4OR 4: 47–59 (2006)

DOI: 10.1007/s10288-005-0074-x

Martins' algorithm revisited for multi-objective shortest path problems with a MaxMin cost function

Xavier Gandibleux[1,2], **Frédéric Beugnies**[2], **and Sabine Randriamasy**[3]

[1] LINA – FRE CNRS 2729, Université de Nantes, 2 rue de la Houssiniere BP 92208, 44322 Nantes cedex 03, France (e-mail: Xavier.Gandibleux@univ-nantes.fr)
[2] LAMIH-ROI – UMR CNRS 8530, Université de Valenciennes, Campus "Le Mont Houy", 59313 Valenciennes cedex 9, France (e-mail: Frederic.Beugnies@univ-valenciennes.fr)
[3] ALCATEL R&I - ISR project, Route de Nozay, 91460 Marcoussis, France (e-mail: Sabine.Randriamasy@alcatel.fr)

Received: February 2005 / Revised version: June 2005

Abstract. This paper presents a direct extension of the label setting algorithm proposed by Martins in 1984 for the shortest path problem with multiple objectives. This extended version computes all the efficient paths from a given source vertex, to all the other vertices of the network. The algorithm copes with problems in which the "cost values" associated with the network arcs are positive. The proposed extension can handle objective functions that are either of the "sum" type or of the "bottleneck" type. The main modifications to Martins' algorithm for multi-objective shortest path problems are linked to the dominance test and the procedure for identifying efficient paths. The algorithmic features are described and a didactic example is provided to illustrate the working principle. The results of numerical experiments concerning the number of efficient solutions produced and the CPU time consumed for several configurations of objectives, on a set of randomly generated networks, are also provided.

Key words: Multi-objective combinatorial optimization – shortest path problem – labelling algorithm.

MSC classification: 90C29, 90C27, 05C38, 90B18, 68M12

1 Introduction

The Multi-Objective Shortest Path (MOSP) problem is one of the most intensively studied problems in Multi-Objective Combinatorial Optimization (Ehrgott

and Gandibleux 2002). Two types of objectives formulated by a linear function or a max-min function are usually considered. These objectives correspond respectively to the "sum" problem and the "bottleneck" problem, which are denoted in the general case as ($\sigma-\mathrm{S}\,|\,\mu-\mathrm{M}$), meaning that a situation concerns respectively σ objectives of the first type and μ objectives of the second. For example, cost or delay are *S-type* objectives to be minimized, and quality or bandwidth are *M-type* ones to be maximized.

There are two principal classes of algorithms for solving multi-objective shortest path problems with linear functions: labelling-based algorithms and ranking path-based algorithms. The former may be split into two main families: label setting algorithms (see Hansen 1979; Martins 1984a) in which one label is set permanently at each iteration, and label correcting algorithms (see Brumbaugh-Smith and Shier 1989; Mote et al. 1991; Skriver and Andersen 2000) in which all the labels become permanent only at the last iteration. Ranking-based techniques, like those proposed in Clímaco and Martins (1982), use a k-shortest path routine to solve the problem. In addition, specific network problems may be solved by using dynamic programming-based algorithms (see Henig 1985; Kostreva and Wiecek 1993; Sniedovich 1988).

When all the objectives lead to sum problems ($\sigma-\mathrm{S}$) only, the set of efficient paths can, under assumptions concerning the cost values, be computed by applying Martins' label setting algorithm. In this paper, we propose an extension of Martins' algorithm for solving ($\sigma-\mathrm{S}\,|\,1-\mathrm{M}$)-class multi-objective shortest path problems. Several real-world contexts are addressed by this ($\sigma-\mathrm{S}\,|\,1-\mathrm{M}$) problem. For example, Internet traffic routing could be enhanced if based on a multi-objective routing procedure which would prevent network congestion. MOSP between one router and all the other routers of the network must be computed in real-time, by simultaneously optimizing ($\sigma-\mathrm{S}\,|\,1-\mathrm{M}$) (Randriamasy et al. 2002; Randriamasy and Gandibleux 2003). The routing operation includes a procedure for computing the MOSPs. In the context of vehicle routing, selecting a route in a transportation network, according to the objectives of capacity, distance or time, is the same type of problem (Moore et al. 1978).

This paper is organized as follows. Section 2 provides the mathematical background (definitions and notation) and recalls the principles of the Martins label setting algorithm. Section 3 describes our revision of this algorithm for ($\sigma-\mathrm{S}\,|\,1-\mathrm{M}$) shortest path problems. In Sect. 4, the results of numerical experiments on a set of randomly generated networks are reported and discussed. Finally, Sect. 5 presents our conclusions and suggests several directions for future research.

2 Mathematical background

2.1 Fundamental definitions

A *network* is defined as a directed and connected *graph* $G = (V, A)$, where $V = \{1, \ldots, n\}$ is the set of *vertices* with cardinality $|V| = n$ and $A =$

$\{(i_1, j_1), \ldots, (i_m, j_m)\}$ is the set of *arcs* with cardinality $|A| = m$. The directed arc linking vertices i and j is denoted by (i, j), and the vector $(c_1(i, j), \ldots, c_k(i, j))$ represents the "cost values" associated with the arc (i, j). C refers to the *cost matrix* for all arcs (i, j) in G. In the set V, we identify a *source vertex* s and a *sink vertex* t. A *path* r from s to t in G is a sequence of arcs and vertices from s to t, where the tail vertex of a given arc coincides with the head vertex of the next arc in the path. The *decision space* is denoted by $R_{s,t}$, the set of all paths from s to t in G, or by $R_{s,\bullet}$ the set of all the paths from s to all others vertices $V \setminus \{s\}$ in G. Let $z_p(r)$ denote the value of a path r with respect to the objective p, for $p = 1, \ldots, k$, with $k = \sigma + \mu$, the number of objectives. Hence, the vector $z(r) = (z_1(r), \ldots, z_k(r))$ represents the *performance vector* of path $r \in R_{s,t}$ in *the objective space* $Z = z(R_{s,t})$. The formulation of the objectives is $z_p(r) = \sum_{(i,j)\in r} c_p(i, j)$ for the linear function, and $z_p(r) = \min_{(i,j)\in r} c_p(i, j)$ for the max-min function. These functions correspond respectively to the $(1-\mathrm{S})$ problem $\min \sum_{(i,j)\in r} c_p(i, j)$, and the $(1-\mathrm{M})$ problem $\max \min_{(i,j)\in r} c_p(i, j)$.

Let r^1, r^2 be two paths, and $z^1 = z(r^1), z^2 = z(r^2)$ the performance vectors. When all the objectives are to be minimized, z^1 *dominates* z^2 iff $z_p^1(r^1) \leq z_p^2(r^2),\ p = 1, \ldots, k$, and $z^1 \neq z^2$. z is *non-dominated* iff $z \in Z$ and $\nexists z' \in Z$ such that z' dominates z. A path r^e is *efficient* iff $r^e \in R_{s,t}$ and $\nexists r \in R_{s,t}$ such that $z(r)$ dominates $z(r^e)$. When r^e is an efficient solution then $z(r^e)$ is *a non-dominated vector*. When a path is not efficient, it is dominated. Also, z^1 *dominates strictly* z^2 iff $z_p^1(r^1) < z_p^2(r^2),\ p = 1, \ldots, k$. According to this definition of dominance, a path r^e is *weakly efficient* iff $r^e \in R_{s,t}$ and $\nexists x \in R_{s,t}$ such that $z(x) < z(r^e)$. When r^e is a weakly efficient solution then $z(r^e)$ is *a weakly non-dominated vector*. When a path is not weakly efficient, it is dominated.

2.2 Multi-objective shortest paths

The multi-objective shortest path problem is a well-known NP-hard (Serafini 1986) MOCO problem. The version considered in this paper can be summarized as:

$$\text{“}\min_{r\in R_{s,t}}\text{”}(z_1(r), \ldots, z_k(r)) \quad \forall t \in V \setminus \{s\} \tag{MOSP}$$

Let $E(R_{s,\bullet})$ denote the *maximal complete set* of efficient paths in $R_{s,\bullet}$ for a given source vertex s, i.e. the set of all paths of $R_{s,\bullet}$ in which each path corresponds to a non-dominated point. Several distinct efficient paths r^1, r^2, r^3 can correspond to the same non-dominated point $z(r^1) = z(r^2) = z(r^3)$ in the objective space. In this case, the paths r^1, r^2, r^3 are said to be *equivalent* in the objective space (Hansen 1979). The *minimal complete set* of efficient paths is a subset of $E(R_{s,\bullet})$ that contains no equivalent solutions, and for any $r \in E(R_{s,\bullet})$, there exists r' in the minimal complete set, such that r and r' are equivalent.

Hansen (1979) studied ten shortest path problems with two objectives. In that paper, the author reported the following important results:

- Minimizing or maximizing a (1−M) problem is equivalent (one problem can be transformed to the other). Thus, (1−M) problem can be considered either as max min or min max.
- For the (2−S) problem, the number of solutions $|E(R_{s,t})|$ is equal to $2^{|A|}$ for some specific networks.
- An algorithm yielding a minimal complete set of efficient paths $E(R_{s,t})$ for a given t for the (1−S | 1−M) problem in $O(m^2 \log n)$ operations is provided. This problem is also studied in Martins (1984b).

In terms of complexity, the enumeration of all efficient paths in a multi-objective shortest path problem may not be tractable in polynomial time. However, the example considered in Hansen makes use of exponential arc values. In practice, arc values are far from that specific case, and the number of efficient paths is generally not exponential as illustrated in Sect. 4 (which provides experimental results on the number of efficient paths collected for randomly generated networks). Martins (1984a) proposed a label setting algorithm for computing the maximal complete set of efficient paths when all the cost values are non negative ($c_p(i, j) \geq 0$ for all $(i, j) \in A$, and $p = 1, \ldots, k$) with at least one strict inequality holding for some value of p. It is a generalisation for (σ −S) objectives of the algorithm presented by Hansen in 1979.

The Martins algorithm is a multiple objective extension of the Dijkstra algorithm in which the 'min' operator is replaced by a dominance test. This modification is valid since the Bellman principle of optimality can be applied to the multiple objective path case: for the networks being considered, any subpath of an efficient path is an efficient subpath. Martins' algorithm ensures the computation of the maximal complete set of efficient paths from one vertex to all the other vertices of a network.

The main idea of Martins algorithm is quite simple. At each iteration, for each vertex, there are two different sets of labels to qualify paths: *permanent* labels and *temporary* labels. The algorithm selects the lexicographically smallest label from all the sets of temporary labels, converts it to a permanent label, and propagates the information contained in this label to all the temporary labels of its successors. The procedure stops when there are no more temporary labels. Each permanent label corresponds to a unique efficient path. A proof of the validity of the algorithm is given in Ehrgott (2000), p. 171.

The characteristics of Martins' algorithm have motivated us to extend this algorithm to simultaneously optimise S-type and M-type objectives. S-type objectives are to be minimized and M-type ones are to be maximized in the continuation. The revised version presented in the next section deals with (σ −S | μ −M) in which $\sigma \geq 1$ and $\mu = 1$, and computes *the maximal complete set* of efficient paths $E(R_{s,\bullet})$.

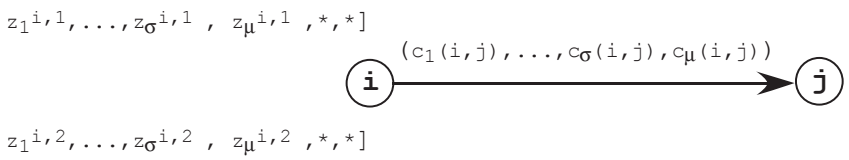

Fig. 1. View of two labels on a vertex

3 The revised Martins algorithm for (σ − S | 1 − M)

The main change concerns the dominance test for ensuring that the maximal complete set of efficient paths is computed for (σ−S | 1−M). For short, the indices noted as $1, \ldots, \sigma$ refer to the S-type objectives and μ refers to the M-type objective in the following description.

A label l^j can be represented as $l^j = [\ z_1, \ldots, z_k,\ i,\ h\ ]$ where $(z_1, \ldots, z_k)$ is the performance vector extracted; i is the preceding adjacent vertex from which it was possible to label the vertex j; and h is the position of the label in the list of labels on vertex i. Let us focus on a vertex i connected to a successor vertex j (Fig. 1).

The costs of the arc (i, j) are given by $(c_1(i, j), \ldots, c_\sigma(i, j), c_\mu(i, j))$. Consider two temporary labels $[\ z_1^{i,h}, \ldots, z_\sigma^{i,h}, z_\mu^{i,h},\ *,\ *\ ]$ where $h = 1, 2$ on vertex i. When $z_p^{i,1} = z_p^{i,2}\ \forall p = 1, \ldots, \sigma$ and $z_\mu^{i,1} > z_\mu^{i,2}$, label 2 is weakly non-dominated by label 1. According to the dominance test in Martins algorithm, label 2 should be deleted. Now, consider these two labels on vertex j according to the value of $c_\mu(i, j)$. If $c_\mu(i, j) \leq \min(z_\mu^{i,1}, z_\mu^{i,2})$, then the two labels become equivalent on vertex j and thus, label 2 must be kept; else, label 2 remains weakly non-dominated on vertex j. Thus, the dominance test used in the original version of the algorithm cannot find all the efficient paths for this configuration of objectives. To do so, a modification is necessary to handle this case.

Table 1 recapitulates the five situations to consider using a $(2 - \text{S} \mid 1 - \text{M})$ problem as illustration. For a given objective, let us denote by symbols $\{+, =, -\}$, which mean respectively 'better', 'equal' and 'worse', the result of the comparison between z^1 and z^2, the vectors extracted from two labels l^1, l^2.

Because some weakly non-dominated labels (situation C) can be made permanent, not every permanent label corresponds to an efficient path when the labelling phase is completed. However these weakly non-dominated labels cannot be deleted because they contribute to the determination of the efficient paths. Such a label can correspond to an intermediate vertex for an efficient path; however, it cannot be a sink vertex of an efficient path because, by definition, there is another label on this vertex with a better performance. Weakly non-dominated labels (crossed out twice in Fig. 2) are thus *hidden* in the list of permanent labels that will be used for the construction procedure as a sink vertex of an efficient path. The others remain *visible* and correspond to individual efficient paths.

Table 1. In A, z^1 is better than z^2 at least on one objective. In B, the vectors are equivalent for all objectives. In C, according to the value of $c_\mu(i, j)$, these two labels can become equivalent on vertex j. In D, under the assumptions of non-negative costs, these two labels cannot become equivalent in the next iterations. Finally, in E, z^1 is worse than z^2 for all objectives

p=1	p=2	p=3	Situation	l^1 is deleted
+	+, =, −	+, =, −	A	no
+, =, −	+	+, =, −	A	no
+, =, −	+, =, −	+	A	no
=	=	=	B	no
=	=	−	C	no
−	=, −	=, −	D	yes
=, −	−	=, −	D	yes
−	−	−	E	yes

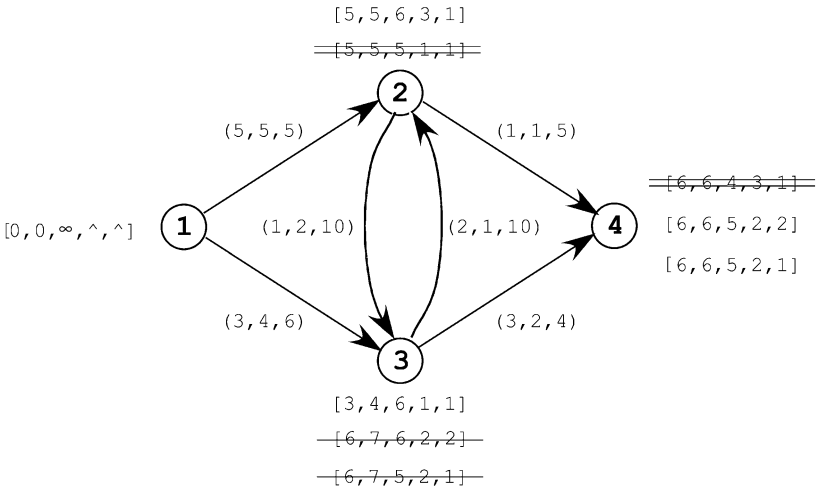

Fig. 2. Example of (2−S | 1−M). The deleted labels are crossed out. The labels that have been crossed out twice correspond to the hidden labels for the determination of a sink vertex of an efficient path. Paths $r^e \in E(R_{1,t})$ are, for $t = 2$: the path $1 \to 3 \to 2$ with $z(r^e)$=(5, 5, 6); for $t = 3$: the path $1 \to 3$ with $z(r^e)$=(3, 4, 6); for $t = 4$: the paths $1 \to 2 \to 4$ and $1 \to 3 \to 2 \to 4$, both with performances $z(r^e)$ equal to (6, 6, 5)

Thus, to determine any efficient paths, choose one visible permanent label on a vertex j and extract the values corresponding to i and h for this label. So, i is the vertex just before j in the efficient path. To determine the label on vertex i that has produced this current path, the value of h is needed. This value indicates the hth label on vertex i that has produced the current path. By moving backwards, the first vertex of the path (s) will be reached. Figure 2 illustrates the shortest paths for (2−S | 1−M) from source vertex $s = 1$ to sink vertices $t = \{2, 3, 4\}$.

1. The temporary label $[0, 0, \infty, \perp, \perp]$, assigned to vertex $s = 1$, is selected and becomes permanent. Two temporary labels are computed: [5, 5, 5, 1, 1] for vertex 2 and [3, 4, 6, 1, 1] for vertex 3.
2. Of these temporary labels, the smallest lexicographically is [3, 4, 6, 1, 1]. It is selected and becomes permanent. The vertex 3 successors are labelled, yielding temporary labels [5, 5, 6, 3, 1] for vertex 2 and [6, 6, 4, 3, 1] for vertex 4. Label [5, 5, 5, 1, 1] on vertex 2 is weakly non-dominated by [5, 5, 6, 3, 1], case where performances on the sum objectives are equal. Therefore, this label is not deleted.
3. [5, 5, 6, 3, 1] is the next temporary label selected to become permanent. The vertex 2 successors are labelled, yielding temporary labels: [6, 7, 6, 2, 2] for vertex 3 and [6, 6, 5, 2, 2] for vertex 4. The new temporary label for vertex 3 is weakly non-dominated by permanent label [3, 4, 6, 1, 1]. This situation corresponds to the case D presented earlier. Consequently, label [6, 7, 6, 2, 2] is deleted.
4. In the next iteration, [5, 5, 5, 1, 1] is the temporary label selected and made permanent. The vertex 2 successors are labelled, yielding two temporary labels: [6, 7, 5, 2, 1] on vertex 3 and [6, 6, 5, 2, 1] on vertex 4. The new temporary label on vertex 3 is dominated by [3, 4, 6, 1, 1] (case E), and consequently, is deleted.
5. [6, 6, 5, 2, 1] and [6, 6, 5, 2, 2] are lexicographically the smallest temporary labels. Assuming that [6, 6, 5, 2, 1] is selected and becomes permanent, vertex 4 has no successor, and thus no new temporary label is created.
6. The list now contains only a single temporary label, [6, 6, 5, 2, 2]. This label is selected and becomes permanent. Again, vertex 4 has no successor, and no new temporary label is created. The list of temporary labels now being empty, the labelling phase is concluded.

To summarize, the revision affects three aspects of the original algorithm. First, it replaces the value 0 in the initial label with ∞ for the corresponding max$-$min objective. Second, it extends the dominance test on labels, and third, it disables the weakly non-dominated permanent labels for the determination of efficient paths at the end of the labelling phase. The following section presents the numerical experiments.

4 Numerical experiments

4.1 Experimental conditions

The aim of these experiments is to provide feedback on the number of efficient solutions observed and the CPU time required in practice, for selected configurations of randomly generated networks. The networks are generated for a given number of vertices n and a given density d (with $d = 100 \times m/(n \times (n-1))$). In all networks, each node can be reached from the source node. The network topology is

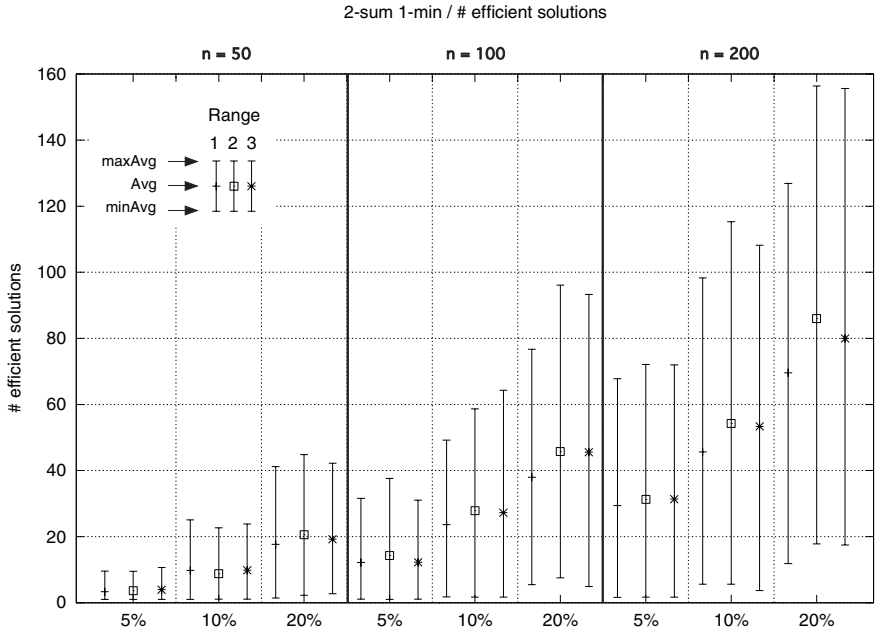

Fig. 3. Number of efficient paths observed for the (2−S | 1−M) configuration

built as follows: first a rooted tree is elaborated, and then, arcs are added randomly to the tree in order to reach the desired density.

Twenty-seven parameter configurations are selected for generating the random networks: (i) the number of vertices: 50, 100, and 200; (ii) the density of the network: 5%, 10%, and 20%; (iii) three classes of ranges for cost values $c_p(i, j)$:

- range 1: [1, 100] for $p = 1, 2, 3$
- range 2: [1, 1000] for $p = 1, 2, 3$
- range 3: [1, 255] for $p = 1$; [1, 50000] for $p = 2$; [1, 1000000] for $p = 3$

Ten numerical instances are generated for each configuration. Four configurations of objectives are considered for computing the efficient paths: (2−S), (1−S | 1−M), (3 − S), and (2 − S | 1 − M), with all S-type objectives minimized and the M-type maximized. For each configuration of objectives, the maximal complete set of efficient paths from node $s = 1$ to all other network vertices is computed. For each network configuration, the experiments yield results on: (i) the CPU time (minimum/avg/maximum); (ii) the number of efficient paths observed (avg); for the (2−S | 1−M) objective configuration, (iii) the number of efficient paths observed, in regard to the node with the minimum number of solutions (min avg), and (iv) the number of efficient paths observed, in regard to the node with the maximum number of solutions (max avg).

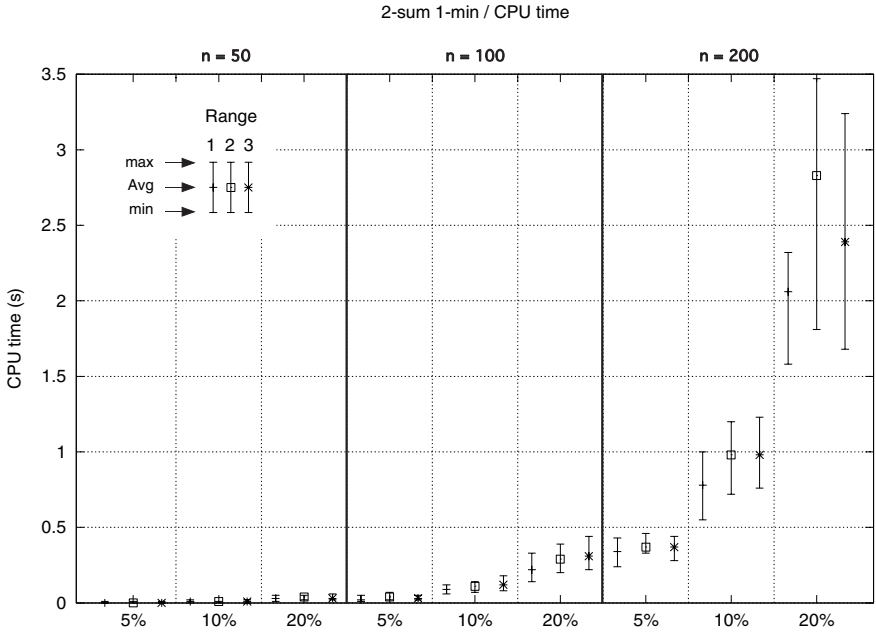

Fig. 4. CPU time comsumed for the $(2-\mathrm{S} \mid 1-\mathrm{M})$ configuration

The computer used for the experiments is a laptop equipped with a PowerPC G4 1GHz processor, 512 Mb of RAM memory installed, running under the MacOSX 10.3.2 operating system. The algorithms are implemented in C. The binary code is obtained using the gcc-3.3 compiler without any compiler optimization.

4.2 Results and discussion

Figures 3 and 4 present, respectively, the number of efficient paths and the CPU time observed for the (2–S | 1–M) configuration. We found no significant difference between the three value range sets used in the experiment. Our algorithm does not seem to be sensitive to the ranges.

The number of efficient paths grows with the density and network size. In general, the average minimum number of efficient paths is low. Thus, there are apparently some vertices in the network where few efficient backup paths are available from s to these vertices. On the other hand, the maximum number is generally high, almost twice the average value. Thus, there are also several vertices with a huge number of backup paths. Consequently, it is not surprising to observe that more CPU time is needed in large networks with a high density of arcs.

Figures 5 and 6 summarize the average values of the efficient paths and the CPU time observed for the four configurations of objectives. Remember that all S-type objectives were minimized and the M-type objective was maximized in

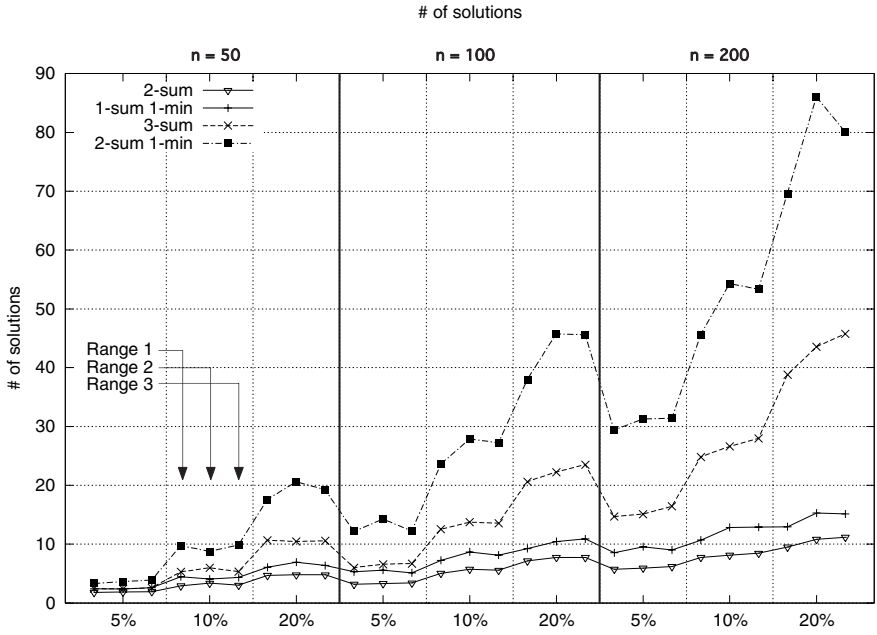

Fig. 5. Average number of efficient paths observed for all configurations

these experiments. For the configurations with two or three objectives, plus one M-type objective enabled, the number of efficient paths is generally greater than the configurations with only S-type objectives. In examining the 1080 resolutions performed, it appears that this is true for 96,30 % of the configurations.

Compared to the $(2-\mathrm{S}\,|\,1-\mathrm{M})$ configuration, all the other configurations need less CPU time. The expected use of this algorithm in real world problems related to computer communication, involves a target configuration for the network of 100 vertices with a density of less than 20 %, and typical cost values in "range 3". To be used in this environment, the algorithm must be able to produce solutions in less than one second, using a conventional computer. Given that the most pessimistic measurements are far under this temporal limit, the results shown in Figs. 4 and 6 appear very promising. Thus, the algorithm satisfies the ressource constraints of the real world problem, for all of the configurations in the experiment.

5 Conclusion

We have presented a revised version of the Martins algorithm for multi-objective shortest path problems with $(\sigma-\mathrm{S}\,|\,1-\mathrm{M})$ objectives. Our revision concerns the initial label, the dominance test for labels during the labelling procedure, and the extraction procedure for the labelled efficient paths. Numerical experiments allowed

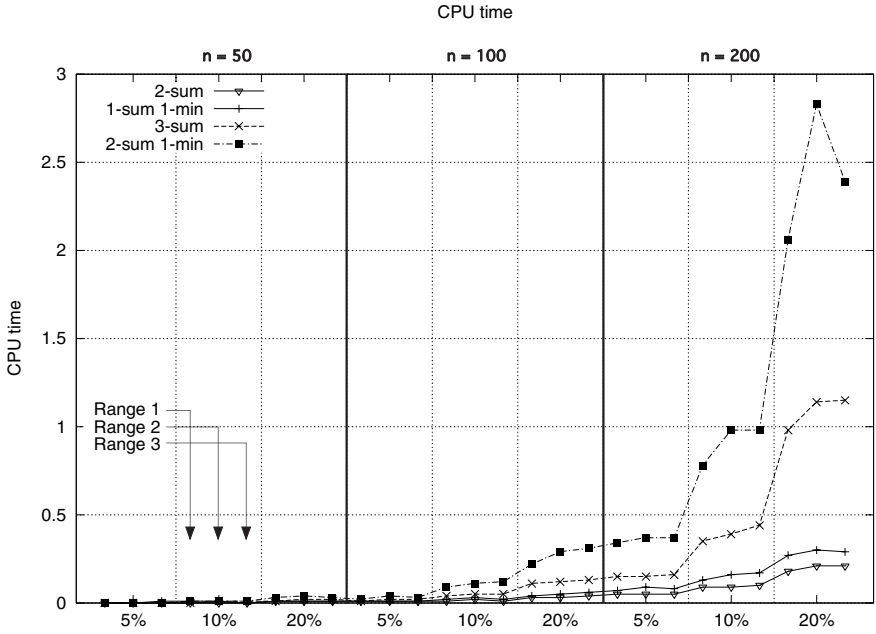

Fig. 6. Average CPU time for computing all solutions for all configurations

us to verify that the number of efficient paths generated is not exponential in practice, even when simultaneously optimizing these S- and M- type objectives. In addition, the computational effort required by the revised algorithm is comparable to the Martins version although more labels are handled in the revised version.

By combining two types of objectives, this revised version provides a flexible algorithmic solution to deal with real world problems in which the configuration of objectives to be considered depends on external factors. It can be used in a real-time environment, such as the computer communication problem mentioned in this paper. Finally, although this paper examined the configuration with only one M-type objective, our revised version of the Martins algorithm is capable of handling more than one of these objectives.

Acknowledgements. The authors would like to thank José Figueira, guest researcher at the University of Valenciennes for one month in 2001, for fruitful discussions concerning shortest path problems with multiple objectives.

References

Brumbaugh-Smith J, Shier D (1989) An empirical investigation of some bicriterion shortest path algorithms. *European Journal of Operational Research* 43(2): 216–224

Clímaco JCM, Martins EQV (1982) A bicriterion shortest path algorithm. *European Journal of Operational Research* 11: 399–404

Ehrgott M (2000) *Multicriteria optimization. Lecture Notes in Economics and Mathematical Systems*, 491. Springer, Berlin Heidelberg New York

Ehrgott M, Gandibleux X (2002) Multiobjective combinatorial optimization. In: Ehrgott M, Gandibleux X (eds) *Multiple criteria optimization: state of the art annotated bibliographic survey. Int. Series in Operations Research and Management Science* Kluwer Academic Pub., 52:8 pp. 369–444

Hansen P (1979) Bicriterion path problems. In: Fandel G, Gal T (eds) *Multiple criteria decision making theory and application. Lecture Notes in Economics and Mathematical Systems*, 177. Springer, Berlin Heidelberg New York, pp. 109–127

Henig MI (1985) The shortest path problem with two objective functions. *European Journal of Operational Research* 25: 281–291

Kostreva MM, Wiecek MM (1993) Time dependency in multiple objective dynamic programming. *Journal of Mathematical Analysis and Applications* 173(1): 289–307

Martins EQV (1984a) On a multicriteria shortest path problem. *European Journal of Operational Research* 16: 236–245

Martins EQV (1984b) On a special class of bicriterion path problems. *European Journal of Operational Research* 17: 85–94

Moore LJ, Taylor BW, Lee SM (1978) Analysis of a transshipment problem with multiple conflicting objectives. *Computers and Operations Research* 5: 39–46

Mote J, Murthy I, Olson DL (1991) A parametric approach to solving bicriterion shortest path problems. *European Journal of Operational Research* 53: 81–92

Randriamasy S, Gandibleux X, Figueira J, Thomin Ph (2002) Brevet 03291744.5-2416 *Dispositif et procédé de détermination de chemins de routage dans un réseau de communications, en présence d´attributs de sélection (A device and a method for determining routing paths in a communication network, in the presence of selection attributes.)* déposé le 15 Juil. 2002, France

Randriamasy S, Gandibleux X (2003) Routage multiobjectif dans les réseaux IP (Multiobjective routing in IP networks). *ROADEF'03*, pp 26–28 février 2003. Avignon, France

Serafini P (1986) Some considerations about computational complexity for multi objective combinatorial problems. In: Jahn J, Krabs, W (eds) *Recent advances and historical development of vector optimization. Lecture Notes in Economics and Mathematical Systems*, 294. Springer, Berlin Heidelberg New York, pp 222–232

Skriver AJV, Andersen KA (2000) A label correcting approach for solving bicriterion shortest path problems. *Computers and Operations Research* 27(6): 507–524

Sniedovich M (1988) A multi-objective routing problem revisited. *Engineering Optimization* 13: 99–108