

Instituto de Engenharia de Sistemas e Computadores de Coimbra  
Institute of Systems Engineering and Computers  
INESC – Coimbra

Marta Pascoal

**Labeling algorithms for minimum-hop bicriteria path problems**

No. 15

2008

ISSN: 1645-2631

Instituto de Engenharia de Sistemas e Computadores de Coimbra  
INESC – Coimbra  
Rua Antero de Quental, 199; 3000 - 033 Coimbra; Portugal  
[www.inescc.pt](http://www.inescc.pt)

# LABELING ALGORITHMS FOR MINIMUM-HOP BICRITERIA PATH PROBLEMS

MARTA M. B. PASCOAL

Departamento de Matemática da Universidade de Coimbra  
Apartado 3008, EC Universidade, 3001-454 Coimbra, Portugal  
E-mail: [marta@mat.uc.pt](mailto:marta@mat.uc.pt)

Instituto de Engenharia de Sistemas e Computadores – Coimbra (INESCC), Rua Antero de Quental, 199,  
3000-033 Coimbra, Portugal

Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT), Canada

December 2008

---

**Abstract:** The number of hops (or arcs) of a path is a frequent objective function with applications to problems where network resources utilization is to be minimised. In this paper we deal with bicriteria path problems involving that objective function and two other common metrics, the path cost and the path capacity. We introduce labeling algorithms using a breadth-first search tree in order to compute the maximal and the minimal sets of non-dominated paths. Computational experiment results on randomly generated test problems are presented and discussed.

**Keywords:** Hop, Cost, Capacity, Bicriteria path problem, Labeling algorithms.

---

## 1 Introduction

Optimal problems arise in different contexts and with different types of objective functions. Often a single objective function is not sufficient to completely characterise a problem, which leads to consider more than one objective. In general the objectives are conflictual and there is no solution that optimises all them simultaneously. In those cases the determination of the optimal solution can be replaced by the determination of the set of non-dominated paths.

Various references can be found about the shortest path problem, when an additive cost function is considered, but also variants like the maximum capacity path problem, involving a bottleneck function, or the minimum hop path problem, aiming to minimise the number of arcs in a path (or hops, borrowing the telecommunications terminology), have applications. The works by Ahuja, Magnanti and Orlin or by Deo and Pang present numerous references on these problems [1, 4]. The latest of these problems, that can be seen as a particular case of the shortest path problem when all arc costs are equal to 1, is particularly interesting to telecommunications, as it is often the case that a route that uses the least network resources has to be found. In such problems it is also common to look for a route that uses the maximum available bandwidth, or that has the minimum cost<sup>1</sup>. When combining the objective function number of hops with the cost or with the bandwidth, particular cases

---

<sup>1</sup>Sometimes also related with the arcs bandwidth.

of the bicriteria shortest path problem or the bicriteria shortest - maximum capacity path problem are obtained. Labeling algorithms for these general problems have been studied by authors like Hansen, Martins or Gandibleux, Beugnies and Randriamasy [5, 6, 7, 8], to name a few. This manuscript focuses on labeling algorithms for bicriteria path problems involving the number of hops and the cost minimization or the number of hops minimization and the capacity maximization, and proposes to use a breadth-search tree to store the labels, where nodes are scanned by order of the level they belong to.

The manuscript is organised as follows. Section 2 introduces notation, preliminary concepts and the problems. Section 3 proposes labeling algorithms for the minimum hop - shortest path problem and the minimum hop - maximum capacity path problem from two points of view, the determination of all non-dominated paths or simply of those with distinct objective values. Section 4 presents the results of computational experiments and Section 5 is reserved for conclusions.

## 2 Some bicriteria path problems

Let  $(\mathcal{N}, \mathcal{A})$  be a directed network consisting of a set  $\mathcal{N} = \{1, \dots, n\}$  of nodes and a set  $\mathcal{A} = \{1, \dots, m\}$  of arcs. Let  $s$ , the initial node, and  $t$ , the terminal node, be two distinct nodes in  $(\mathcal{N}, \mathcal{A})$ . A path from  $s$  to  $t$  in  $(\mathcal{N}, \mathcal{A})$  is a sequence of the form  $p = \langle v_1, v_2, \dots, v_\ell \rangle$  where  $v_1 = s$ ,  $v_\ell = t$ ,  $v_i \in \mathcal{N}$ ,  $i = 1, \dots, \ell$ , and  $(v_i, v_{i+1}) \in \mathcal{A}$ ,  $i = 1, \dots, \ell - 1$ . Let  $\mathcal{P}$  denote the set of all paths from  $s$  to  $t$  in  $(\mathcal{N}, \mathcal{A})$ .

Associated with each arc  $(i, j) \in \mathcal{A}$  we may consider the values  $c_{ij} \in \mathbb{R}$ , representing its cost, and  $u_{ij} \in \mathbb{R}^+$ , representing its capacity. Then several functions can assign a value to each path  $p$ , we stress the path cost, given by:

$$c(p) = \sum_{(i,j) \in p} c_{ij},$$

the path capacity, given by:

$$u(p) = \min_{(i,j) \in p} \{u_{ij}\}$$

and the path number of arcs, represented by  $h(p)$ . Usually functions  $c$  and  $h$  are minimised, while  $u$  is maximised. Some of the most common bicriteria path problems result from the combination of these objective functions, see Table 1.

Denomination	Aim
MinHop-MinSum	$\min_{\mathcal{P}}\{h(p)\}, \max_{\mathcal{P}}\{c(p)\}$
MinSum-MaxMin	$\min_{\mathcal{P}}\{c(p)\}, \max_{\mathcal{P}}\{u(p)\}$
MinHop-MaxMin	$\min_{\mathcal{P}}\{h(p)\}, \max_{\mathcal{P}}\{u(p)\}$

Table 1: Bicriteria path problems

In general, the two objective functions are not correlated and there is not a solution that optimises them simultaneously. Instead, the set of non-dominated paths, for which there is no other solution that improves one of the objectives without worsening the other, is computed. In the following definitions we borrow some terminology used by Gandibleux, Beugnies and Randriamasy in [5], also for multicriteria path problems. As we will focus on the optimization of different objective functions we present a general definition of dominance.

**Definition 1.** Let  $p_1, p_2$  be two paths between the same pair of nodes in  $(\mathcal{N}, \mathcal{A})$  and  $f_1, f_2$  two functions defined for any path.

1. It is said that  $p_1$  dominates  $p_2$  ( $p_1 \mathcal{D} p_2$ ) if and only if  $f_i(p_1)$  is better than or equal to  $f_i(p_2)$ ,  $i = 1, 2$ , and it is strictly better for at least one of the objective functions. In that case it is also said that  $(f_1(p_1), f_2(p_1))$  dominates  $(f_1(p_2), f_2(p_2))$  ( $(f_1(p_1), f_2(p_1)) \mathcal{D} (f_1(p_2), f_2(p_2))$ ).
2. Path  $p_1$  strictly dominates path  $p_2$  if and only if it is better than  $p_2$  for  $f_1$  and  $f_2$ .  $p_1, p_2$  are equivalent when  $f_i(p_1) = f_i(p_2)$ ,  $i = 1, 2$ .

**Definition 2.** A path  $p \in \mathcal{P}$  is non-dominated, or efficient, if and only if it is not dominated by any other. If there is no path that strictly dominates  $p$ , then  $p$  is said to be weakly non-dominated, or weakly efficient.

**Definition 3.** Let  $\mathcal{P}_D$  be the subset of dominated paths in  $\mathcal{P}$ . Then  $\mathcal{P}_N = \mathcal{P} - \mathcal{P}_D$  denotes the maximal complete set of non-dominated paths in  $\mathcal{P}$ , while  $\bar{\mathcal{P}}_N$ , the minimal complete set of non-dominated paths denotes the largest subset of  $\mathcal{P}_N$  that contains no equivalent solutions.

### 3 MinHop-MinSum and MinHop-MaxMin path problems

In 1980 Hansen [6] presented a list with several bicriteria path problems, studied their complexity orders and introduced labeling algorithm adaptations. That list included the MinSum-MinSum path problem, as well as a problem involving one additive and one bottleneck functions, the MinSum-MaxMin path problem. Later Martins [7] generalised labeling algorithms for the MinSum path problem with more than two objective functions, and in [8] the same author studied the MinSum-MaxMin case and developed both an algorithm for finding the maximal set of non-dominated paths and another for finding simply the minimum set of non-dominated paths, thus computing the non-dominated objective values. Recently the algorithm presented by Martins in [7] was extended by Gandibleux *et al.* [5] in order to cope with more than one cost function and one of bottleneck type. Other labeling algorithms have also been presented for the MinSum-MinSum case, like [2, 9].

In this type of labeling algorithms, several labels can be assigned to a network node, each corresponding to a path in the network starting from  $s$ , thus constructing a tree of paths rooted at  $s$ . Some branches of this tree are pruned by means of testing the dominance of new nodes and of those that are already in the tree, corresponding to paths. Like for the single criteria case, label setting and label correcting algorithms for bicriteria shortest path problems differ on the strategy they use to pick the next label to scan, which should be the lexicographically smallest in the first case. As a consequence such a label is permanent, i.e., non-dominated, after being scanned, while with label correcting algorithms non-dominated paths can only be known when all labels have been scanned.

Similarly labeling algorithms can be designed for the MinSum-MaxMin path problem, by including the cost and the capacity values in each label and modifying the label dominance test. However, as pointed out in [5] non-dominated paths may contain weakly non-dominated subpaths, therefore weakly non-dominated labels might be necessary to determine the maximal set of paths. As an example the paths  $\langle 1, 2, 4, 5 \rangle$  and  $\langle 1, 3, 4, 5 \rangle$  in Figure 1 are both non-dominated with respect to the MinHop-MaxMin path problem, although  $\langle 1, 2, 4 \rangle \mathcal{D} \langle 1, 3, 4 \rangle$ .

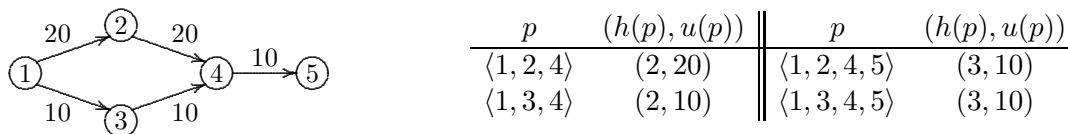


Figure 1: Efficient solutions formed by weakly efficient subpaths

The MinHop-MaxMin path problem can be seen as a special case of the problems solved by Martins and by Gandibleux *et al.* in [5, 7] considering one cost function and  $c_{ij} = 1$  for any  $(i, j) \in \mathcal{A}$ . In the following we first focus on the MinHop-MinSum and then on the MinHop-MaxMin path problems and consider a label associated with a node  $x$  in the search tree formed by paths from  $s$  to other nodes that have the form  $l_x = [\pi_x^h, \pi_x^c, \xi_x, \beta_x]$ , or  $l_x = [\pi_x^h, \pi_x^u, \xi_x, \beta_x]$  where:

- $\pi_x^h$  denotes the number of arcs in the path from the root node to  $x$ ,
- $\pi_x^c$  its cost, or  $\pi_x^u$  its capacity,
- $\xi_x$  is the node preceding  $x$  in that tree, and
- $\beta_x$  is the network node that corresponds to  $x$ .

Given  $\beta_x = i \in \mathcal{N}$  and  $(i, j) \in \mathcal{A}$ , a new node,  $y$ , can be considered in the tree, with the label

$$l_y = [\pi_x^h + 1, \pi_x^c + c_{ij}, x, j], \text{ or } l_y = [\pi_x^h + 1, \min\{\pi_x^u, u_{ij}\}, x, j],$$

depending on the involved objective functions.  $X$  will represent the set of labels that are eligible to be scanned.

### 3.1 MinHop-MinSum path problem

In the MinHop-MinSum path problem, given two labels  $l_x, l_y$  correspondent to a path ending at the same network node, we say  $l_x$  is dominated by  $l_y$  if and only if:

$$(\pi_x^h > \pi_y^h \text{ and } \pi_x^c \geq \pi_y^c) \text{ or } (\pi_x^h \geq \pi_y^h \text{ and } \pi_x^c > \pi_y^c). \quad (1)$$

Therefore, a new label  $l_y$  can be discarded if there is another node  $x$  already in the tree such that  $\beta_x = \beta_y$  and

$$(\pi_x^h < \pi_y^h \text{ and } \pi_x^c \leq \pi_y^c) \text{ or } (\pi_x^h \leq \pi_y^h \text{ and } \pi_x^c < \pi_y^c). \quad (2)$$

On the other hand, whenever (1) holds for some label  $l_x$  in the search tree, then  $l_x$  can be replaced by  $l_y$ .

Now, considering the maximal set of non-dominated paths computation and assuming  $X$  is manipulated as a First In First Out (FIFO) list the number of arcs of the analysed paths, i.e. the  $\pi_x^h$  values, form a non-decreasing sequence, [3]. Thus for a new label  $l_y$  condition  $\pi_x^h \leq \pi_y^h$  always holds, so the first part of (1) is never satisfied and (1) can be replaced by

$$\pi_x^h = \pi_y^h \text{ and } \pi_x^c > \pi_y^c. \quad (3)$$

Furthermore the labels associated with a certain node have particular properties, as shown in Lemma 1.

**Lemma 1.** *If  $X$  is a FIFO, then for each level of the tree of paths rooted at  $s$  a tree node is associated with several labels, iff all have the same objective function values.*

**Proof.** Let  $l_x$  and  $l_y$  be two labels at the same level in  $X$ , that correspond to the same node, i.e.  $\beta_x = \beta_y$  and  $\pi_x^h = \pi_y^h$ . By contradiction, assuming that  $\pi_x^c \neq \pi_y^c$ , then:

1. either  $\pi_x^c < \pi_y^c$ , which implies  $l_x \mathcal{D} l_y$ , so  $y$  should not belong to  $X$ ,
2. or else  $\pi_x^c > \pi_y^c$ , therefore  $l_y \mathcal{D} l_x$ , so  $x$  should not belong to  $X$ .

Moreover, if a network node has several non-dominated labels with the same objective values, then they share the same value of  $h$  and belong to the same paths tree level.  $\square$

This result allows to simplify the acceptance condition of a new label  $l_y$ , that can be restated as

$$(\pi_x^h < \pi_y^h \text{ and } \pi_x^c > \pi_y^c) \text{ or } (\pi_x^h = \pi_y^h \text{ and } \pi_x^c \geq \pi_y^c), \quad (4)$$

for any  $x \in X$  such that  $\beta_x = \beta_y$ , while if (3) holds for some  $x \in X$ , that node can be removed from the tree as its label is dominated. Using the FIFO data structure to represent  $X$ , and conditions (3) and (4) as dominance rules, the following result holds.

**Corollary 1.** *If  $X$  is a FIFO, then the sequence of labels extracted from  $X$  and associated with a network node is in lexicographic order.*

Another consequence of Lemma 1 and Corollary 1 is that each label picked in  $X$  is non-dominated.

**Corollary 2.** *If  $X$  is a FIFO, then a label chosen in  $X$  is permanent.*

Two conclusions can be drawn from this result, first non-dominated paths from  $s$  to  $t$  can be known as labels that correspond to  $t$  are chosen in  $X$ , and second for checking the dominance of a new label it is sufficient to compare it with the last label inserted in  $X$  associated with the same node. In the pseudo-code presented below this information is maintained in the  $n$  elements array *Last*.

Although  $X$  is manipulated as a FIFO in the sense that new elements are inserted at the end of the queue while the first one is scanned, an adaptation might have to be done when a new label  $l_y$  dominates a previous one,  $l_x$ . If  $l_x$  is the only label at level  $\pi_x^h$ , it is enough to replace  $l_x$  by  $l_y$ , however if there are multiple labels associated with  $\beta_x$  at level  $\pi_x^h$ , then they are all dominated and should be deleted. An alternative is to add an additional comparison between labels that are picked in  $X$  and the correspondent *Last*, in order to check their dominance.

The pseudo-code of the method just described is presented in Algorithm 1.

**Algorithm 1.** *MinHop-MinSum maximal set of paths determination*

```

For  $i \in N$  Do  $Last_i \leftarrow 0$ 
 $Last_s \leftarrow 1$ ;  $nX \leftarrow 1$ ;  $l_{nX} \leftarrow [0, 0, -, s]$ ;  $X \leftarrow \{1\}$ ;  $\mathcal{P}_N \leftarrow \emptyset$ 
While  $X \neq \emptyset$  Do
   $x \leftarrow$  first node in  $X$ ;  $X \leftarrow X - \{x\}$ ;  $i \leftarrow \beta_x$ 
  If  $i = t$  Then  $\mathcal{P}_N \leftarrow \mathcal{P}_N \cup \{x\}$ 
  For  $j \in N$  such that  $(i, j) \in A$  Do
     $y \leftarrow Last_j$ 
    If  $(y = 0)$  or  $(y \neq 0$  and  $\pi_x^h + 1 > \pi_y^h$  and  $\pi_x^c + c_{ij} < \pi_y^c)$  Then
       $nX \leftarrow nX + 1$ ;  $l_{nX} \leftarrow [\pi_x^h + 1, \pi_x^c + c_{ij}, x, j]$ ; Insert  $nX$  at the end of  $X$ 
       $Last_j \leftarrow nX$ 
    Else If  $\pi_x^h + 1 = \pi_y^h$  and  $\pi_x^c + c_{ij} = \pi_y^c$  Then
       $nX \leftarrow nX + 1$ ;  $l_{nX} \leftarrow [\pi_y^h, \pi_y^c, x, j]$ ; Insert  $nX$  at the end of  $X$ 
    Else If  $\pi_x^h + 1 = \pi_y^h$  and  $\pi_x^c + c_{ij} < \pi_y^c$  Then
       $l_y \leftarrow [\pi_y^h, \pi_x^c + c_{ij}, x, j]$ 
      Remove from  $X$  other labels associated with  $j$  with the objective values of  $x$ 
  EndIf
EndFor
EndWhile

```

If the goal is to find the non-dominated objective values, i.e. the minimal set of non-dominated paths, it is enough to compute a single path for each objective values pair, which allows to include certain modifications in the method above. The main difference now is that in such a case at most one label is used for each node and each number of arcs, so Lemma 1 is now replaced by the following result.

**Lemma 2.** *If  $X$  is a FIFO, then at most one label per level is associated with a network node.*

Moreover, like for the maximal set of paths determination Corollaries 1 and 2 are still valid. Thus, concerning the minimal set of paths computation a new label  $l_y$  should be inserted in  $X$  iff

$$(\pi_x^h < \pi_y^h \text{ and } \pi_x^c > \pi_y^c) \text{ or } (\pi_x^h = \pi_y^h \text{ and } \pi_x^c > \pi_y^c), \quad (5)$$

for any  $x \in X$  such that  $\beta_x = \beta_y$ , and in the second case, i.e., if

$$\pi_x^h = \pi_y^h \text{ and } \pi_x^c > \pi_y^c, \quad (6)$$

$l_x$  can be discarded. The fact that there is a single label associated with each pair of objective values also allows to replace labels, with no need for deletions, if they become dominated by others. The pseudo-code in Algorithm 2 is a simplified version of Algorithm 1 for finding the minimal set of paths.

**Algorithm 2. MinHop-MinSum minimal set of paths determination**

```

For  $i \in N$  Do  $Last_i \leftarrow 0$ 
 $Last_s \leftarrow 1$ ;  $nX \leftarrow 1$ ;  $l_{nX} \leftarrow [0, 0, -, s]$ ;  $X \leftarrow \{1\}$ ;  $\bar{\mathcal{P}}_N \leftarrow \emptyset$ 
While  $X \neq \emptyset$  Do
   $x \leftarrow$  first node in  $X$ ;  $X \leftarrow X - \{x\}$ ;  $i \leftarrow \beta_x$ 
  If  $i = t$  Then  $\bar{\mathcal{P}}_N \leftarrow \bar{\mathcal{P}}_N \cup \{x\}$ 
  For  $j \in N$  such that  $(i, j) \in A$  Do
     $y \leftarrow Last_j$ 
    If  $(y = 0)$  or  $(y \neq 0 \text{ and } \pi_x^h + 1 > \pi_y^h \text{ and } \pi_x^c + c_{ij} < \pi_y^c)$  Then
       $nX \leftarrow nX + 1$ ;  $l_{nX} \leftarrow [\pi_x^h + 1, \pi_x^c + c_{ij}, x, j]$ ; Insert  $nX$  at the end of  $X$ 
       $Last_j \leftarrow nX$ 
    Else If  $\pi_x^h + 1 = \pi_y^h$  and  $\pi_x^c + c_{ij} < \pi_y^c$  Then  $l_y \leftarrow [\pi_y^h, \pi_x^c + c_{ij}, x, j]$ 
  EndFor
EndWhile

```

The MinHop-MinSum path problem has up to  $n(n-2)$  non-dominated pairs of objective values, if there is a path with any number of arcs between 1 and  $n-2$  from  $s$  to any other node  $i$ . Therefore the tree of paths obtained by Algorithm 2 can have at most  $n-2$  levels. In the worst-case every of the  $m$  network arcs has to be scanned for each of those levels. Analysing an arc implies the insertion and deletion of an element in  $X$ , which can be made in constant order, therefore the worst-case time complexity of Algorithm 2 is  $\mathcal{O}((n-2)m)$ , that is  $\mathcal{O}(nm)$ .

The theoretical complexity order of Algorithm 1 also depends on the number of levels the paths tree can have and on the total number of non-dominated labels. Even though the tree can have up to  $n-2$  levels, as there may be multiple labels with the same objective values the number of labels cannot be predicted. Denoting it by  $k$ , the number of operations performed by this method is  $\mathcal{O}(knm)$ .

### 3.2 MinHop-MaxMin path problem

Let us now consider the MinHop-MaxMin path problem, where instead of minimizing a cost function we maximise a capacity function. Given the labels  $l_x, l_y$  correspondent to a path starting at  $s$  and

ending at the same network node, we say  $l_x$  is dominated by  $l_y$  if and only if:

$$(\pi_x^h > \pi_y^h \text{ and } \pi_x^u \leq \pi_y^u) \text{ or } (\pi_x^h \geq \pi_y^h \text{ and } \pi_x^u < \pi_y^u). \quad (7)$$

Taking into account that in this problem weakly non-dominated labels can be used to obtain non-dominated solutions, namely solutions with the same number of hops but different capacities, as shown in Figure 1, a new label  $l_y$  should only be discarded if there is another  $x$  in  $X$  such that  $\beta_x = \beta_y$  and

$$\pi_x^h < \pi_y^h \text{ and } \pi_x^u \geq \pi_y^u. \quad (8)$$

On the other hand,  $l_x$  can be replaced by  $l_y$  whenever

$$\pi_x^h > \pi_y^h \text{ and } \pi_x^u \leq \pi_y^u. \quad (9)$$

Back to the determination of the maximal set of non-dominated paths using a FIFO, as the number of arcs of the scanned paths is non-decreasing (9) never holds, which means no label should be deleted and the dominance test whenever a new label is formed can be replaced simply by (8).

The acceptance of weakly non-dominated labels that may be dominated implies that Corollaries 1 and 2 are no longer valid. For this reason the set  $\mathcal{P}_N$  is only known after the labeling process is over. Moreover, it is not enough to compare a new label with the last one observed for that node, and two cases should be distinguished:

- $\pi_x^h = \pi_y^h$  for some  $x$  in  $X$ , then node  $\beta_x$  already has a label at level  $\pi_y^h$  and neither  $l_x$  strictly dominates  $l_y$  nor  $l_y$  strictly dominates  $l_x$ , therefore  $y$  is inserted in  $X$ .
- $\pi_x^h < \pi_y^h$  for every  $x$  in  $X$ , then  $l_y$  belongs to a different level than  $l_x$  and it should be inserted iff  $\pi_x^u < \pi_y^u$ , that is,

$$\max\{\pi_x^u : x \in X \text{ and } \pi_x^h < \pi_y^h\} < \pi_y^u. \quad (10)$$

In short, a candidate label will only be accepted if its capacity improves the capacity of the labels that have less arcs. An auxiliary array storing the best capacity value found for each network node until the latest scanned level,  $Best$ , is used in Algorithm 3, that summarises the pseudo-code for finding the maximal set of MinHop-MaxMin paths.

**Algorithm 3. *MinHop-MaxMin maximal set of paths determination***

```

For  $i \in \mathcal{N}$  Do
   $Best_i \leftarrow 0$ ;  $Last_i \leftarrow 0$ ;  $\pi_i^u \leftarrow 0$ 
EndFor
 $Last_s \leftarrow 1$ ;  $nX \leftarrow 1$ ;  $l_{nX} \leftarrow [0, +\infty, -, s]$ ;  $X \leftarrow \{1\}$ 
While  $X \neq \emptyset$  Do
   $x \leftarrow$  first node in  $X$ ;  $X \leftarrow X - \{x\}$ ;  $i \leftarrow \beta_x$ 
  For  $j \in \mathcal{N}$  such that  $(i, j) \in \mathcal{A}$  Do
    If  $\min\{\pi_x^u, u_{ij}\} > Best_j$  Then
       $y \leftarrow Last_j$ 
      If  $(y = 0)$  or  $(y \neq 0 \text{ and } \pi_x^h + 1 > \pi_y^h)$  Then
         $Best_j \leftarrow \pi_{Last_j}^u$ 
        If  $\min\{\pi_x^u, u_{ij}\} > Best_j$  Then
           $nX \leftarrow nX + 1$ ;  $l_{nX} \leftarrow [\pi_x^h + 1, \min\{\pi_x^u, u_{ij}\}, x, j]$ ; Insert  $nX$  at the end of  $X$ 
           $Last_j \leftarrow nX$ 
        EndIf
      EndIf
    EndFor
  EndWhile

```

```

Else If  $\pi_x^h + 1 = \pi_y^h$  Then
     $nX \leftarrow nX + 1$ ;  $l_{nX} \leftarrow [\pi_y^h, \min\{\pi_x^u, u_{ij}\}, x, j]$ ; Insert  $nX$  at the end of  $X$ 
    If  $\min\{\pi_x^u, u_{ij}\} > \pi_y^u$  Then  $Last_j \leftarrow nX$ 
EndIf

EndFor
EndWhile
 $\mathcal{P}_N \leftarrow \{non\text{-dominated paths from } s \text{ to } x \in X \text{ where } \beta_x = t\}$ 

```

While for nodes in general weakly non-dominated solutions have to be generated, only non-dominated labels associated with  $t$  are of interest for the final solution. A different dominance test can be applied then, as a new label  $l_y$ ,  $\beta_y = t$ , should be discarded if and only if there is another one  $l_x$ ,  $\beta_x = t$ , such that

$$(\pi_x^h < \pi_y^h \text{ and } \pi_x^u \geq \pi_y^u) \text{ or } (\pi_x^h = \pi_y^h \text{ and } \pi_x^u > \pi_y^u), \quad (11)$$

and it should replace  $l_x$  if

$$\pi_x^h = \pi_y^h \text{ and } \pi_x^u < \pi_y^u. \quad (12)$$

This variation of Algorithm 3 can assure that  $t$  labels are non-dominated the moment they are chosen in  $X$  thus allowing to obtain non-dominated paths between  $s$  and  $t$  along the labeling process.

If again the aim is the determination of the minimal set of non-dominated paths at most one label is stored for each node in a tree level, therefore no dominated subpaths need to occur in non-dominated solutions and the following result can be proved.

**Proposition 1.** *Let  $p^* \in \mathcal{P}_N$  for the MinHop-MaxMin path problem, then there is  $p \in \mathcal{P}$  formed by non-dominated subpaths from  $s$  to any node such that  $(h(p), u(p)) = (h(p^*), u(p^*))$ .*

This result can be used to tighten the dominance test, as a new label  $l_y$  can be discarded if there is  $l_x$  such that  $\beta_x = \beta_y$  and

$$(\pi_x^h < \pi_y^h \text{ and } \pi_x^u \geq \pi_y^u) \text{ or } (\pi_x^h \leq \pi_y^h \text{ and } \pi_x^u > \pi_y^u),$$

that is, as  $X$  is manipulated as a FIFO, if

$$\pi_x^u \geq \pi_y^u. \quad (13)$$

The same label shall replace  $l_x$  whenever

$$\pi_x^h = \pi_y^h \text{ and } \pi_x^u < \pi_y^u. \quad (14)$$

Moreover, in this case the labels associated with a network node  $i$  correspond to a non-dominated path from  $s$  to  $i$ , and thus a non-dominated path from  $s$  to  $t$  is obtained whenever a label associated with  $t$  is chosen in  $X$ . The resulting method is very similar to Algorithm 2, nevertheless its pseudo-code is shown in Algorithm 4 for completeness.

**Algorithm 4.** *MinHop-MaxMin minimal set of paths determination*

```

For  $i \in N$  Do  $Last_i \leftarrow 0$ 
 $Last_s \leftarrow 1$ ;  $nX \leftarrow 1$ ;  $X \leftarrow \{1\}$ ;  $l_{nX} \leftarrow [0, +\infty, -, s]$ ;  $\mathcal{P}_N \leftarrow \emptyset$ 
While  $X \neq \emptyset$  Do
     $x \leftarrow$  first node in  $X$ ;  $X \leftarrow X - \{x\}$ ;  $i \leftarrow \beta_x$ 
    If  $i = t$  Then  $\mathcal{P}_N \leftarrow \mathcal{P}_N \cup \{x\}$ 
    For  $j \in N$  such that  $(i, j) \in A$  Do

```

```

y ← Lastj
If (y = 0) or (y ≠ 0 and πxh + 1 > πyh and min{πxu, uij} > πyu) Then
    nX ← nX + 1; lnX ← [πxh + 1, min{πxu, uij}, x, j]; Insert nX at the end of X
    Lastj ← nX
Else If πxh + 1 = πyh and min{πxu, uij} > πyu Then ly ← [πyh, min{πxu, uij}, x, j]
EndFor
EndWhile

```

The worst-case time complexity orders of Algorithms 3 and 4 can be studied in the same manner used for the MinHop-MinSum path problem. In the case of Algorithm 3 the bound is  $\mathcal{O}(knm)$ , if  $k$  is the maximum number of multiple labels with the same objective values a network node can have, while Algorithm 4 has  $\mathcal{O}(nm)$ .

## 4 Computational results

Computational experiments have been carried out to evaluate the empirical performance of the methods described in the previous section. With this purpose a set of random networks with 1 000, 3 000, 5 000 and 7 000 nodes,  $dn$  arcs, for densities  $d = 5, 10, 20, 30$ , and uniformly integer cost (capacity) values generated in  $[1, M]$ , with  $M = 100$  and  $M = 10000$ , was considered. The results presented in the following were obtained on 30 different instances generated for each dimension of this data set. Tests were carried out on a Dual Core AMD Opteron at 2 GHz, with 4 Gb of RAM running over SUSE Linux 10.3.

### 4.1 MinHop-MinSum path problem

In order to evaluate the methods proposed for finding the maximal set of MinHop-MinSum paths two programs have been coded in C language, namely a labeling algorithm where  $X$  is as FIFO list with a standard dominance test, F1, and Algorithm 1, A1. Similarly, when concerning only the determination of the minimal set of paths a standard labeling algorithm using a FIFO, F2, and Algorithm 2, A2.

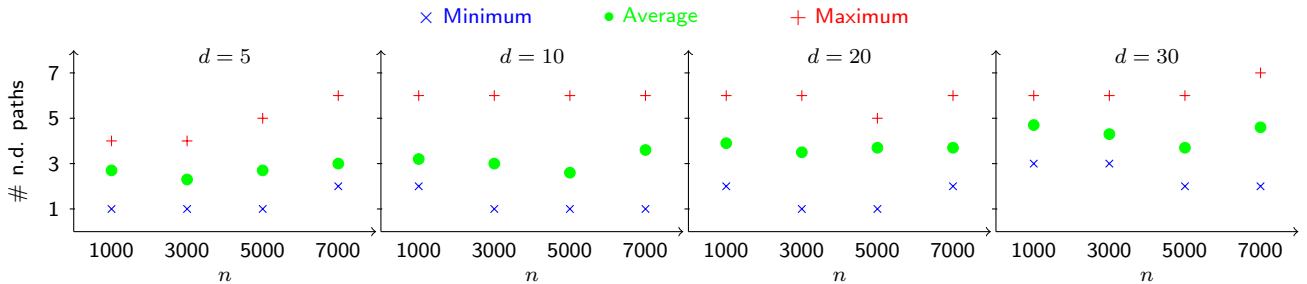


Figure 2: Number of MinHop-MinSum non-dominated paths with  $M = 100$

Figure 2 presents the minimum, average and maximum number of non-dominated paths from  $s$  to  $t$  concerning the case of costs in  $[1, 100]$ . Even though  $t$  can have  $n - 2$  different non-dominated labels in the worst-case, the results show the actual number of labels to be much smaller. The average number of  $\mathcal{P}_N$  elements is between 2 and 5, increasing slowly with the instance density. When  $M = 10000$  the results are very similar in general, only with some higher minimum and average number of non-dominated paths.

A comparison between the average running times of standard versions of a labeling algorithm, F1 and F2, and the methods introduced for finding the maximal and the minimal sets of paths, A1 and

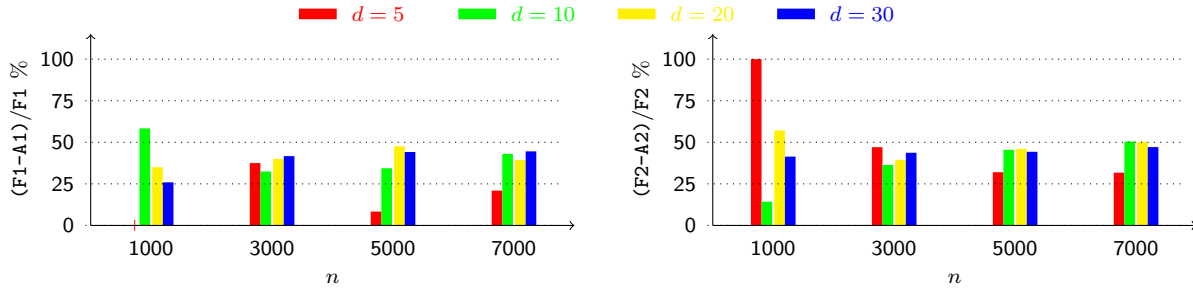


Figure 3: Average CPU times improvement of MinHop-MinSum non-dominated paths with  $M = 100$

A2, is presented in Figure 3, with average values of the improvement obtained by the new algorithms. In most cases the CPU time is approximately reduced by half, although in some small size instances (with low density when finding the minimal) the results are better for the standard versions. The algorithms performance is very similar for the two cost ranges, although the times are slightly greater for the wider range. It is still worth notice that the minimal set determination is easier than the maximal's one, the difference seems to increase with the instances dimension. It is worth noticing the CPU times taken by any of the implemented algorithms were very small for small size instances. Figures 4 and 5 show those average running times for the A1 and A2 codes, the first two plots concern the variation depending on the number of network nodes, while the two others depend on the network density. There is an increase of running times with  $n$  as well as with  $d$ , both for A1 and A2 as the theoretical complexity bound suggests. The determination of the maximal, and of the minimal, sets of MinHop-MinSum paths in the considered data set was made in short times. A1 and A2 were able to solve problems with 7000 nodes and 21000 arcs in less than 0.043 seconds.

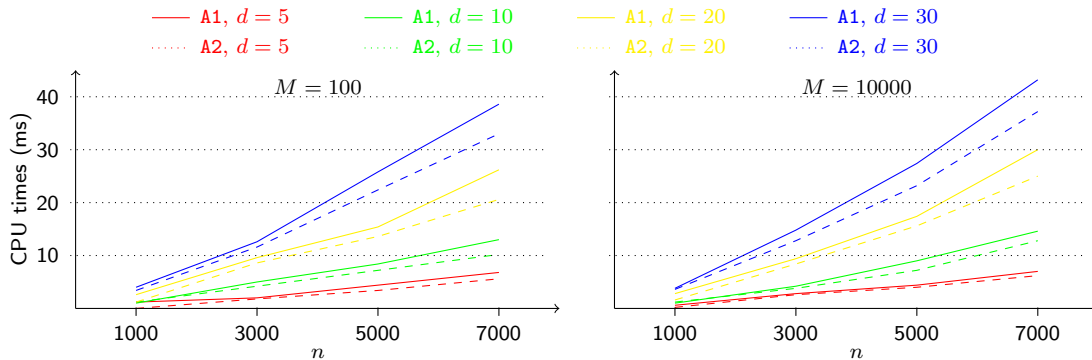


Figure 4: Average CPU times MinHop-MinSum non-dominated paths versus  $n$

## 4.2 MinHop-MaxMin path problem

The procedure followed for this problem was analogous to the one used in the previous section, two standard labeling algorithm using a FIFO were coded one for finding the maximal and the other for finding the minimal sets of MinHop-MaxMin paths, F3 and F4, as well as Algorithms 3 and 4, A3 and A4, respectively.

The minimum, average and maximum number of solutions in the tested instances with capacity values in  $[1, 100]$  is indicated in Figure 6. These values are greater than for the latter problem, in average the maximal set of paths has between 4 and 8 non-dominated solutions, but are still far from the upperbound  $n - 2$ . For instances with  $M = 10000$  the average and the maximum values can

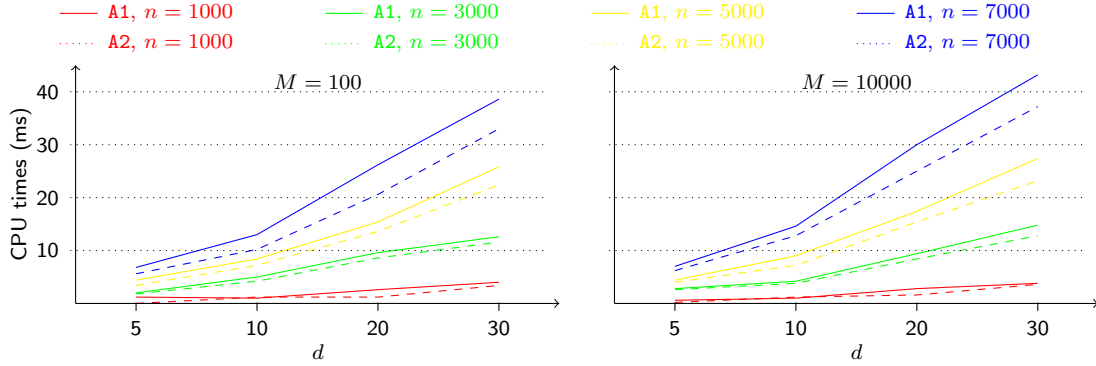


Figure 5: Average CPU times MinHop-MinSum non-dominated paths versus  $d$

increase by 1 to 2 solutions.

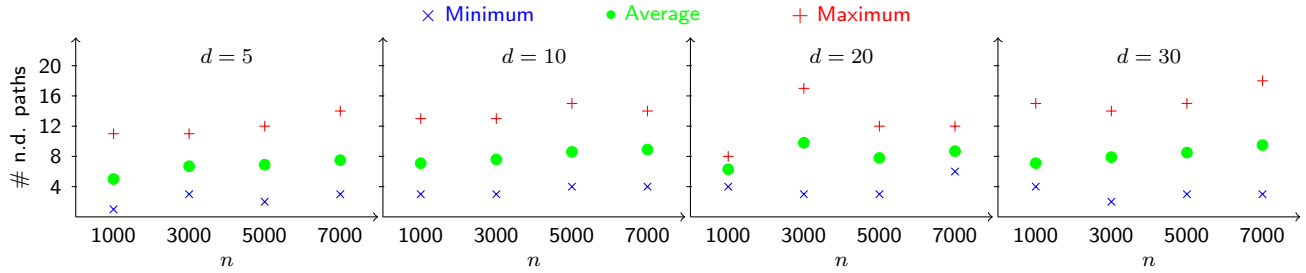


Figure 6: Number of MinHop-MaxMin non-dominated paths with  $M = 100$

Figure 7 shows the relation between the original algorithms, F3 and F4, and the introduced methods, A3 and A4, for the minimal and the maximal sets determination. The improvement on running times is around 75% for finding all non-dominated paths and between 75% and 100% for finding the non-dominated objective values, regardless of the problems dimension. As expected these values are greater than the observed for the MinHop-MaxSum path problem, as much more weakly non-dominated labels have to be stored now. The general tendency is the same when  $M = 10000$ , but the improvement is around 80% for codes F3 and A3, and around 70% for F4 and A4.

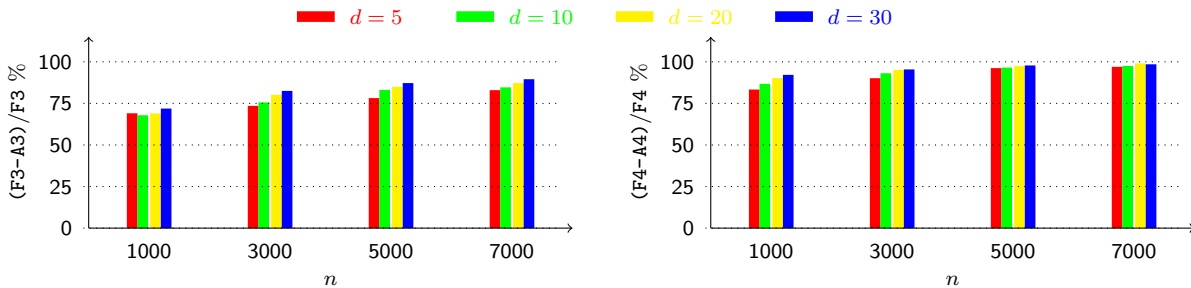


Figure 7: Average CPU times improvement of MinHop-MaxMin non-dominated paths with  $M = 100$

The difference in performance when determining the maximal and the minimum sets of paths is also evident on the plots in Figures 8 and 9, that show the running times of A3 and A4 variation with  $n$  and with  $d$ , respectively. Solid lines, depicting A3 results, grow much faster than dashed lines, for A4. The growth seems to present a linear behaviour with  $n$  and like in the previous section, quadratic with  $d$ . For bigger instances it took in average 0.374 seconds to find the maximal set of paths and 0.076 seconds to find the minimal.

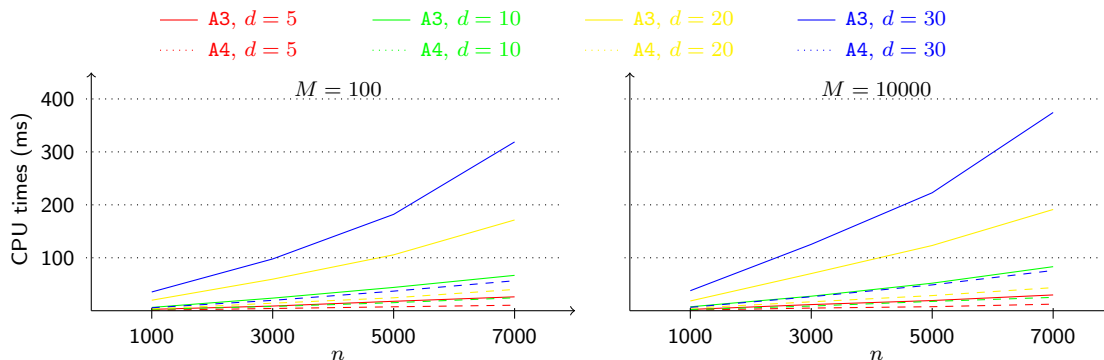


Figure 8: Average CPU times MinHop-MaxMin non-dominated paths versus  $n$

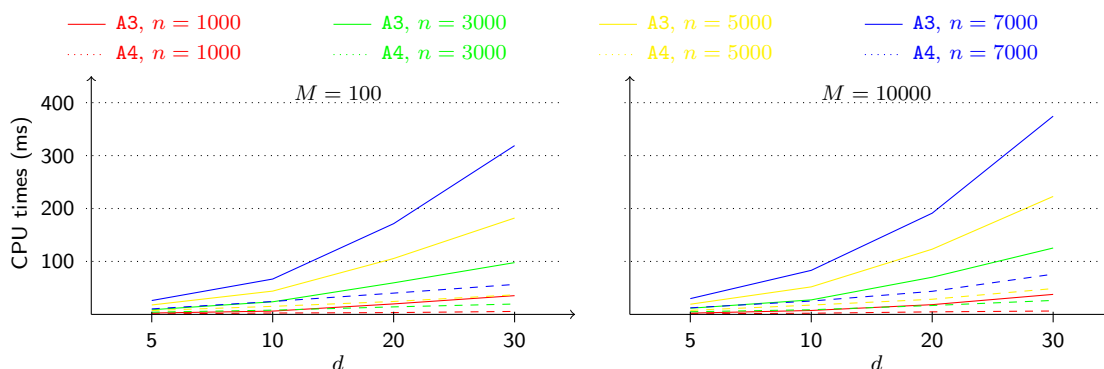


Figure 9: Average CPU times MinHop-MaxMin non-dominated paths versus  $d$

## 5 Conclusions

Labeling algorithms for bicriteria path problems minimising the number of hops and either the path cost or the path capacity have been described, aiming the maximal and the minimal sets of non-dominated paths computation. The introduced methods make use of a breadth-first search tree, by managing the set of labels as a FIFO, and list node labels by non-decreasing order of the number of hops. Tuning the dominance test according with this structure leads to an improvement of about 50% for the MinHop-MinSum path problem running times and between 75% and 100% for the MinHop-MaxMin path problem over randomly generated instances. Numerical results report up to 0.043 seconds for finding the whole set of non-dominated paths in the first case and 0.037 seconds in the second for instances with 7 000 nodes and 210 000 arcs. The determination of the non-dominated objective values was made in up to 0.374 seconds for the MinHop-MinSum and 0.076 seconds for the MinHop-MaxMin path problems over the same test bed.

## References

- [1] R. Ahuja, T. Magnanti, and J. Orlin. Some recent advances in network flows. *SIAM Review*, 33(2):175–219, June 1991.
- [2] J. Brumbaugh-Smith and D. Shier. An empirical investigation of some bicriterion shortest path algorithms. *European Journal of Operational Research*, 43(2):216–224, 1989.

- [3] T. Cormen, C. Leiserson, R. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, Cambridge, MA, 2001.
- [4] N. Deo and C. Pang. Shortest paths algorithms: taxonomy and annotation. *Networks*, 9:275–323, 1979.
- [5] X. Gandibleux, F. Beugnies, and S. Randriamasy. Multi-objective shortest path problems with a maxmin cost function. *4OR – Quarterly Journal of the Belgian, French and Italian Operations Research Societies*, 4:47–59, 2006.
- [6] P. Hansen. Bicriterion path problems. In *Multiple Criteria Decision Making: Theory and Applications*, editors: G. Fandel and T. Gal, Lectures Notes in Economics and Mathematical Systems, 177, 109–127, Springer Heidelberg, 1980.
- [7] E. Martins. On a multicriteria shortest path problem. *European Journal of Operational Research*, 16:236–245, 1984.
- [8] E. Martins. On a special class of bicriterion path problems. *European Journal of Operational Research*, 17:85–94, 1984.
- [9] A. Skriver and K. Andersen. A label correcting approach for solving bicriterion shortest-path problems. *Computers and Operations Research*, 27(6):507–524, 2000.