema do trajecto mais curto por unidade de tempo é uma solução do problema biobjectivo atrás referido, podendo-se assim resolver, indirectamente, o problema inicial.

3 - Problema do Trajecto Mais Curto Multiobjectivo

Nesta secção generaliza-se para vários objectivos, o problema do trajecto mais curto assim como algumas das suas propriedades (por exemplo, a finitude e o facto de ser ou não limitado).

Esta generalização consiste em associar um vector $\mathbf{c_{i,j}} = (c^1_{i,j},\ldots, c^k_{i,j}) \in \mathbb{R}^k$ a cada arco $(i,j) \in A$ (portanto, um vector com a mesma dimensão da função objectivo), sendo a ℓ-ésima componente da função objectivo definida por

$$f_\ell : \mathfrak{T}_{\mathcal{G}} \to \mathbb{R}$$
$$p \mapsto f_\ell(p) = \sum_p c^\ell_{i,j},\ (\ell = 1,\ldots, k).$$

Como já foi referido, uma questão importante é impor condições para que o problema seja finito e limitado. Em [4] provam-se os seguintes teoremas que garantem estas propriedades no problema do trajecto mais curto multiobjectivo.

Teorema 4: O problema mais curto em $\mathbb{R}^k$ é finito se e só se não existem ciclos negativos em $\mathcal{G}$, isto é, $f_1(C) \geq 0,\ldots, f_k(C) \geq 0$, qualquer que seja o ciclo $C \in \mathfrak{T}_{\mathcal{G}}$.

Teorema 5: O problema do trajecto mais curto em $\mathbb{R}^k$ é ilimitado se e só se é finito.

Por vezes o problema do trajecto mais curto multiobjectivo, onde $\mathfrak{T}^*$ coincide com $\mathfrak{T}$, é confundido com o problema do caminho mais curto multiobjectivo, isto é, o problema onde $\mathfrak{T}^*$ é o subconjunto de $\mathfrak{T}$ constituído pelos trajectos sem ciclos (isto é, caminhos). De facto, e ao contrário do problema do trajecto mais curto multiobjectivo, este é sempre finito (e como tal limitado) sendo também sempre finito o número de soluções óptimas. O facto destes dois problemas serem, muitas vezes, confundidos resulta do teorema 6 que é consequência da demonstração do teorema 4.

Teorema 6: Para cada valor $v \in f(\bar{D})$, existe um trajecto não dominado com esse valor que é caminho se e só se não existem ciclos negativos na rede.

No entanto, estes problemas só são equivalentes em algumas situações, como se refere no seguinte teorema, cuja demonstração pode ser vista em [4].

Teorema 7: Admita-se que $f_i(C) \geq 0$, $i = 1,\ldots, k$, e $\exists j \in \{1,\ldots, k\}$ tal que $f_j(C) > 0$ qualquer que seja o ciclo C da rede. Sob esta condição, o problema do trajecto mais curto em $\mathbb{R}^k$ é equivalente ao problema do caminho mais curto em $\mathbb{R}^k$; isto é, todo o trajecto não dominado é caminho não dominado e vice-versa.

Já foi referido anteriormente um algoritmo para determinar as soluções óptimas do problema multiobjectivo quando este verifica o Princípio de Optimalidade. O teorema 8 estabelece condições para que o problema do trajecto mais curto multiobjectivo verifique este princípio.

Teorema 8: Admita-se que não existem ciclos negativos numa rede. Então o problema do trajecto mais curto em $\mathbb{R}^k$ verifica o Princípio de Optimalidade.

Demonstração: Ver [4]. ♦

Note-se que a condição estabelecida no teorema 8 é apenas suficiente ao contrário do que acontece no caso real (k = 1), onde a condição é também necessária. Em [4] ilustra-se esta situação com um exemplo, onde a condição não é necessária no caso multiobjectivo.

Note-se que o teorema 8 também é válido para o problema do caminho mais curto. No entanto, e ao contrário do problema do trajecto mais curto em $\mathbb{R}^k$, esta condição não é necessária para k = 1, como se prova em [4] com um contraexemplo.

Quando se verifica o Princípio de Optimalidade, o algoritmo de rotulação é simplificado (ver algoritmo 2).

Tal como no caso real, as várias formas de manipular o conjunto X originam as formas lifo, fifo e dijkstra das quais a que se mostrou mais eficiente foi uma versão dijkstra com endereçamento calculado, [1], onde se escolhe, em cada iteração o rótulo lexicograficamente menor. Em [4] apresenta-se uma descrição detalhada sobre a estrutura de dados utilizada e a sua implementação. Também se apresenta um estudo comparativo entre as várias versões do algoritmo de rotulação que foram implementadas.

Algoritmo 2
(Algoritmo geral de rotulação para o problema do trajecto mais curto em $\mathbb{R}^k$)

{X_j: conjunto de rótulos, ainda não testados, associados ao nó j}

{X: conjunto de rótulos ainda não testados, isto é, $X = \bigcup_{j \in \mathcal{N}} X_j$}

{Y_i: subconjunto de X_i}

{Π_j: conjunto dos rótulos não dominados associados ao nó j}

$X_s \leftarrow \{0_{\mathbb{R}^k}\}$; $\Pi_j \leftarrow \{0_{\mathbb{R}^k}\}$

$X \leftarrow \bigcup_{\ell \in \mathcal{N}} X_\ell$

enquanto $X \neq \phi$ **faz**

escolher $Y_i \subseteq X_i$

para todo o arco $(i,j) \in A$ **faz**

$Z \leftarrow \{y_i + c_{i,j} : y_i \in Y_i\}$

$\Pi_j \leftarrow$ rótulos não dominados de $\Pi_j \cup Z$

$X_j \leftarrow \Pi_j \cap (X_j \cup Z)$

$X \leftarrow \bigcup_{\ell \in \mathcal{N}} X_\ell$

fim_para

fim_enquanto

4 - Algoritmos baseados no "ranking"

Os algoritmos apresentados nesta secção baseiam-se em várias técnicas para realizar o "ranking" e constituem a parte mais inovadora deste artigo. Nesta secção descrevem-se, de uma forma resumida, as várias versões implementadas. Estas versões foram elaboradas unicamente para o caso biobjectivo, não sendo a sua generalização para $\mathbb{R}^k$ ($k > 2$) aqui apresentada.

A classe de algoritmos baseados no "ranking" é justificada pela grande eficiência de alguns algoritmos para a enumeração de trajectos, de entre os quais se destacam o algoritmo MS, [3], e o algoritmo MPS, [2]. De um modo muito resumido, dir-se-á que o primeiro desses algoritmos é baseado na determinação da melhor alternativa (isto é, o melhor trajecto ainda não determinado) para cada subtrajecto $p^{\ell}_{s,i}$ do ℓ-ésimo trajecto de s para t com melhor custo, p^{ℓ}, onde i é um nó intermédio de p^{ℓ}.

O algoritmo MPS é bem mais recente, não estando ainda suficientemente estudado aquando da elaboração de [4]. Por esta razão, só foi utilizado o algoritmo MS na elaboração de todas as versões implementadas para os algoritmos baseados no "ranking" para o problema multiobjectivo.

O "ranking" só é possível quando se utiliza uma relação de ordem total; no caso presente, utilizar-se-á a ordem lexicográfica já que, pelo teorema 3, um trajecto só poderá ser dominado por trajectos lexicograficamente menores; dado que a relação "$<$" é transitiva, facilmente se conclui que basta realizar o teste da dominância com trajectos lexicograficamente menores e não dominados.

Uma questão muito importante ao realizar o "ranking" é o de saber quando é que este pode terminar, isto é, quando se pode garantir que todos os trajectos não dominados foram determinados. Em [4] é proposta uma condição de paragem nova que se baseia no seguinte teorema:

Teorema 9: Seja p um dado trajecto de s para t. Para determinar todos os trajectos não dominados de s para t basta determinar todos os trajectos q tais que $f_i(q) \leq f_i(p)$, para algum $i = 1,\ldots, k$; isto é, basta realizar k "rankings", um em cada objectivo, onde o limite superior do i-ésimo "ranking" é $f_i(p)$.

Põe-se então o problema da escolha do trajecto p que indicará um limite superior do "ranking" em cada objectivo. Pode provar-se que, se p é o trajecto lexicograficamente menor em algum dos objectivos, então não é necessário realizar o "ranking" nesse objectivo, pelo que bastaria realizar o "ranking" segundo k - 1 objectivos. No entanto, algum estudo computacional deste problema, permite concluir que esta escolha não é mais vantajosa. De facto, em [4] reportam-se alguns resultados computacionais, para o caso biobjectivo, onde se tentou determinar todos os trajectos não dominados numa rede aleatória com 100 nós e 500 arcos, escolhendo-se para p o trajecto lexicograficamente menor. Nessa rede, foi possível realizar o "ranking" de quase um milhão de trajectos em menos de seis segundos, embora só se tenham

determinado cinco dos seis trajectos não dominados. Não foi possível determinar o último trajecto não dominado apesar de se ter usado todo o espaço de memória disponível do computador (cerca de 120Mb RAM). Por esta razão, optou-se por se escolher o trajecto p para a condição de paragem de uma outra forma. Resumidamente, esta forma consiste em realizar uma "parte" do "ranking" (α%) no primeiro objectivo, obter o último trajecto não dominado determinado e finalmente utilizar esse trajecto como condição de paragem no segundo "ranking". Realizam-se deste modo dois "rankings", embora sejam menos extensos. Na prática foram implementadas duas versões, ambas para o caso biobjectivo; qualquer delas começa por determinar um trajecto lexicograficamente menor em cada objectivo, obtendo assim a amplitude do "ranking" a realizar em cada objectivo. Tem-se pois:

1. **muda1**: realiza α% do "ranking" no primeiro objectivo e seguidamente o "ranking" no segundo objectivo.
2. **muda2**: realiza o "ranking" no primeiro objectivo até garantir que só é necessário realizar α% do "ranking" no segundo objectivo.

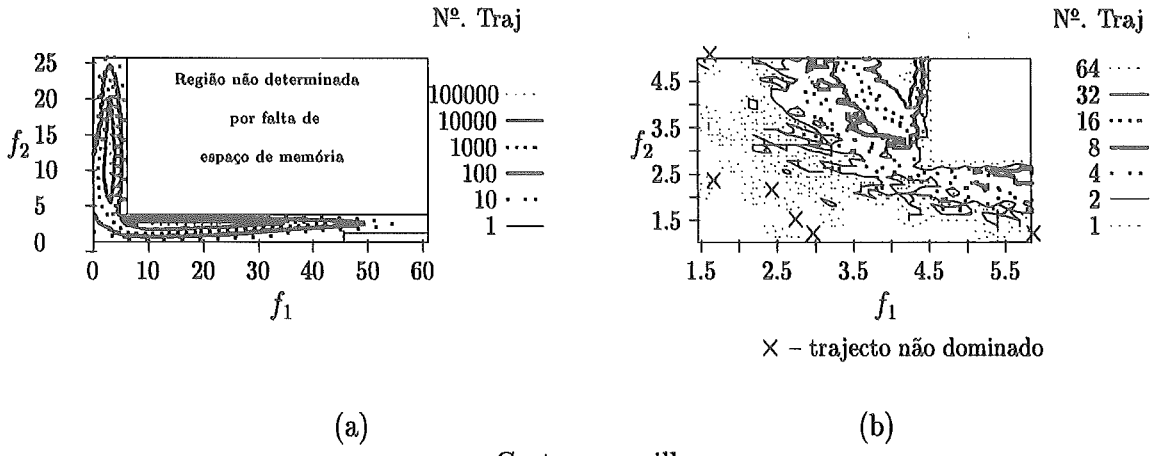


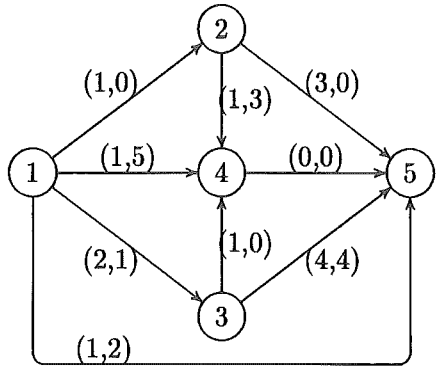
Figura 1:
(a) Número de trajectos determinados segundo os "ranking"
(b) Ampliação da região onde se encontram os trajectos não dominados

Um dos problemas destas versões é encontrar o valor óptimo do parâmetro α, isto é, o valor de α com o qual é possível determinar todos os trajectos não dominados, minimizando ainda o número de trajectos dominados que são determinados. Alguns (poucos) resultados computacionais, sugerem um valor de α no intervalo [20,30]; no entanto, está a ser realizado[3] um estudo mais exaustivo sobre este assunto.

[3] Na verdade, recentemente foi desenvolvido um novo critério para decidir quando trocar o objectivo onde se está a realizar o "ranking" que apresenta, em geral, uma melhor eficiência que as duas versões descritas.

Na figura 1(a) apresenta-se o número máximo de trajectos que foi possível determinar pelo "ranking" em cada objectivo. Facilmente se pode imaginar que, neste caso, uma pequena variação do parâmetro α pode conduzir a uma grande variação no número de trajectos determinados, tornando a escolha do valor óptimo para α muito difícil e extremamente dependente do problema. No entanto, o custo dos trajectos não dominados encontra-se numa região muito restrita do gráfico, perto da origem dos eixos; por esta razão, realizou-se uma ampliação dessa região, ver 1(b), para saber qual a distribuição desses custos. Assim, surge a ideia de realizar um "ranking" usando uma função que é combinação linear das funções objectivo, isto é, um "ranking" oblíquo. Neste caso, determina-se um trajecto lexicograficamente menor em cada objectivo, define-se uma função que é uma combinação linear dos dois objectivos e de modo a que essa nova função tenha o mesmo valor nos dois trajectos inicialmente determinados (de forma informal, e do ponto de vista gráfico, realiza-se o "ranking" deslocando a recta, paralelamente a si mesma, definida pelas coordenadas dos custos dos trajectos inicialmente determinados). Esta versão apresentou os melhores resultados relativamente às várias versões que foram implementadas tendo por base métodos do "ranking".

Foi implementada ainda outra versão. A ideia que lhe deu origem foi o facto de que, em certas condições, o problema do trajecto mais curto multiobjectivo verifica o Princípio de Optimalidade Forte (isto é, todo o trajecto não dominado é constituído por subtrajectos não dominados) assim como o problema do "ranking" de trajectos lexicográficos menores (isto é, o i-ésimo trajecto lexicográfico menor é constituído por j-ésimos subtrajectos lexicográficos menores, com $j \leq i$). Deste modo, será lícito "fundir" os dois problemas (que verificam ambos o Princípio de Optimalidade Forte) e determinar a alternativa lexicográfica menor não dominada? A resposta é negativa pois este novo problema não verifica o Princípio de Optimalidade (fraca), isto é, o i-ésimo trajecto lexicográfico menor não dominado pode ser constituído por j-ésimos subtrajectos lexicográficos menores com $j > i$, como se pode constatar com o seguinte exemplo.



$\mathcal{T}_{1,5}$	custo	
$\langle 1,5\rangle$	$(1,2)$	ND
$\langle 1,4,5\rangle$	$(1,5)$	D
$\langle 1,2,4,5\rangle$	$(2,3)$	D
$\langle 1,3,4,5\rangle$	$(3,1)$	ND
$\langle 1,2,5\rangle$	$(4,0)$	ND
$\langle 1,3,5\rangle$	$(6,5)$	D

$\mathcal{T}_{1,4}$	custo	
$\langle 1,4\rangle$	$(1,5)$	ND
$\langle 1,2,4\rangle$	$(2,3)$	ND
$\langle 1,3,4\rangle$	$(3,1)$	ND

Figura 2 - Exemplo onde não se verifica o Princípio de Optimalidade

Na figura 2 apresenta-se uma tabela com os trajectos que se podem definir de 1 para 5 na rede dessa mesma figura, indicando-se quais os não dominados. Pode-se então constatar que o segundo trajecto lexicográfico menor não dominado de 1 para 5, $\langle 1,3,4,5\rangle$, é constituído por um terceiro subtrajecto lexicográfico menor não dominado de 1 para 4, $\langle 1,3,4\rangle$, pelo que o problema do trajecto lexicográfico menor não dominado não verifica o Princípio de Optimalidade.

Poder-se-ia pensar num algoritmo que realizasse o "ranking" dos trajectos segundo a ordem lexicográfica e que, sempre que encontrasse uma alternativa dominada de um nó, pesquisasse a alternativa seguinte desse nó até encontrar uma alternativa não dominada. Porém, se existirem ciclos na rede, esta definição pode ser recursiva e põe em causa a finitude do algoritmo. Em [4] apresenta-se um exemplo onde ocorre esta situação. Contudo, foi possível elaborar uma versão onde, quando não é possível determinar a alternativa lexicograficamente menor não dominada de um nó, se determina a alternativa lexicograficamente menor (podendo esta ser dominda). Esta versão, embora determine menos trajectos de s para t, é mais lenta e não diminui o espaço de memória que é utilizado.

5. Resultados Computacionais

Nesta secção apresentar-se-á um resumo dos reultados computacionais obtidos com as várias versões implementadas, comparando-se apenas as melhores versões dos algoritmos clássicos (algoritmos de rotulação) e dos algoritmos baseados no "ranking". A justificação para as versões escolhidas, assim como um estudo computacional mais completo, pode ver-se em [4].

Os resultados correspondem à medida aritmética dos valores obtidos para 10 problemas do mesmo tipo e foram obtidos no servidor DEC AXP 7610 do Laboratório de Cálculo do Departamento de Matemática, com velocidade de processador de 275 MHz sobre DEC Unix 3.2, tendo 128 Mbytes de RAM e 200.9SPECint92. As várias estruturas consideradas para as redes foram as seguintes: aleatórias, euclidianas, grelhas e completas. As do primeiro tipo não obedecem a qualquer restrição, a não ser garantir a existência de um ciclo hamiltoniano[4]. As redes euclidianas são em tudo idênticas às redes aleatórias; no entanto, a primeira componente do custo de cada arco (i,j) da rede corresponde ao arredondamento da distância euclidiana entre as coordenadas que definem os nós i e j e que são determinadas aleatoriamente. As redes com estrutura em grelha (grelhas), são definidas em malhas com h nós em altura e ω nós em largura. Os nós correspondem aos pontos de intersecção na malha, existindo portanto $h \times \omega$ nós. Os arcos são também definidos pela malha e os custos dos arcos são aleatórios. Finalmente, uma rede completa corresponde a uma rede onde existem todos os arcos da forma (i,j), $i \neq j, i,j \in \mathfrak{N}$. Note-se que os custos dos arcos são aleatórios em todos os tipos redes, excepto para a primeira componente nas redes euclianas.

---

[4] Esta condição, que será válida em todos os tipos de redes, permite garantir que na rede não existem nós excedentes, isto é, permitem garantir que $\Im_{i,j} \neq \phi, \forall i,j \in A$

Em [4] é também realizado um estudo computacional sobre a variação do número de trajectos não dominados para os vários tipos de rede. Os resultados obtidos nesse estudo sugerem o agrupar das redes em duas classes. A primeira é constituída pelas redes aleatórias e euclidianas, onde o número de trajectos não dominados é pequeno, tendo as várias versões resolvido a quase totalidade dos problemas; além disso, parece que a estrutura definida nas redes euclidianas facilita ainda mais o problema, diminuindo o número de trajectos não dominados existentes, assim como o tempo de execução e o espaço de memória ocupado pelas várias versões apresentadas. A segunda classe é constituída pelas redes do tipo grelha e completas. Nestes tipos de redes, existem muitos trajectos não dominados, pelo que as versões baseadas no "ranking" não determinaram, em geral, todos os trajectos não dominados, ao contrário das versões baseadas nos algoritmos de rotulação. Entre estes dois últimos tipos de redes, as grelhas, pela sua rígida estrutura nos arcos, constituem os problemas mais difíceis de resolver.

As versões para as quais se apresentarão alguns resultados são as que mostraram maior eficiência na sua classe e descrevem-se, resumidamente, a seguir. Uma descrição mais detalhada destas versões, assim como das outras que já foram referidas, pode ser consultada em [4].

- **rotTemp**: este código foi implementado com base no algoritmo dos rótulos não definitivos. O conjunto X, dos rótulos ainda não utilizados, é manipulado utilizando-se uma estrutura em lista do tipo fifo. Resumidamente, escolhe-se o primeiro rótulo de X, a partir do qual se tentam gerar novos rótulos; além disso, para cada nó, mantém o conjunto dos rótulos não dominados ordenados lexicograficamente.
- **rotDef1**: Utiliza uma estrutura com endereçamento calculado para manter o conjunto X ordenado. Para cada nó, mantém ordenado lexicograficamente o conjunto dos rótulos não dominados que lhe estão associados. Resumidamente, escolhe o rótulo lexicograficamente menor em X e tenta gerar novos rótulos a partir deste.
- **torta**: este código determina o trajecto mais curto em cada objectivo e, de seguida, a recta definida pelos custos destes trajectos. Após este passo, é realizado o "ranking" segundo esta recta.

Todos os códigos são para o problema biobjectivo e foram implementados na linguagem C.

Nos gráficos da figura 3 apresentam-se os resultados obtidos em redes aleatórias, onde d representa a densidade da rede, isto é, o quociente entre o seu número de arcos e o seu número de nós. O primeiro destes gráficos refere-se ao tempo de execução (em segundos) e o segundo refere-se ao espaço de memória ocupado (em Mbytes). Estes gráficos apontam para uma maior eficiência, embora ligeira, do código **torta** relativamente ao tempo de execução (excepto para densidades elevadas); no entanto, relativamente ao espaço de memória ocupada, este código parece ser mais ineficiente.

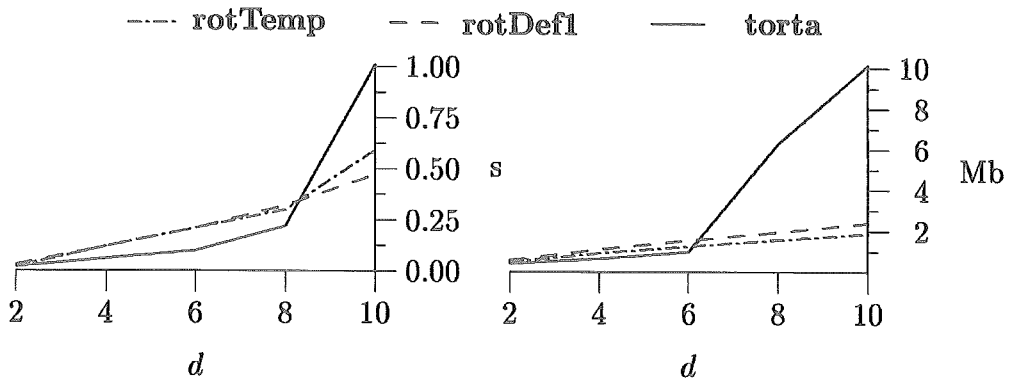


Figura 3 - Redes aleatórias

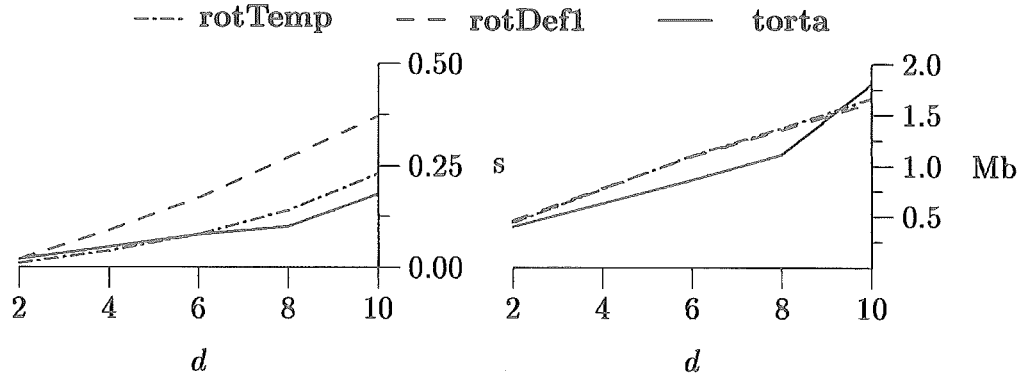


Figura 4 - Redes euclidianas

Nós	100	500	1000
Tempo médio de execução (torta)	0.8s	53.53s	23.85s
% de trajectos ND determinado	98.64	91.59	91.76
Memória ocupada (torta)	34.51Mb	103.07Mb	88.85Mb
Tempo médio de execução (rotDefl)	0.04s	2.06s	11.92s
Memória ocupada (rotDefl)	0.16Mb	3.33Mb	12.53Mb

Tabela 1 - Redes completas

Nos gráficos da figura 4 apresentam-se os resultados obtidos em redes euclidianas. Neste tipo de redes, o código **torta** apresenta uma maior eficiência do que os algoritmos de rotulação; no entanto, como já foi referido, este tipo de redes facilita a resolução do problema (note-se que foi alterada a escala nos eixos do tempo e do espaço de memória). Curiosamente, o código **rotDefl** piorou significativamente a sua eficiência relativamente aos outros códigos, o que talvez se justifique pela complexa estrutura de dados que utiliza (endereçamento calculado) a qual só lhe permite obter vantagens em problemas mais complexos. Destes resultados parece

poder concluir-se que, para problemas com uma estrutura simples, o código **torta** apresenta maior eficiência do que os códigos baseados nos algoritmos de rotulação. Contudo, essa diferença não é muito significativa (por exemplo, a diferença máxima dos tempos de execução é sempre inferior ou igual a 0.1 segundos para redes aleatórias e de 0.2 segundos para redes euclidianas).

Na tabela 1 apresentam-se os resultados obtidos para os códigos **torta** e **rotDef1** em redes completas. Os resultados do código **rotTemp** não foram incluídos, uma vez que o código **rotDef1** se apresentou muito mais eficiente do que o código **rotTemp**. Estes resultados permitem concluir que o código **rotDef1** é claramente mais eficiente que o código baseado no "ranking", quer ao nível de tempo de execução quer ao nível de espaço de memória ocupado; mais, note-se que o código **torta** não conseguiu resolver a totalidade dos problemas (embora tenha determinado mais de 90% dos trajectos não dominados).

Finalmente, na tabela 2 apresentam-se os resultados obtidos em grelhas que, como já foi referido, é o tipo de redes onde as várias versões tiveram maiores dificuldades. Nesta tabela só foram consideradas redes com 2500 nós (aproximadamente); o parâmetro q indica o quociente entre a largura e a altura da grelha (desta forma, para q = 1, obtemos uma grelha quadrada). O código **rotDef1** é o que apresenta melhores resultados e além disso parece não ser afectado significativamente pela variação do parâmetro q. Por outro lado, o código **torta** piorou significativamente a sua eficiência, não conseguindo determinar mais do que um terço dos trajectos não dominados, razão pela qual os tempos de execução não são significativos.

Da análise destes resultados parece ser evidente que o código **rotDef1** é claramente mais eficiente em problemas mais difíceis. Além disso, como este código não era significativamente pior do que o código **torta** nas redes com estruturas mais simples (a diferença do tempo de execução era sempre inferior a 0.2 segundos) então, num cômputo geral, o código **rotDef1** parece ser o que apresenta melhores resultados. Resultados mais recentes, envolvendo novas versões, apontam, no entanto, para um melhor desenvolvimento dos códigos baseados no "ranking", nomeadamente quando o algoritmo **MPS** é utilizado.

q	1	2	5	10
Tempo médio de execução (torta)	7.37s	8.20s	4.35s	3.21s
% de trajectos ND determinado	30.64	29.08	23.65	17.41
Memória ocupada (torta)	107.05Mb	107.05Mb	107.05Mb	107.05MB
Tempo médio de execução (rotDef1)	8.23s	8.43s	7.26s	8.29s
Memória ocupada (rotDef1)	28.05Mb	29.61Mb	32.00Mb	39.03Mb

Tabela 2 - Redes do tipo grelha

O autor pertence à Linha Ciências da Computação do Centro de Informática e Sistemas da Universidade de Coimbra, parcialmente subsidiado pelo Ministério das Ciências e Tecnologia através do Projecto PRAXIS XXI da Junta Nacional de Investigação Científica e Tecnológica.

Ao meu orientador, Prof. Doutor Ernesto de Queirós Vieira Martins, agradeço o ter sido incansável no apoio e colaboração prestada na elaboração da minha tese de mestrado bem como na elaboração deste artigo.

Referências

[1] Dial, R., Glover, G., Karney, D. and Klingman, D., A computational analysis of alternative algorithms and labelling techniques for finding shortest path trees, Netwroks 9 (1979) 215-348.

[2] Martins, E.Q.V., Pascoal, M.M.B. and Santos, J.L.E., A new algorithm for ranking loopless paths, CISUC Technical Report, Maio 1997.

[3] Martins, E.Q.V. and Santos, J.L.E., A new shortest paths ranking algorithm, aceite para publicação em *Investigação Operacional.*

[4] Santos, J.L.E., O problema do trajecto óptimo multiobjectivo, 1997 (Dissertação de Mestrado, Departamento de Matemática, Universidade de Coimbra).

NOTA DO EDITOR PRINCIPAL

Um lamentável lapso impediu que o artigo "Integer Solutions of Multicriteria Network Flow Problems", de M. Ehrgott, do número 1 do volume 19, 1999, fosse publicado na sua totalidade. O Editor Principal da Investigação Operacional assume por inteiro a responsabilidade do sucedido e apresenta as maiores desculpas ao autor do artigo e aos leitores da revista. A versão completa do referido artigo é apresentada nas páginas seguintes deste número.

NOTE FROM THE EDITOR-IN-CHIEF

In the number 1 of volume 19, 1999, the article "Integer Solutions of Multicriteria Network Flow Problems", by M. Ehrgott, has been printed out without some of its contents. The Editor-in-Chief of Investigação Operacional apologizes to the author of the article and the readers of the journal for such inconvenience. The complete version of the article is included in the next pages of this number.

INTEGER SOLUTIONS OF MULTICRITERIA NETWORK FLOW PROBLEMS

Mathias Ehrgott

Fachbereich Mathematik
Universität Kaiserslautern
67653 Kaiserslautern
Germany

Abstract

This paper is concerned with the solution of integer multicriteria network flow problems. The single criterion network flow problem and a sketch of the out-of-kilter method are presented. Important results from (linear) multicriteria optimization are stated and their importance for network flow problems discussed. The main topic of the paper is the presentation of a method to solve integer multicriteria network flow problems. The set of all efficient solutions in objective space is determined in two steps: first the set of all maximal efficient faces of the linear relaxation is determined. The second step consists in finding all integer efficient points. Finally the lexicographic max-ordering theory is proposed to choose a compromise solution.

Keywords

Network flow problems, multicriteria optimization, integer programming.

1. The Network Flow Problem

The general formulation of the (minimum cost) network flow problem is the following:

$$\min \sum_{(i,j)\in A} c_{ij}x_{ij}$$

$$\text{subject to} \sum_{j\in N} x_{ij} - \sum_{j\in N} x_{ji} = 0 \qquad \forall i\in N \qquad \text{(NFP)}$$

$$x_{ij} \geq l_{ij} \qquad \forall (i,j)\in A$$

$$x_{ij} \leq u_{ij} \qquad \forall (i,j)\in A$$

Here, $D = (N,A)$ is a network with node set N, arc set A and variable x_{ij} representing flow along arc $(i,j)\in A$. Many types of algorithms are available for the solution of this problem. Below we will outline the idea of the out-of-killer algorithm, upon which the method for the multicriteria network flow problem developed later is based.

Using dual variables Π_i for the nodes $i\in N$, reduced costs for the arcs are defined as $\bar{c}_{ij} = \Pi_i - \Pi_j - c_{ij}$. From complementary slackness conditions of linear programming it is

straightforward to derive the optimality conditions for the network flow problem: A feasible flow x is optimal if for each arc (i,j) $\bar{c}_{ij} = 0$,

$$x_{ij} = l_{ij} \quad \text{if} \quad \bar{c}_{ij} < 0, \text{ or}$$
$$x_{ij} = u_{ij} \quad \text{if} \quad \bar{c}_{ij} > 0$$

(Nonbasic) variables satisfying these conditions are called "in kilter". The out-of-kilter algorithm is based on the idea of bringing all edges in kilter in order to obtain an optimal solution of the NFP. Because we use the ideas of this algorithm throughout the rest of this paper, we will give a sketch of it now, for details see e.g. [Bazaraa and Jarvis, 1977].

Sketch of the out-of-kilter algorithm

1 **Let** x := 0 and $\prod$:= 0
2 **If** there is an "out-of-kilter" edge (u,v) **goto 3, else STOP**: x is an optimal flow
3 Primal phase:
If possible change flow along a cycle containing (u,v), **else goto 4**
4 Dual phase:
Change dual variables $\prod$ and **goto 2**

In this paper we will discuss the solution of multicriteria integer (NFP). Integrality is an issue that does not create any difficulty in the single objective case: due to the total unimodularity of the constraint matrix of the (NFP) (the node-arc incidence matrix of the network N), it is well known that all basic feasible solutions of NFP are integral. Thus, solving the integer (NFP) is equivalent to solving the linear (NFP). In Section 3 below we will see why the integrality requirement will make the multicriteria NFP much more difficult to solve.

For this paper we shall assume that all problems are non-degenerate. I.e. for all basic feasible solutions, we do neither have $x_{ij} = l_{ij}$ or $x_{ij} = u_{ij}$ for basic variables (primal degeneracy), nor $\bar{c}_{ij} = 0$ for nonbasic variables (dual degeneracy). We will also assume that (NFP) is feasible and (w.l.o.g.) that $l_{ij} \geq 0$ and $u_{ij} < \infty$. Thus the feasible set of (NFP) will be a compact set. Assuming nondegeneracy will avoid technical complications, e.g. in Lemma 4. However, the method can be modified to allow degenerate solutions.

2. Multicriteria (Linear) Optimization

In this section we will briefly summarize the basic definitions and some important results in (linear) multicriteria optimization. Because the multicriteria NFP is a special case, we will refer to these results later.

A linear multiobjective program is given by

$$\begin{aligned} \min \quad & Cx \\ \text{subject to} \quad & Ax = b \\ & x \geq 0 \end{aligned} \qquad \text{(MOLP)}$$

Here, C denotes a Q×n objective or criteria matrix, A is an m×n constraint matrix of full row rank. The right hand side vector is $b \in \mathbb{R}^m$ and $x \in \mathbb{R}^n$ is the vector of variables. Individual

objectives (rows of C) are denoted by C^q. As usual, we will denote the set of feasible solutions of (MOLP) by

$$X = \{x \in \mathbb{R}^n : Ax = b, x \geq 0\}.$$

The following is the fundamental definition of optimality in multicriteria optimization, replacing the single objective notion of minimality.

Definiton 1

- *$x^0 \in X$ is called Pareto optimal if there is no $x \in X$, $x \neq x^0$ such that $Cx \leq Cx^0$ componentwise, with strict inequality for at least one component.*
- *If $x^0 \in X$ is Pareto optimal, then the corresponding image point $y^0 = Cx^0$ is called efficient.*

We will here consider the case of a compact feasible set X, as we encounter in the (NFP), due to our assumptions on the bounds for arc capacities, l_{ij} and u_{ij}. We shall also use the notation Y = CX for the image of the feasible set in criterion space.

The set of efficient points in Y is often referred to as the efficient frontier, a notion we shall adhere to, too. It is well known that the efficient frontier is indeed located on the boundary of Y. Figure 1 depicts Y and the efficient frontier (solid lines) for an (MOLP). Many authors have investigated (MOLP). We will now summarize, without proofs, some of the most important results in the area, which are relevant for this paper.

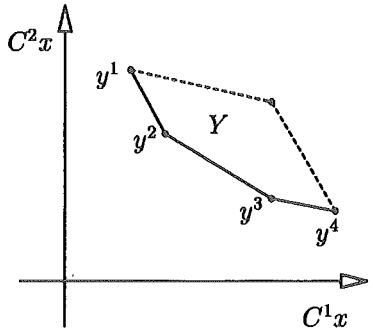


Figure 1 - Feasible set in objective space and efficient frontier

Theorem 1 ([Geoffrion, 1968, Isermann, 1974])

If x^0 is a Pareto optimal solution of (MOLP) then there exists some $\lambda \in \mathbb{R}^Q$, $0 < \lambda_q < 1$,

$\sum_{q=1}^{Q} \lambda_q = 1$, *such that*

$$\lambda^T C x^0 \leq \lambda^T C x \quad \forall x \in X.$$

In other words, all Pareto optimal points can be found by solving weighted sum scalarizations of (MOLP) with appropriate positive weights. The reader can easily visualize this result from the drawing in Figure 1.

The next results refer to topological properties of the efficient frontier, and the set of Pareto optimal solutions (the preimage of the efficient frontier under C).

Theorem 2 ([Naccache, 1978])

The efficient frontier of Y is connected.

Theorem 3 ([Warburton, 1983])

The set of Pareto optimal solutions of (MOLP) is connected.

These results are very important as they imply that neither in decision nor in objective space we can have isolated Pareto optimal/efficient solutions. It is important to note that these results are not true for general (nonconvex) objectives. Theorems 2 and 3 emphasize the continuous nature of linear programming. However, due to basic theory of linear programming and the widespread use of simplex type algorithms, basic feasible solutions are the ones which are most important. For compact X this is a finite set. Thus linear programming is on the boundary between continuous and combinatorial optimization. The combinatorial counterpart of Theorem 3 is the following result, due to Isermann.

Theorem 4 ([Isermann, 1977])

For an (MOLP) define a graph G = (N,E) such that the set of nodes N represents the set of Pareto optimal basic feasible solutions. The set of edges E is defined in such a way that an edge exists between two nodes if and only if the corresponding basic solutions can be obtained from each other by one single pivot step. Then G is connected.

Note that G is naturally an undirected graph. The important implication of Theorem 4 is, that the whole continuum of Pareto optimal solutions can be generated by exploring basic feasible solutions alone, since Pareto optimal basic feasible solutions represent extreme points of the feasible set in decision space, Y.

There is a drawback, however. Experiments recently reported by Benson [Benson, 1997] indicate that even for medium sized problems ((MOLP) with 4 objectives, 50 variables and 50 constraints) the number of Pareto optimal basic feasible solutions may be prohibitively large (80,000 on average). It may well be demanding too much of decision makers to choose among these. Therefore, the need for compromise solutions is evident.

But how to choose a compromise solution? We propose the following lexicographic max-ordering approach. The reasons will be illustrated below.

Definition 2

- *$x^0 \in X$ is called max-ordering optimal if*

$$\max_{1 \le q \le Q} (C^q)^T x^0 \le \max_{1 \le q \le Q} (C^q)^T x$$

  for all $x \in X$.
- *For $y \in \mathbb{R}^Q$ let $sort(x) = (sort_1(x),\ldots, sort_Q(x))$ be such that $sort_1(x) \ge sort_2(x) \ge \ldots \ge sort_Q(x)$. $x^0 \in X$ is called lexicographic max-ordering (lex-MO) optimal if $sort(Cx^0) \le_{lex} sort(Cx)$ for all $x \in X$, where $\le_{lex}$ denotes the lexicographic order.*

In other words, we first choose a solution which minimizes the worst objective (some kind of "conservative planning" idea). But, there may be many such max-ordering solutions, and

most of them need not be Pareto optimal. The idea of lexicographic max-ordering is to iterate this idea to the second worst, third worst objective, etc.

Why does this define a reasonable compromise solution? The answer is given in Theorem 5 from [Ehrgott, 1977b] (see also [Ehrgott, 1995] and [Ehrgott, 1997a]).

Theorem 5

- *If x^0 is a lex-MO optimal solution then x^0 is also Pareto optimal solution and it is a max-ordering solution.*
- *x^0 is a lexicographic max-ordering solution if and only if x^0 satisfies the normalization, regularity and reduction property.*

The properties of Theorem 5 need some explanation. Normalization simply means, that when we have only one objective the optimal solution should define the usual minimum. Regularity is defined as the conservativeness or robustness in the sense of max-ordering optimality. Finally, by reduction we mean that, should the value of one objective for an optimal solution be known, then we can drop this criterion from minimization and impose it as a constraint instead, but still obtain the same optimal solutions for this reduced problem.

From the point of view of decision makers, given the task to choose from such a huge number of possibly "good" (Pareto optimal) alternatives, these three properties seem reasonable. So, once they are accepted as valid, there is only one choice for choosing a solution, namely the lexicographic max-ordering optimal solution.

3. Multicriteria Network Flow Problems

In this main section we will introduce the multicriteria network flow problem. After reviewing existing literature on the topic, we will present the outline of a method to solve the continuous problem in Section 3.1. Section 3.2 describes a method to solve the integer problem. The method will be illustrated by means of an example. Finally, we discuss the problem of determining a lexicographic max-ordering solution as a compromise solution.

The multicriteria network flow problem can be written as follows

$$\min \begin{pmatrix} \sum_{(i,j)\in A} c^1_{ij} x_{ij} \\ \vdots \\ \sum_{(i,j)\in A} c^Q_{ij} x_{ij} \end{pmatrix}$$

$$\sum_{j\in N} x_{ij} - \sum_{j\in N} x_{ji} = 0 \qquad \forall i \in N \qquad \text{(MONFP)}$$

$$x_{ij} \geq l_{ij} \qquad \forall (i,j)\in A$$

$$x_{ij} \leq u_{ij} \qquad \forall (i,j)\in A$$

The feasible set is again denoted by X, the feasible set of the integer (MONFP) is denoted by $X_{int} := X \cap \mathbb{Z}^{|A|}$.

Several authors have investigated this problem. Solutions which can be obtained by a weighted sum scalarization have been considered in [Klingman and Mote, 1982]. Burkard and others [Burkard et al., 1989, Ruhe and Fruhwirth, 1990] have used a sandwich approximation algorithm for convex curves to approximate the efficient frontier of continuous bicriteria network flow problems. Lee and Pulat and later Pulat and Huarng developed a method to completely determine the efficient frontier and all Pareto optimal solutions in the continuous bicriteria case [Lee and Pulat, 1991, Pulat et al., 1992]. Another study by Lee and Pulat treated the bicriteria integer case [Lee and Pulat, 1993]. It is this paper upon which our research builds up. We generalize the method to the case of more than two objectives. However, we will not give any technical proofs, which would be very similar to those given in [Lee and Pulat, 1993]. The structure of integer solutions in the bi- and tricriteria case was also studied in [Mustafa and Goh, 1997] and [Mustafa and Goh, 1998]. Several authors considered (MONFP) with lexicographic optimality [Calvete and Mateo, 1995, Calvete and Mateo, 1996] or max-ordering optimality [Hamacher, 1995]. Results of the latter paper will be used in Section 3.3. Finally, the method is illustrated by examples in Section 4.

Figure 2 illustrates why solving the continuous problem is not enough to find all integer Pareto optimal solutions. The point y is an integer efficient point, because there is no integer point on the edge connecting y^2 and y^3.

Due to the results mentioned above (Theorem 1 and the total unimodularity of the incidence matrix) we know that all extreme points in Y are integer. However, there may be integer points between extreme points, if a pivot step between the corresponding adjacent Pareto optimal basic solutions represents a flow change of more than one unit. In Figure 2 the points $y^2_{(1)}$, $y^2_{(2)}$, and $y^3_{(3)}$ between extreme points y^2 and y^3 are such points. These are also efficient points for the continuous problem. However, introducing the integrality requirement results in points $y^2_{(1,0)}, \ldots, y^2_{(1,3)}$ being efficient. These cannot be found solving a problem with a weighted sum of objectives. Additional problems only encountered with more than two objectives will be explained later.

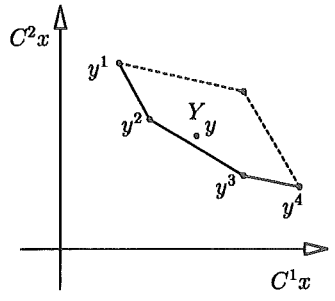


Figure 2 - A bicriteria example for integer efficient points

The general idea to solve the integer (MONFP) contains the following steps.

1. Find an initial efficient extreme point y^1.
2. Starting from y^1 find all efficient extreme points.
3. Determine all maximal efficient faces constituting the efficient frontier.

4. Find the integer solutions on the efficient frontier.
5. Find the integer efficient solutions "inside" Y.

Points 1. to 3. determine the set of efficient points of the continuous problem, points 3. and 4. describe the extra effort to solve the integer problem.

3.1 Efficient extreme points and maximal efficient faces

The determination of an initial Pareto optimal basic feasible solution and an efficient extreme point is quite easy: due to our assumptions and Theorem 1 it is enough to solve

$$\min \sum_{q=1}^{Q} \sum_{(i,j)\in A} \lambda_q c_{ij}^q x_{ij}$$

$$x \in X$$

with $\lambda_q = \frac{1}{Q}$ for q = 1,..., Q. Using the out-of-kilter algorithm may not result in a basic optimal solution. Then a step of flow changes has to be applied to find a basic solution with the same (optimal) cost.

Let us now introduce adjacency of extreme points and basic feasible solutions. Two basic feasible solutions x^1 and x^2 are called adjacent, if x^2 can be obtained from x^1 by a single pivot step. Two extreme points y^1 and y^2 of Y are called adjacent, if they are connected by an edge (a one dimensional face) of Y.

In order to find all efficient extreme points (Pareto optimal basic feasible solutions) we check all extreme points adjacent to those determined, and compare the values. In order to do that, we keep a list of all adjacent basic feasible solutions to be checked for each possibly Pareto optimal basic solution. During the procedure of checking adjacent basic solutions a pointer structure for all efficient points (corresponding to the basic solutions) will be built up. This is the objective space counterpart of the graph G in Theorem 4, which we denote by G_{eff}.

In Figure 3 an example of a face of a two-dimensional efficient frontier with an extreme point y^1 and two adjacent extreme points y^2 and y^5 is shown.

Note that throughout the procedure, we need only keep those vectors which are locally efficient with respect to the set of all points checked so far.

Definition 3

Let $Y' \subset Y$. Then $y' \in Y'$ is called locally efficient, if it is efficient with respect to Y'.

At an efficient extreme point (and its corresponding Pareto optimal basic solution) we can move to an adjacent extreme point (basic solution) by introducing a variable into the basis (and excluding another one). However, we need not check all possible nonasic variables, as the following Lemma shows.

Lemma 1

If y^t is an efficient extreme point, $y^{t'}$ is obtained by introducing (u,v) into the basis at y^t and (u,v) is in kilter for all c^q. Then $y^{t'}$ is not efficient.

Proof:

Obvious from the definition of an arc being in kilter. ♦

After having determined the efficient extreme point of Y, we use G_{eff} to determine the whole efficient frontier, which is composed of maximal efficient faces.

Definiton 4

Consider the continuous (MONFP). The convex hull of a subset of extreme points of Y, which is obtained in the boundary of Y is called a face of Y. A face F of Y is called efficient face, if all $y \in F$ are efficient. An efficient face F is maximal if there is no other efficient face F' such that $F \subset F'$.

In order to determine all maximal efficient faces we check all possible faces (which correspond to cyles in G_{eff}) for efficiency. This makes use of the result:

Lemma 2

A face F is efficient if and only if there exists an inner point $\hat{y}$ of F which is efficient.

Proof:

If a face is efficient, all points of F are efficient. On the other hand suppose $\hat{y}$ is an efficient inner point of F. By Theorem 4 this implies that there exists a $\lambda \in \mathbb{R}^Q$ such that $\hat{y}$ solves the (NFP) with objective $\lambda^T C$. By the linearity of the problem, the whole face F is optimal for this same scalarized problem. ♦

The idea of the efficiency check is illustrated in Figure 3 and formally stated in Lemma 3. An inner point $\hat{y}$ can be obtained by

$$\hat{y} = y^t + 0.5\,(\bar{c}_{(u,v)_1} + \ldots + \bar{c}_{(u,v)_t}).$$

Here y^t is an efficient extreme point of a t-dimensional face and $(u,v)_i$ are the edges introduced into the basis at y^t to move to adjacent extreme point y^i i = 1,..., t.

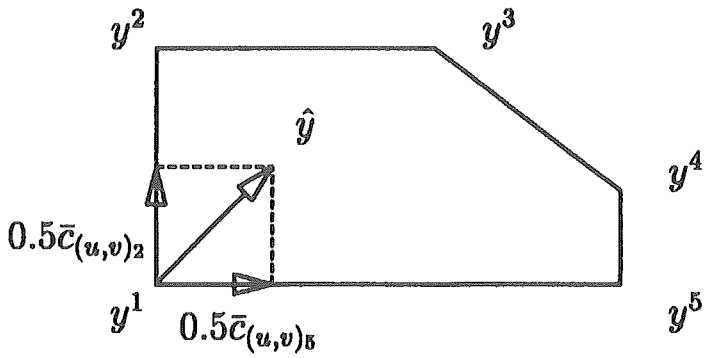


Figure 3 - An inner point of a face

Lemma 3 ([Ecker and Kouada, 1975])

Let $\hat{y}$ be an inner point of a face F of Y and let (x^0,s^0) be an optimal solution of the problem (TP) below. Then $\hat{y}$ is efficient if and only if $\sum_{q=1}^{Q} s_i^0 = 0$.

$$\begin{aligned} \min \quad & \sum_{q=1}^{Q} s_q \\ \text{s.t.} \quad & x \in X \\ & s \geq 0 \\ & Cx - \hat{y} = -Is \end{aligned} \qquad \text{(TP)}$$

3.2 Integer efficient points

In order to find all integer efficient solutions of (MONFP), we proceed in two steps. First, the integer solutions on the efficient frontier are determined. Then, modifying our original (MONFP), we determine solutions in the interior of Y.

During the determination of the efficient extreme points of Y (which are integer efficient points themselves) all those lying on efficient edges can be determined immediately. Note that there may exist integer efficient points on edges leading to non efficient extreme points. These will be determined as points "inside" or "behind" the efficient frontier.

After the determination of the maximal efficient faces, the integer points on the efficient frontier can be computed. Let F' be a t-dimensional maximal efficient face, $2 \leq t \leq Q-1$, and let y^1 be an extreme point of F'. Then select t adjacent efficient extreme points $y^2,\ldots, y^{t+1}$ such that $\{y^1,\ldots, y^{t+1}\}$ are affinely independent. This identifies t nonbasic variables x_{uv_i} at y^1.

Then all integer combinations of flow changes of the variables x_{uv_i} (and the respective cycles) leading from y^1 to the adjacent extreme points $y^2,\ldots, y^{t+1}$ define the set of integer efficient points on F'. The unit flow changes to be considered are between 1 and the flow change to reach y^i, respectively. For an illustration, see Figure 4 with y^1 and adjacent extreme points y^2 and y^5.

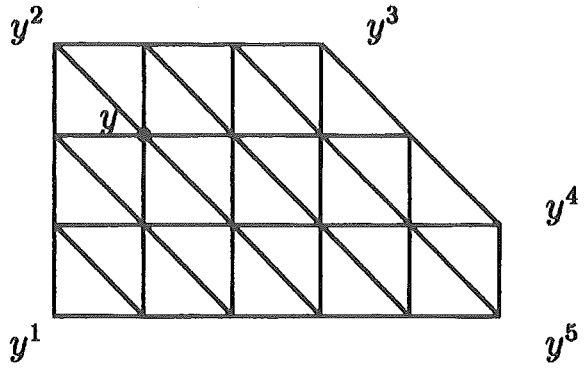


Figure 4 - Integer efficient points on a two-dimensional face

If we call two Pareto optimal integer flows adjacent, if one can be obtained from the other by flow change along a single cycle, we have the following theorem.

Theorem 6

The set of Pareto optimal integer flows corresponding to integer efficient point on the efficient frontier defines a connected graph.

Proof:

According to Theorem 2 the efficient frontier is connected. Theorem 4 says that the graph of Pareto optimal basic solutions is connected. We can therefore restrict ourselves to a maximal efficient face, say F. Let y be an efficient integer point on face F. According to the algorithm, y is obtained by combining flow changes along several cycles. Performing these backwards one after another will generate a sequence of integer points eventually ending in an efficient extreme point $\hat{y}$ of face F (see Figure 4 and the text before). According to the choice of nonbasic variables at $\hat{y}$ none of the flow changes can yield points outside F, thus all intermediate integer points (and corresponding flows) are efficient (resp. Pareto optimal). ♦

The crucial step in determining integer efficient solutions is the computation of those which are behind the continuous efficient frontier. The idea to obtain these is to change the bounds (decrease the upper bound or increase the lower bound) of some nonbasic variables by 1 at some efficient basic solution.

Lemma 4

Let y^t be an extreme point of a face F. Changing the bound of a nonbasic variable one at y^t leads to a parallel translation of a face F containing y^t.

Proof:

The slope of a face is determined by the reduced costs $\bar{c}_{ij}$. Changing the bound of a basic variable by one does not change the structure of the basis, thus $\bar{c}_{ij}$ are not changed and the slope remains the same. That the translation is nonzero follows from the fact that the original solution becomes infeasible by the bound change. ♦

Note that in the degenerate case, we may have $\bar{c}_{ij} = 0$ and therefore the result is not true in general.

The effect of changing bounds is illustrated in Figure 5.

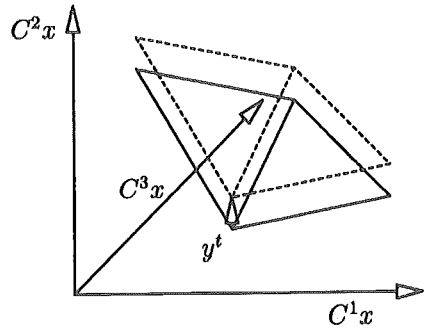


Figure 5 - Parallel translation of faces

Again, not all nonbasic variables have to be considered for bound changes.

Lemma 5

Let y^t be an efficient extreme point. Changing bounds for arcs (u,v) which are in kilter for all objectives C^q does not yield an efficient point.

As a consequence we have:

Theorem 7

Every integer efficient point of (MONFP) in the interior of Y is an integer efficient point of some (MONFP)' with modified bounds, which is on the boundary of Y'.

3.3 Lexicographic max-ordering solutions

As mentioned earlier, one will often encounter a big set of integer efficient points. Even the small example in Section 3.2 has 43 integer efficient points. Therefore a decision maker, facing such a problem will have to choose one of those. As we have proposed in Section 2 we will now illustrate how to find a lexicographic max-ordering solution as a compromise. The lex-MO optimal solution in the example (Figures 6 and 10) is highlighted by a circle around the point. It can be seen that it is located centrally in the integer efficient set: A lex-MO solution has its objectives values as equal as possible.

The main result establishes a ranking approach to the integer lex-MO (MONFP).

$$\text{lexmin}_{x\in X_{int}} \text{sort}(C^1x,\ldots, C^Qx)$$

Theorem 8

Let $\lambda\in\mathbb{R}^Q$ be such that $\lambda_{q^*} = 1$ and $\lambda_q = 0$ for all $q \neq q^*$. Furthermore, let $x^1,\ldots, x^p$ be such that

$$\lambda Cx^1 \leq \lambda Cx^2 \leq \ldots \leq \lambda Cx^p \leq Cx$$

for all $x\in X_{int}$. ($x^1,\ldots, x^p$ are the p best solutions of the integer NFP with objective C^{q^*}). Let

$$F^* := \min_{x\in X_{int}} \max_{q=1,\ldots,Q} C^qx$$

and assume $C^{q^*}x^p > F^*$. Then all max-ordering solutions are contained in $\{x^1,\ldots, x^p\}$.

Proof:

Suppose x' is a max-ordering solution of (MONFP), i.e. $\max_{q=1}^{Q} C^qx' = F^*$ and $x'\notin \{x^1,\ldots, x^p\}$. Since $C^{q^*}x^p > F^*$ we must have

$$C^{q^*}x' \geq C^{q^*}x^p > F^*,$$

a contradiction. ♦

Note that, due to the definition of max-ordering solutions, there must exist a pair p, q* such that $C^{q^*}x^p > F^*$ holds. The lex-MO solution can then be chosen from $\{x^1,\ldots, x^p\}$ by sorting and selecting the lexicographic minimum. An efficient way to do that is described in [Ehrgott, 1998].

The essential step in determining p best solutions is to find a second best solution. Then by applying branch and bound methods, further solutions can be determined. Theorem 8 provides the background of an algorithm for the integer lex-MO (MONFP). The details of the ranking method for (NFP) can be found in [Hamacher, 1995] and need not be repeated here.

4. Examples

We illustrate the proposed method by means of two examples with two and three criteria, respectively. Both will be defined for the same network of Figure 6.

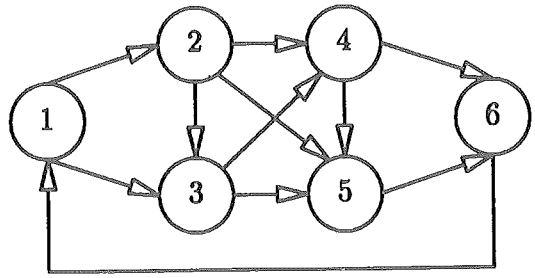


Figure 6 - Network for the examples

The cost coefficients as well as lower and upper bounds are given in the table below.

arc	(1,2)	(1,3)	(2,3)	(2,4)	(2,5)	(3,4)	(3,5)	(4,5)	(4,6)	(5,6)	(6,1)
C^1	5	8	2	6	1	2	6	5	1	2	0
C^2	1	2	7	1	7	4	2	3	9	8	0
l	0	2	0	2	0	0	3	0	0	4	10
u	9	12	8	10	6	10	10	7	10	9	10

Determining an initial extreme point using weights $\lambda_1 = \lambda_2 = \frac{1}{2}$ will yield either y^2 or y^3, depending on the implementation of the algorithm. Exploring adjacent basic feasible solutions, we eventually discover the complete efficient frontier, containing extreme points $y^1,\ldots, y^4$.

In the second step, we observe that only moving from y^2 to y^3 a flow change of more than one unit is performed: nonbasic variables $x_{(2,4)}$ is increased by 4 units. Stepwise increase at this point, together with regions where integer efficient points may be found is illustrated in Figure 7.

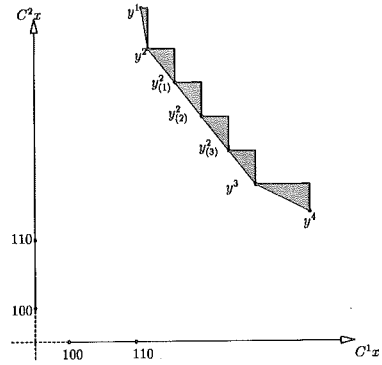


Figure 7 - Integer points on the efficient frontier

To find integer efficient points behind the efficient frontier, we choose y^2. Noting that $x_{(3,4)}$ is not in kilter for both objectives, we increase the lower bound on arc (3,4) from 0 to 1. We have to change the flow in order to obtain a feasible solution. As a result, y^2 is translated to $y^2_{(3,4)}$ [1,0]. Here the 1 denotes flow change along arc (3,4), the 0 flow change along arc (2,4), leading to y^3. When we now perform the same flow changes at the translated point, that we did at y^2 we find $y^2_{(3,4)}$ [1,1] and $y^2_{(3,4)}$ [1,2]. Finally, we note that a further increase of the lower bound does not yield more integer efficient solutions. The same is observed for all other nonbasic variables at the extreme points, thus the problem is solved. Note that conv{$y^2_{(3,4)}$ [1,0],$y^2_{(3,4)}$ [1,3]} is the translation of the efficient face conv{y^2,y^3}.

Calculating a lexicographic max-ordering solution in this problem yields $y^2_{(3)}$ = (124,123) as a compromise solution. Note that this is not a basic feasible solution. The result is shown in Figure 8.

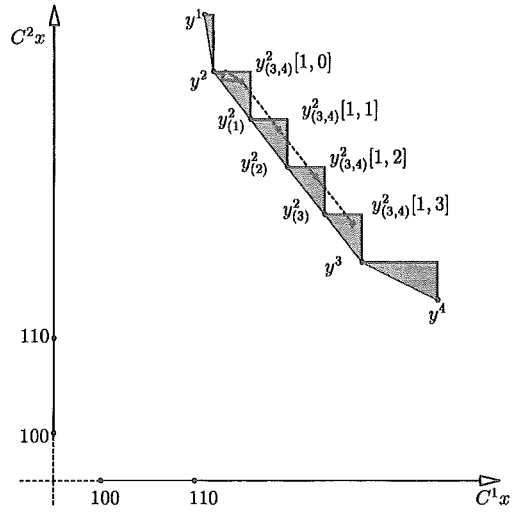


Figure 8 - All integer efficient solutions

For a tricriteria example, we simply add another objective function to the bicriteria example: C^3 = (1,3,5,9,7,6,2,3,1,2,0). We restrict ourselves to the presentation of the results.

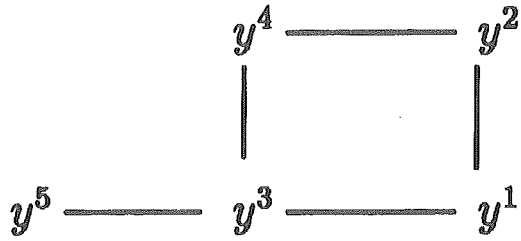


Figure 9 - Graph of efficient extreme points

The graph G_{eff} is shown in Figure 9. The objective space with all 43 integer efficient points (21 on the efficient frontier and 22 others) is shown in Figure 10. The efficient frontier itself consists of a one and a two dimensional face. Integer points on the efficient frontier are black dots, those in the interior of Y are empty dots, the lex-MO solution is indicated in Figure 10 by a circle around the point.

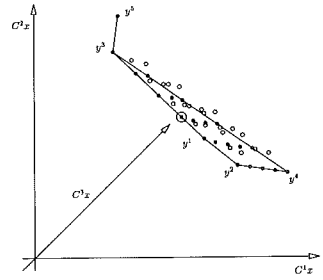


Figure 10 - The (integer) efficient set for the tricriteria example

Acknowledgment

The author wants to express his gratitude to Christina Soltau, who carried out the computations for the examples.

References

[1] Bazaraa, M. and Jarvis, J., *Linear Programming and Network Flows*, Wiley, New York (1977).

[2] Benson, H., *An outer approximation algorithm for generating all efficient points in the outcome set of a multiple objective linear programming problem*, Journal of Global Optimization 13 (1998) 1-24.

[3] Bukard, R., Rote, G., Ruhe, G. and Sieber, N., *Algorithmische Untersuchungen zu bikriteriellen kostenminimalen Flüssen in Netzwerken*, Wissenschaftliche Zeitung der technischen Hochschule Leipzig 13 (1989) 333-341.

[4] Calvete, H. and Mateo, M., *An approach for the network flow problem with multiple objectives*, Computers and Operations Research 22 (1995) 971-983.

[5] Calvete, H. and Mateo, M., *A sequential network-based approach for the multiobjective network flow problem with preemptive priorities*, in Tamiz, M. editor, Multi-Objective Programming and Goal Programming - Theory and Applications (1996) 74-86, Springer-Verlag, Berlin.

[6] Ecker, J. and Kouada, I., *Finding efficient points for linear multiple objective programs*, Mathematical Programming 8 (1975) 375-377.

[7] Ehrgott, M., *Lexicographic max-ordering - a solution concept for multicriteria combinatorial optimization*, In Schweigert, D. editor, Methods of Multicriteria Decision Theory, Proceedings of the 5th Workshop of the DGOR-Working Group Multicriteria Optimization and Decision Theory (1995) 55-66.

[8] Ehrgott, M., *A characterization of lexicographic max-ordering solutions*, In Göpfert, A., Seeländer, J. and Tammer, C., editors, Methods of Multicriteria Decision Theory, Proceedings of the 6th Workshop of the DGOR-Working Group Multicriteria Optinization and Decision Theory Alexisbad 1996, vol. 2389 (1997) 193-202, Deutsche Hochschulschriften, Hänsel-Hohenhausen, Egelsbach.

[9] Ehrgott, M., *Multiple Criteria Optimization - Classification and Methodology*, Shaker Verlag, Aachen (1997).

[10] Ehrgott, M., *Discrete decision problems, multiple criteria optimization classes and lexicographic max-ordering*, In Stewart, T. and van den Honert, E. editors, Trends in Multicriteria Decision Making, Lecture Notes in Economics and Mathematical Systems 465 (1998) 31-44, Springer-Verlag, Berlin.

[11] Geoffrion, A., *Proper efficiency and the theory of vector maximization*, Journal of Mathematical Analysis and Applications 22 (1968) 618-630.

[12] Hamacher, H., *K best network flows*, Annals of Operations Research 57 (1995) 65-72.

[13] Isermann, H., *Proper efficiency and the linear vector maximum problem*, Operations Research 22 (1974) 189-191.

[14] Isermann, H., *The enumeration of the set of all efficient solutions for a linear multiple objective program*, Operational Research Quarterly 28 (1977) 711-725.

[15] Klingman, D. and Mote, J., *Solution approaches for network flow problems with multiple criteria*, Advances in Management Science 1 (1982) 1-30.

[16] Lee, H. and Pulat, P., *Bicriteria network flow problems: continuous case*, European Journal of Operational Research 51 (1991) 119-126.

[17] Lee, H. and Pulat, P., *Bicriteria network flow problems: integer case*, European Journal of Operational Research 66 (1993) 148-157.

[18] Mustafa, A. and Goh, M., *Characteristics of the efficient solutions of bicriteria and tricriteria network flow problems*, In Caballero, R., Ruiz, F. and Steuer, R. editors, Advances in Multiple Objective and Goal Programming, Lecture Notes in Economics and Mathematical Systems 455 (1997) 131-139, Springer-Verlag, Berlin.

[19] Mustafa, A. and Goh, M., *Finding integer efficient solutions for bicriteria and tricriteria network flow problems*, Computers and Operations Research 25 (1998) 139-157.
[20] Naccache, P., *Connectedness of the set of nondominated outcomes in multicriteria optimization*, Journal of Optimization Theory and Applications 25 (1978) 459-467.
[21] Pulat, P., Huarng, F. and Lee, H., *Efficient solutions for the bicriteria network flow problem*, Computers and Operations Research 19 (1992) 649-655.
[22] Ruhe, G. and Fruhwirth, B., *ε-optimality for bicriteria programs and its application to minimum cost flows*, Computing 44 (1990) 21-34.
[23] Warburton, A., *Quasiconcave vector maximization: connectedness of the sets of Pareto-optimal and weak Pareto-optimal alternatives*, Journal of Optimization Theory and Applications 40 (1983) 537-557.

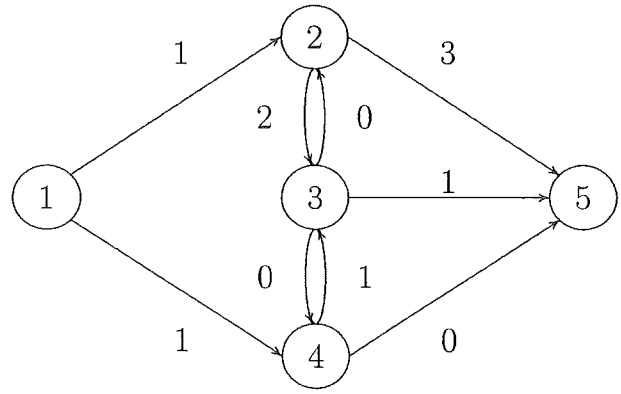


Figura 2 - Rede

Para esta rede, e considerando K = 4, obtemos os seguintes trajectos mais curtos,

$p_1 = \langle 1,4,5 \rangle$, $p_2 = \langle 1,4,3,4,5 \rangle$, $p_3 = \langle 1,2,3,4,5 \rangle$, $p_4 = \langle 1,4,3,5 \rangle$,

com custos $c(p_1) = 1$, $c(p_2) = 2$ e $c(p_3) = c(p_4) = 3$. É de notar que podem existir trajectos distintos com o mesmo custo (o que acontece com p_3 e p_4). Isto significa que o conjunto dos K trajectos mais curtos numa rede pode não ser único, o que no entanto não interfere no que se segue, dado que pretendemos apenas calcular um qualquer conjunto de K trajectos mais curtos de s para t em (N,A).

O trajecto p_1 é uma árvore de acordo com a sua definição usual - Figura 3(a). Consideremos agora os dois trajectos mais curtos, p_1 e p_2. O trajecto p_2 é constituído por uma parte inicial comum a p_1, o subtrajecto $\langle 1,4 \rangle$, "desviando-se" de p_1 no segundo nó, 4. Estes dois trajectos formam a árvore representada na Figura 3(b). Analogamente, o terceiro trajecto mais curto, p_3, contém um subtrajecto comum a p_1 e um subtrajecto comum a p_2, iguais e constituídos apenas pelo nó inicial. Deste modo, p_3 "desvia-se" dos trajectos anteriores, p_1 e p_2, naquele nó - Figura 3(c). Por fim, p_4 coincide com p_1 em $\langle 1,4 \rangle$, com p_2 em $\langle 1,4,3 \rangle$, e com p_3 em $\langle 1 \rangle$, ou seja, tem um subtrajecto inicial comum à árvore formada por p_1, p_2 e p_3, $\langle 1,4,3 \rangle$, "desviando-se" destes trajectos no nó 3 - Figura 3(d).

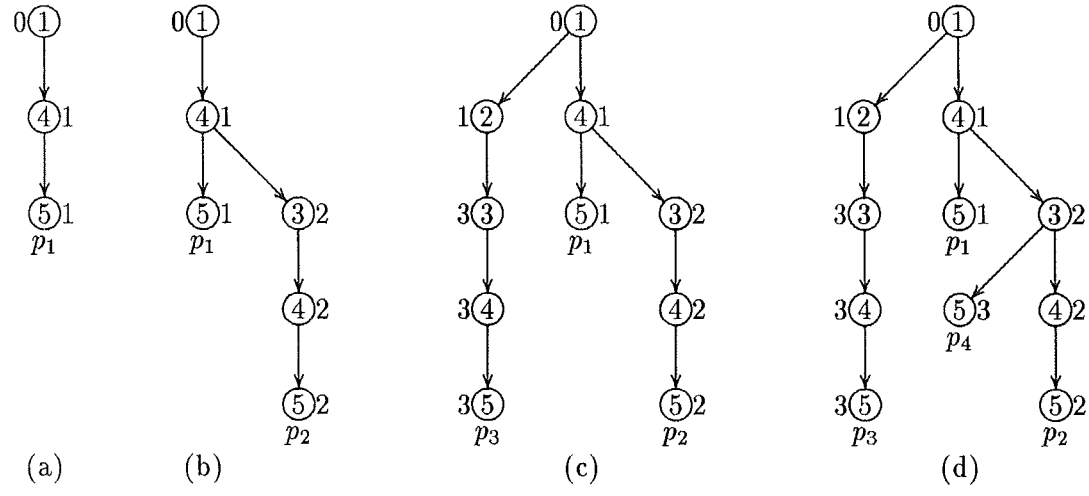


Figura 3 - Árvores de trajectos mais curtos de 1 para 5

Na realidade a estrutura formada, tal como a apresentámos, não é a de uma árvore de acordo com a definição usual uma vez que alguns dos nós e arcos aparecem repetidos. No entanto, de modo a evitar estas situações, podemos utilizar uma função que a cada nó de um trajecto p_i, $i \in \{1,\ldots, K\}$, associa um número natural, por forma a que as imagens do mesmo nó em trajectos diferentes sejam também diferentes. Sendo um arco definido por um par de nós, pode ocorrer uma situação análoga para arcos pertencendo a mais que um trajecto de uma árvore. A utilização de uma tal função permite também distinguir tais arcos. No presente trabalho não será utilizada esta função para não sobrecarregar a notação, pelo que a distinção dos nós e arcos deverá ser feita mediante o contexto.

Denotaremos por T_k, para $k \in \{1,\ldots, K\}$, a árvore constituída pelos k trajectos mais curtos de s para t em (N,A), $p_1,\ldots, p_k$. O resultado que se apresenta em seguida formaliza o conceito, introduzido de forma intuitiva, de árvore dos k trajectos mais curtos entre dois nós de uma rede.

Teorema 5 Seja $k \in \{1,\ldots, K^1\}$; então o (k+1)-ésimo trajecto mais curto de s para t em (N,A) é da forma $p_{k+1} = sub_j(s,u)\Diamond\langle u,v\rangle\Diamond p$, para algum $j \in \{1,\ldots, k\}$, com $(u,v) \in (N,A)$ não pertencente à árvore T_k e onde $p \in P_{vt}$.

Demonstração: Utilizando indução sobre k comecemos por analisar o caso k = 1.

Sendo p_1 o trajecto mais curto de s para t e p_2 o segundo trajecto mais curto entre os mesmos nós, p_1 e p_2 são distintos. Seja então $u \neq t$ o nó de p_1 e p_2 mais distante (relativamente ao número de arcos) de s, tal que $sub_1(s,u) = sub_2(s,u)$ e portanto, sendo (u,v') e (u,v") os arcos de p_1 e p_2, respectivamente, com início em u, e consequentemente distintos, $sub_1(s,u)\Diamond\langle u,v'\rangle = sub_1(s,v') \neq sub_2(s,v") = sub_2(s,u)\Diamond\langle u,v"\rangle$. A árvore T_1 é constituída apenas pelo trajecto p_1 e portanto não contém o arco (u,v). Concluímos pois que $p_2 = sub_j(s,u)\Diamond\langle u,v\rangle\Diamond p$, com $j = 1 < 2$ e $p \in P_{vt}$.

Consideremos agora que, para todo o $i \leq k$, $p_i = sub_{j_i}(s,u_i)\Diamond\langle u_i,v_i\rangle\Diamond q_i$, com $j_i \in \{1,\ldots, i-1\}$ e (u_i,v_i) não pertencente a T_{i-1} e $q_i \in P_{v_it}$.

Seja p_{k+1} o (k+1)-ésimo trajecto mais curto de P; então p_{k+1} coincide com cada p_j, para todo o $j \in \{1,\ldots, k\}$, desde o nó inicial até um nó x_j distinto do nó t uma vez que $p_{k+1} \notin \{p_1,\ldots, p_k\}$. De entre estes nós $x_1,\ldots, x_k$, denotemos por $u = x_j$, para algum $j \in \{1,\ldots, k\}$, aquele que está mais distante de s em p_{k+1}. Então $p_{k+1} = sub_j(s,u)\Diamond\langle u,v\rangle\Diamond p$, com $p \in P_{vt}$, onde, de acordo com a escolha de u, (u,v) não está na árvore T_k. ♦

O nó u de p_k é denominado por nó desvio de p_k e é denotado por $v^k_{\alpha_k}$, onde α_k é a ordem respectiva no trajecto p_k (e p_j). O arco (u,v) é, geralmente, denominado arco desvio de p_k e p_j o seu trajecto pai. Por convenção, o nó desvio de p_1 é o nó inicial, s.

Com base nas Figuras 2 e 3 podemos verificar que os nós desvio de p_1, p_2, p_3 e p_4 são, respectivamente, $v^1_1 = 1$, $v^2_2 = 4$, $v^3_1 = 1$ e por fim $v^4_3 = 3$.

Deste modo a determinação dos K trajectos mais curtos pode ser feita através da construção de uma árvore.

4.2.2 Um algoritmo baseado na construção da árvore de trajectos

Ao estudar este problema, facilmente se conclui que o ideal seria determinar o menor número possível de trajectos, isto é, determinar somente os K trajectos pretendidos. No entanto, como veremos no que se segue, os algoritmos que propomos determinam uma sobreárvore da árvore, T_K, dos K trajectos mais curtos.

O primeiro trajecto a determinar é o mais curto de s para t em (N,A), sendo para isso utilizado um qualquer algoritmo. A partir deste trajecto, p_1, como poderemos obter p_2? De acordo com o Teorema 5, p_2 é da forma $sub_1(s,u)\Diamond\langle u,v\rangle\Diamond p$, onde $(u,v)\notin p_1$ e $p\in P_{vt}$, pelo que poderíamos tentar determinar todos os trajectos com esta forma, o mais curto dos quais seria p_2. Seguindo este raciocínio seria necessário, para cada nó $u\in p_1$, hipotético nó desvio de p_2, e cada arco $(u,v)\notin p$, conhecer todos os trajectos de P_{vt}. Este procedimento pode ser melhorado se atendermos ao Teorema 6.

Dada a rede (N,A) e o nó terminal, t, denotaremos por T_t a árvore dos caminhos mais curtos com raiz em t, constituída pelos caminhos mais curtos de todos os nós de (N,A) para t. Nesta árvore, dado um nó u, denotamos por $T_t(u)$ o caminho de u para t em T_t. A Figura 4 representa a árvore T_5 para a rede (N,A) da Figura 2, associando a cada nó u o valor $c(T_5(u))$.

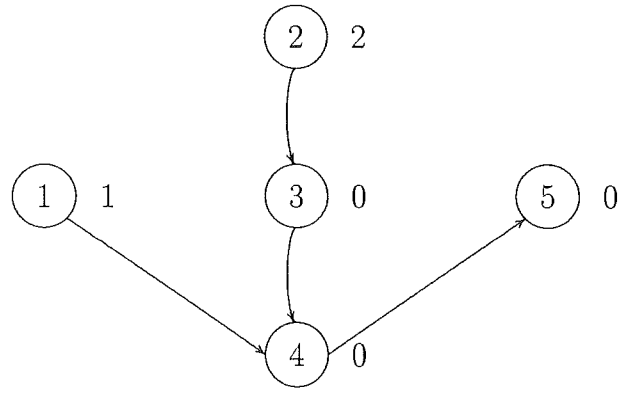


Figura 4 - Árvore dos trajectos mais curtos com raiz em t = 5

Teorema 6 Seja $k\in\{1,\ldots, K\}$; então p_{k+1} é da forma $sub_j(s,v_{\alpha_{k+1}}^{k+1})\Diamond\langle v_{\alpha_{k+1}}^{k+1},v\rangle\Diamond T_t(v)$, para algum $j\in\{1,\ldots, k\}$, onde $(v_{\alpha_{k+1}}^{k+1},v)$ é um arco não pertencente à árvore T_k.

Demonstração: No Teorema 5 vimos que $p_{k+1} = sub_j(s,v_{\alpha_{k+1}}^{k+1})\Diamond\langle v_{\alpha_{k+1}}^{k+1},v\rangle\Diamond p$, pelo que basta mostrar que p tem que ser um trajecto mais curto de v para t. Suponhamos que isto não acontecia, isto é, que $c(T_t(v)) < c(p)$ e consideremos o trajecto $q = sub_{k+1}(s,v)\Diamond T_t(v)\in P$. Então, como o arco de q com início em $v_{\alpha_{k+1}}^{k+1}$ não está na árvore T_k, $q\notin\{p_1,\ldots, p_k\}$. Além disso

$$c(q) = c(sub_{k+1}(s,v)) + c(T_t(v)) < c(sub_{k+1}(s,v)) + c(p) = c(p_{k+1}),$$

pelo que p_{k+1} não seria o (k+1)-ésimo trajecto mais curto de s para t, como queríamos mostrar. ♦

Deste modo, voltando à determinação de p_2 a partir de p_1, bastaria calcular para cada nó u de p_1 o trajecto $sub_1(s,u)\Diamond\langle u,v\rangle\Diamond T_t(v)$, para todos os arcos $(u,v)\notin p_1$. De acordo com este procedimento são gerados vários elementos de P, o mais curto dos quais é p_2. Ou seja, se a partir de p_1 forem gerados estes trajectos, candidatos a p_2, e os colocarmos num conjunto X, após a análise de todos os nós de p_1, o trajecto mais curto de entre os trajectos de X é p_2.

Este raciocínio, depois de generalizado de modo a determinar p_k, foi o utilizado no algoritmo proposto por Hoffman e baseado na árvore de trajectos, [7]. Tem no entanto o inconveniente de aumentar muito rapidamente o número de trajectos na árvore em construção, uma vez que, para cada nó u de um trajecto nessa árvore, são colocados todos os arcos não repetidos com início em u.

Para tentarmos diminuir o número de trajectos determinados a partir do nó u de p_1 basta observar que se u for de facto o nó desvio de p_2, então devemos escolher, entre todos aqueles trajectos, aquele que for o mais curto. Por outras palavras, basta determinar $sub_1(s,u)\Diamond\langle u,v\rangle\Diamond T_t(v)$, onde $v\in N$ é tal que $(u,v)\notin p_1$, e minimiza o custo de $\langle u,v\rangle\Diamond T_t(v)$, isto é, minimiza $c_{uv} + c(T_t(v))$. Deste modo, para cada nó u são analisados todos os arcos $(u,v)\notin p_1$, sendo escolhido um nó v, e consequentemente um único trajecto $sub_1(s,u)\Diamond\langle u,v\rangle\Diamond T_t(v)$. Os trajectos gerados são também acrescentados ao conjunto X de candidatos.

É de notar que, num passo k, o conjunto $X\cup\{p_1,\ldots, p_k\}$ contém todos os trajectos calculados até então, ou seja, contém os trajectos da árvore construída pelo algoritmo até àquele momento, constituída pelos k trajectos mais curtos já enumerados e por alguns candidatos a j-ésimos trajectos mais curtos, com $j > k$.

A enumeração de trajectos é conseguida a partir de sucessivas escolhas (e consequente remoção) de um trajecto no conjunto X. Deste modo, apesar dos trajectos não serem calculados por ordem dos seus custos, repetindo este procedimento, num passo genérico k, com $k\in\{1,\ldots, K\}$, será escolhido o elemento de X com menor custo, que será o k-ésimo trajecto mais curto de s para t em (N,A). Ao retirar p_k do conjunto X devem ser gerados novos trajectos, de forma análoga à indicada para $k = 1$, que serão candidatos a p_j, com $j > k$.

De modo a que um dado trajecto não seja determinado mais do que uma vez, ao escolher p_k no conjunto X são analisados apenas os nós que se seguem ao nó desvio, inclusivé, e, para cada um destes nós, digamos u, procura-se $v\in N$ tal que $(u,v)\in A$, não pertence à sub-árvore com raiz em u construída até ao momento e que, nestas condições, minimiza o custo do trajecto $\langle u,v\rangle\Diamond T_t(v)$.

Denotemos por $A_{T_k}(u)$ o conjunto dos arcos da árvore dos trajectos no passo k do algoritmo com início no nó u. Pretendemos pois determinar outro nó v tal que $(u,v)\notin A_{T_k}(u)$ e v minimiza $c_{uv}+c(T_k(v))$. Mais uma vez, cada trajecto obtido deste modo é acrescentado ao conjunto X. É de notar que quando $k = 1$ o único trajecto na árvore é p_1, pelo que $(u,v)\notin A_{T_k}(u)$ se e somente se $(u,v)\notin p_1$, como havíamos exigido.

Os passos principais do algoritmo acima descrito são apresentados resumidamente em seguida.

Algoritmo 1: Algoritmo baseado na construção da árvore dos trajectos

Determinar T_t

$k \leftarrow 0$

$X \leftarrow \{T_1(s)\}$

Enquanto ((k < K) e ($X \neq \phi$)) **Fazer**

$k \leftarrow k+1$

$p_k \leftarrow$ trajecto com menor custo em X

$\alpha_k \leftarrow$ ordem do nó desvio, $v^k_{\alpha_k}$, de p_k

$X \leftarrow X - \{p_k\}$

Para (todo o $u \in sub_k(v^k_{\alpha_k}, t)$) **Fazer**

Se ($A - A_{T_k}(u) \neq \phi$) **Então**

$v \leftarrow$ nó tal que $c_{uv}+c(T_t(v))$ é mínimo em $A - A_{T_k}(u)$

$p \leftarrow sub_k(s,u) \Diamond \langle u,v \rangle \Diamond T_t(v)$

$X \leftarrow X \cup \{p\}$

FimSe

FimPara

FimEnquanto

Nas Figuras 5 e 6 encontram-se representadas as árvores de trajectos construídas no final dos primeiros quatro passos do Algoritmo 1, quando aplicado à rede da Figura 2.

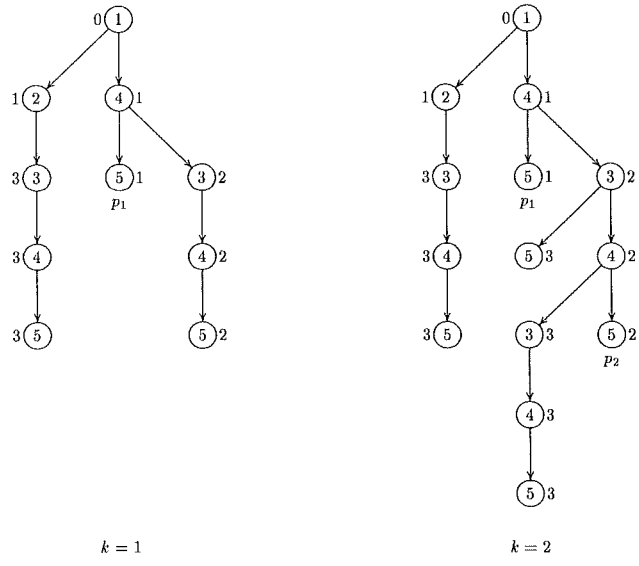


Figura 5 - Árvores de trajectos obtidas no final dos passos 1 e 2

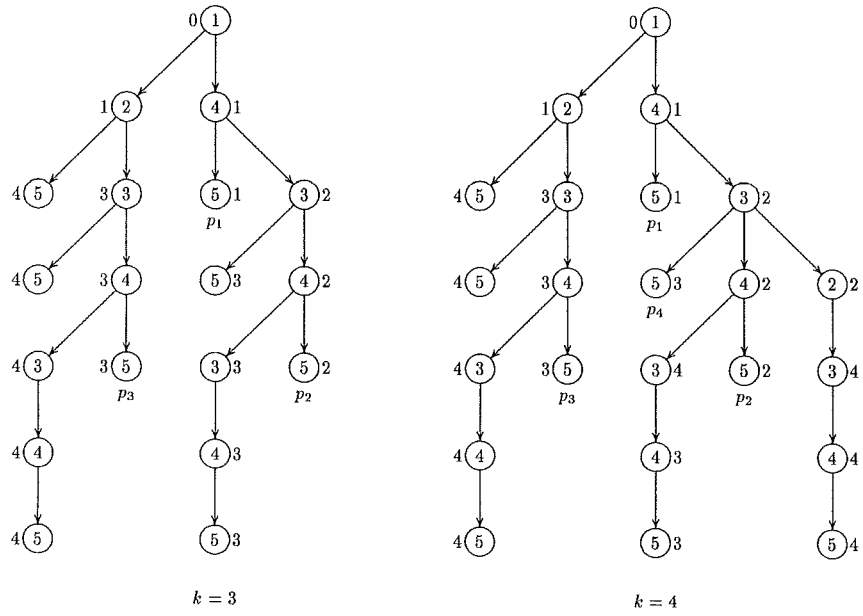


Figura 6 - Árvores de trajectos obtidas no final dos passos 3 e 4

Como referimos, o primeiro trajecto determinado, p_1, é o mais curto de s = 1 para t = 5, escolhido quando k = 1. Sendo o seu nó desvio $v_1^1 = 1$, serão analisados sucessivamente os nós 1, 4 e 5. A partir do nó 1 apenas podemos escolher o arco (1,2) (uma vez que (1,4) pertence a p_1) e então o novo trajecto, $q_1 = \langle 1,2 \rangle \Diamond T_t(2) = \langle 1,2,3,4,5 \rangle$ é acrescentado ao conjunto X. Em seguida é analisado o nó 4, a partir do qual somos levados a escolher o único arco possível, (4,3), obtendo então o trajecto $q_2 = \langle 1,4 \rangle \Diamond \langle 4,3 \rangle \Diamond T_t(3) = \langle 1,4,3,4,5 \rangle$. A partir de t = 5 não é gerado qualquer trajecto, dado não existirem arcos com início neste nó. No final deste passo a árvore determinada contém os trajectos p_1, q_1 e q_2, e $X = \{q_1, q_2\}$. No passo seguinte k = 2 e p_2 é o trajecto mais curto em X, isto é, $p_2 = \langle 1,4,3,4,5 \rangle$, cujo nó desvio é $v_2^2 = 4$. Depois de analisado o nó 4 é analisado o nó 3, a partir do qual podemos escolher $\langle 3,2 \rangle \Diamond T_t(2)$ ou $\langle 3,5 \rangle$, com custos 3 ou 1, respectivamente. Uma vez que o mais curto entre estes é $\langle 3,5 \rangle$, o novo trajecto será $\langle 1,4,3,5 \rangle$. O algoritmo continua do mesmo modo até serem escolhidos K trajectos em X. Para K = 4 são ainda realizados dois passos do algoritmo, para k = 3 e k = 4, representados, como se referiu, na Figura 6. Quando k = 3, é escolhido $p_3 = \langle 1,2,3,4,5 \rangle$, a partir do qual são acrescentados a X os trajectos $\langle 1,2,5 \rangle$, $\langle 1,2,3,5 \rangle$ e $\langle 1,2,3,4,3,4,5 \rangle$. Analogamente, para k = 4 é escolhido $p_4 = \langle 1,4,3,5 \rangle$, a partir do qual é acrescentado a X um único trajecto, $\langle 1,4,3,2,3,4,5 \rangle$.

4.2.3 O algoritmo MPS

O algoritmo que apresentamos nesta subsecção, também ele novo, combina o algoritmo anterior com uma substituição do custo dos arcos e uma diferente representação da rede (N,A). O objectivo desta combinação é construir exactamente a mesma árvore obtida com o algoritmo da subsecção anterior, mas fazê-lo de forma mais eficiente. Para obter mais pormenores acerca

deste algoritmo sugerimos a consulta de [6] ou [11]. Na última destas referências não só se aborda todo o algoritmo mas também a forma utilizada para representação de (N,A); na primeira pode ver-se mais pormenorizadamente a substituição de custos que é realizada.

Custos reduzidos

Voltemos a considerar uma árvore dos trajectos mais curtos de todos os nós para t, T_t, e dado $u \in N$ denotemos por π_u^t o custo do trajecto de u para t na árvore T_t. Com base em T_t associemos a cada $(i,j) \in A$ o respectivo custo reduzido, definido por $\bar{c}_{ij} = \pi_j^t - \pi_i^t + c_{ij}$. Utilizando estes valores definimos custo reduzido de um trajecto, de modo análogo ao custo de um trajecto, por $\bar{c}(p) = \sum_p \bar{c}_{ij}$.

A Figura 7 representa a rede (N,A) da Figura 2, associando-se a cada arco o respectivo custo reduzido. Sendo $p = \langle 1,2,5 \rangle$ um trajecto de 1 para 5 em (N,A), de acordo com a definição apresentada, o custo reduzido de p é $\bar{c}(p) = \bar{c}_{12} + \bar{c}_{25} = 3$.

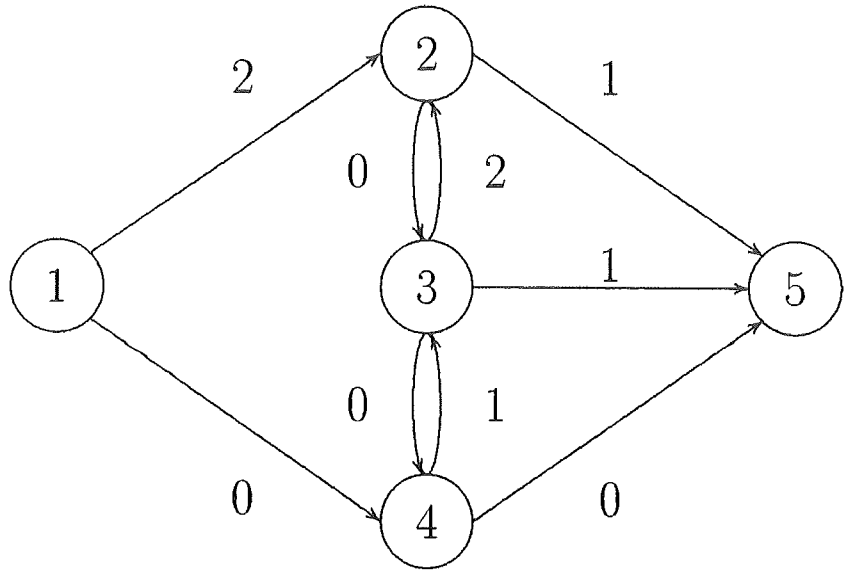


Figura 7 - Rede (N,A) com custos reduzidos

É possível demonstrar que é indiferente enumerar trajectos quer utilizando os respectivos custos quer os custos reduzidos, o que é feito seguidamente, no Lema 1 e no Teorema 7.

Lema 1 Seja p um trajecto qualquer de P e p_1 um trajecto mais curto, também elemento de P. Então $\bar{c}(p) = c(p) - c(p_1)$.

Demonstração: Seja $p = \langle v_1 = s, \ldots, v_\ell = t \rangle \in P$, então

$$\bar{c}(p) = \sum_p \bar{c}_{ij} = \sum_p (\pi_j^t - \pi_i^t + \bar{c}_{ij}) = \pi_{v_2}^t - \pi_{v_1}^t + c_{v_1v_2} + \pi_{v_3}^t - \pi_{v_2}^t + c_{v_2v_3} + \ldots +$$

$$+ \pi_{v_\ell}^t - \pi_{v_{\ell-1}}^t + c_{v_{\ell-1}v_\ell} = c(p) + \pi_t^t - \pi_s^t.$$

Sendo p_1 um trajecto mais curto em P, $c(p_1) = \pi_s^t - \pi_t^t$, e então $\bar{c}(p) = c(p) - c(p_1)$, como pretendíamos mostrar. ♦

Teorema 7 Sejam p e q dois trajectos quaisquer de P. Então:

1. $\overline{c}(p) = \overline{c}(q)$ se e somente se $c(p) = c(q)$;
2. $\overline{c}(p) < \overline{c}(q)$ se e somente se $c(p) < c(q)$.

Demonstração: As demonstrações dos pontos 1 e 2 são análogas, pelo que apresentaremos apenas a demonstração de 1.

Consideremos pois dois trajectos $p,q \in P$; então, pelo Lema 1, $\overline{c}(p) = \overline{c}(q)$ se e somente se $c(p) - c(p_1) = c(q) - c(p_1)$ e $c(p) - c(p_1) = c(q) - c(p_1)$ o que ocorre se e só se $c(p) = c(q)$, como queríamos demonstrar. ♦

Deste modo, é possível enumerar trajectos, quer pelos seus custos quer pelos seus custos reduzidos, pelo que é também possível aplicar o algoritmo anterior à rede (N,A) realizando a substituição $\overline{c}_{ij} = \pi_j^t - \pi_i^t + c_{ij}$, para todos os arcos $(i,j) \in A$. De facto, assim continuamos a determinar os K trajectos com menor custo reduzido, e consequentemente os K trajectos com menor custo. Mas, qual a utilidade deste procedimento? Isto é, existirá alguma vantagem na utilização de custos reduzidos em vez da utilização dos custos usuais de cada um dos arcos? Para responder a esta questão comecemos por notar que os custos reduzidos verificam algumas propriedades importantes, que enunciamos e demonstramos no Teorema 8.

Teorema 8 Seja $\overline{c}_{ij} = \pi_j^t - \pi_i^t + c_{ij}$ o custo reduzido associado a T_t, para todo o $(i,j) \in A$. Então:

1. $\overline{c}_{ij} \geq 0$, para todo o $(i,j) \in A$;
2. $\overline{c}_{ij} = 0$, para todo o $(i,j) \in A \cap T_t$.

Demonstração:

1. Seja $(i,j) \in A$ e sejam $T_t(i)$ e $T_t(j)$, respectivamente, trajectos mais curtos de i para t e de j para t em (N,A) (e simultaneamente na árvore T_t). Então, sendo $\langle i,j \rangle \Diamond T_t(j)$ também um trajecto de i para t, $c(\langle i,j \rangle \Diamond T_t(j)) = c_{ij} + \pi_j^t \geq \pi_i^t = c(T_t(i))$, e portanto $\overline{c}_{ij} = \pi_j^t - \pi_i^t + c_{ij} \geq 0$.
2. Consideremos agora que é dado $(i,j) \in A$, pertencente também a T_t. Então $\langle i,j \rangle \Diamond T_t(j) = T_t(i)$ e $c_{ij} + \pi_j^t = \pi_i^t$, logo $\overline{c}_{ij} = 0$. ♦

Como consequência deste Teorema temos imediatamente o seguinte resultado.

Corolário 8.1 Seja p um trajecto qualquer pertencente a T_t; então $\overline{c}(p) = 0$.

Com base neste Corolário podemos responder à questão que atrás foi colocada. De facto, no algoritmo apresentado na subsecção anterior, dado um nó u do k-ésimo trajecto mais curto é necessário encontrar um nó v tal que $(u,v) \in A\text{-}A_{T_k}(u)$ e que minimize $c_{uv}+c(T_t(v))$. Utilizando custos reduzidos em substituição dos custos usuais, isto é o mesmo que minimizar $\overline{c}_{uv} + \overline{c}(T_t(v))$. Como $T_t(v)$ é um trajecto de T_t, então $c(T_t(v)) = 0$, bastando portanto encontrar v tal que $(u,v) \in A\text{-}A_{T_k}(u)$ que minimize $\overline{c}_{uv}$. Podemos então concluir que a utilização dos custos reduzidos dos arcos da rede no Algoritmo 1 simplifica realmente a determinação de cada nó v neste algoritmo.

Representação da rede na sorted forward star form

Como vimos, em cada passo do algoritmo são analisados todos os nós u de p_k que se seguem ou coincidem com o seu nó desvio e, para cada u, é necessário determinar $v \in A$ tal que $(u,v) \in A$ tenha um custo reduzido mínimo de entre os sucessores de arcos com início em u que não se encontrem na árvore construída até ao momento. Além disso, como se demonstra em seguida, $\bar{c}_{ij}$ representa o custo a acrescentar ao trajecto obtido do trajecto mais curto de i para t, quando se obriga a que seja utilizado o arco (i,j).

Teorema 9 Seja $(i,j) \in A$ e $p = \langle i,j \rangle \Diamond T_t(j)$; então $c(p) = \bar{c}_{ij} + c(T_t(i))$.

Demonstração: Seja $p = \langle i,j \rangle \Diamond T_t(j)$, para $(i,j) \in A$. Então,

$$c(p) = c_{ij} + c(T_t(j)) = \pi_i^t - \pi_j^t + \bar{c}_{ij} + \pi_j^t = \bar{c}_{ij} + \pi_i^t = \bar{c}_{ij} + c(T_t(i)),$$

como pretendíamos demonstrar. ♦

Podemos deste modo concluir que, ao analisar um nó u, é de toda conveniência escolher "mais cedo" sucessores de arcos com início em u que tenham um menor custo reduzido. Com esse objectivo podemos representar (N,A) na sorted forward star form, [6]. Nesta forma, o conjunto A encontra-se ordenado pelos antecessores dos arcos da rede, e os arcos com iguais antecessores encontram-se ordenados por ordem não decrescente dos respectivos custos reduzidos.

Utilizando a sorted forward star form para a rede representada na Figura 7 obteríamos o conjunto de arcos $A = \{(1,4),(1,2),(2,3),(2,5),(3,4),(3,5),(3,2),(4,5),(4,3)\}$.

Considerando (N,A) na sorted forward star form, um nó u de p_k e sendo (u,x) o arco de p_k com início em u, ao analisar u será escolhido o nó v tal que (u,v) é o arco de (N,A) (representada na sorted forward star form) que se segue a (u,x), se tal existir. O arco (u.v) será aquele que tem custo reduzido imediatamente superior, ou eventualmente igual, ao de (u,x).

Algoritmo

Como referimos, o algoritmo MPS segue os passos do Algoritmo 1, utilizando não só custos reduzidos em vez dos custos usuais, mas também a rede (N,A) representada na sorted forward star form. Os passos principais deste algoritmo são apresentados resumidamente em seguida.

Algoritmo 2: Algoritmo de Martins, Pascoal e Santos (MPS)

Determinar T_t

Para (todo o $(i,j) \in A$) **Fazer** $\bar{c}_{ij} \leftarrow \pi_j^t - \pi_i^t + c_{ij}$

Colocar (N,A) na sorted forward star form

$k \leftarrow 0$

$X \leftarrow \{T_t(s)\}$

Enquanto $((k < K)$ e $(X \neq \phi))$ **Fazer**

$k \leftarrow k+1$

$p_k \leftarrow$ trajecto com menor custo reduzido em X
$\alpha_k \leftarrow$ ordem do nó desvio, $v^k_{\alpha_k}$, de p_k
$X \leftarrow X - \{p_k\}$
Para (todo o $u \in sub_k(v^k_{\alpha_k}, t)$) **Fazer**
$v \leftarrow$ nó tal que (u,v) é o arco que se segue ao arco com início em u de p_k
Se (v está definido) **Então**
$p \leftarrow sub_k(s,u) \Diamond \langle u,v \rangle \Diamond T_t(v)$
$X \leftarrow X \cup \{p\}$
FimSe
FimPara
FimEnquanto

A árvore obtida com este algoritmo coincide, como foi referido, com a que é obtida pelo Algoritmo 1, sendo no entanto construída de forma mais eficiente.

5. A Enumeração de Caminhos

Podendo o Problema dos K Caminhos mais Curtos ser considerado um subproblema do Problema dos K Trajectos mais Curtos, poderíamos também ser levados a sugerir, para aquele problema, o tipo de algoritmos e a classificação que foi mencionada na secção anterior. No entanto, os problemas são suficientemente diferentes para que não possa ser feita uma tal analogia. De facto, como vimos atrás, no problema dos K Caminhos mais Curtos os trajectos determinados devem restringir-se apenas a caminhos, isto é, a trajectos sem repetição de nós. Isto reflecte-se, desde logo, nas condições em que cada um dos problemas verifica o Princípio de Optimalidade.

Analogamente ao que foi feito na secção 4, $p_k = \langle s = v^k_1, \ldots, t = v^k_{\ell_k} \rangle$ e $sub_k(i,j)$ denotarão o k-ésimo caminho mais curto de s para t e o subcaminho de p_k do nó i para o nó j, respectivamente.

5.1 O Princípio de Optimalidade

Como veremos em seguida, ao contrário do que acontece para K = 1 em que podemos afirmar que o Princípio de Optimalidade é satisfeito se (N,A) não contiver ciclos negativos, mesmo sob estas condições não é possível garantir o mesmo para K > 1.

Comecemos, como foi feito para trajectos, por considerar o caso K = 1. Sob certas condições podemos garantir, como se demonstra em seguida, que o Problema do Caminho mais Curto verifica o Princípio de Optimalidade.

Teorema 10 Se (N,A) não contém ciclos de custo negativo, então todos os caminhos mais curtos de s para t em (N,A), são constituídos por subcaminhos mais curtos.

Demonstração: Sabemos já (ver secção 3) que o Problema do Caminho mais Curto é finito qualquer que seja a rede considerada. Suponhamos que existe um caminho mais curto de s para t, p, e que dados u e v, dois nós de p, o subtrajecto $sub_p(u,v)$ não é óptimo. Então existe um

caminho $q \in P_{uv}$ tal que $c(q) < c(sub_p(u,v))$. A partir da concatenação dos caminhos $sub_p(s,u)$, q e $sub_p(v,t)$ obtemos o trajecto de s para t, $q' = sub_p(s,u) \Diamond q \Diamond sub_p(v,t)$.

Usando um raciocínio análogo ao da demonstração do Teorema 3, $c(q') < c(p)$. Se q' for um caminho fica demonstrado o pretendido; senão q' tem nós repetidos. Mas então é possível retirar os ciclos de q' obtendo-se um caminho, q", tal que, sendo por hipótese $c(C) \geq 0$ para todo o ciclo C de (N,A), o seu custo é não superior ao de q', isto é, $c(q") \leq c(q') < c(p)$; portanto p não seria o caminho mais curto. ♦

A implicação contrária pode ou não ser verdadeira. De facto, na rede representada na Figura 8(a) o ciclo C = ⟨4,3,2,4⟩ tem custo -2. Considerando s = 1 e t = 4 o caminho mais curto é ⟨1,2,4⟩ que contém o subcaminho ⟨1,2⟩ de custo 0. No entanto, ⟨1,4,3,2⟩ é também caminho de 1 para 2 e tem custo menor que o anterior.

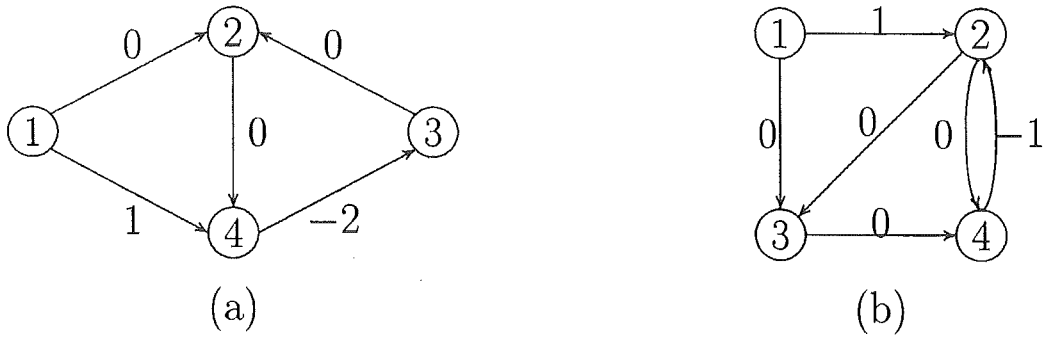


Figura 8 - Redes

Usando agora a rede da Figura 8(b), também com um ciclo negativo C = ⟨2,4,2⟩, o caminho mais curto de 1 para 4 é ⟨1,3,4⟩, constituído apenas por subcaminhos óptimos.

Assim, uma vez que estamos a considerar somente redes sem ciclos negativos, os algoritmos de rotulação referidos para trajectos podem ser utilizados também para a determinação do caminho mais curto.

Para K > 1 diz-se que o Problema dos K Caminhos mais Curtos verifica o Princípio de Optimalidade quando todo o k-ésimo caminho mais curto de s para t em (N,A), $k \in \{1,\ldots, K\}$, é constituído por j-ésimos subcaminhos mais curtos, com $j \in \{1,\ldots, k\}$.

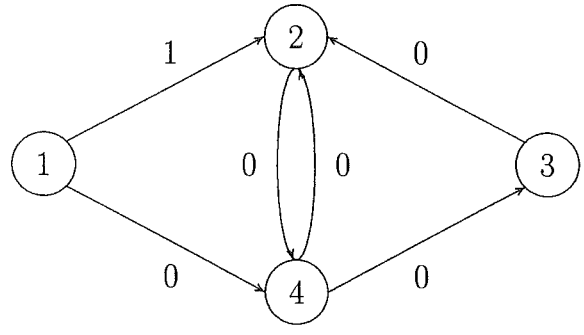


Figura 9 - Rede

Como referimos atrás, sendo K > 1, este prinípio pode não ser satisfeito. Um exemplo disto pode ser obtido atendendo a que o caminho mais curto de s = 1 para t = 4 na rede (N,A) da Figura 9 é $p_1 = \langle 1,4 \rangle$, enquanto que o segundo é $p_2 = \langle 1,2,4 \rangle$. No entanto existem em (N,A) três caminhos de 1 para 2, nomeadamente e por ordem não decrescente de custos, $t_1 = \langle 1,4,2 \rangle$, $t_2 = \langle 1,4,3,2 \rangle$ e $t_3 = \langle 1,2 \rangle$. Daqui concluímos que o terceiro caminho mais curto de 1 para 2 é subcaminho de p_2, ou seja, este problema não verifica o Princípio de Optimalidade.

5.2 Árvore dos K caminhos mais curtos

Sendo um caminho também um trajecto, o conjunto dos caminhos entre s e t em (N,A) é um subconjunto de P. Assim, K caminhos mais curtos entre dois nós de uma rede formam uma árvore análoga à do problema anterior.

Consideremos novamente a rede representada na Figura 2. A árvore dos seis caminhos mais curtos de 1 para 5 em (N,A) encontra-se representada na Figura 10.

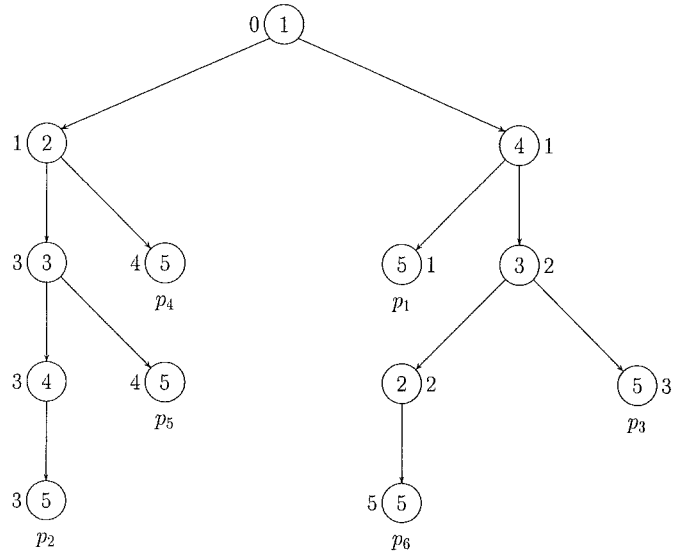


Figura 10 - Árvore de caminhos de 1 para 5

Deste modo, é possível utilizar esta estrutura para a obtenção de algoritmos que enumeram os K caminhos mais curtos entre dois nós de uma rede.

5.3 Adaptação de algoritmos baseados na construção da árvore

Analogamente ao que foi feito para o Problema dos K Trajectos mais Curtos, o nosso objectivo é desenvolver algoritmos que construam uma árvore que contenha os K caminhos mais curtos entre dois nós, s e t, de (N,A).

Um algoritmo conhecido para a resolução do Problema dos K Caminhos mais Curtos, e que podemos considerar ser baseado na determinação da árvore dos caminhos mais curtos, é o de Yen, [14]. De facto, sendo X um conjunto de candidatos a k-ésimo caminho mais curto, no passo k é escolhido o caminho mais curto em X, p_k. A partir deste são analisados os nós

seguintes ou coincidentes com o seu nó desvio e são gerados novos caminhos que são desvios de p_k, guardados em seguida no conjunto X. Uma vez que se pretendem determinar somente caminhos, ao analisar cada nó u de p_k são apagados de (N,A) todos os nós e arcos que não podem ser repetidos, determinando-se posteriormente o caminho mais curto de u para t, p, na rede alterada. Deste modo, o novo trajecto será $sub_k(s,u)\Diamond p$ que, pela forma como é calculado, é também um caminho.

Dois inconvenientes deste algoritmo são desde logo a necessidade, em cada um dos seus passos, de:

- alterar a rede (N,A), alterações essas a repor posteriormente;
- resolver um Problema do Caminho mais Curto na rede alterada.

É de notar que o Algoritmo 1 pode ser visto como uma generalização do algoritmo de Yen, onde é admitida a determinação de trajectos com e sem ciclos.

Tentaremos em seguida adaptar o algoritmo MPS, apresentado na subsecção 4.2.3 para a enumeração de trajectos, com vista à resolução deste problema.

5.3.1 Adaptação do algoritmo MPS

Como vimos, neste algoritmo começamos por determinar T_t, a árvore dos trajectos mais curtos de todos os nós para t, trajectos estes que são também caminhos, como foi referido na subsecção 3.2. Deste modo determinamos $p_1 = T_t(s)$, caminho mais curto de s para t em (N,A).

Ao gerar candidatos a segundo caminho mais curto obtemos, de acordo com o mesmo algoritmo, trajectos da forma $sub_1(s,u)\Diamond\langle u,v\rangle\Diamond T_t(v)$. Apesar de quer $sub_1(s,u)$ quer $\langle u,v\rangle$ e $T_t(v)$ serem caminhos, facilmente se verifica que a sua concatenação pode não o ser, isto é, pode conter ciclos. Para isso basta notar que na rede representada na Figura 9, quer $\langle 1,2,4\rangle$ quer $\langle 4,2,3\rangle$ são caminhos e, no entanto, a sua concatenação, $\langle 1,2,4,2,3\rangle$, contém o ciclo $\langle 2,4,2\rangle$, pelo que não é caminho.

Assim, em cada passo do algoritmo ao escolher, p', o trajecto com menor custo reduzido no conjunto de candidatos a k-ésimos caminhos mais curtos, para $k\in\{2,\ldots, K\}$, este pode não ser um caminho. No entanto, ainda que p' contenha ciclos, deverá ser analisado uma vez que a partir dele podem eventualmente vir a ser gerados outros caminhos. Deste modo analisam-se os nós de p' seguintes ou coincidentes com o respectivo nó desvio, $v'_{\alpha'}$, mas anteriores ao primeiro nó que forma um ciclo (caso contrário seriam gerados trajectos que saberíamos à partida conterem ciclos). Podemos assim reduzir o número de trajectos com ciclos determinados pela adaptação do algoritmo MPS. Se o último nó analisado for o terminal, então podemos concluir que p' não contém ciclos, sendo portanto o k-ésimo caminho mais curto de s para t em (N,A), para algum $k\in\{1,\ldots, K\}$.

Além disso, ao analisar um nó u de p' escolhendo para v um nó de $sub_{p'}(s,u)$, sabemos que obteremos um trajecto com ciclo pelo que vamos evitar fazê-lo. Ou seja, o nó v escolhido será

tal que (u,v) é o arco que se segue ao arco com início em u de p' na rede (N,A) representada na sorted forward star form e tal que $v \notin sub_{p'}(s,u)$.

Os passos principais desta adaptação encontram-se resumidos no Algoritmo 3.

Com um raciocínio idêntico poderíamos adaptar outros algoritmos para determinação dos K trajectos mais curtos que se baseiem na construção de uma árvore que os contém, nomeadamente o Algoritmo 1 ou o algoritmo de Hoffman, [7].

Algoritmo 3: Adaptação do algoritmo de Martins, Pascoal e Santos para a enumeração de caminhos

Determinar T_t
Para (todo o $(i,j) \in A$) **Fazer** $\bar{c}_{ij} \leftarrow \pi_j^t - \pi_i^t + c_{ij}$
Colocar (N,A) na sorted forward star form
$k \leftarrow 0$
$X \leftarrow \{T_t(s)\}$
Enquanto ($(k < K)$ e $(X \neq \phi)$) **Fazer**
    $p' \leftarrow$ trajecto com menor custo reduzido em X
    $\alpha' \leftarrow$ ordem do nó desvio, $v'_{\alpha'}$, de p'
    $X \leftarrow X - \{p'\}$
    Para (todo o $u \in sub_{p'}(v'_{\alpha'},t)$ tal que $sub_{p'}(s,u)$ é um caminho) **Fazer**
        $v \leftarrow$ nó tal que (u,v) é o arco que se segue ao arco com início em u de p' e $v \notin sub_{p'}(s,u)$
        Se (v está definido) **Então**
            $p \leftarrow sub_{p'}(s,u) \Diamond \langle u,v \rangle \Diamond T_t(v)$
            $X \leftarrow X \cup \{p\}$
        FimSe
        Se (u = t) **Então**
            $k \leftarrow k + 1$
            $p_k \leftarrow p'$
        FimSe
    FimPara
FimEnquanto

6. Complexidade e Experiência Computacional

De entre os algoritmos referidos para o Problema dos K Trajectos mais Curtos, aquele que apresenta a melhor complexidade, no pior dos casos, é o de Eppstein, com O(m+nlogn+KlogK), [11]. Relativamene aos novos algoritmos apresentados, podemos afirmar que, no pior dos casos, o Algoritmo 1 tem uma complexidade de O(Km) (ver [11]), enquanto que o Algoritmo 2 tem complexidade de O(Kn+m) (ver também [11]). Relativamente ao algoritmo de Yen podemos afirmar que, no pior dos casos, são resolvidos Kn problemas do caminho mais curto.

Quanto à adaptação do algoritmo MPS para o Problema dos K Caminhos mais Curtos, uma vez que em geral não conhecemos, à partida, um limite superior para o número de trajectos que é necessário determinar neste algoritmo, não foi possível estabelecer uma ordem de complexidade polinomial. No entanto, testes computacionais efectuados têm revelado um bom comportamento deste algoritmo na resolução de problemas.

Alguns dos algoritmos, quer para trajectos quer para caminhos, foram testados computacionalmente. Os resultados podem ver-se em [9,10,11]. Dos testes efectuados concluiu-se que o algoritmo MS parece ser o mais eficiente para a enumeração de trajectos, enquanto que o algoritmo MPS se mostrou bastante eficiente relativamente à enumeração de caminhos. Um estudo computacional mais completo será assunto de um futuro trabalho.

A autora pertence à Linha Ciências da Computação do Centro de Informática e Sistemas da Universidade de Coimbra, parcialmente subsidiado pelo Ministério das Ciências e Tecnologia através do Projecto PRAXIS XXI da Junta Nacional de Investigação Científica e Tecnológica.

Ao meu orientador, Prof. Doutor Ernesto de Queirós Vieira Martins, agradeço o incansável apoio e a colaboração prestada na elaboração da minha tese de mestrado bem como na elaboração deste artigo.

Ao revisor agradeço os úteis comentários e sugestões.

References

[1] Azevedo, J.A, Costa, M.E.O.S, Madeira, J.J.E.R.S. e Martins, E.Q.V., An algorithm for the ranking of shortest paths, European Journal of Operational Research 69 (1993) 97-106.

[2] Azevedo, J.A., Madeira, J.J.E.R.S., Martins, E.Q.V. e Pires, F.M.A., A shortest paths ranking algorithm, Proceedings of the Annual Conference AIRO'90, Models and Methods for Decision Support, Operational Research Society of Italy (1990) 1001-1011.

[3] Azevedo, J.A., Madeira, J.J.E.R.S., Martins, E.Q.V. e Pires, F.M.A., A computational improvement for a shortest paths ranking algorithm, European Journal of Operational Research 73 (1994) 188-191.

[4] Cherkassky, B.V., Goldberg, A.V. e Radzik, T., Shortest paths algorithms: Theory and experimental evaluation, Mathematical Programming 73 (1996) 129-196.

[5] Dreyfus, S.E., An appraisal of some shortest-path algorithms, Operations Research 17 (1969) 395-412.

[6] Eppstein, D., Finding the k shortest paths, SIAM Journal on Computing (aceite para publicação).

[7] Hoffman, R. e Pavley, R.R., A method for the solution of the n th best path problem, Journal of the Association for Computing Machinery 6 (1959) 506-514.

[8] Martins, E.Q.V., Pascoal, M.M.B., Rasteiro, D.M.L.D. e Santos, J.L.E., The optimal path problem, Investigação Operacional 19 (1999) 43-60.

[9] Martins, E.Q.V., Pascoal, M.M.B. e Santos, J.L.E., A new algorithm for ranking loopless paths, Research Report, CISUC (1997) submetido.

[10] Martins, E.Q.V. e Santos, J.L.E., A new shortest paths ranking algorithm, Investigação Operacional (aceite para publicação).

[11] Pascoal, M.M.B., Algoritmos para a enumeração dos k trajectos mais curtos, Dissertação de Mestrado, Departamento de Matemática, Universidade de Coimbra (1997).

[12] Shier, D., Interactive methods for determining the k shortest paths in a network, Networks 6 (1976) 151-159.

[13] Shier, D., On algorithms for finding the k shortest paths in a network, Networks 9 (1979) 195-214.

[14] Yen, J.Y., Finding the k shortest loopless paths in a network, Management Science 17 (1971) 712-716.

MÉTODOS EFICIENTES PARA OPTIMIZAÇÃO DO DIMENSIONAMENTO NÃO-LIMITADO DE LOTES A PRODUTO ÚNICO

João Luís de Miranda

Escola Superior de Tecnologia e Gestão
Instituto Politécnico de Portalegre
Lugar da Abadessa
7301 Portalegre Codex
Portugal

Miguel Casquilho

Instituto Superior Técnico (UTL)
Av. Rovisco Pais
1049-001 Lisboa
Portugal

Abstract

Several resolution methods for the "economic lot sizing", as presented by Wagner and Whitin [15], are described and compared. Their applicability supports discontinuous production processes, frequently found for instance in the chemical industry and useful in other areas. The underlying calculation efficiency, which is made clear, results from the mathematical structure of the problem and is characterized by its speed, although this feature is not readily generalizable. Therefore, this descriptive analysis focuses a problem that is able to become a building element for other, more complex problems, which keeps granting it intense research activity.

Resumo

Perspectivam-se diferentes formas de resolução de um problema que mantém actualidade, o dimensionamento óptimo de lotes ("economic lot sizing"), tal como desenvolvido por Wagner e Whitin [15], procedendo-se também a uma sua avaliação comparativa. A aplicabilidade directa desta ferramenta é evidente quanto aos processos de produção descontínuos, com abundantes concretizações, por exemplo na indústria química, mas a sua utilidade ultrapassa aquele contexto. Evidencia-se a eficiência de resolução, decorrente das propriedades da estrutura matemática do problema e caracterizada por grande rapidez de cálculo, mas esta qualidade revela-se insusceptível de uma generalização directa. Daí, esta análise descritiva focalizar um problema que persiste atractivo para se constituit como elemento componente de outros problemas, mais complexos, e mantendo-se, como tal, sujeiro a inúmera investigação.

Keywords

Economic lot sizing, Mixed Integer Linear Programming, reformulations, dual ascend method, dynamic programming.

O interesse pela estrutura matemática do problema clássico de dimensionamento de lotes deriva do facto de, devido a serem pouco numerosas as soluções que admite [15], o algoritmo de resolução utilizado poder atingir o ponto óptimo com notável eficiência. Neste problema, observa-se que as variáveis binárias de decisão existentes condicionam de tal maneiras as variáveis contínuas de produção que estas assumem, de forma exclusiva,

- um valor nulo; ou
- um valor (dum conjunto discreto) correspondente ao somatório das procuras em sucessivos períodos

Complementarmente, as variáveis contínuas referentes às quantidades armazenadas também apresentam um conjunto limitado de valores discretos possíveis.

Esta característica evoca a Programação Linear, cuja resolução através do algoritmo do *simplex* de Dantzig se torna bastante eficiente, visto que este apenas calcula pontos extremos do espaço de procura (vértices) para encontrar o óptimo do problema (eventualmente associado a soluções múltiplas). A citada propriedade revela-se crucial, decisivamente vantajosa para a determinação da solução deste tipo de problemas de Programação Linear Mista, e deriva duma simples transformação - reformulação ou redefinição de variáveis.

Após a citada transformação matemática, a relaxação linear das restrições de integralidade nas variáveis decisórias resulta exacta e óptima (como se verá adiante), pois que, se as variáveis contínuas [de produção e posse (i. é, armazenagem, entre outros)] se situarem sobre um dos limites, inferior ou superior, precisamente o mesmo acontecerá às variáveis discretas relaxadas: situar-se-ão no seu limite inferior (zero) ou no seu limite superior (um).

Dado que a situação óptima assim obtida, mediante transformação, apresenta resultados precisamente sobre a fronteira dos valores admissíveis para as variáveis binárias relaxadas, o respectivo problema dual definirá, necessariamente, uma solução óptima com valor nulo nas variáveis de folga ("slack") das restrições duais correspondentes. Logo, o conhecimento dest facto possibilita uma estratégia de resolução através do problema dual, evitando-se o inconveniente de métodos ditos enumerativos, nos quais se inclui o algoritmo do branch-and-bound.

Considerar-se-á, ainda, a resolução através de Programação Dinâmica, como problema de rota mínima, dado que o problema foi primeiramente resolvido de forma eficiente por Wagner e Whitin utilizando uma via afim para o problema de "economic lot sizing" (dimensionamento óptimo de lotes de produção).

Para uma melhor apreciação das principais características diferenciadoras das várias estratégias de resolução do problema em análise, inclui-se um exemplo numérico simples, a título ilustrativo.

Enquadramento do problema

A peculiaridade do problema de "economic lot sizing" não cessa de suscitar o interesse dos investigadores, pois não só ele constitui o suporte para a construção de modelos cada vez mais complexos, como também tem sido profícua a investigação para melhorar quer a sua aplicabilidade a diferentes problemas, quer a sua eficácia em diferentes situações.

Van Hoesel e Wagelmans [14] publicaram um algoritmo para resolução do problema de "economic lot sizing", considerando custos de produção côncavos e limites definidos e

constantes para as capacidades utilizadas. Mantendo a forma linear para os custos de posse, os autores demonstram que o seu algoritmo resolve o descrito problema num tempo de $O(T^3)$.

Wolsey [16] apresenta uma resenha actualizada dos progressos efectuados no âmbito do problema de "economic lot sizing", como sejam a sua aplicação a problemas multiproduto e recentes avanços nos métodos utilizados para a sua resolução.

Sahinidis e Grossmann [12,13] reformulam os seus modelos de planeamento ("planning") e calendarização ("scheduling") aplicáveis a processos químicos e representáveis como problemas de Programação Linear Mista, reconhecendo substruturas matemáticas do problema de "economic lot sizing" como componentes da sua estrutura e reformulando-as a contento.

Karanicolas [4], na análise da localização multiperíodo de unidades de produção, com ou sem restrições de capacidade, desenvolve um algoritmo que, através duma relaxação lagrangeana, procede a uma decomposição do problema inicial em múltiplos problemas independentes de ELS, que dependem dum problema mestre de Programação Linear Mista.

Gopalakrishnan et al. [2] consideram o problema de "economic lot sizing" capacitado, bem como a produção de múltiplos produtos associada aos respectivos custos de preparação ou mudança ("setup carryover"). É utilizado um método de decomposição que procura tirar partido da resolução eficiente característica do "economic lot sizing", para assim resolver até à optimalidade um problema de elevada complexidade.

Gunasekaran et al. [3] utilizam a estrutura matemática analisada neste trabalho para determinar os níveis óptimos dos parâmetros de qualidade e produtividade associados ao dimensionamento dos processos "batch", bem como para optimizar o investimento em operações de controlo de produção.

A formulação clássica

O modelo clássico de Economic Lot Size (ELS) visa minimizar o custo total associado à satisfação da procura ao longo do horizonte temporal, considerando custos fixos - de aprontamento - e custos variáveis, dependendo dos níveis de produção e posse. Os dados do problema, necessariamente conhecidos para cada um dos T períodos de tempo ($t = 1,\ldots, T$), serão:

p_t - custo (marginal) unitário de produção

h_t - custo (marginal) unitário de posse (hold)

f_t - custo fixo de aprontamento ("steup") de cada unidade de produção

d_t - quantidade de produto procurada (demand)

Os referidos custos correspondem, respectivamente, ás seguintes variáveis:

x_t - quantidade produzida

s_t - quantidade de produto em armazém (stock)

y_t - decisão binária quanto à produção (0, não; 1, sim)

No caso particular em que se considerem valores constantes para os diversos parâmetros do problema, este reduz-se ao problema da Quantidade Óptima de Encomenda ("Economic Order

Quantity", EOQ). As conhecidas "fórmulas da raiz quadrada", para estimativa da quantidade, $Q^* = \sqrt{2d\ f/h}$, ou tempo, $t^* = \sqrt{2f/dh}$, óptimos, serão válidas apenas sob essa presunção. Duma forma mais geral e logo que os dados do problema variem - por exemplo, por actualização financeira dos diversos custos -, teremos, obviamente, de recorrer a abordagens diferentes. Nomeadamente, através da Programação Linear Mista, este problema de "economic lot sizing" pode formular-se num modelo como o que designámos por ELS.

Modelo ELS:

$$[\min]\ z = \sum_{t=1}^{T} (p_t x_t + h_t s_t + f_t y_t) \qquad (1)$$

sujeito a, para t = 1..T,

$$s_{t-1} + x_t = d_t + s_t \qquad (1a)$$

$$x_t \leq W_t y_t \qquad (1b)$$

$$W_t = \sum_{i=t}^{T} d_i \qquad (1c)$$

$$s_0 = s_T = 0 \qquad (1d)$$

$$x_t,\ s_t \geq 0 \qquad y_t \in \{0;\ 1\} \qquad (1e,f)$$

O probema apresenta uma função objectivo côncava, porquanto, para cada período o custo será:

(i) linear, quando apenas se incorrer em custos de armazenagem;

(ii) linear acrescido de uma parte fixa, quando considerarmos também a produção.

Consequentemente, obtém-se, segundo Nemhause [8], uma solução típica do problema de "minimum spanning tree", com dois aspectos característicos: no ponto óptimo, sempre que há produção, produz-se em quantidade suficiente para satisfazer a procura num número inteiro de períodos contíguos subsequentes; e em algum período produzir-se-á apenas parcialmente, complementando-se a quantidade procurada com o produto existente em armazém. Assim, é possível verificar que a política de custo mínimo conduz a soluções características, em que a produção, x_t (em cada período, t = 1..T), corresponderá a um dos elementos do conjunto X_t (com T - t + 2 elementos), conforme Eq. (2).

$$X_t = \left\{0;\ d_t;\ d_t+d_{t+1};\ldots;\sum_{i=t}^{T} d_i\right\} = \left\{0;\ \sum_{i=t}^{t'} d_i,\ t' = t..T\right\} \qquad (2)$$

São apresentados, para as diferentes formas de resolução equacionadas, os resultados de um caso ilustrativo para o qual, na Tabela 1, se listam os dados necessários

Período, t	C.to de produção, p_t	C.to de armazenagem, h_t	C.to fixo, f_t	Quant.procurada, d_t
1	12	3	72	41
2	10	5	95	23
3	15	2	61	35
4	11	2	84	30
5	11	3	50	21

Tabela 1 - Dados do caso ilustrativo, com 5 períodos de tempo

Nomeadamente, para o modelo ELS, consubstanciam-se na Figura 1 os resultados correspondentes.

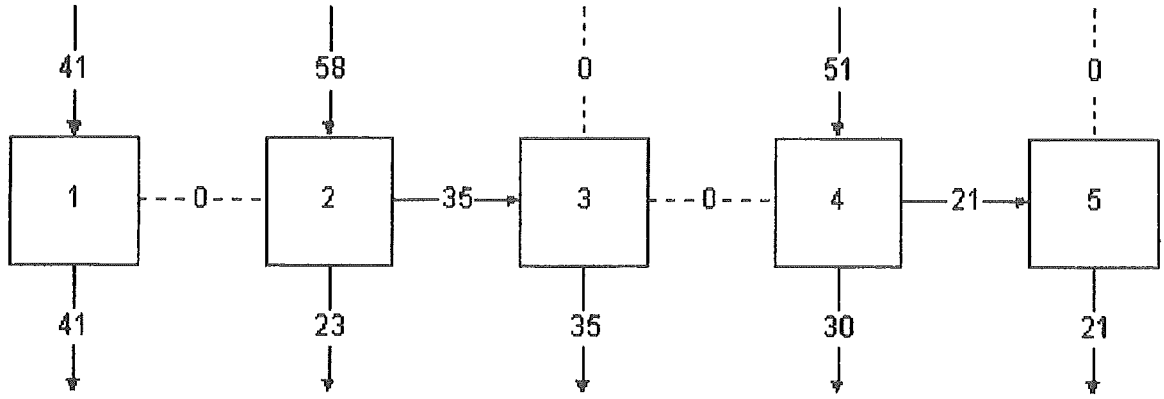


Figura 1 - Diagrama da solução do caso ilustrativo segundo o modelo ELS

Desagregação de variáveis

Uma altermativa possível à apresentação clássica anteriormente descrita é a de formular o problema de maneira a desagregar cada variável de produção segundo o período a que é destinada [5], ou seja, definindo-se umanova variável, $q_{t\tau}$, que corresponde à quantidade produzida no período t destinada a ser consumida no período τ.

Assim, o modelo *Desagregado*, composto pelas Eqs. (3), constitui uma diferente perspectiva do problema de "economic lot sizing", forma esta que revelará também características próprias: o resultado determinado a partir da respectiva relaxação linear corresponde à solução inteira óptima. Logo, caso se utilize o algoritmo de branch-and-bound, apenas se necessitará de calcular o nó inicial, pois teremos obtido imediatamente a solução do problema.

Modelo Desagregado:

$$[\min]\ z = \sum_{t=1}^{T} \left[\sum_{t=1}^{T} \left(p_i + \sum_{t=1}^{T} h_{t'} \right) q_{it} + p_t q_{tt} \right] + \sum_{t=1}^{T} f_t y_t \qquad (3)$$

sujeito a, pata t = 1..T:

$$\sum_{i=1}^{T} q_{it} = d_t \qquad (3a)$$

$$q_{it} \leq d_t y_t \qquad (3b)$$

$$q_{it} \geq 0 \qquad y_t \in \{0; 1\} \qquad (3c)$$

O significado (em termos económicos) dos parâmetros p_t, h_t, f_t, d_t e das variáveis y_t é idêntico ao referido no modelo ELS, ou seja, ambos os modelos utilizam o mesmo conjunto de dados e, correspondentemente, apresentam similares decisões. O resultado obtido através do modelo Desagregado ilustra-se na Figura 2.

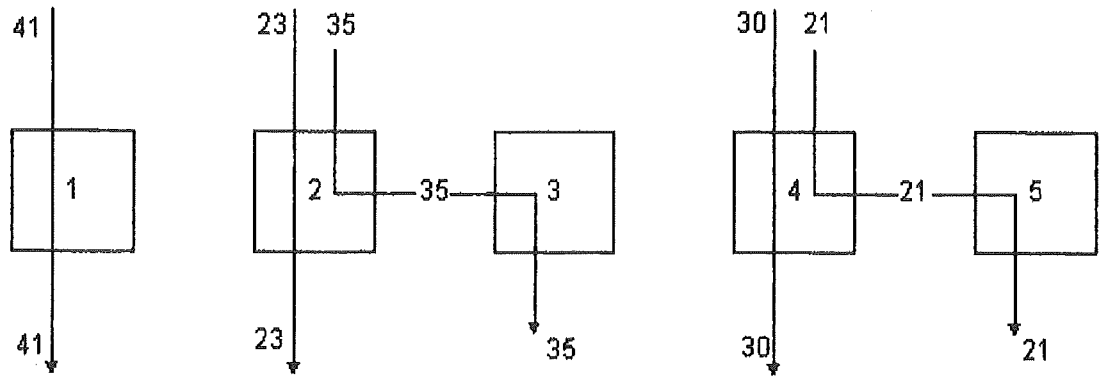


Figura 2 - Diagrama da solução do caso ilustrativo segundo o modelo Desagregado

A reformulação de Martin

Esta reformulação do modelo ELS é efectuada através da introdução de novas variáveis contínuas, $\lambda_{t\tau}$, que representam a produção num dado período t, de maneira a satisfazer cumulativamente a procura do produto até ao período τ. Esta nova perspectiva do problema em análise visa utilizar eficazmente o conhecimento das características da respectiva solução, pois verifica-se que a produção óptima corresponde a quantidades produzidas de forma a satisfazer integralmente a procura em períodos de tempo consecutivos.

A reformulação deste problema de "economic lot sizing", apresenta juntamente com outros tipos de reformulações por Martin [6], consiste essencialmente no modelo ELS acrescido das novas variáveis de produção cumulativa, $\lambda_{t\tau}$, associando-se também os constrangimentos respectivos.

Modelo Reformulado:

$$[\min]\ z = \sum_{t=1}^{T} (p_t x_t + h_t s_t + f_t y_t) \qquad (4)$$

sujeito a, pata t = 1..T,

$$s_{t-1} + x_t = d_t + s_t \qquad (4a)$$

$$x_t \leq C_{tT} y_t \tag{4b}$$

$$\lambda_{t\tau} \leq C_{t\tau} y_t \tag{4c}$$

$$C_{t\tau} = \sum_{i=t}^{\tau} d_i \tag{4d}$$

$$x_t \geq \lambda_{t\tau} \tag{4e}$$

$$\sum_{\tau=1}^{t} \lambda_{\tau t} \geq C_{1t} \tag{4f}$$

$$s_0 = s_T = 0 \tag{4g}$$

$$x_t,\ s_t,\ \lambda_{t\tau} \geq 0 \qquad y_t \in \{0;\ 1\} \tag{4h,i}$$

O modelo *Reformulado* apresenta a propriedade de relaxação linear inteira óptima, similarmente ao modelo *Desagregado*. Ou seja: quando se utiliza o algoritmo de branch-and-bound na resolução do problema de Programação Linear Mista que o modelo *Reformulado* constitui, observa-se uma optimização imediata, logo ao primeiro nó de procura, de igual forma ao que sucede para o modelo *Desagregado*.

Esta propriedade típica apenas se afigura possível devido à existência de uma situação de equivalência entre o conjunto das soluções possíveis de ambos os referidos modelos com o modelo original (ELS).

A Figura 3 mostra a solução do caso ilustrativo pelo modelo Reformulado.

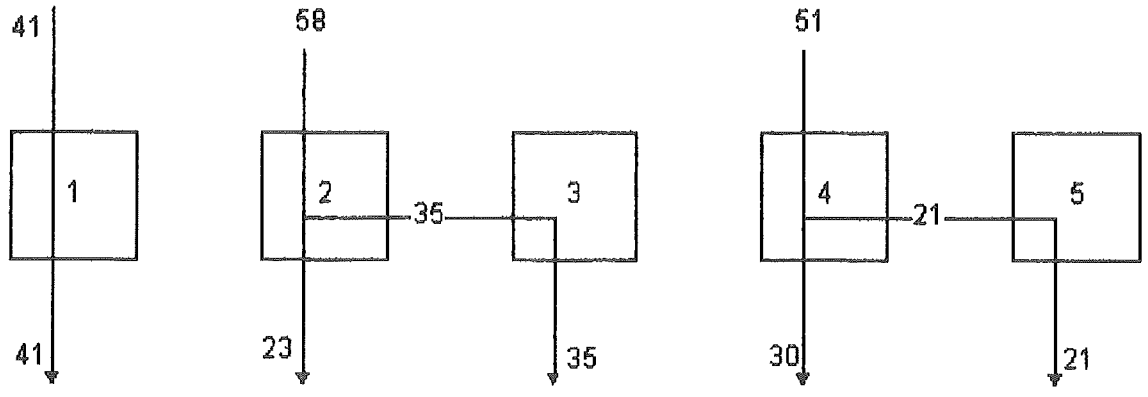


Figura 3 - Diagrama da solução do caso ilustrativo segundo o modelo Reformulado

Um método ascendente dual

Nas formulações referentes aos modelos *Desagregado* e *Reformulado*, consegue-se a optimalidade da relaxação linear de ambas as reformulações do mesmo problema ELS de Programação Linear Mista, com obtenção de valores inteiros (zero ou um) para as variáveis de decisão relaxadas, y_t. De forma sintética, verifica-se que, relaxando os constrangimentos de

integralidade destas variáveis, mediante substituição do *conjunto* de valores {0; 1} pelo *intervalo* [0; 1], o valor óptimo das variáveis em análise se situará precisamente num dos pontos extremos. Consequentemente, as variáveis de folga ("slack") das restrições do problema dual, correspondentes às variáveis de decisão binária do problema primal, apresentarão valores nulos, o que representa uma das principais características da solução da relaxação linear do problema de Programação Linear Mista.

Atendendo a esta especificidade, Erlenkotter [1] e, independentemente, Krarup e Bilde [5], desenvolveram um procedimento dual para a resolução do modelo *Desagregado*, equivalente ao problema de "economic lot sizing" em questão.

A formulação dual obtém-se da correspondente formulação primal, tendo agora como seu objectivo o de maximizar uma função soma constituída por parcelas cujos coeficientes são os coeficientes independentes, d_t, do problema primal. A matriz de coeficientes das restrições obtém-se da transposição da respectiva matriz do problema inicial e os termos independentes serão formados pelos coeficientes c_{it} da função objectivo primal:

$$c_{it} = p_i + h_i + h_{i+1} + \ldots + h_{t-1} = p_i + \sum_{t'=i}^{t-1} h_{t'} \qquad (5)$$

Assim, da dualização do modelo *Desagregado* obtem-se o modelo *Dual.*

Modelo Dual:

$$[\max]\ z = \sum_{k=1}^{T} u_k d_k \qquad (6)$$

sujeito a, pata t = 1..T e k = 1..T

$$\sum_{k=t}^{T} d_k . w_{tk} \leq f_t \qquad (6a)$$

$$u_k - w_{tk} \leq c_{tk} \qquad (6b)$$

$$w_{tk} \geq 0 \qquad u_k \geq 0 \qquad (6c,d)$$

Este modelo (dual) possibilita um método de resolução bastante eficiente, pois os valores duais, calculados a título definitivo num dado período de tempo, serão utilizados para obtenção dos valores correspondentes ao período seguinte, e assim ascendentemente até ao período final (procedimento dual). De seguida, aplica-se um método de leitura dos valores assim determinados (procedimento primal), retrocedendo nos tempos, de forma converter aqueles valores duais nos respectivos valores primais.

Algoritmo ascendente dual:

Procedimento Dual

1) Inicializa, fazendo

$$z_{dual} = 0$$

$$k = 1$$

$$s_{tt} = f_t \qquad t = 1..T$$

2) Calcular, sequencialmente, para k = 1..T e t = 1..k

$$u_k = \min_t \left(c_{tk} + \frac{s_{tk}}{d_k} \right)$$

$$w_{tk} = \max(0; u_k - c_{tk})$$

$$s_{t,k+1} = s_{tk} - d_k w_{tk}$$

$$z_{dual} = z_{dual} + d_k u_k$$

3) Se k < t, então fazer k := k+1 e voltar ao passo 2); senão ir para o passo 4).

Procedimento Primal

4) Inicializa, colocando

$$t = T$$

$$j = T + 1$$

$$x = 0 \qquad y = 0$$

5) Se $s_{tj} = 0$, então

$$y_t = 1$$

$$x_{tk} = d_k, \qquad k = t..j - 1$$

$$j = t$$

6) Se t > 1, então fazer t := t - 1 e voltar ao passo 5); senão, terminar procedimento.

Similares procedimentos têm sido aplicados exaustivamente por Rajagopalan [9,10,11], respectivamente, em problemas de expansão de capacidades sujeitos a determinados condicionamentos, como a deterioração de equipamentos, a escolha de tecnologias, ou supondo expansões discretas de capacidades, tal como referido na bibliografia.

Programação Dinâmica por rota mínima

Como já referido, o problema de determinar a dimensão óptima de lotes foi desenvolvido por Wagner e Whitin [15], através da Programação Dinâmica, por desenvolvimento duma formulação de *fluxos em rede*. Assim, uma formulação equivalente à de Programação Linear Mista para o problema de "economic lot sizing" será a de o considerar como um problema de *rota mínima* ("shortest route"), onde:

Nós - representam a produção acumulada até cada um dos períodos de tempo;

Arcos - representam as decisões, às quais se associa o respectivo custo.

O conjunto dos nós e arcos considerados corresponde a todos os percursos possíveis, que constituem um conjunto necessariamente limitado, pois as variáveis contínuas (quantidades produzidas e quantidades armazenadas) apenas poderão assumir um número limitado de valores, como se observou anteriormente.

As principais condições a observar na reformulação do problema de Programação Linear Mista (que o ELS constitui) como um problema de definição de percurso a custo mínimo envolvem a precaução de que não sejam nem criadas novas soluções nem negligenciada qualquer solução do problema original. Por conseguinte, o método de resolução adoptado envolve três etapas diferentes - correspondendo as duas primeiras à construção do espaço de procura (nós e arestas), enquanto a terceira determinará o percurso de custo mínimo:

i) Definição de todo e qualquer nó associado às diferentes hipóteses "óptimas" de produção em cada período, cujos valores se resumem a valores acumulados da procura de produto ao longo do tempo, como se verificou oportunamente;

ii) Determinação dos custos associados a cada um dos possíeis arcos, a partir dos valores unitários dos custos variáveis e das quantidades produzidas e armazenadas em cada período; dever-se-á também associar o respectivo custo fixo sempre que se verifique efectiva produção num dado período de tempo.

iii) Determinação do percurso que incorre no mínimo custo global ("rota mínima"), utilizando o algoritmo de Dijkstra, por exemplo.

Algumas características passíveis de implementação directa através deste método de resolução, no sentido de uma aproximação maior à realidade, traduzem-se num aumento da complexidade do problema, como por exemplo:

- Imposição de limites de capacidades às diferentes unidades, sejam de produção ou armazenagem;
- Poderem, em cada unidade de produção, ser fabricados diversos produtos, os quais estarão interligados por custos comuns de produção (operação, manutenção), custos de mudança ("changeover"), considerando-se também a eventualidade de compartilharem recursos limitados.

Graficamente, para o exemplo ilustrativo adoptado, obtém-se o "percurso" de rota mínima da Figura 4.

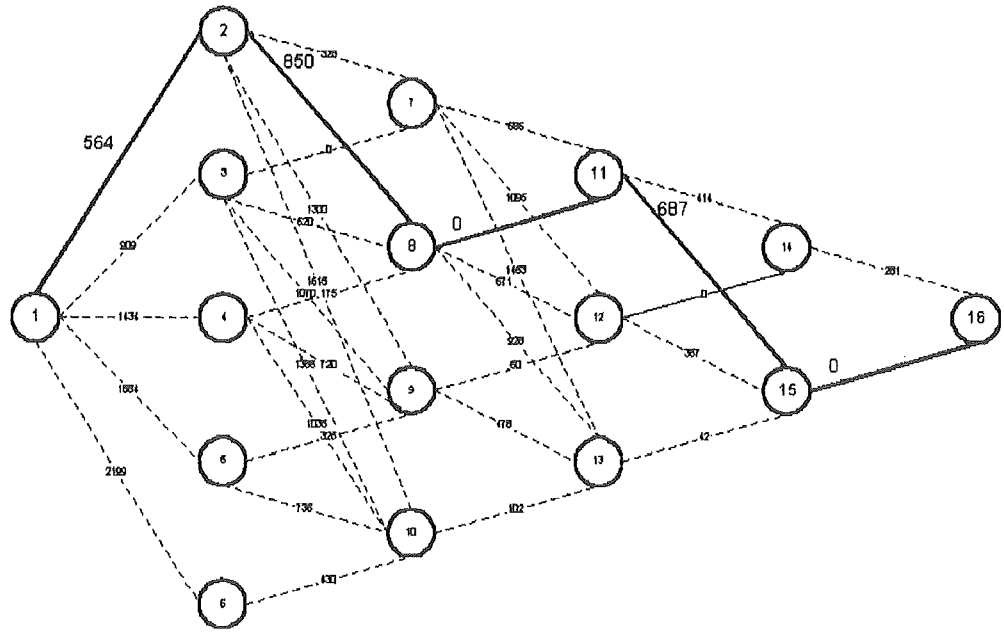


Figura 4 - Grafo produções-tempo do caso ilustrativo

Comparação dos métodos de resolução

Os elementos característicos, relativamente a cada método de resolução, apresentam-se condensados na Tabela 2, permitindo assim uma visão englobante.

Modelo/Modelo	Nº de variáveis	Nº de restrições
ELS	3 T	2 T
Desagregado	T(T+3)/2	T(T+3)/2
Reformulado	T(T+7)/2	T(T+4)
Dual	T(T+3)/2	T(T+3)/2
Rota Mínima	$1 + \frac{T(T+1)}{2}$ (nós)	$\frac{T(T+1)(T+2)}{6}$ (arcos)

Tabela 2 - Elementos caracterizadores dos modelos e métodos utilizados na resolução do problema de "economic lot sizing"

Através da resolução de um número elevado de problemas, com dimensões variadas (pois se equacionaram diversos horizontes de planeamento), possibilitou-se a comparação dos correspondentes tempos de resolução, número de ietarções e espaço de memória necessários. Os resultados foram obtidos num computador DEC 7620, em ambiente Open VMS, e sintetizam-se, quanto a tempos de resolução e a espaço de memória, respectivamente, nos gráficos da Figura 5 e da Figura 6.

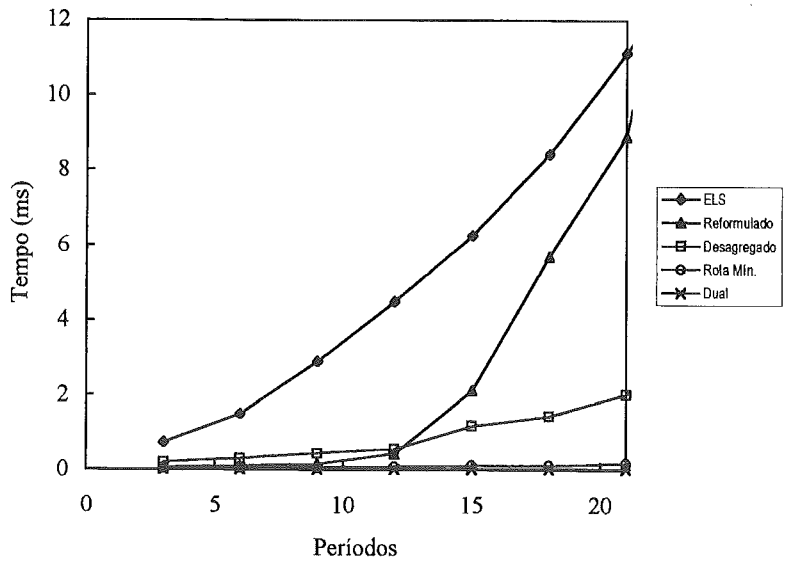


Figura 5 - Tempo de cálculo (CPU) resultante dos vários modelos, em função do número de períodos considerado

Os modelos ELS, Reformulado e Desagregado foram resolvidos através duma rotina da biblioteca matemática NAG (1991), direccionada para resolução de problemas de Programação Linear Mista, a rotina H02BBF (Mark 14, 1991), que utiliza um método "branch-and-bound" eral, ou seja, sem recurso a técnicas de pré-processamento, nem sendo especificamente para os modelos em análise.

Nos modelos ELS, mesmo quando considerámos problemas de igual dimensão em número de períodos, o tempo de resolução mostrou-se irregular (desde alguns segundos até vários minutos) ou mesmo limitante (não-resolução por excesso de tempo), tal como irregular se mostrou o número de iterações. Por conseguinte, observou-se uma notória sensibilidade aos dados do problema, sendo o tempo de reoslução típico o mais alto dos métodos estudados. A dimensão da memória exigível é relativamente grande, dado o espaço de trabalho necessário ao funcionamento da rotina citada.

O modelo Desagregado necessita de muito maior espaço de memória, a qual aumenta com o quadrado do número de variáveis (e este com o quadrado do número de períodos). No entanto, o cálculo processa-se rapidamente, já que apenas se pesquisa o primeiro nó da árvore do branch-and-bound, devido à optimalidade da relaxação linear característica deste modelo.

O modelo Reformulado, similarmente ao anterior, também resolve ao nó inicial da árvore, mas exige o maior espaço de memória, pois apresenta o maior número de variáveis e de restrições (seneivelmente o dobro das do Modelo Desagregado). O tempo gasto é ligeiramente maior que o deste último, mas, comparativamente, muito inferior ao do ELS. Estes métodos, Reformulado e Desagregado, resolvem um só problema de Programação Linear (o do "primeiro nó"), pelo que apenas se utiliza uma pequena parte do espaço de memória automaticamente reservado.

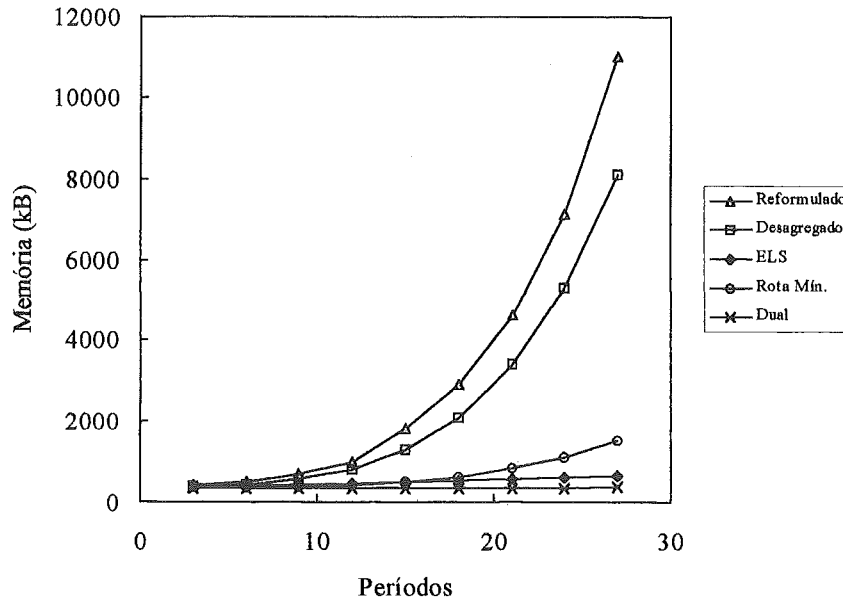


Figura 6 - Memória ocupada pelos vários modelos, em função do número de períodos considerado

O algoritmo ascendente dual, correspondente à formulação dual complementar ao Modelo Desagregado do problema "economic lot sizing" em estudo, foi desenvolvido em Froten 77, conforme já descrito. Este método de natureza dual - não-iterativo e exacto - revela-se bastante rápido, mesmo para problemas com horizonte temporal da ordem das centenas, e necessita de espaço de memória apenas para as variáveis do problema. É, pois, pouco exigente em termos computacionais e de fácil implementação.

Finalmente, também em Fortran 77, desenvolveu-se um método de resolução através da Programação Dinâmica, que incluiu o algoritmo de Dijkstra para definir a rota mínima. Este método, não tão rápido como o Dual dada a não-especificidade do algoritmo utilizado, também ocupa um espaço de memória relativamente diminuto e suplanta, nos aspectos rapidez e espaço de trabalho, os modelos que recorrem ao branch-and-bound.

Para cada um destes métodos alternativos, não só se utilizou um variado número de períodos do tempo, como se diversificou a relação de valor entre os diferentes custos: fixo, de produção e armazenagem. Sendo considerados 9 valores diferentes (varrimento logarítmico decimal da gama -4 a +4) para cada um dos 4 parâmetros do problemas, foram equacionados 9^4 (i. é, 6561) problemas diferentes, através de cada um dos métodos de resolução já descritos. Os modelos ou métodos aqui apresentados encontram-se com maior detalhe e pormenorizadamente discriminados em Miranda [7].

Dada a elevada eficiência observada nos métodos exactos aqui analisados, não se achou necessário rever outros métodos, como os heurísticos, obviamente baseados em cálculos aproximados e afectados, nomeadamente, pelo número crescente de períodos do tempo a considerar.

Conclusões

A partir de vários métodos de resolução do problema de "economic lot sizing", perspectivam-se os níveis de eficiência e robustez daqueles métodos através de uma análise comparativa, definindo-se mais rigorosamente as suas características.

Analisam-se as várias formulações-tipo do modelo, com números próprios de variáveis discretas e contínuas, bem como de constrangimentos, que lhes induzem características específicas, através da apresentação de um caso numérico ilustrativo. Desta maneira, complementa-se a apresentação das diferentes formulações e métodos de resolução aplicáveis ao problema de Programação Linear Mista em estudo, com uma análise comparativa entre si, permitindo uma melhor compreensão dos pontos cruciais para a sua resolução numérica eficiente.

Também, ficou bem patente a importância que uma simples reformulação ou redefinição das variáveis do problema poderá ter na melhoria do cálculo de optimização, especialmente no caso de problemas de Programação Linear Mista - como este que se analisou -, cuja resolução apresenta regularmente um elevado grau de dificuldade.

Dos diferentes métodos de resolução equacionados, comprova-se a qualidade das diversas reformulações, porquanto, tomando como ponto de referência o problema de "economic lot sizing" modelizando através da Programação Linear Mista, verifica-se que todas as outras modelizações lhe são manifestamente superiores.

Em termos de desenvolvimento futuros, algumas generalizações passíveis de discussão associam-se como:

- Maximização de lucros como objectivo último;
- Custos, de produção ou posse, não proporcionais às quantidades;
- Procuras ou vendas como função do preço considerado.

Referências

[1] Erlenkotter, D., A dual based procedure for uncapacitated facility location, Operations Research 26 (1978) 992-1009.

[2] Gopalakrishnan, M., Miller, D.M. and Schmidt, C.P., A framework for modelling setup carryover in the capacitated lot sizing problem, Int. J. Prod. Res. 33 (1995) 1973-1988.

[3] Gunasekaran, A., Korukonda, A.R., Virtanen, I. and Yli-Olli, P., Optimal investment lot sizing policies for improved productivity and quality, Int. J. Prod. Res. 33 (1995) 261-278.

[4] Karanicolas, P.C., Multiperiod Plant Location Problems, Ph. D. Thesis, Syracuse University (1979).

[5] Krarup, J. and Bilde, O., Plant location, set covering and economic lot sizes: An O(mn) algorithm for structured problems, in: Collatz, L. (ed.), Optimierung bei Graphentheoretischen und Ganzzähligen Probleme (Birkhauser Verlag, Basel) (1977) 155-180.

[6] Martin, R.K., Generating Alternative Mixed-Integer Programming Models Using Variable Redefinition, Operations Researcg 35 (1987).

[7] Miranda, J.L., O Planeamento Industrial e o Dimensionamento Óptimo de Lotes, Tese de Mestrado, IST-UTL, Lisboa (1996).

[8] Nemhauser, G.L., Rinnoy Kan, A.H.G. and Todd, M.J. (ed.), Handbooks in Operations Research and Management Science, Optimization, (North-Holland, Amsterdam) (1989).

[9] Rajagopalan, S., Deterministic Capacity Expansion Under Deterioration, Management Science 38 (1992).

[10] Rajagopalan, S. Capacity Expansion with Alternative Technology Choices, European Journal of Operational Research 77 (1994a) 392-403.

[11] Rajagopalan, S. and Soteriou, A.C., Capacity Acquisition and Disposal with Discrete Facility Sizes, Management Science 40 (1994b).

[12] Sahinidis, N.V. and Grossmann, I.E., Reformulation of Multiperiod MILP Models for Planning and Scheduling of Chemical Processes, Computers & Chemical Engineering 15 (1991) 255-272.
[13] Sahinidis, N.V. and Grossmann, I.E., Reformulation of the Multiperiod MILP Model for Capacity Expansion of Chemical processes, Operations Research 40 (1992).
[14] Van Hoesel, C. and Wagelmans, A., A O(T3) Algorithm for the Economic Lot-sizing Problem with Constant Capacities, Management Science 42 (1996).
[15] Wagner, H.M. and Whitin, T.M., Dynamic version of the economic lot size model, Management Science 5 (1958) 89-96.
[16] Wolsey, L.A., Progress with single-item lot-sizing, European Journal of Operational Research 86 (1995) 395-401.

MÉTODOS PARA PROBLEMAS DE MÍNIMOS QUADRADOS NÃO LINEARES E SUA CONVERGÊNCIA

Gisela C.G.V. Ramadas

Departamento de Matemática
Instituto Superior de Engenharia do Porto
4200 Porto

Edite M.G.P. Fernandes

Departamento de Produção e Sistemas
Universidade do Minho
4710 Braga

Abstract

The nonlinear least squares problem is of practical importance specially in curve fitting and parameter estimation. This problem can be solved either by general optimization methods or by specialized algorithms which take into account its structure. If this is explored more efficient algorithms can be developed.

The purpose of this paper is to survey for methods to be used for the solution of nonlinear least squares problems and to study their local convergence rate.

The iterative solution of the normal equations defines the Newton's method. To save derivative and arithmetic calculations Gauss-Newton, Levenberg-Marquardt and Quasi-Newton structured algorithms have been proposed. In order to simplify the calculus, to improve the conditioning and to maintain good local convergence properties, as the Newton's method, a variety of implementations are available at present.

Resumo

O problema que consiste em minimizar uma soma de quadrados de funções não lineares tem grande importância prática, principalmente no campo do ajuste de curvas e da estimativa de parâmetros. Este problema é solucionável quer pelo uso de algoritmos genéricos de optimização, quer através de algoritmos especializados que têm com consideração a sua estrutura. Explorando esta estrutura especial resultam métodos mais eficientes.

O objectivo deste trabalho consiste em investigar os métodos disponíveis para a resolução do problema de mínimos quadrados não linear e estudar as suas propriedades, designadamente no que respeita à razão e domínio de convergência.

O método de Newton surge directamente da resolução iterativa do sistema das equações normais. Para economizar cálculos de derivadas e operações aritméticas, têm sido desenvolvidos outros métodos tais como o método de Gauss-Newton, o de Levenberg-Marquardt e os Quasi-Newton estruturados. De cada um destes, existem variadas implementações que surgiram da necessidade de simplificar os cálculos, melhorar o condicionamento do problema e conservar a razão de convergência próxima da quadrática, como se verifica no método de Newton.

Keywords

Nonlinear programming, least squares, Newton and Quasi-Newton algorithms.

1. Introdução

O problema de mínimos quadrados não linear surge principalmente nas aplicações em que se pretende ajustar um modelo $Y(t_i;x)$ a um conjunto de dados y_i, $i = 1,\ldots, m$, obtidos

experimentalmente. A função $Y(t_i;x)$ é não linear nas variáveis $x_1, x_2,\ldots, x_n$ que formam o vector x dos parâmetros.

Seja $Y:\Re^p \times \Re^n \rightarrow \Re$ uma função modelo de p variáveis reais t_j, j = 1,..., p, e n parâmetros reais x_j, j = 1,..., n, de tal forma que $t = [t_1,\ldots, t_p]^T$ e $x = [x_1,\ldots, x_n]^T$. Suponha serem dados p vectores de m elementos t_{ij} (i = 1,..., m e j = 1,..., p) e m números reais y_i, i = 1,..., m.

Considerando a função $S:\Re^n \rightarrow \Re$ definida por

$$S(x) = \sum_{i=1}^{m} \{Y_i(t_{i1},\ldots, t_{ip}; x) - y_i\}^2, \quad x \in \Re^n \tag{1}$$

em que $m \geq n$, o ajuste do modelo

$$Y(x) = [Y_1(x),\ldots, Y_m(x)]^T \text{ e } Y_i(x) = Y(t_{i1},\ldots, t_{ip}; x), \quad i = 1,\ldots, m, \tag{2}$$

aos valores tabelados y_i, i = 1,..., m, no sentido dos mínimos quadrados, corresponde à minimização da função S(x):

$$\min_{x \in \Re^n} \sum_{i=1}^{m} \{Y_i(x) - y_i\}^2 = \|Y(x) - y\|_2^2. \tag{3}$$

A solução $x^* = [x_1^*, x_2^*,\ldots, x_n^*]^T$ do problema (3) ajusta os dados tabelados no sentido dos mínimos quadrados. O objectivo é, pois, calcular o mínimo local da função S(x), que é não linear nas variáveis $x_1, x_2,\ldots, x_n$. A condição necessária e suficiente de primeira ordem para que x^* seja um mínimo local de S(x) estabelece que

$$\frac{\partial S(x^*)}{\partial x_i} = 0 \quad (i = 1,\ldots, n). \tag{4}$$

Se $f:\Re^n \rightarrow \Re^m$ for o vector das m funções residuais definido por

$$f(x) = Y(x) - y \tag{5}$$

com $y = [y_1,\ldots, y_m]^T$ e Y(x) definido em (2), então

$$S(x) = f(x)^T f(x), \tag{6}$$

$$\frac{\partial S(x)}{\partial x_i} = 2 \sum_{j=1}^{m} f_j(x) \frac{\partial f_j(x)}{\partial x_i} \quad (i = 1,\ldots, n)$$

e a condição (4) define o seguinte sistema de equações não lineares em x

$$\sum_{j=1}^{m} f_j(x) \frac{\partial f_j(x)}{\partial x_i} = 0 \quad (i = 1,\ldots, n). \tag{7}$$

Este sistema de equações é designado por sistema das equações normais do problema de mínimos quadrados. Se definirmos a matriz do Jacobiano, A(x), com elementos $a_{ji}(x)$, por

$$a_{ji}(x) = \frac{\partial f_j(x)}{\partial x_i} \quad (j = 1,\ldots, m;\ i = 1,\ldots, n) \tag{8}$$

então o sistema das equações normais, em termos da matriz A(x), é

$$A(x)^T f(x) = 0. \tag{9}$$

De (6), tira-se que o vector gradiente de S(x), g(x), é dado por

$$g(x) = 2A(x)^T f(x).$$

Os métodos para resolver o problema de mínimos quadrados não linear são iterativos e calculam a solução do sistema não linear (9), isto é, calculam um ponto estacionário da função S(x). A partir do sistema das equações normais (7), se derivarmos novamente em ordem a x, obtemos a matriz Hessiana de S(x), ou a matriz do Jacobiano de (9),

$$\sum_{j=1}^{m} \frac{\partial f_j(x)}{\partial x_k}\frac{\partial f_j(x)}{\partial x_i} + \sum_{j=1}^{m} f_j(x)\frac{\partial^2 f_j(x)}{\partial x_k \partial x_i} \quad (i = 1,\ldots, n;\ k = 1,\ldots, n) \tag{10}$$

e de acordo com (8), o primeiro termo de (10) é definido por

$$A(x)^T A(x). \tag{11}$$

Seja $B_j(x)$ a matriz das segundas derivadas da função $f_j(x)$, j = 1,..., m, com elementos

$$b^j_{ki}(x) = \frac{\partial^2 f_j(x)}{\partial x_k \partial x_i} \quad (i = 1,\ldots, n;\ k = 1,\ldots, n),$$

então o segundo termo de (10) é definido por

$$R(x) = \sum_{j=1}^{m} f_j(x)\ B_j(x) \tag{12}$$

Usando (11) e (12), obtém-se a matriz Hessiana de S(x), na forma matricial,

$$J(x) = A(x)^T A(x) + R(x). \tag{13}$$

Uma maneira de classificar o problema de mínimos quadrados tem em conta o valor do vector das funções residuais, f(x), na solução x^*. Assim, pode ter-se um problema de resíduos nulos, se $f(x^*) = 0$, o que significa um ajuste perfeito do modelo $Y(t_i,x)$ aos dados y_i. Os problemas de pequenos resíduos definem-se por $S(x^*)$ atingir valores próximos de zero e finalmente quando $\|f(x^*)\|$ é grande o problema designa-se por problema de grandes resíduos. Um valor enorme de S(x) pode não definir problema de grandes resíduos mas o seu valor

dever-se apenas ao escalonamento da função S(x). Definimos, assim, um problema de grandes resíduos se a quantidade $S(x)/\|A(x)^T A(x)\|$ for grande.

Este artigo está estruturado da seguinte maneira. Na Secção 2 são descritos o método de Newton e as suas variantes. Destas, a mais conhecida é o método de Gauss-Newton (em 2.1). Em 2.2 faz-se uma introdução ao método de Levenberg-Marquardt e em 2.3 e 2.4 surgem duas implementações desta última técnica. Na Secção 3 são introduzidos os métodos Quasi-Newton estruturados. Em 3.1 surgem as fórmulas de actualização mais gerais, em 3.2 referem-se duas fórmulas que têm em consideração factores de escalonamento de matrizes, em 3.3 introduzem-se as actualizações factorizadas, em 3.4 faz-se referência ao factor de escalonamento usado em 3.3 e finalmente em 3.5 aparece um esquema de actualização das matrizes Hessianas das funções f_j. Na Secção 4 apresenta-se um resumo de alguns resultados computacionais e a última secção é reservada aos comentários finais onde é feita uma retrospectiva sucinta sobre a convergência dos métodos descritos.

2. Método de Newton e suas Variantes

Quando o sistema das equações normais (9) é não linear, deve usar-se um método iterativo para o resolver. Quando as segundas derivadas de $f_j(x)$, $j = 1,\ldots, m$, são fáceis de calcular o método de Newton para a resolução do sistema (9) é o mais aconselhável. Este constrói uma sequência de aproximações $\{x^{(k)}\}$, à solução, de modo que

$$x^{(k+1)} = x^{(k)} + \Delta^{(k)}$$

sendo $\Delta^{(k)}$ a direcção de procura que satisfaz a equação

$$[A(x^{(k)})^T A(x^{(k)}) + R(x^{(k)})]\, \Delta^{(k)} = -A(x^{(k)})^T f(x^{(k)}) \quad \text{para } k = 0, 1,\ldots \tag{14}$$

em que $A(x)^T f(x)$ é o vector das equações normais e a matriz dos coeficientes deste sistema é a matriz do Jacobiano de (9). Para simplificar a notação, usaremos a partir de agora $A^{(k)}$, $R^{(k)}$, $J^{(k)}$, $g^{(k)}$ e $f^{(k)}$ em vez de, respectivamente, $A(x^{(k)})$, $R(x^{(k)})$, $J(x^{(k)})$, $g(x^{(k)})$ e $f(x^{(k)})$. Uma resolução estável do sistema (14) recorre à decomposição dos valores singulares da matriz A(x),

$$A = U \begin{bmatrix} C \\ 0 \end{bmatrix} W^T$$

em que $C = \mathrm{diag}(c_1, c_2,\ldots, c_n)$ é uma matriz diagonal com os valores singulares de A(x), U é uma matriz ortonormal do tipo $m \times m$ e W é uma matriz ortonormal $n \times n$. Esta implementação é válida quer a matriz do sistema seja definida positiva quer seja indefinida (veja-se em Gill e Murray [8]). O correspondente algoritmo que também inclui uma estratégia de procura unidimensional baseada numa interpolação cúbica é apresentado em Ramadas [17].

Da análise de convergência do método de Newton para a resolução de sistemas de equações não lineares, sabe-se que o método é localmente convergente e a razão de convergência é dois:

Teorema 1: Seja $g:\Re^n \rightarrow \Re^n$ continuamente diferenciável num conjunto aberto convexo $D \subset \Re^n$. Suponha que existem $x^* \in \Re^n$ e r, $\beta > 0$, tais que $N(x^*,r) \subset D$, $g(x^*) = 0$, que $J(x^*)^{-1}$ existe com $\|J(x^*)^{-1}\| \leq \beta$, e que $J \in Lip_\gamma (N(x^*,r))$ (Lipschitz contínua no conjunto $N(x^*, r)$ com constante γ). Então existe $\varepsilon > 0$ tal que para qualquer $x^{(0)} \in N(x^*, \varepsilon)$ a sequência x_1, x_2,... gerada por

$$x^{(k+1)} = x^{(k)} - J^{(k)^{-1}} g^{(k)}, \quad k = 0, 1,...$$

é bem definida, converge para x^*, e obedece a

$$\|x^{(k+1)} - x^*\| \leq \beta\gamma \|x^{(k)} - x^*\|^2, \quad k = 0, 1,...$$

Prova: (Dennis e Schnabel [5]).

$N(x^*,r)$ é uma vizinhança de x^*, de raio r. Este método é pouco utilizado nas aplicações. A matriz dos coeficientes, em (14), nem sempre se encontra disponível analiticamente a um custo razoável. Implementações baseadas na aproximação numérica das derivadas são apresentadas em Bartels [1] e Wolfe [18] e alguns resultados de experiências realizadas surgem em Gill e Murray [8] e McKeown [13].

2.1 Método de Gauss-Newton

O método de Gauss-Newton foi o primeiro a explorar a estrutura especial da matriz Hessiana e do vector gradiente de S(x) que ocorre nos problemas de mínimos quadrados. A direcção de procura Gauss-Newton é a solução do sistema

$$[A^{(k)^T} A^{(k)}] \Delta^{(k)} = -A^{(k)^T} f^{(k)}. \tag{15}$$

Estas equações obtêm-se ignorando a matriz R(x) que depende das segundas derivadas dos $f_j(x)$, j = 1,..., m. Se o valor de S(x) na solução é igual ou próximo de zero, os valores de $f_j(x^*)$, j = 1,..., m também são nulos ou muito pequenos e, consequentemente, o termo R(x) em (13) pode ser ignorado. O Jacobiano é então aproximado por

$$J(x) \approx A(x)^T A(x). \tag{16}$$

para valores de x suficientemente próximos de x^*. Se a aproximação inicial $x^{(0)}$ estiver perto de x^*, pode-se esperar que o método baseado em (16) seja eficiente.

O método de Gauss-Newton é pois direccionado para problemas em que $\|R(x)\|$ é pequena quando comparada com $\|A(x)^T A(x)\|$, como acontece com o problema de resíduos nulos, pois $f(x) \rightarrow 0$ quando $x \rightarrow x^*$. Para este tipo de problemas o método Gauss-Newton converge com a mesma razão de convergência do método de Newton, apesar de apenas serem utilizadas primeiras derivadas nos cálculos.

Teorema 2: Sejam $f:\Re^n \rightarrow \Re^m$ e $S(x) = f(x)^T f(x)$ duas vezes continuamente diferenciáveis num conjunto aberto convexo $D \subset \Re^n$. Suponha que existem $A(x) \in Lip_\gamma (D)$ com $\|A(x)\| \leq \eta$

para todo $x \in D$, e que existe $x^* \in D$ e $\lambda, \sigma \geq 0$ (λ é o valor próprio mais pequeno de $A(x^*)^T A(x^*)$), tais que $A(x^*)^T f(x^*) = 0$ e

$$\|(A(x) - A(x^*))^T f(x^*)\|_2 \leq \sigma \|x - x^*\|_2 \qquad (17)$$

para todo $x \in D$. Se $\sigma < \lambda$, então para qualquer $c \in (1, \frac{\lambda}{\sigma})$, existe $\varepsilon > 0$ tal que para todo o $x^{(0)} \in N(x^*, \varepsilon) \subset D$, a sequência gerada pelo método de Gauss-Newton

$$x^{(k+1)} = x^{(k)} - (A^{(k)^T} A^{(k)})^{-1} A^{(k)^T} f^{(k)}$$

é bem definida, converge para x^*, e obedece a

$$\|x^{(k+1)} - x^*\|_2 \leq \frac{c\sigma}{\lambda} \|x^{(k)} - x^*\|_2 + \frac{c\eta\gamma}{2\lambda} \|x^{(k)} - x^*\|_2^2$$

e

$$\|x^{(k+1)} - x^*\|_2 \leq \frac{c\sigma + \lambda}{2\lambda} \|x^{(k)} - x^*\|_2 < \|x^{(k)} - x^*\|_2.$$

Prova: em Dennis e Schnabel [5].

O corolário seguinte estabelece que o método de Gauss-Newton tem convergência quadrática local se $R(x^*) = 0$. Esta situação ocorre quando o vector $f(x)$ é formado por funções lineares ou o problema não linear é de resíduos nulos.

Corolário 1: Suponha que as hipóteses do Teorema 2 são satisfeitas. Se $f(x^*) = 0$, então existe $\varepsilon > 0$ tal que para qualquer $x^{(0)} \in N(x^*, \varepsilon)$ a sequência $\{x^{(k)}\}$ gerada pelo método Gauss-Newton é bem definida e converge quadraticamente para x^*.

Prova: (Dennis e Schnabel [5]).

A constante σ desempenha um papel importante na prova de convergência pois mede a não linearidade de $f(x)$ e o tamanho do vector resíduo $f(x^*)$. Se $f(x)$ é linear ou $f(x^*) = 0$, a partir de (17), tira-se que $\sigma = 0$.

Quando a sequência de aproximações obtidas pelo método Gauss-Newton converge, geralmente, fá-lo rapidamente. No entanto, para muitos problemas de mínimos quadrados, a sequência diverge ou torna-se mesmo indefinida. O método de Gauss-Newton não é bem definido se $A^{(k)}$ tem característica deficitária, isto é, se a característica de A é menor do que n, a matriz $A^{(k)^T} A^{(k)}$ é singular e a direcção $\Delta^{(k)}$ dada por (15) não é definida. Embora a implementação do método de Gauss-Newton possa levantar alguns problemas, ele é a base de alguns dos métodos mais importantes e bem sucedidos para a resolução de um problema de mínimos quadrados. Se a aproximação ao Jacobiano definida em (16) for definida positiva, então o vector $\Delta^{(k)}$ do sistema (15) verifica a condição

$$g^{(k)^T} \Delta^{(k)} < 0, \tag{18}$$

isto é, $\Delta^{(k)}$ é de descida em relação à função S(x), no ponto $x^{(k)}$. Nestas condições existe um escalar $\alpha^{(k)} > 0$, que dá o comprimento do passo a efectuar ao longo da direcção $\Delta^{(k)}$, de tal forma que

$$S(x^{(k)} + \alpha^{(k)} \Delta^{(k)}) < S(x^{(k)}).$$

Esta relação significa que, a partir do ponto $x^{(k)}$, pode caminhar-se ao longo de $\Delta^{(k)}$ e encontrar um novo ponto, $x^{(k)} + \alpha^{(k)} \Delta^{(k)}$, para o qual a função S toma um valor mais baixo do que em $x^{(k)}$. Também é possível que $\Delta^{(k)}$ seja bem definida mas que $g^{(k)^T} \Delta^{(k)} = 0$. Nesta situação, pode acontecer que não seja possível calcular um valor de $\alpha^{(k)}$ que origine um decréscimo do valor de S(x) e o método de Gauss-Newton falha. Resultados de experiências computacionais realizadas com o método Gauss-Newton baseado no cálculo do comprimento do passo são apresentados em Fernandes [6], Gill e Murray [8] e Wolfe [18]. Em McKeown [13] é feita um estudo comparativo entre três implementações do método de Gauss-Newton e duas de métodos Quasi-Newton. Em Gulliksson e Soderkvist [9] e McKeown [12] são introduzidos pesos na formulação do problema. Este facto pode influenciar positivamente o seu condicionamento. Além dos pesos, Gulliksson e Soderkvist [*ibidem*] consideram também problemas formulados implicitamente. As implementações dos dois tipos de problemas, explícito e implícito, são apresentadas em paralelo e de uma maneira equivalente. O caso implícito é caracterizado pela introdução de um conjunto de restrições o que origina um aumento da dimensão do problema.

2.2 Método de Levenberg-Marquardt

O método de Levenberg-Marquardt (Marquardt [11]) tem como objectivo superar os problemas provocados pelo cálculo das segundas derivadas, no método de Newton, usando uma matriz diagonal $\mu^{(k)}I$ como aproximação a $R^{(k)}$. A primeira implementação mais elaborada deste método é devida a Fletcher [7] (também descrita em Wolfe [18]). Além disso, este método também soluciona os problemas causados pela singularidade da matriz $A^{(k)^T} A^{(k)}$ no método Gauss-Newton. A equação iterativa do método é

$$(A^{(k)^T} A^{(k)} + \mu^{(k)} I) \Delta^{(k)} = -A^{(k)^T} f^{(k)}$$

com $\mu^{(k)} \geq 0$, para $k = 0, 1, 2, \ldots$ com $x^{(k+1)} = x^{(k)} + \Delta^{(k)}$.

As propriedades de convergência local do método Levenberg-Marquardt são similares às do método de Gauss-Newton e são dadas pelo seguinte teorema.

Teorema 3: Suponha que as condições do Teorema 2 são satisfeitas e que a sequência $\{\mu^{(k)}\}$, de números reais não negativos, é limitada por $b > 0$. Se $\sigma < \lambda$ então para qualquer

$c \in (1,(\lambda+b)/(\sigma+b))$, existe $\varepsilon > 0$, de tal forma que, para todo $x^{(0)} \in N(x^*,\varepsilon)$, a sequência gerada pelo método de Levenberg-Marquardt,

$$x^{(k+1)} = x^{(k)} - (A^{(k)^T} A^{(k)} + \mu^{(k)} I)^{-1} A^{(k)^T} f^{(k)}$$

é bem definida, obedece a,

$$\|x^{(k+1)} - x^*\|_2 \leq \frac{c(\sigma+b)}{\lambda+b} \|x^{(k)} - x^*\|_2 + \frac{c\eta\gamma}{2(\lambda+b)} \|x^{(k)} - x^*\|_2^2$$

e

$$\|x^{(k+1)} - x^*\|_2 \leq \frac{c(\sigma+b)+(\lambda+b)}{2(\lambda+b)} \|x^{(k)} - x^*\|_2 < \|x^{(k)} - x^*\|_2.$$

Se $f(x^*) = 0$ e $\mu^{(k)} = O(\|A^{(k)^T} f^{(k)}\|_2)$, então $\{x^{(k)}\}$ converge quadraticamente para x^*.

Prova: ver em Dennis e Schnabel [5].

O Teorema 3 sugere que o algoritmo Levenberg-Marquardt pode tornar-se muito lento em problemas de grandes resíduos ou fortemente não lineares. Quando o passo Gauss-Newton é demasiado longo, o passo Levenberg-Marquardt encontra-se perto da direcção de descida máxima, $-A^{(k)^T} f^{(k)}$, sendo portanto superior ao passo de Gauss-Newton. Dennis e Schnabel [5] sugerem a consulta doutros trabalhos para a prova de convergência global dos algoritmos do tipo Levenberg-Marquardt, por exemplo Moré [14].

2.3 Sequência MDLS

Usaremos também, a partir de agora, $V^{(k)}$ em vez de $V(x^{(k)})$ e $D^{(k)}$ em vez de $D(x^{(k)})$.

Uma variante do método de Levenberg-Marquardt define uma sequência de aproximações, conhecida por sequência MDLS (de Modified Damped Least Squares), $\{x^{(k)}\}$, a partir de

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} \Delta^{(k)} \tag{19}$$

em que $\Delta^{(k)}$ satisfaz

$$(V^{(k)} + \mu^{(k)} D^{(k)}) \Delta^{(k)} = -A^{(k)^T} f^{(k)}, \; k = 0, 1, 2,\ldots \tag{20}$$

sendo $V^{(k)} = A^{(k)^T} A^{(k)}$, $\mu^{(k)} > 0$ e $D^{(k)} = \text{diag}(v_{11}^{(k)},\ldots, v_{nn}^{(k)})$ em que v_{ii} é o elemento (i,i) da matriz V. O escalar $\alpha^{(k)}$ define o comprimento do passo.

No Teorema 4 e Corolário 2 estão contidas condições suficientes para a convergência superlinear da sequência MDLS.

Teorema 4: Se $\{x^{(k)}\}$ for uma sequência MDLS que converge para $x^* \in \Re^n$; se λ for o menor valor próprio de $V(x^*)$; $s = \max_{1 \leq i \leq n} \{|s_i|\}$, em que os $s_i (i = 1,\ldots, n)$ são os valores próprios da

matriz $R(x^*)$ (12); $b = \frac{s}{\lambda} < 1$; $\beta < \frac{(1-b)}{2}$; $\mu^{(k)} \to 0$ quando $k \to \infty$, então existe um inteiro positivo $\hat{k}$ de tal forma que, para $\forall k \geq \hat{k}$, $\alpha^{(k)} = 1$;

$$\lim_{k\to\infty} \sup \frac{\|x^{(k+1)} - x^*\|_2}{\|x^{(k)} - x^*\|_2} \leq b$$

e x^* é um mínimo isolado de S.

Prova: (Wolfe [18]).

Note-se que, se $\mu^{(k)} = 0$ ($\forall k \geq 0$), então este teorema é aplicável ao método de Gauss-Newton com procura unidimensional.

Corolário 2: Se as hipóteses do Teorema 4 forem válidas e $S(x^*) = 0$, então a sequência MDLS gerada por (19) e (20) converge superlinearmente para x^*.
Prova: veja-se em Wolfe [18].

2.4 Problema estruturado de mínimos quadrados

A matriz R(x) (em (12)) pode ser aproximada por outra matriz diagonal diferente de μD (do algoritmo MDLS). Considerando $D^{(k)} = \text{diag}(d_1^{(k)}, d_2^{(k)}, \ldots, d_n^{(k)})$, com

$$d_i^{(0)} = \left\| \frac{\partial f(x^{(0)})}{\partial x_i} \right\|$$

(norma do vector coluna i de A(x) calculada no ponto $x^{(0)}$) e

$$d_i^{(k)} = \max\{d_i^{(k-1)}, \left\| \frac{\partial f(x^{(k)})}{\partial x_i} \right\|\}, \text{ para } k \geq 1,$$

a nova modificação do algoritmo de Levenberg-Marquardt define a seguinte sequência $\{x^{(k)}\}$ de aproximações à solução x^*,

$$x^{(k+1)} = x^{(k)} + \Delta^{(k)} \quad k = 0, 1, 2, \ldots$$

com $\Delta^{(k)}$ calculado a partir

$$(V^{(k)} + \mu^{(k)} D^{(k)^T} D^{(k)}) \Delta^{(k)} = -A^{(k)^T} f^{(k)}, \tag{21}$$

com $V^{(k)} = A^{(k)^T} A^{(k)}$ e $\mu^{(k)} \geq 0$.

O sistema (21) corresponde às equações normais do seguinte problema estruturado de mínimos quadrados

$$\begin{pmatrix} A \\ \mu^{1/2} D \end{pmatrix} D \cong - \begin{pmatrix} f \\ 0 \end{pmatrix}. \tag{22}$$

A resolução do sistema (21), para $\mu > 0$, é duas vezes mais rápida do que a resolução do problema formulado em (22). No entanto, a formação de $A^T A$ ou $D^T D$ pode originar

"underflows" ou "overflows" desnecessários, situações que não ocorrem em (22). Assim, a perda na velocidade é compensada pelo ganho na confiança e robustez. Além disso, podem surgir dificuldades nas equações normais quando $\mu = 0$ e A é de característica deficitária. A resolução a partir da formulação (22), é feita em duas fases (Moré [14]) e ambas incluem a decomposição QR de matrizes. Moré [*ibidem*] sugere a implementação de matrizes de rotação de Givens (veja-se em Hager [10]) na decomposição QR.

Sempre que for necessário modificar μ, apenas a segunda fase do processo terá de ser repetida. Detalhes da implementação deste método podem também ser consultados em Ramadas [17].

3. Métodos Quasi-Newton Estruturados

O cálculo do vector gradiente, $A^T f$, em cada ponto x, pode ser utilizado para fornecer uma aproximação Quasi-Newton à matriz R(x) do Jacobiano J(x).

Uma aplicação directa dos métodos Quasi-Newton (Wolfe [18] e também em McKeown [13]) ao problema de mínimos quadrados não linear não é aconselhável pois tem como objectivo aproximar toda a matriz Hessiana da função objectivo, S. Como a matriz A(x) é facilmente calculável, a parte $A(x)^T A(x)$ de (13) está disponível e parece mais vantajoso aproximar apenas a segunda parte, ou seja, R(x).

3.1 Fórmulas de actualização

A aplicação dos métodos Quasi-Newton, ao problema tratado neste artigo, considera que a direcção de procura, $\Delta^{(k)}$, pode ser calculada resolvendo

$$(A^{(k)^T} A^{(k)} + H^{(k)}) \Delta^{(k)} = -A^{(k)^T} f^{(k)}$$

em que a matriz $H^{(k)}$ é a aproximação à segunda parte da matriz Hessiana de S, $R^{(k)}$. Para a iteração seguinte, k+1, a matriz $H^{(k+1)}$ deve verificar

$$(A^{(k+1)^T} A^{(k+1)} + H^{(k+1)}) \Delta^{(k)} = A^{(k+1)^T} f^{(k+1)} - A^{(k)^T} f^{(k)} \qquad (25)$$

em que $\Delta^{(k)} = x^{(k+1)} - x^{(k)}$. Esta condição é conhecida por condição secante para aproximar a matriz Hessiana de S(x). A aproximação Quasi-Newton $H^{(k+1)}$, a $R^{(k+1)}$, é obtida pela adição de uma matriz de correcção, $CC^{(k)}$, de característica um ou dois, à matriz $H^{(k)}$,

$$H^{(k+1)} = H^{(k)} + CC^{(k)}. \qquad (26)$$

Este tipo de técnicas define os métodos Quasi-Newton estruturados. Um destes esquemas Quasi-Newton que satisfaz (25) e (26) é (Gill e Murray [8] e Yabe [20])

$$H^{(0)} = 0$$

$$CC^{(k)} = \frac{1}{\alpha^{(k)} y^{(k)T} \Delta^{(k)}} y^{(k)} y^{(k)T} - \frac{1}{\Delta^{(k)T} W^{(k)} \Delta^{(k)}} W^{(k)} \Delta^{(k)} \Delta^{(k)T} W^{(k)} \quad (27)$$

onde $W^{(k)} = A^{(k+1)T} A^{(k+1)} + H^{(k)}$ e

$$y^{(k)} = A^{(k+1)T} f^{(k+1)} - A^{(k)T} f^{(k)}. \quad (28)$$

Esta fórmula é baseada no esquema actualizador BFGS (de Broyden, Fletcher, Goldfarb e Shanno) muito conhecido e usado na minimização de funções não lineares.

Se algum algoritmo para o cálculo do comprimento do passo for usado de modo a assegurar que $y^{(k)T} \Delta^{(k)} > 0$, a fórmula de actualização BFGS tem a propriedade de gerar uma matriz $A^{(k+1)T} A^{(k+1)} + H^{(k+1)}$ definida positiva, se $A^{(k+1)T} A^{(k+1)} + H^{(k)}$ também o for. Os métodos Quasi-Newton estruturados não exigem que cada matriz $H^{(k)}$ seja definida positiva, o que é importante pois a própria matriz R(x) pode não ser definida positiva, até mesmo na solução.

Directamente da condição (25) surge também a seguinte matriz de correcção

$$CC^{(k)} = \frac{(u^{(k)} - H^{(k)}\Delta^{(k)})\Delta^{(k)T} + \Delta^{(k)}(u^{(k)} - H^{(k)}\Delta^{(k)})^T}{\Delta^{(k)T}\Delta^{(k)}} - \frac{(u^{(k)} - H^{(k)}\Delta^{(k)})^T \Delta^{(k)}}{(\Delta^{(k)T}\Delta^{(k)})^2} \Delta^{(k)} \Delta^{(k)T} \quad (29)$$

em que

$$u^{(k)} = y^{(k)} - A^{(k+1)T} A^{(k+1)} \Delta^{(k)}$$

e $y^{(k)}$ foi definido em (28). Esta fórmula de actualização é conhecida por definir o algoritmo de Broyden e Dennis (Yabe e Takahashi [19] e Yabe [20]).

O algoritmo de Betts (em Gill e Murray [8]) usa o esquema de actualização de característica um, usando, na condição secante (25), a matriz $A^{(k)}$ em vez de $A^{(k+1)}$. Isto origina a seguinte correcção, $CC^{(k)}$, para $H^{(k)}$

$$CC^{(k)} = \frac{1}{q^{(k)T} \Delta^{(k)}} q^{(k)} q^{(k)T} \quad (30)$$

em que

$$q^{(k)} = y^{(k)} - (A^{(k)T} A^{(k)} + H^{(k)}) \Delta^{(k)}$$

e $y^{(k)}$ já foi definido em (28).

Nos problemas gerais de optimização sem restrições a condição secante é apenas uma das restrições impostas à aproximação da matriz Hessiana. Estas restrições podem ser usadas para assegurar que se a função objectivo é quadrática, e consequentemente, o objectivo é aproximar uma matriz constante, a aproximação Quasi-Newton deve convergir para a Hessiana em n iterações. No contexto dos problemas de mínimos quadrados não lineares, R(x) é uma matriz constante apenas quando for nula. Uma vez que se tenta aproximar uma matriz, que depende de

x mesmo no caso quadrático, propriedades como a terminação quadrática não têm qualquer significado.

3.2 Fórmulas de actualização com factor de escalonamento

Além da condição secante referenciada em (25) Bartholomew-Biggs (ver em Yabe e Takahashi [19]) e Dennis, Gay e Welsch [4], propuseram outra que é baseada na estrutura especial da segunda parte da Hessiana de S:

$$H^{(k+1)}\,\Delta^{(k)} = (A^{(k+1)} - A^{(k)})^{T} f^{(k+1)}. \tag{31}$$

A partir desta condição e usando um factor de escalonamento propuseram algoritmos robustos que resolvem todo o tipo de problemas. Quando o problema é de grandes resíduos, estes algoritmos funcionam tão bem como o de Broyden e Dennis e para problemas de pequenos resíduos, têm um comportamento semelhante ao método Gauss-Newton. Em problemas de resíduos nulos (ou de pequenos resíduos) a matriz $H^{(k)}$ deve tender para a matriz nula, quando k aumenta, uma vez que aproxima a parte R(x) do Jacobiano (veja-se em (13)) e esta matriz tende para a matriz nula à medida que $x^{(k)} \rightarrow x^{*}$ (pois $f(x^{*}) = 0$). No entanto, as fórmulas gerais de actualização conhecidas não geram matrizes nulas. Há assim necessidade de introduzir nos esquemas de actualização de matrizes factores de escalonamento que forcem essa convergência.

A fórmula de actualização sugerida por Bartholomew-Biggs (BB) é

$$H^{(k+1)} = \beta^{(k)} H^{(k)} + \frac{(v^{(k)}-\beta^{(k)}H^{(k)}\Delta^{(k)})(v^{(k)}-\beta^{(k)}H^{(k)}\Delta^{(k)})^{T}}{(v^{(k)}-\beta^{(k)}H^{(k)}\Delta^{(k)})^{T}\Delta^{(k)}} \tag{32}$$

com

$$\beta^{(k)} = \frac{f^{(k+1)^{T}} f^{(k)}}{f^{(k)^{T}} f^{(k)}} \quad \text{e} \quad v^{(k)} = (A^{(k+1)} - A^{(k)})^{T} f^{(k+1)},$$

onde $\beta^{(k)}$ é o factor de escalonamento de Biggs e a actualização de Dennis, Gay e Welsch (DGW) também inclui o factor de escalonamento e é definida por:

$$H^{(k+1)} = \beta^{(k)} H^{(k)} + \frac{(v^{(k)}-\beta^{(k)}H^{(k)}\Delta^{(k)})y^{(k)^{T}} + y^{(k)}(v^{(k)}-\beta^{(k)}H^{(k)}\Delta^{(k)})^{T}}{\Delta^{(k)^{T}} y^{(k)}} -$$

$$- \frac{\Delta^{(k)^{T}}(v^{(k)}-\beta^{(k)}H^{(k)}\Delta^{(k)})}{(\Delta^{(k)^{T}} y^{(k)})^{2}}\, y^{(k)} y^{(k)^{T}}, \tag{33}$$

com

$$\beta^{(k)} = \min\left(\left| \frac{\Delta^{(k)^{T}} v^{(k)}}{\Delta^{(k)^{T}} H^{(k)} \Delta^{(k)}} \right|, 1 \right).$$

3.3 Actualizações factorizadas

Das fórmulas de actualização para os problemas gerais de optimização, as mais eficientes conservam a matriz definida positiva, ao longo do processo iterativo. São exemplos, as actualizações BFGS e DPF (devida a Davidon, Fletcher e Powell). Esta propriedade garante que a direcção de procura é uma direcção de descida em relação à função objectivo. Por outro lado, para os problemas de mínimos quadrados e com as actualizações Quasi-Newton estruturadas, não parece fácil arranjar uma estratégia que construa fórmulas actualizadoras para $H^{(k)}$, por forma a garantir que a matriz $A^{(k)^T} A^{(k)} + H^{(k)}$ seja definida positiva. Para ultrapassar este inconveniente, foi proposta uma estratégia baseada na factorização da matriz $A^{(k)^T} A^{(k)} + H^{(k)}$.

A direcção de procura é calculada através da resolução do sistema

$$(L^{(k)} + A^{(k)})^T (L^{(k)} + A^{(k)}) \Delta^{(k)} = -A^{(k)^T} f^{(k)}$$

em que a matriz $L^{(k)}$ é uma matriz de correcção do Jacobiano, de forma que $L^{(k)^T} L^{(k)} + L^{(k)^T} A^{(k)} + A^{(k)^T} L^{(k)}$ é a aproximação à segunda parte da matriz Hessiana de S, $R^{(k)}$. Para se chegar à fórmula actualizadora da matriz $L^{(k)}$, considera-se a condição secante (25) para $H^{(k+1)}$, que pode ser reduzida à seguinte condição para $L^{(k+1)}$,

$$(L^{(k+1)} + A^{(k+1)})^T (L^{(k+1)} + A^{(k+1)}) \Delta^{(k)} = z^{(k)} \tag{34}$$

em que

$$z^{(k)} = y^{(k)}. \tag{35}$$

No caso, de se tomar a condição (31) em vez de (25), tem-se também a condição (34), mas agora

$$z^{(k)} = v^{(k)} + A^{(k+1)^T} A^{(k+1)} \Delta^{(k)} \tag{36}$$

e os vectores $\Delta^{(k)}$, $y^{(k)}$ e $v^{(k)}$ são os definidos anteriormente.

É possível mostrar que, para $\Delta^{(k)}$ e $z^{(k)}$ diferentes de zero, a matriz dos coeficientes de (34) é consistente apenas se

$$L^{(k+1)^T} h = z^{(k)} - A^{(k+1)^T} h \quad \text{e} \quad L^{(k+1)} \Delta^{(k)} = h - A^{(k+1)} \Delta^{(k)} \tag{37}$$

para algum vector h de dimensão m. Além disso, as equações (37) têm uma matriz solução comum, $L^{(k+1)}$, se e só se cada equação separadamente tem uma solução e $h^T h = \Delta^{(k)^T} z^{(k)}$. Assim, o objectivo é encontrar uma matriz rectangular $L^{(k+1)}$ que satisfaça as equações (37) pressupondo que $\Delta^{(k)^T} z^{(k)} > 0$.

Uma vez que as equações (37) não determinam uma matriz solução $L^{(k+1)}$ única, usa-se a técnica de variação mínima no sentido de Dennis e Schnabel [5], para determinar a correspondente fórmula de actualização.

Minimizando a norma de Frobenius da variação verificada, na matriz L, min $\|L^{(k+1)} - L^{(k)}\|_F$, em relação a $L^{(k+1)}$, sujeita à primeira condição de (37), surge a seguinte fórmula de actualização, de característica um, para $L^{(k)}$,

$$L^{(k+1)} = L^{(k)} + \left(\frac{(L^{(k)}+A^{(k+1)})\Delta^{(k)}}{\Delta^{(k)T}P^{(k)}\Delta^{(k)}}\right)\left(\left(\frac{\Delta^{(k)T}P^{(k)}\Delta^{(k)}}{\Delta^{(k)T}z^{(k)}}\right)^{1/2} z^{(k)} - P^{(k)}\Delta^{(k)}\right)^{T} \qquad (38)$$

com

$$P^{(k)} = (L^{(k)} + A^{(k+1)})^{T}(L^{(k)} + A^{(k+1)}).$$

Detalhes da dedução destas fórmulas podem ser consultados ou no trabalho de Yabe e Takahashi [19] ou em Ramadas [17]. O Teorema 5 e Corolário 3 estabelecem a convergência local desta actualização factorizada da matriz H.

Seja D um subconjunto convexo e aberto de $\Re^n$ que contém o mínimo. Consideremos as seguintes hipóteses:

H1: Sejam ξ_1, ξ_2 e p constantes positivas, tais que

$$\|J(x) - J(x^*)\| \leq \xi_1 \|x - x^*\|^p \text{ para qualquer } x \in D$$

e

$$\|A(x_1) - A(x_2)\| \leq \xi_2 \|x_1 - x_2\|^p \text{ para quaisquer } x_1 \text{ e } x_2 \in D.$$

H2: J é simétrica e definida positiva em x^*.

Teorema 5: Suponha que as hipóteses H1 e H2 são satisfeitas. Seja a matriz $L^{(k)}$ actualizada pela fórmula (38), onde $z^{(k)}$ é dado por (35) ou (36). Seja a sequência $\{x^{(k)}\}$ gerada por

$$x^{(k+1)} = x^{(k)} - ((L^{(k)} + A^{(k)})^{T}(L^{(k)} + A^{(k)}))^{-1} A^{(k)T} f^{(k)}. \qquad (39)$$

Então, para qualquer $r \in (0,1)$, existem constantes positivas $\varepsilon(r)$ e $\delta(r)$ tais que se $\|x^{(0)} - x^*\| \leq \varepsilon(r)$ e $\|((L^{(0)} + A^{(0)})^{T}(L^{(0)} + A^{(0)}))^{-1} - J(x^*)^{-1}\|_{F,\infty} \leq \delta(r)$, a sequência $\{x^{(k)}\}$ gerada pela equação (39) é bem definida, converge para x^*, com

$$\|x^{(k+1)} - x^*\| \leq r\|x^{(k)} - x^*\|, k \geq 0.$$

Além disso, as sequências

$$\{\|(L^{(k)} + A^{(k)})^{T}(L^{(k)} + A^{(k)})\|\}, \{\|((L^{(k)} + A^{(k)})^{T}(L^{(k)} + A^{(k)}))^{-1}\|\},$$

$$\{\|(L^{(k)} + A^{(k+1)})^{T}(L^{(k)} + A^{(k+1)})\|\} \text{ e } \{\|((L^{(k)} + A^{(k+1)})^{T}(L^{(k)} + A^{(k+1)}))^{-1}\|\}$$

são uniformemente limitadas.

Prova: veja-se em Yabe e Takahashi [19].

Corolário 3: Suponha que todas as condições do Teorema 5 são válidas. Então a sequência $\{x^{(k)}\}$ gerada pela equação (39) converge superlinearmente para x^*.
Prova: veja-se em Yabe e Takahashi [19].

As provas do Teorema 5 e Corolário 3 não exigem que a factorização $(L^{(k)} + A^{(k)})^T (L^{(k)} + A^{(k)})$ convirja para $J(x^*)$, à medida que $x^{(k)} \to x^*$. A caracterização desta convergência superlinear, tal como foi definida por Dennis e Moré [3] e confirmada por Nocedal [6], também aqui é baseada na seguinte condição necessária e suficiente:

$$\lim_{k\to\infty} \frac{\left\| \left[\left((L^{(k)} + A^{(k)})^T (L^{(k)} + A^{(k)}) \right)^{-1} - J(x^*)^{-1} \right] z^{(k)} \right\|}{\|z^{(k)}\|} = 0,$$

isto é, a direcção Quasi-Newton estruturada, definida por (39), aproxima-se assimptoticamente da direcção Newton.

Uma outra fórmula de actualização da matriz L pode ser obtida, tentando minimizar a seguinte norma $\|W_L(L^{(k+1)} - L^{(k)}) W_R\|_F$, em relação a $L^{(k+1)}$, sujeita à segunda condição de (37), para as matrizes não singulares $W_L \in \Re^{m\times m}$ e $W_R \in \Re^{n\times n}$. Se a matriz definida positiva W, $W = W_R^{-T} W_R^{-1}$, for escolhida por forma a que $W\Delta^{(k)} = z^{(k)}$, obtém-se a seguinte actualização de característica um para $L^{(k)}$,

$$L^{(k+1)} = L^{(k)} + (L^{(k)} + A^{(k+1)}) \left(\left(\frac{\Delta^{(k)T} z^{(k)}}{z^{(k)T} (P^{(k)})^{-1} z^{(k)}} \right)^{1/2} (P^{(k)})^{-1} z^{(k)} - \Delta^{(k)} \right) \left(\frac{z^{(k)}}{\Delta^{(k)T} z^{(k)}} \right)^T. \qquad (40)$$

Os teoremas que estabelecem a convergência local da fórmula de actualização (40) são idênticos aos da fórmula (38) (Yabe e Takahashi [19]).

3.4 A importância do factor de escalonamento

Nos problemas de resíduos nulos, a matriz $L^{(k)T} L^{(k)} + L^{(k)T} A^{(k)} + A^{(k)T} L^{(k)}$ deve convergir para zero à medida que k aumenta. Se os seus elementos não diminuem, ao longo do processo iterativo, então os métodos Quasi-Newton estruturados não conseguem competir com o método de Gauss-Newton. Como as fórmulas de actualização nunca geram uma matriz nula, há necessidade de introduzir esquemas que forçam a convergência através do escalonamento das matrizes a actualizar. As fórmulas BB (em (32)) e DGW (em (33)) (Dennis, Gay e Welsch [4], Yabe e Takahashi [19]) são exemplos deste tipo de esquemas.

A introdução do factor de escalonamento nas actualizações factorizadas (38) e (40) origina, respectivamente, as seguintes versões,

$$L^{(k+1)} = \beta^{(k)} L^{(k)} + \left(\frac{(\beta^{(k)}L^{(k)}+A^{(k+1)})\Delta^{(k)}}{\Delta^{(k)T}P^{(k)}\Delta^{(k)}}\right)\left(\left(\frac{\Delta^{(k)T}P^{(k)}\Delta^{(k)}}{\Delta^{(k)T}z^{(k)}}\right)^{1/2} z^{(k)}-P^{(k)}\Delta^{(k)}\right)^{T} \quad (41)$$

e

$$L^{(k+1)} = \beta^{(k)} L^{(k)} + (\beta^{(k)} L^{(k)} + A^{(k+1)})\left(\left(\frac{\Delta^{(k)T}z^{(k)}}{z^{(k)T}(P^{(k)})^{-1}z^{(k)}}\right)^{1/2} (P^{(k)})^{-1}z^{(k)}-\Delta^{(k)}\right)\left(\frac{z^{(k)}}{\Delta^{(k)T}z^{(k)}}\right)^{T} \quad (42)$$

em que $z^{(k)}$ é dado por (36) e o escalar $\beta^{(k)}$ é o factor de escalonamento de Biggs (em (32)). A matriz $P^{(k)}$ é agora definida por

$$P^{(k)} = (\beta^{(k)} L^{(k)} + A^{(k+1)})^{T} (\beta^{(k)} L^{(k)} + A^{(k+1)}).$$

Note-se que se $f^{(k+1)}$ se anula, $\beta^{(k)}$ e $v^{(k)}$ também se anulam e consequentemente a nova matriz $H^{(k+1)}$ também se anula (ver (31)).

3.5 Actualização das matrizes B_j

Um outro esquema de actualização de matrizes do tipo Quasi-Newton, específico dos problemas de mínimos quadrados não lineares, aproxima as matrizes $B_j(x^{(k)})$ (j = 1,..., m), que contêm as segundas derivadas das funções $f_j(x)$. Com as m matrizes $H_1^{(k)},\ldots, H_m^{(k)}$, do tipo n×n, que aproximam as Hessianas das funções f_j, j = 1,..., m, o sistema das equações fica com a forma

$$\left(\sum_{j=1}^{m} f_j^{(k)} H_j^{(k)} + A^{(k)T} A^{(k)}\right)\Delta^{(k)} = -A^{(k)T} f^{(k)}, \quad (43)$$

para o cálculo do vector $\Delta^{(k)}$, a partir do qual se calcula $x^{(k+1)} = x^{(k)} + \Delta^{(k)}$. Para k = 0, as matrizes iniciais $H_j^{(0)}$ podem ser calculadas através do uso de diferenças finitas para aproximar as segundas derivadas que compõem as matrizes $B_j(x)$, j = 1,..., m, no ponto $x^{(0)}$.

Com os novos valores de $x^{(k+1)}$, $A^{(k+1)}$ e $f^{(k+1)}$, as matrizes $H_j^{(k)}$, j = 1,..., m, são actualizadas de acordo com o seguinte esquema

$$H_j^{(k+1)} = H_j^{(k)} + [\nabla f_j^{(k+1)T} - \nabla f_j^{(k)T} - H_j^{(k)} (x^{(k+1)} - x^{(k)})]\frac{(x^{(k+1)} - x^{(k)})^{T}}{\|x^{(k+1)} - x^{(k)}\|_2^2} \quad (44)$$

em que $\nabla f_j^{(k)}$ é apenas a j-ésima linha de A que contém as n primeiras derivadas de f_j em ordem aos $x_1, x_2,\ldots, x_n$, calculada no ponto $x^{(k)}$. A fórmula (44) é uma generalização, adequada a este tipo de problema, do esquema de actualização de característica um, de Broyden, para aproximar $B_j^{(k)}$ e é devida a Brown e Dennis [2].

As propriedades de convergência local do algoritmo são caracterizadas pelos Teoremas 6 e 7.

O lema seguinte limita o erro das aproximações, às Hessianas, dadas por (44).

Lema 1: Seja D uma vizinhança convexa de x^*. Seja $\gamma \geq 0$ uma constante e ∇f um vector de funções diferenciáveis com derivada Fréchet ($B = \nabla^2 f$) de D em $\Re^n$, de modo que, para cada $x \in D$

$$\|B(x) - B(x^*)\| \leq \gamma \|x - x^*\|.$$

Seja $H^{(k)}$ uma matriz $n \times n$ e seja $x^{(k)}, x^{(k+1)} \in D$. Defina-se $H^{(k+1)}$ por

$$H^{(k+1)} \equiv H^{(k)} + [\nabla f^{(k+1)} - \nabla f^{(k)} - H^{(k)} (x^{(k+1)} - x^{(k)})] \frac{(x^{(k+1)} - x^{(k)})^T}{\|x^{(k+1)} - x^{(k)}\|^2}$$

então

$$\|H^{(k+1)} - B^{(k+1)}\| \leq \|H^{(k)} - B^{(k)}\| + 2\gamma (\|x^{(k)} - x^*\| + \|x^{(k+1)} - x^*\|).$$

Este lema estabelece a propriedade da deterioração limitada do esquema actualizador definido por (44) e significa que se a aproximação $H^{(k+1)}$, à matriz $B^{(k+1)}$, não é melhor do que a da iteração anterior, pelo menos essa deterioração é limitada superiormente.

Os dois teoremas seguintes estabelecem as condições para uma convergência local. O Teorema 7 determina ainda uma convergência quadrática mas apenas para problemas de resíduos nulos (Brown e Dennis [2]).

Teorema 6: Se x^* é um zero de $A^T f$ e $\gamma_j \geq 0$, $j = 1,\ldots, m$, são constantes tais que, para cada $x \in D$,

$$\|B_j(x) - B_j(x^*)\| \leq \gamma_j \|x - x^*\|.$$

Então, se $A(x^*)^T A(x^*)$ é uma matriz não singular, existem constantes $\delta > 0$ e $\varepsilon > 0$, tais que, se $\|x^{(0)} - x^*\| < \varepsilon$ e $\|H_j^{(0)} - B_j^{(0)}\| \leq \delta$, $j = 1,\ldots, m$, o método baseado na actualização das matrizes B_j (equações (43) e (44)) converge para x^* a partir de $x^{(0)}$.

Teorema 7: Se as hipóteses do Teorema 6 são satisfeitas e $\|f(x^*)\| = 0$, então o processo definido pelas equações (43) e (44) converge no mínimo quadraticamente.

Se for assumida a condição de continuidade mais forte

$$\|B_j(x) - B_j(y)\| \leq \gamma \|x - y\| \text{ para todo o } x \text{ e } y \in D$$

então não é necessário considerar a existência de um zero, x^*, do gradiente de S(x), isto é, basta fazer considerações sobre o comportamento da função e das suas derivadas num subconjunto convexo aberto de $\Re^n$, para que seja possível provar o "Teorema de Kantorovich" (Dennis e Schnabel [15]) para o processo iterativo (43), no qual a existência de x^* é deduzida como uma parte da prova.

4. Resultados Computacionais

Nesta secção são apresentadas tabelas com resultados computacionais. O objectivo é poder comparar a eficiência e a robustez dos métodos descritos anteriormente. Foram seleccionados

dezanove problemas dos mais conhecidos na literatura (veja-se, por exemplo, Moré, Garbow e Hillstrom [15]). Nem todos os artigos têm resultados para todos os problemas escolhidos.

As referências onde se pode encontrar a descrição dos problemas usados nos diversos artigos, os seus nomes, as siglas usadas nas tabelas 3 e 4, as dimensões dos problemas e a respectiva classificação, de acordo com o que foi referido na secção 1, são apresentados na tabela 1.

A tabela 2 descreve resumidamente, e relativamente aos trabalhos que apresentam resultados, as características da implementação e o ambiente de teste.

Os resultados apresentados em Moré, Garbow e Hillstrom [15] não são incluídos neste trabalho uma vez que os autores não especificam as características dos métodos usados e da sua implementação, tornando difícil a sua comparação.

Na globalização dos métodos e nos trabalhos onde é usada a pesquisa unidimensional, na condição de Armijo para aceitação de um escalar α, comprimento do passo, que gera um decréscimo significativo do valor da função S(x),

$$S(x^{(k)}) - S(x^{(k)} + \alpha\Delta^{(k)}) \geq -2\,\eta\alpha\, A^{(k)^T} f^{(k)},$$

o valor de η está indicado na coluna 5 da tabela 2. No critério de paragem e em todos os trabalhos analisados, é incluída a condição

$$\|\Delta^{(k)}\|/\|x^{(k+1)}\| \leq \varepsilon$$

com ε entre 10^{-8} e 10^{-4}. Alguns utilizam também uma condição que analisa o progresso da função S(x) e noutros também se inclui um teste ao gradiente $A^T f$.

Sigla	Nome	Número de variáveis, n	Número de observações, m	Resíduos	Referências
Bar	Bard	2	2	Nulos	[8] e [15]
Be	Beale	2	3	Nulos	[8] e [15]
Box	Box	3	10	Nulos	[8] e [15]
Eng	Engvall	3	5	Nulos	[8] e [15]
F e R	Freudenstein e Roth	2	2	Grandes	[8] e [15]
Har	Hartley	3	6	Grandes	[18]
J e S	Jennrich e Sampson	2	10	Grandes	[8] e [15]
K e O	Kowalik e Osborne	4	11	Pequenos	[8] e [15]
Mad	Madsen	2	3	Pequenos	[8] e [15]
M e R 1	Meyer e Roth 1	3	5	Pequenos	[18]
M e R 2	Meyer e Roth 2	3	16	Grandes	[18]
M e R 3	Meyer e Roth 3	3	23	Nulos	[18]
O 1	Osborne 1	5	33	Pequenos	[8] e [15]
O 2	Osborne 2	11	65	Pequenos	[8] e [15]
Per	Pereyra	3	13	Nulos	[18]
P S	Powell Singular	4	4	Nulos	[8],[15] e [18]
Ros	Rosenbrock	2	2	Nulos	[8],[15] e [18]
W 6	Watson 6	6	31	Pequenos	[8] e [15]
W 9	Watson 9	9	31	Pequenos	[8] e [15]

Tabela 1 - Problemas teste

Referência	Aritmética	Linguagem do código	Ambiente	Condição de Armijo	Globalização do método
Gill e Murray [8]	Precisão simples	Fortran	CDC 6500	10^{-4}	Pesquisa unidimensional Interpolação cúbica
Dennis, Gay e Welsch [4]	Precisão dupla	IBM's Forthx	IBM 370/168	-	Regiões de confiança e esquema 'dogleg'
Fernandes [6]	Precisão dupla	C++	COMPAQ Pentium 166 Mhz	10^{-5}, 10^{-4} e 10^{-1}	Pesquisa unidimensional Armijo
Wolfe [18]	Precisão dupla	Fortran	IBM 360/44	10^{-5}	Pesquisa unidimensional Armijo
Yabe e Takahashi [19]	Precisão dupla	Fortran 77	NEC PC-9801 VX	10^{-1}	Pesquisa unidimensional Armijo

Tabela 2 - Características das implementações

As tabelas 3 e 4 apresentam para diversos métodos duas quantidades. A primeira é o número de iterações e a segunda, o número de cálculos da função.

Na tabela 3 e nas colunas 2, 3, 4 e 5 são apresentados os resultados obtidos por Gill e Murray [8] pelas seguintes versões: o método de Newton tendo como base a decomposição dos valores singulares da matriz A(x) (secção 2), o método de Newton baseado nas diferenças finitas para aproximar as segundas derivadas, na matriz R, e dois métodos Quasi-Newton estruturados para aproximar a parte R da Hessiana. A coluna 4 corresponde ao esquema actualizador BFGS (fórmulas (26) e (27)) e a coluna 5 ao esquema Betts (fórmulas (26) e (30)). As colunas 6 e 7 da tabela 3 foram obtidas através da utilização da aplicação CONUM presente no trabalho de Fernandes [6]. Foram experimentados três valores de η na condição de Armijo (ver tabela 2). Em relação aos resultados das colunas 6 e 7 da tabela 3, surgiram diferenças apenas em quatro problemas e com o método Gauss-Newton, para $\eta = 10^{-1}$: Har (24/36), K e O (8/12), O1 (6/9) e Ros (11/39). A resolução do método de Newton é baseada no método directo de eliminação de Gauss com pivotagem parcial. Os resultados das três últimas colunas desta tabela foram retirados do livro de Wolfe [18]. Referem-se ao método Gauss-Newton, a uma implementação do método de Levenberg-Marquardt (subsecção 2.2) e à implementação da sequência MDLS. Veja-se em (19) e (20).

Sigla	Newton [8]	Diferenças finitas [8]	Quasi-Newton (BFGS) [8]	Quasi-Newton (Betts) [8]	Newton [6]	Gauss-Newton [6]	Gauss-Newton [18]	Levenberg-Marquardt (início:μ=0) [18]	MDLS (início:μ=0) [18]
Bar	5/6	5/7	8/9	9/11	7/8	4/5	-	-	-
Be	7/14	7/14	7/14	7/14	c)	6/7	-	-	-
Box	5/6	5/6	4/5	5/6	7/8	4/5	-	-	-
Eng	9/15	9/15	9/15	9/15	c)	9/14	-	-	-
FeR	8/18	8/24	18/27	19/31	7/8	44/45	-	-	-
Har	-	-	-	-	8/10	23/24	12/13	12/13	7/20
JeS	10/35	10/43	22/45	23/44	9/10	Overflow	-	-	-
KeO	8/11	8/21	18/21	19/23	c)	8/14	-	-	-
Mad	9/11	9/16	22/24	21/23	8/9	24/26	-	-	-
MeR1	-	-	-	-	c)	5/6	5/7	8/11	5/19
MeR2	-	-	-	-	c)	9/14 d)	c)	c)	8/26
MeR3	-	-	-	-	8/10	Singular	Singular	Singular	24/100 a)
O1	8/12	8/13	11/23	8/12	21/27	6/7	-	-	-
O2	10/18	10/32	23/32	68/135 b)	Overflow	8/14	-	-	-
Per	-	-	-	-	3/4	3/4	c)	c)	5/18
PS	13/14	13/14	13/14	13/14	13/14	8/9	-	-	-
Ros	12/31	12/31	12/31	12/31	20/28	10/34	11/34	22/30	11/34
W6	6/7	6/14	20/23	10/12	-	-	-	-	-
W9	5/6	5/6	5/6	5/6	-	-	-	-	-

Tabela 3 - Resultados computacionais

Comparação entre Newton, Gauss-Newton, do tipo Levenberg-Marquardt e Quasi-Newton

a) Início do processo com $\mu = 0.01$
b) Não atingiu a precisão desejada
c) Não convergiu em 100 iterações
d) $\varepsilon = 10^{-3}$ (não atinge a precisão desejada)

Na tabela 4 as versões BB e DGW com escalonamento referem-se aos métodos Quasi-Newton estruturados, respectivamente, com as fórmulas de actualização BB, (32), e DGW, em (33), com os respectivos factores de escalonamento. Em [19], Yabe e Takahashi introduzem a factorização modificada de Cholesky (detalhes da factorização em [18, p. 99]) no cálculo da direcção de procura sempre que A^TA+H não é definida positiva. Com a fórmula BB esta factorização foi precisa no problema K e O e com DGW foi preciso nos problemas O1 e Ros. As colunas 5, 6, 7 e 8 da tabela 4 apresentam os resultados referentes às implementações das fórmulas de actualização factorizadas (subsecção 3.3). As versões BFGS, sem e com factor de escalonamento, referem-se respectivamente às fórmulas (38) e (41). O factor utilizado é o de Biggs, com o numerador igual a $|f^{(k+1)^T} f^{(k)}|$. As versões DFP correspondem às fórmulas (40) e (42). Yabe e Takahashi ([19]) também aqui usaram o factor de escalonamento de Biggs, com o numerador positivo. Os resultados das colunas 5 e 7 foram obtidos tendo como base o vector $z^{(k)}$ dado por (36). Não incluímos neste artigo os resultados referentes à fórmula (35) por estes serem muito piores. Com a fórmula DFP, em 25% dos 16 problemas testados, Yabe e Takahashi não conseguiram convergência. Nas versões com escalonamento o $z^{(k)}$ é dado por (36). As três últimas colunas desta tabela contêm os resultados retirados do artigo de Dennis, Gay e Welsch [4] e correspondem a experiências computacionais realizadas com o código

NL2SOL que implementa a versão Quasi-Newton estruturada DGW com regiões de confiança, e com o código SUMSOL. Esta implementação é do tipo Quasi-Newton, não estruturada, e usa a fórmula geral BFGS para aproximar a inversa da Hessiana da função objectivo S(x). A penúltima coluna corresponde à versão em que a matriz inicial do processo é a matriz A^TA. A última coluna considera a matriz identidade como a primeira matriz do processo.

Não foram publicados resultados para os problemas Har, M e R1, M e R2, M e R3 e Per com os métodos referidos na tabela 4.

Sigla	Gauss-Newton [19]	BB com escalonamento [19]	DGW com escalonamento [19]	BFGS factorizada [19]	BFGS factorizada com escalonamento [19]	DFP factorizada [19]	DFP factorizada com escolonamento [19]	NL2SOL (DGW) [4]	SUMSOL $H=A^TA$ (Quasi-Newton) [4]	SUMSOL H=I (Quasi-Newton) [4]
Bar	5/24	12/53	10/45	9/41	8/37	9/41	8/37	7/7	16/20	18/22
Be	6/26	8/33	9/36	9/39	8/36	11/42	8/33	8/10	17/21	14/17
Box	4/20	6/28	5/24	5/24	5/24	5/24	5/24	7/7	15/16	35/48
Eng	-	-	-	-	-	-	-	15/17	30/35	30/33
FeR	c)	6/21	6/21	7/31	7/31	6/21	6/21	9/9	9/9	11/13
JeS	c)	9/32	9/32	11/57	9/64	89/272	9/32	12/16	14/16	22/34
KeO	19/103	10/59	12/70	10/61	9/56	10/60	9/55	12/14	19/27	33/42
Mad	-	-	-	-	-	-	-	12/12	15/15	16/16
O1	6/44	27/172	21/148	27/181	18/128	57/349	139/843	26/34	42/56	59/83
O2	9/125	13/171	12/159	24/310	15/200	41/507	13/170	13/15	32/37	59/75
PS	9/50	14/75	14/75	14/75	14/75	14/75	14/75	20/20	45/45	75/80
Ros	11/62	20/73	17/87	22/78	13/48	54/180	19/68	18/22	37/50	36/40
W6	6/49	9/70	9/71	15/117	14/114	27/196	8/63	8/8	21/25	34/41
W9	80/810	82/830	82/831	28/295	26/280	47/480	23/240	9/10	22/22	72/81

Tabela 4 - Resultados computacionais

Comparação entre Gauss-Newton, Quasi-Newton e Quasi-Newton estruturados

c) Não convergiu em 100 iterações

Se compararmos as duas implementações do método de Newton, pode concluir-se que a versão baseada na decomposição dos valores singulares é mais robusta, principalmente porque se introduz a decomposição modificada Cholesky no algoritmo. A matriz dos coeficientes do sistema Newton não é necessariamente definida positiva e é por isso que a resolução directa do sistema é pouco robusta. Quando se introduzem diferenças finitas para aproximar as segundas derivadas, o desempenho do algoritmo não é afectado, embora haja um aumento muito significativo de cálculos da matriz A.

As implementações Gauss-Newton são eficientes. No entanto, quando o problema é de grandes resíduos podem falhar. A singularidade da matriz A^TA é também um inconveniente. Os algoritmos do tipo Levenberg-Marquardt respondem razoavelmente bem a estas questões, desde que o valor inicial a atribuir ao parâmetro μ não seja nulo.

Se compararmos os resultados apresentados nas tabelas com os métodos do tipo Quasi-Newton, aqueles que têm em consideração a estrutura especial do problema em estudo, conhecidos por Quasi-Newton estruturados, são os mais eficientes. Veja-se na tabela 5 um

resumo referente aos números médios de iterações e de cálculos da função obtidos pelos métodos Quasi-Newton. No entanto, os Quasi-Newton estruturados são mais sensíveis à técnica de procura unidimensional utilizada. Enquanto que o trabalho de Gill e Murray [8] apresenta valores aceitáveis de cálculos da função e é baseado na condição de Armijo com $\eta = 10^{-4}$ (colunas 2 e 3 da tabela 5), no trabalho de Yabe e Takahashi [9] constata-se que esses números chegam a atingir valores exagerados (colunas 4, 5, 6, 7, 8 e 9 da tabela 5). Na condição de Armijo, Yabe e Takahashi usam $\eta = 10^{-1}$. Um valor desta ordem pode usar-se se daí resultar uma diminuição significativa do número de iterações. O que neste caso não se verifica. A implementação NL2SOL ([4]) é, entre os métodos Quasi-Newton estruturados, a que apresenta melhores resultados, no sentido de que necessita de menos iterações e também de menos cálculos da função.

Nas actualizações factorizadas pode concluir-se que a introdução do factor de escalonamento traz vantagens (compare-se as colunas 6 e 8 respectivamente com 7 e 9 da tabela 5). Diminuem o número de iterações e o número de cálculos da função. As diferenças são menos significativas (em média uma redução de 20%) quando a fórmula de actualização do tipo BFGS é a escolhida. Com a fórmula DFP há uma redução de 30% em média. E, entre as fórmulas DFP e BFGS, tal como em optimização geral, a segunda é mais eficiente.

Nas actualizações factorizadas pode concluir-se que a introdução do factor de escalonamento traz vantagens (compare-se as colunas 6 e 8 com 7 e 9 da tabela 5). Diminuem o número de iterações e o número de cálculos da função. As diferenças são menos significativas (em média uma redução de 20%) quando a fórmula de actualização do tipo BFGS é a escolhida. Com a fórmula DFP há uma redução de 30% em média. E, entre as fórmulas DFP e BFGS, tal como em optimização geral, a segunda é mais eficiente.

	QNE (BFGS) [8]	QNE (Betts) [8]	QNE (BB) c/e [19]	QNE (DGW) c/e [19]	QNEF (BFGS) [19]	QNEF (BFGS) c/e [19]	QNEF (DPF) [19]	QNEF (DPF) c/e [19]	QNE NL2SOL (DGW) [4]	QN SUMSOL $H=A^TA$ [4]	QN SUMSOL H=I [4]
nmi	14	16	18	17	15	12	31	22	13	24	37
nmcf	21	27	135	133	109	91	187	138	14	28	45

Tabela 5 - Número médio de iterações e de cálculos da função dos métodos Quasi-Newton[1]

nmi - número médio de iterações
nmcf - número médio de cálculos da função
QN - Quasi-Newton
QNE - Quasi-Newton estruturado
QNEF - Quasi-Newton estruturado e factorizado
c/e - com factor de escalonamento

[1] Só foram considerados nestes cálculos os problemas efectivamente resolvidos pelos respectivos métodos

5. Comentários Finais

Os métodos disponíveis para a resolução de problemas de mínimos quadrados não lineares podem ser agrupados em três classes. A primeira classe envolve o método de Newton e aproximações baseadas em diferenças finitas, a segunda classe envolve o método de Gauss-Newton e as variantes do método de Levenberg-Marquardt e o último grupo envolve os métodos Quasi-Newton. Pretendia-se também com este artigo analisar a eficiência e a robustez dos métodos Quasi-Newton quando comparados com as variantes do método de Newton. Do estudo realizado pode concluir-se que os métodos Quasi-Newton estruturados trazem vantagens quando comparados com os métodos Quasi-Newton mais gerais. A implementação da técnica de globalização baseada em regiões de confiança parece ser a mais aconselhada. No entanto, com as fórmulas factorizadas, que garantem matrizes definidas positivas e consequentemente direcções de procura de descida, a globalização dos algoritmos pode ser feita de uma maneira mais simples, recorrendo, por exemplo, a critérios do tipo Armijo. Também se aconselha a introdução de um factor de escalonamento. Este trabalho já originou, entretanto, um estudo onde foram deduzidas novas fórmulas de actualização para as matrizes factorizadas de métodos Quasi-Newton estruturados. Os resultados que serão obtidos com as novas fórmulas serão comparados com os já existentes e apresentados noutro artigo.

Para finalizar resta-nos comentar um pouco sobre a questão da convergência. Do estudo conclui-se que se obtém convergência quadrática local com o método de Newton se a matriz do Jacobiano do sistema for Lipschitz contínua numa vizinhança da solução, se for não singular e a sua inversa for limitada superiormente na solução. Também é possível conseguir-se uma convergência quadrática e local para o método de Gauss-Newton, mas só no caso do problema ser de resíduos nulos.

Para o método Levenberg-Marquardt e para aproximações iniciais próximas da solução e na resolução de problemas de resíduos nulos, o Teorema 3, estabelece condições para a convergência quadrática. A condição mais importante exige que a sequência de parâmetros $\mu^{(k)}$, tenda para zero, à medida que nos aproximamos do ponto estacionário da função $S(x)$.

O Teorema 4 e o Corolário 2 estabelecem condições para que o método que define uma sequência MDLS tenha convergência superlinear para um mínimo isolado de $S(x)$ e estas são válidas apenas para problemas de resíduos nulos. A convergência é estabelecida apenas nos problemas em que o valor próprio de menor módulo da matriz A^TA é maior do que o raio espectral da matriz R.

Para os métodos Quasi-Newton estruturados baseados na factorização de J, é possível obter uma convergência local e superlinear, segundo o que é estabelecido no Teorema 5 e Corolário 3 como já foi referido em 3.3, mas é necessário que a sequência de factorizações seja uniformemente limitada.

Quando cada matriz a actualizar aproxima uma das matrizes B_j, das segundas derivadas das funções f_j, o método Quasi-Newton resultante pode apresentar uma convergência local e

quadrática se o problema for de resíduos nulos. A proximidade das matrizes de aproximação $H^{(0)}$, em relação às $B_j^{(0)}$, bem como a condição das matrizes $B_j(x)$ serem Lipschitz contínuas são necessárias para que haja convergência para um ponto estacionário de S (veja-se os Teoremas 6 e 7).

Agradecimentos

As autoras gostariam de agradecer ao revisor as críticas feitas e as sugestões apresentadas à primeira versão do artigo.

Referências

[1] Bartels, R.H., The Use of Factorizations in Derivative - Free Nonlinear Least Squares Algoritmos, em O.L. Mangasarian, R.R. Meyer e S.M. Robinson, (Eds.), Nonlinear Programming 2 (1975) 231-253, Academic Press.

[2] Brown, K.M. and Dennis, Jr., J.E., A New Algorithm for Nonlinear Least-Square Curve Fitting, em J.R. Rice (Ed.), Mathematical Software (1971) 391-396, Academic Press.

[3] Dennis, Jr., J.E. and Moré, J.J., A Characterization of Superlinear Convergence and its Application to Quasi-Newton Methods, Mathematics of Computation 28 (1974) 549-560.

[4] Dennis, Jr., J.E., Gayd, D.M. and Welsch, R.E., An Adaptive Nonlinear Least Squares Algorithm, ACM Transactions on Mathematical Software 7 (1981) 348-368.

[5] Dennis, Jr., J.E. and Schnabel, R.B., Numerical Methods for Unconstrained Optimization and Nonlinear Equations, Prentice-Hall, Inc. (1983).

[6] Fernandes, E.M.G.P., Computação Numérica, 2ª edição, Universidade do Minho, Braga (1998).

[7] Fletcher, R., Modified Marquardt Subroutine for Non-Linear Least Squares, Theoretical Physics Division, Atomic Energy Research Establishment, Harwell, Berkshire, Report AERE-R 6799 (1971).

[8] Gill, P.E. and Murray, W., Algorithms for the Solution of the Non-Linear Least-Squares Problem, Division of Numerical Analysis and Computing, National Physical Laboratory, Teddington, Middlesex, NPL Report NAC 71 (1976).

[9] Gulliksson, M. and Soderkvist, I., Surface Fitting and Parameter Estimation with Nonlinear Least-Squares, Optimization Methods and Software 5 (1995) 247-269.

[10] Hager, W.W., Applied Numerical Linear Algebra, Prentice-Hall (1988).

[11] Marquardt, D.W., An Algorithm for Least-Squares Estimation of Nonlinear Parameters, J. Soc. Industry and Appl. Mathematics 11 (1963) 431-441.

[12] McKeown, J.J., Experience with Peckham's Algorithm for Minimising Sums of Squared Functions, Numerical Optimisation Centre, The Hatfield Polytechnic, Technical Report 29 (1972).

[13] McKeown, J.J., Specialised versus General-Purpose Algorithms for Minimising Functions that are Sums of Squared Terms, Numerical Optimisation Centre, The Hatfield Polytechnic, Technical Report 50 (1974).

[14] Moré, J.J., The Levenberg-Marquardt Algorithm: Implementation and Theory, em G.A. Watson (Ed.), Numerical Analysis, Lecture Notes in Mathematics 630 (1978) 105-116, Springer-Verlag.

[15] Moré, J.J., Garbow, B.S. and Hillstrom, R.E., Testing Unconstrained Optimization Software, ACM Transactions on Mathematical Software 7 (1981) 17-41.

[16] Nocedal, J., Theory of Algorithms for Unconstrained Optimization, Acta Numerica (1992) 199-242, Cambridge University Press.

[17] Ramadas, G.C.G.V., Problemas de Mínimos Quadrados Não Lineares. Resolução de Algumas Questões Numéricas, Tese de Mestrado, Universidade do Minho, Braga (1997).

[18] Wolfe, M.A., Numerical Methods for Unconstrained Optimization, Van Nostrand Reinhold Company (1978).

[19] Yabe, H. and Takahashi, T. Factorized Quasi-Newton Methods for Nonlinear Least-Squares Problems, Mathematical Programming 51 (1991) 75-100.

[20] Yabe, H., Variations of Structured Broyden Families for Nonlinear Least Squares Problems, Optimization Methods and Software 2 (1993) 107-144.

INSTRUÇÕES AOS AUTORES

Os autores que desejam submeter um artigo à Investigação Operacional devem enviar três cópias desse trabalho para:

Prof. Joaquim J. Júdice
Departamento de Matemática
Universidade de Coimbra
3000 Coimbra, Portugal

Os artigos devem ser escritos em Português ou Inglês. A primeira página deve conter a seguinte informação:

- Título do artigo
  - Autor(es) e instituição(ões) a que pertence(em)
  - Abstract (em inglês)
  - Resumo
  - Keywords (em inglês)
  - Título abreviado

As figuras devem aparecer em separado de modo a poderem ser reduzidas e fotocopiadas. As referências devem ser numeradas consecutivamente e aparecer por ordem alfabética de acordo com os seguintes formatos:

Artigos: autor(es), título, título e número da revista (livro com indicação dos editores), ano, páginas.

Livros: autor(es), título, editorial, local de edição, ano.

ÍNDICE

Apdio

CESUR - Instituto Superior Técnico
Av. Rovisco Pais - 1049 - 001 LISBOA
Telef. 21 840 74 55 - Fax. 21 840 98 84
http://www.inescc.pt/~apdio