

A BICRITERION APPROACH FOR ROUTING PROBLEMS IN MULTIMEDIA NETWORKS

JOÃO C. N. CLÍMACO^(1,2), JOSÉ M. F. CRAVEIRINHA ^(2,3), and MARTA M. B. PASCOAL^(4,5) ¹

⁽¹⁾ Faculdade de Economia da Universidade de Coimbra
Avenida Dias da Silva, 165,
3004-512 COIMBRA, Portugal

⁽²⁾ Instituto de Engenharia de Sistemas e Computadores – Coimbra
Rua Antero de Quental, 199,
3000-033 COIMBRA, Portugal
E-mail: *jclimaco@inescc.pt*

⁽³⁾ Departamento de Engenharia Electrotécnica e de Computadores
Polo II da Universidade de Coimbra,
Pinhal de Marrocos,
3030-290 COIMBRA, Portugal
E-mail: *jcra@dee.uc.pt*

⁽⁴⁾ Centro de Informática e Sistemas

⁽⁵⁾ Departamento de Matemática
Polo I da Universidade de Coimbra,
Apartado 3008,
3001-454 COIMBRA, Portugal
E-mail: *marta@mat.uc.pt*

November 2001
Revised in June 2002

Abstract

Routing problems in communication networks supporting multiple services, namely multimedia applications, involve the selection of paths satisfying multiple constraints (of a technical nature) and seeking simultaneously to “optimise” the associated metrics. Although traditional models in this area were single-objective, in many situations it is important to consider different, eventually conflicting, objectives. In this paper we consider a bicriterion model dedicated to calculating non-dominated paths for specific traffic flows (namely associated with video services), in multiservice high-speed networks. The mathematical formulation of the problem and the bicriterion algorithmic approach developed for its resolution will be presented together with extensive computational results regarding an application to video traffic in a high-speed network. The algorithmic approach is an adaptation of recent works by Ernesto Martins and his collaborators, namely the MPS algorithm. ²

¹Authors' names in alphabetic order.

²This is a preprint of an article published in *A bicriterion approach for routing problems in multimedia networks*, João C. N. Clímaco, José M. F. Craveirinha, Marta M. B. Pascoal, Networks, Volume 41, Issue 4, Pages: 206-220, 2003.
(<http://www3.interscience.wiley.com/cgi-bin/jtoc?ID=32046>)

1 Introduction

Routing problems in communication networks supporting multiple services, namely multimedia applications, involve the selection of paths satisfying multiple constraints of a technical nature, designated as QoS (Quality of Service) requirements and seeking simultaneously to “optimise” the chosen objective functions. The objective functions are concerned with the necessity of minimizing the consumption of (transmission) resources along a path and to obtain a minimum negative impact in all other traffic flows that may use the network. The specific models of these cost functions and of the QoS constraints depend on the type of multimedia service associated with the “calls” which are being routed from origin to destination. In this context the term traffic flow is the representation (usually of a stochastic nature) of the calls associated with a given application/service, which are being offered and transported by the network. Typical objective functions are the number of arcs (usually designated in telecommunications as hops or links) and the cost of accepting a call in each arc, as measured by an appropriate traffic model related with the bandwidth available in each link. As for the constraints on the paths, in the case of multimedia applications, these are typically the minimum bandwidth required by the call and the maximum allowed delay and jitter.

Although traditional models in this area were single-objective, in many situations it is important to consider different, eventually conflicting objectives. Routing algorithms that have been employed in current networks or proposed for this type of problem, are heuristics based on Dijkstra or Bellman-Ford shortest path algorithm. Significant examples of this type of approaches are in Kompella *et al.* [5], Lee *et al.* [6] and Pornavalai *et al.* [16].

Having in mind to explore the multicriterion nature of this type of problem we consider in this paper a bicriterion model dedicated to calculating the whole set of non-dominated paths for traffic flows associated with multimedia type services in multiservice networks. For this purpose an exact algorithmic approach is developed based on the bicriterion shortest path algorithm by Clímaco and Martins [8] and on the MPS algorithm [11, 12]. Note that both algorithms belong to a research stream headed by Ernesto Martins at the University of Coimbra during the last two decades. As outlined in the conclusions the fastness of the proposed approach in calculating the non-dominated solutions makes it also rather appropriate for adaptation to a real-time selection of an adequate non-dominated solution, as required in many practical situations.

The major contributions of this paper are the following. It is an application of a bicriterion shortest path model to a multimedia network routing problem (concerning the relevance of multicriterion shortest path models in practical applications see [15]). As far as we know is the first time that an exact algorithm is used for obtaining the solutions of a bicriterion model of this specific type, for this purpose it was necessary to adapt a ranking algorithm for generating the set of non-dominated paths. It would be expected that the use of a labeling algorithm could be a better approach; however the explicit consideration of additional constraints to the bicriterion shortest path problem showed that the ranking algorithm leads to a better performance. This new approach was tested to an US inter-city network thereby simulating a realistic type of application. Although the number of obtained non-dominated solutions is not very high the advantages of using a bicriterion approach in many problems of this time, will be made clear.

The notation, basic definitions and the mathematical formulation of the routing problem are presented in section 2 of the paper. The algorithmic approach dedicated to the calculation of the set of non-dominated solutions is described in section 3. An application of this model to a specific routing problem of video traffic in a high-speed network together with extensive computational results, are presented in section 4. Conclusions concerning the inherent advantages of this approach and its applicability are outlined in the final section.

2 Mathematical formulation

In this section we will recall some basic concepts and present the multimedia traffic routing problem and its formulation in terms of a bicriterion shortest path problem.

In a teletraffic routing problem we consider a representation of a communication network the nodes of which may represent routers, servers or switches and the arcs of which represent links in the network with a certain transmission capacity expressed in terms of bandwidth.

Let $(\mathcal{N}, \mathcal{A})$ be a network, where $\mathcal{N} = \{v_1, \dots, v_n\}$ denotes the set of nodes and $\mathcal{A} = \{a_1, \dots, a_m\}$ denotes the set of arcs (or links), assuming a_k corresponds to pair (i, j) , for some $i, j \in \mathcal{N}$. $(\mathcal{N}, \mathcal{A})$ is said to be an undirected network when its arcs are non-ordered pairs.

In the following $(\mathcal{N}, \mathcal{A})$ represents an undirected network and two distinct nodes are considered in this network: s (the initial node) and t (the terminal node).

A path from $i \in \mathcal{N}$ to $j \in \mathcal{N}$ in $(\mathcal{N}, \mathcal{A})$ is an alternating sequence of nodes and arcs in the network, $p = \langle v'_1, a'_1, v'_2, a'_2, \dots, v'_\ell \rangle$, where:

- $i \equiv v'_1$ and $j \equiv v'_\ell$;
- $v'_k \in \mathcal{N}$, for any $k \in \{1, \dots, \ell\}$;
- $a'_k \equiv (v'_k, v'_{k+1}) \in \mathcal{A}$, for any $k \in \{1, \dots, \ell - 1\}$.

A path is said to be a null path if it is formed only by one node. In the following a path will be represented only by its nodes. A cycle (or loop) is a path with no repeated nodes, except the first one which coincides with the last. A path p the nodes of which are all different, that is a path without cycles, is said to be a loopless path.

The set of paths (loopless paths) from i to j in $(\mathcal{N}, \mathcal{A})$ will be denoted by \mathcal{P}_{ij} ($\bar{\mathcal{P}}_{ij}$) and \mathcal{P}_{st} ($\bar{\mathcal{P}}_{st}$) will be denoted by \mathcal{P} ($\bar{\mathcal{P}}$). A subpath of a path p is a sequence of nodes and arcs of p . Let u be a node of p , then $\text{sub}_p(s, u)$ represents its subpath from s to u . Given two paths $p \in \mathcal{P}_{iu}$ and $q \in \mathcal{P}_{uj}$, the concatenation of p and q , denoted by $p \diamond q \in \mathcal{P}_{ij}$, is the path formed by p and followed by q . In order to simplify the notation, sometimes $p \diamond (i, j)$ will be written instead of $p \diamond \langle i, (i, j), j \rangle$.

Let us now introduce the multimedia traffic routing problem as a network problem. Assume three values are associated with each arc (or link) (i, j) in $(\mathcal{N}, \mathcal{A})$, namely $c_{ij} > 0$ representing the cost of (i, j) , $b_{ij} > 0$ representing the available bandwidth of (i, j) and, finally, $d_{ij} > 0$ representing the associated delay. Moreover, let c , b and d be functions which assign to each path p , respectively, $c(p) = \sum_{(i,j) \in p} c_{ij}$, $b(p) = \min_{(i,j) \in p} \{b_{ij}\}$ and $d(p) = \sum_{(i,j) \in p} d_{ij}$. Let us still consider another function h which assigns to each path p its number of arcs. Note that the cost c_{ij} of accepting a call on arc (i, j) is in general a function of some associated link working condition and the objective of minimizing c is to obtain the most favourable traffic distribution in the overall network (maximum traffic carried). In most models c_{ij} is a function of the blocking probability or the available bandwidth of the arc.

Given the values $\Delta_{\text{jitter}} \in \mathbb{N}$, $\Delta_{\text{bandwidth}} \in \mathbb{R}^+$ and $\Delta_{\text{delay}} \in \mathbb{R}^+$, the goal of the problem presented in this work is to determine loopless paths $p \in \bar{\mathcal{P}}$ with, simultaneously, minimum cost and minimum number of arcs, and also satisfying the following constraints:

- $b(p) \geq \Delta_{\text{bandwidth}}$;
- $d(p) \leq \Delta_{\text{delay}}$;
- p has at the most Δ_{jitter} arcs (jitter constraints were expressed in terms of maximum number of arcs).

In other words, considering $f : \mathcal{P} \longrightarrow \mathbb{R}^2$ such that $f(p) = (c(p), h(p))$, we want to:

$$\begin{aligned} \text{“min”} \quad & \{f(p) : p \in \bar{\mathcal{P}}\} & (1) \\ \text{s. a.} \quad & b(p) \geq \Delta_{\text{bandwidth}} & (2) \\ & d(p) \leq \Delta_{\text{delay}} & (3) \\ & h(p) \leq \Delta_{\text{jitter}} & (4) \end{aligned}$$

Thus, this problem may be considered as a bicriterion loopless path problem where the loopless paths to be computed should satisfy several additional constraints.

Initially we will ignore the constraints over the paths being determined and analyse only the underlying bicriterion shortest path problem. In general such a problem does not have a solution, in the sense that if there is a conflict between the considered functions it may happen that no path minimizes both functions simultaneously. Thus, the procedure usually used consists in determining a set of solutions called “efficient” solutions, in the sense that there is no other feasible path which improves one objective function without worsening at least one of the other objective functions. These concepts are summarized in Definitions 1 and 2.

Definition 1 Given $p, q \in \mathcal{P}_{ij}$ it is said that p dominates q or that p is dominated by q , p_Dq , if and only if $c(p) \leq c(q)$, $h(p) \leq h(q)$ and at least one of the inequalities is strict.

Definition 2 A path $q \in \mathcal{P}_{ij}$ is said to be dominated if and only if there is another path $p \in \mathcal{P}_{ij}$ such that p_Dq .

Thus, a path will be non-dominated if there is no other path which dominates this one, and the resolution of problem (1) will consist in computing paths or usually a subset of \mathcal{P}_N which is defined in Definition 3.

Definition 3 The set $\mathcal{P}_N = \{p \in \mathcal{P} : \nexists q \in \mathcal{P} \text{ such that } q_Dp\}$ is called set of non-dominated paths from s to t in $(\mathcal{N}, \mathcal{A})$.

Theorem 1 states that under some assumptions the bicriterion shortest path problem and the bicriterion shortest loopless path problem are equivalent. This theorem's proof is omitted once it is a generalization of a well-known result for the single-objective case.

Theorem 1 Let us assume that for any cycle \mathcal{C} in the network $c(\mathcal{C}) \geq 0$ and $h(\mathcal{C}) \geq 0$. Then the bicriterion shortest path problem and the bicriterion shortest loopless path problem are equivalent.

Using this theorem, and since $c(p) > 0$ and $h(p) > 0$, for any non-null path p in $(\mathcal{N}, \mathcal{A})$, we can conclude that it is sufficient to solve the bicriterion shortest path problem with constraints (2), (3) and (4).

3 Algorithm description

In order to solve this bicriterion shortest path problem with additional constraints we will now introduce an adaptation of an algorithm for the bicriterion shortest path problem.

The first algorithm for bicriterion problems was presented in 1980 by Hansen [4] and it is a labeling algorithm, generalization of Dijkstra's algorithm for the shortest path problem in \mathbb{R} . Later, in 1984, this algorithm was generalized by Martins in [10], for more than two objective functions.

Simultaneously another class of algorithms for the multicriterion problem was developed, based not on labeling algorithms, but on the adaptation of ranking paths algorithms. The first contribution for an algorithm of this class was due to Clímaco and Martins [7] and it concerned multicriterion optimal path problems. Later on this algorithm was particularized by the same authors for the bicriterion case with linear objective functions [8]. In [18] Skriver presented a very complete survey on bicriterion shortest path algorithms.

For the enumeration of paths in [7] and [8] an algorithm due to Martins [9] was considered. Originally this algorithm was conceived to rank optimal paths but in the years following its particularization for the K shortest path problem, it was successively improved and its versions originated the so-called class of deletion algorithms [1, 2, 3, 13, 14].

More recently Martins *et al.* have presented other approaches [11, 12] concerning the K shortest path and the K shortest loopless path problems. In these works another class of algorithms for the determination of paths, (known as deviation algorithms) is described. Among them we can point out the so-called MPS algorithm for its efficiency. Concerning the loopless paths determination those papers also describe the possibility of adapting deviation algorithms in order to decrease the generation of paths with loops hence turning the algorithms more efficient.

Concerning the complexity order, Hansen [4] presented a family of graphs for which the number of non-dominated paths grows exponentially, proving the bicriterion shortest path problem to be intractable. He also showed the adaptation of label setting algorithm to the bicriterion shortest path problem to be pseudo-polynomial, that is, it solves this problem in polynomial time, depending on the instance's characteristics.

This labeling algorithm is supported by an adaptation of the Optimality Principle for the shortest path problem, which states that every non-dominated path is formed by non-dominated subpaths (also valid when determining loopless paths). It may be proved that if the network doesn't contain negative cycles then the bicriterion shortest path problem verifies the Optimality Principle. The algorithm constructs the tree of the non-dominated paths from s to every node, and a set of labels containing the costs of each of those paths is associated with each node. In each step the node with a lexicographically smallest label is chosen, assuring it

is definitive. If that label is not dominated it corresponds to a non-dominated path starting in s and it allows other nodes to be labeled.

Some computational tests comparing the label setting algorithm and the ranking algorithm (using MPS algorithm for loopless paths determination) were made, where the first one showed a better performance (see also [17]). However, for the specific problem to be solved one cannot generalize the Optimality Principle. Consider for instance the network represented in Figure 1(a) and the respective tree of paths represented in Figure 1(b), where π_i^c and π_i^d denote the cost and delay, respectively, of the path from s until i in that tree.

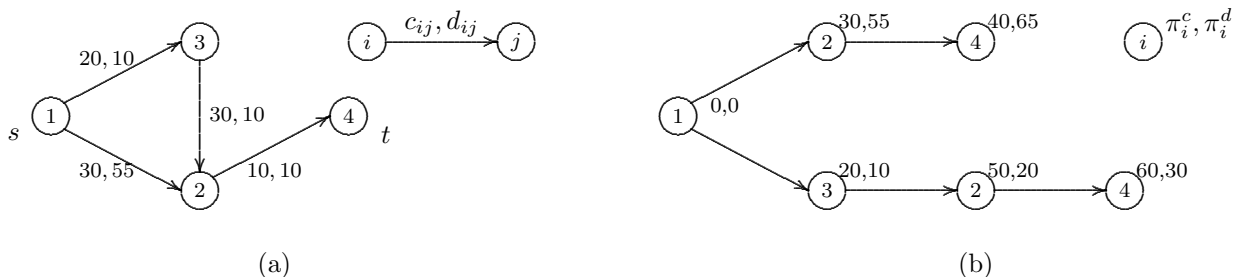


Figure 1: (a) Network $(\mathcal{N}, \mathcal{A})$; (b) Tree of paths rooted at s in $(\mathcal{N}, \mathcal{A})$

Notice that in order to simplify the example only the constraint associated with the path delay is considered, with $\Delta_{delay} = 60$. Recalling that it is intended to compute the path with the least cost and least number of arcs, it can be seen in Figure 1(b) that $p = \langle 1, 3, 2, 4 \rangle$ is the only solution of the problem. Nevertheless, the subpath of p from 1 until 2 is dominated by $\langle 1, 2 \rangle$, which is formed only by one arc and the cost of which is 30, lower than $c(\langle 1, 3, 2 \rangle) = 50$. Therefore one may conclude it is not possible to make the dominance test by the time a node is to become an element of the tree rooted at s , but only when it corresponds to the determination of a path from s until t . More tests were made using this procedure, but now the adaptation of label setting algorithm has shown to be less efficient than the adaptation of ranking loopless paths algorithm. Thus, in the following only the ranking algorithm approach will be considered. This approach consists mainly in using a ranking algorithm to list paths by non-decreasing costs, and using that ordered listing to choose the paths which are non-dominated. Paths are ranked until its cost exceeds an upper bound c^* previously determined, anyway the number of paths that has to be listed is not known in advance, therefore this is also a pseudo-polynomial algorithm, depending on that number.

Concerning the MPS algorithm's adaptation for ranking loopless paths, it has a very efficient performance from a computational point of view, both in terms of the running times and number of paths it generates in order to compute the intended loopless paths. These have been two of the reasons which led us to use this adaptation as a subroutine of the algorithm proposed in [8] for computing the set of non-dominated loopless paths in the multimedia traffic routing problem.

For a better understanding of the resolution of the problem, Clímaco and Martins' algorithm for the bicriterion problem and MPS algorithm will now be briefly described. With no loss of generality, c will be assumed to be the first objective function and h the second one, that is it will be assumed that the paths are ranked according to c .

As mentioned above, ranking paths in a bicriterion problem consists in determining paths by non-decreasing order of their costs, which allows us to partition \mathcal{P}_N in several subsets. Thus, some results the proof of which can be found in [7] and [8] and which support the bicriterion algorithm, will now be stated. The first one, Lemma 1, establishes a ranking stopping condition.

Lemma 1 *Let \mathcal{P}_c be the set of paths from s to t with minimum cost and \mathcal{P}_h be the set of paths from s to t with minimum number of arcs. Let $h^* = \min\{h(p) : p \in \mathcal{P}_c\}$ and $c^* = \min\{c(p) : p \in \mathcal{P}_h\}$. If p is a non-dominated path, then $h(p) \leq h^*$ and $c(p) \leq c^*$.*

The second result allows us to define the dominance test which determines whether a path is non-dominated

or not. Let S_c be the set $S_c = \{p \in \mathcal{P} : c(p) \leq c^*\}$ with a finite number of paths (which can be ordered by their costs c) and consider the following S_c partition

$$S_c = \bigcup_{i=1}^k S_c^i, \text{ where } S_c^i \cap S_c^j = \emptyset \text{ for any } i \neq j,$$

and, given $p \in S_c^i$ and $q \in S_c^j$:

- if $i < j$ then $c(p) < c(q)$,
- if $i = j$ then $c(p) = c(q)$.

Lemma 2 *Let $p^* \in S_c^i$, for some $i \in \{2, \dots, m\}$. Path p^* is non-dominated if and only if $h(p^*) = \min\{h(p) : p \in S_c^i\}$ and $h(q) > \min\{h(p) : p \in S_c^i\}$, for any $q \in S_c^j$ such that $j < i$.*

It is intended to generate the non-dominated solutions by an order, such that the cost values are non-decreasing, and thus the number of arcs should be non-increasing. Therefore, at a given step of the algorithm the non-dominated candidate paths are stored in a set designated by \mathcal{P}_X and when analysing $p \in S_c^i$, the dominance test will consist in comparing $c(p)$ and $h(p)$ with M_c and m_h , respectively, where M_c is the largest cost of the paths determined, while m_h is the smallest number of arcs of those paths. As paths are ranked by non-decreasing order of c , two situations may arise, $c(p) = M_c$ or $c(p) > M_c$. In the first case p is dominated by some other path if $m_h < h(p)$, otherwise it is a potential non-dominated path, thus it is stored in \mathcal{P}_X . In the second case a new set S_c^i starts being determined, therefore p will be the only element in \mathcal{P}_X .

However this problem isn't just a bicriterion shortest path problem, once additional constraints were imposed to the loopless paths, namely (2), (3) and (4). Hereafter the paths from s to t satisfying such constraints will be called feasible paths and its determination can be done by using a ranking paths algorithm adaptation, whether for general paths or loopless paths. In order to decrease the number of generated paths, non-dominated loopless candidates, the MPS algorithm for loopless paths [11] will be used. This procedure is now sketched in Algorithm 1.

Algorithm 1 – *Adaptation of a ranking algorithm for the shortest feasible loopless path problem with two objective functions*

```

 $p_h^* \leftarrow$  feasible loopless path with minimum number of arcs;  $\hat{c} \leftarrow c(p_h^*);$ 
 $p_c^* \leftarrow$  feasible loopless path with minimum cost;  $M_c \leftarrow c(p_c^*); m_h \leftarrow h(p_c^*);$ 
 $\mathcal{P}_X \leftarrow \emptyset; \mathcal{P}_N \leftarrow \emptyset; \text{continue} \leftarrow \text{True}; k \leftarrow 0;$ 
While (continue) Do
   $k \leftarrow k + 1;$ 
   $p_k \leftarrow$   $k$ -th shortest feasible loopless path in  $c$ ;
  If ( $c(p_k) = M_c$ ) Then
    If ( $h(p_k) = m_h$ ) Then  $\mathcal{P}_X \leftarrow \mathcal{P}_X \cup \{p_k\}$ 
    Else
      If ( $h(p_k) < m_h$ ) Then
         $\mathcal{P}_X \leftarrow \{p_k\}; m_h \leftarrow h(p_k)$ 
      EndIf
    EndIf
  Else
    If ( $h(p_k) < m_h$ ) Then
       $\mathcal{P}_N \leftarrow \mathcal{P}_N \cup \mathcal{P}_X; \mathcal{P}_X \leftarrow \{p_k\}; M_c \leftarrow c(p_k); m_h \leftarrow h(p_k);$ 
      If ( $c(p_k) > \hat{c}$ ) Then continue  $\leftarrow$  False
    EndIf
  EndIf
EndWhile

```

Note that since we are considering only feasible loopless paths, in the following \mathcal{P}_N will denote the set of non-dominated feasible solutions. In this description there are still some unanswered questions, such as the determination of the feasible loopless path with less arcs, the determination of the feasible loopless path with minimum cost and the feasible loopless paths ranking. It should also be noticed that in Algorithm 1 \hat{c} is not exactly the value c^* presented in Lemma 1, but an upper bound to the cost of the loopless paths to be determined (that is, $\hat{c} \geq c^*$). Before analysing these points, MPS algorithm for loopless paths ranking will be briefly described, once details can be found in [11].

As it has been referred to, this algorithm belongs to the class of deviation algorithms, which is characterized by using a set X of candidates to k -th shortest loopless path, for some $k \in \mathbb{N}$. The first loopless path being stored in X is the shortest one (obtained by solving the simple problem), after that the shortest path is repeatedly picked up from X . If the path chosen is loopless then it is a k -th shortest loopless path for some $k \in \mathbb{N}$, anyway its nodes are analysed, intending to generate new paths “deviating” from p at one of its nodes, candidates to following shortest loopless paths. Denoting by \mathcal{T}_t the shortest tree rooted at t and by $\mathcal{T}_t(i)$ the loopless path from $i \in \mathcal{N}$ to t in \mathcal{T}_t , each new path q is the shortest one deviating from p in the analysed node and it has the form $q = \text{sub}_p(s, d_q) \diamond (d_q, i) \diamond \mathcal{T}_t(i)$, where d_q , known as the deviation node of q , is the analysed node in p and where the arc (d_q, i) is computed in a specific manner. Path p is said to be the parent of q .

To simplify the determination of q the MPS algorithm begins by computing \mathcal{T}_t and after that computation, each arc’s cost is replaced by the respective reduced cost to allow to choose only the arc with lower reduced cost when finding the shortest deviation. In the following the reduced cost associated with a generic arc’s cost g_{ij} and the tree \mathcal{T}_t will be denoted by $\bar{g}_{ij} = g_{ij} + \pi_i - \pi_j$, where π_i is the cost of the path from i until t in \mathcal{T}_t , while the reduced cost of a path p will be given by $\bar{g}(p) = \sum_{(i,j) \in p} \bar{g}_{ij}$. Moreover the set \mathcal{A} is re-arranged and arcs are ordered according to the tail node, those with the same tail node are ordered according to the reduced cost. This is known as the sorted forward star form (see [12]) and its utilization, together with the reduced costs, allows the number of performed operations to decrease. As mentioned, the paths’ ranking begins by successively picking up the shortest element in X and generating new paths, by analysing its nodes following the deviation node. These new generated paths are also stored in X . For each analysed node x a path with the presented form, defined by the arc with tail node x following the one used in p , in the sorted forward star form, is computed.

Once the algorithms of Clímaco and Martins and MPS were already described, Algorithm 1 for the traffic routing problem will now be analysed with more detail.

Among the non specified steps we begin by pointing out the determination of the feasible loopless path with less arcs. In fact, one could think of using a shortest path algorithm to calculate the loopless path with minimum number of arcs from s to t . However, we cannot guarantee that this path satisfies (2), (3) and (4), therefore this procedure may not be sufficient to compute the feasible loopless path with less arcs. Yet, the computation of this loopless path can be made by using once again the MPS algorithm for ranking loopless paths according to the values of function h , but this algorithm has to be suitably adapted since its application has to end as the first feasible loopless path is computed. Furthermore some steps will also be added in order to decrease the number of generated non feasible paths.

As far as the determination of the least cost feasible loopless path is concerned, a similar procedure will be used, but now using the c values for ranking paths.

It should be noted that in the model used for simulating the traffic routing problem, the upper bound Δ_{jitter} depends on the loopless path with minimum number of arcs. This is the reason why in Algorithm 1 that follows, constraint (4) is not verified from the very beginning. The application model used for the traffic routing problem will be described in the next section.

It still remains to define the changes in each ranking in order to consider constraints (2) and (3), in the case of function h , and constraints (2), (3) and (4), in the case of function c . These constraints can be divided into two types: on one hand we have constraint (2) and on the other we have constraints (3) and (4), where the corresponding function is linear.

Let us start by analysing this second type of constraints, recalling that when computing loopless paths, X may contain paths with loops. However, we try to minimize the number of these kind of paths when generating new candidates and only nodes preceding the first one which forms a cycle in p (if it exists) are analysed, as well as only arcs which do not form a cycle with $\text{sub}_p(s, x)$ are considered. The case of constraints (3) and (4)

can be treated in an analogous way as Lemma 3 proves.

Lemma 3 *Let $p \in \mathcal{P}_{ij}$.*

1. *If $d(p) \geq \Delta_{\text{delay}}$ then $d(p \diamond (j, x)) > \Delta_{\text{delay}}$.*
2. *If $h(p) \geq \Delta_{\text{jitter}}$ then $h(p \diamond (j, x)) > \Delta_{\text{jitter}}$.*

According to this result one may conclude that if $d(\text{sub}_p(s, v_i)) \geq \Delta_{\text{delay}}$ or $h(\text{sub}_p(s, v_i)) \geq \Delta_{\text{jitter}}$ then neither p nor the paths still to be generated from the nodes following v_i in p will be feasible and thus its analysis can be avoided. Moreover a selection of the arcs picked up when analysing a given node can also be made in order to decrease the number of computed non feasible candidate paths. So, when analysing node v_i and considering an arc (v_i, j) it can be ignored if this is not feasible, that is if $d_{v_i, j} > \Delta_{\text{delay}}$. A similar analysis is not necessary if the paths are ranked according to the number of arcs.

With respect to constraint (2) function $b(p) = \min_{(i, j) \in p} \{b_{ij}\}$ is not linear therefore the same procedure cannot be used. However, Lemma 4 allows us to solve this problem.

Lemma 4 *Let $p \in \mathcal{P}_{ij}$. If $b_{uv} < \Delta_{\text{bandwidth}}$ for some arc (u, v) of p , then $b(p) < \Delta_{\text{bandwidth}}$.*

Based on this lemma, one may conclude that no arc verifying $b_{ij} < \Delta_{\text{bandwidth}}$ can belong to the solutions we want to compute and its removal from \mathcal{A} assures that every path in the modified network satisfies constraint (2).

Thus the algorithm for solving the traffic routing problem will be based on the adaptation of MPS algorithm for the bicriterion shortest loopless path problem, where the arcs which do not verify $b_{ij} \geq \Delta_{\text{bandwidth}}$ are removed from $(\mathcal{N}, \mathcal{A})$; where the feasibility of paths is checked and the conditions of arc choices when generating candidate paths are changed (recalling that constraints (3) and (4) are included in the former algorithm).

For simplicity reasons the next algorithm will be divided into two parts, the first of which, Algorithm 2, refers to the computation of the feasible loopless path with minimum cost, when considering a generic cost g .

Algorithm 2 – *MPS algorithm adaptation – Determination of a feasible loopless path with minimum cost g*

```

Delete arcs  $(i, j) \in \mathcal{A}$  such that  $b_{ij} < \Delta_{\text{bandwidth}}$  from the network;
 $\mathcal{T}_t \leftarrow$  tree of the shortest paths from  $i \in \mathcal{N}$  to  $t$  according to  $g$ ;
 $p_g \leftarrow \mathcal{T}_t(s)$ ;
If ( $p_g$  is not defined) Then Stop; /* There are no feasible loopless paths */
 $\bar{g}_{ij} \leftarrow \pi_i - \pi_j + g_{ij}, \forall (i, j) \in \mathcal{A}$ ;
Represent  $\mathcal{A}$  in the sorted forward star form according to  $\bar{g}$ ;
 $d_{p_g} \leftarrow s$ ;  $X \leftarrow \{p_g\}$ ; feasible  $\leftarrow$  False;
While (( $X \neq \emptyset$ ) and (not feasible)) Do
   $p \leftarrow$  path in  $X$  such that  $\bar{g}(p)$  is minimum; /*  $p = \langle s \equiv v_1, v_2, \dots, v_{\ell-1}, v_{\ell} \equiv t \rangle$  */
   $X \leftarrow X - \{p\}$ ;
   $i \leftarrow$  index such that  $v_i = d_p$ ;
  While (( $v_i \neq t$ ) and (sub $_p(s, v_i)$  is loopless) and (constraints are satisfied)) Do
     $l \leftarrow$  index such that  $a_l = (v_i, v_{i+1})$ ;  $p_i \leftarrow \text{sub}_p(s, v_i)$ ;
    While (( $v_i$  is the tail node of  $a_l$ ) and (( $a_{l+1}$  forms a loop with  $p_i$ ) or (constraints aren't satisfied))) Do
       $l \leftarrow l + 1$ 
    EndWhile
    If ( $v_i$  is the tail node of  $a_l$ ) Then
       $v_j \leftarrow$  head node of  $a_l$ ;  $q \leftarrow p_i \diamond a_l \diamond \mathcal{T}_t(v_j)$ ;  $d_q \leftarrow v_i$ ;  $X \leftarrow X \cup \{q\}$ 
    EndIf
     $v_i \leftarrow v_{i+1}$ 
  EndWhile
  If (( $p$  is loopless) and (constraints are satisfied)) Then feasible  $\leftarrow$  True
EndWhile
If (not feasible) Then Stop; /* There are no feasible loopless paths */

```

It should be noticed that for determining the tree of the shortest paths with less arcs in Algorithm 2 (that is, considering $g_{ij} = 1$, for any $(i, j) \in \mathcal{A}$), it is sufficient to use a label correcting algorithm, where the set of nodes' labels is manipulated as a FIFO. Moreover, (assuming that arcs not verifying the bandwidth constraint were deleted) when ranking paths according to their number of arcs constraint (3) has to be verified, and while ranking paths according to c the constraints to be checked are (3) and (4).

The correction of Algorithm 2 follows from the correction of MPS algorithm, noticing that non feasible paths are not considered, and that the generation of a new path is avoided whenever we can assure it is not feasible and it cannot originate any feasible solution (once its subpath from s until the analysed node violates at least one of the constraints).

Also recall that after computing p , the feasible loopless path with minimum number of arcs (when $g_{ij} = 1$, for any $(i, j) \in \mathcal{A}$), Δ_{jitter} is updated and the upper bound \hat{c} , to be used in the enumeration of non-dominated loopless paths, can be obtained by considering $\hat{c} = c(p)$. Analogously, after computing q , the least cost feasible loopless path (when $g_{ij} = c_{ij}$, for any $(i, j) \in \mathcal{A}$), values M_c and m_h can be obtained by using $M_c = c(q)$ and $m_h = h(q)$.

In a general manner Algorithm 3 consists in MPS algorithm when constraints are verified (as in Algorithm 2) adding the dominance test whenever a k -th shortest loopless feasible path is determined. The computation of set \mathcal{P}_N is summarized in Algorithm 3.

Algorithm 3 – MPS algorithm adaptation for the traffic routing problem – Determination of \mathcal{P}_N

```

 $\mathcal{P}_X \leftarrow \{p_c^*\}; \text{continue} \leftarrow \text{True};$ 
While  $((X \neq \emptyset)$  and  $\text{continue})$  Do
   $p \leftarrow$  path in  $X$  such that  $\bar{c}(p)$  is minimum; /*  $p = \langle s \equiv v_1, v_2, \dots, v_{\ell-1}, v_{\ell} \equiv t \rangle$  */
   $X \leftarrow X - \{p\};$ 
   $i \leftarrow$  index such that  $v_i = d_p;$ 
  While  $((v_i \neq t)$  and  $(\text{sub}_p(s, v_i)$  is loopless) and
     $(d(\text{sub}_p(s, v_i)) < \Delta_{\text{delay}})$  and  $(h(\text{sub}_p(s, v_i)) < \Delta_{\text{jitter}}))$  Do
     $l \leftarrow$  index such that  $a_l = (v_i, v_{i+1}); p_i \leftarrow \text{sub}_p(s, v_i);$ 
    While  $((v_i$  is the tail node of  $a_l)$  and
       $((a_{l+1}$  forms a loop with  $p_i)$  or  $(d(p_i \diamond a_{l+1}) > \Delta_{\text{delay}})$  or  $(h(p_i \diamond a_{l+1}) > \Delta_{\text{jitter}}))$  Do
       $l \leftarrow l + 1$ 
    EndWhile
    If  $(v_i$  is the tail node of  $a_l)$  Then
       $v_j \leftarrow$  head node of  $a_l; q \leftarrow p_i \diamond a_l \diamond \mathcal{T}_t(v_j); d_q \leftarrow v_i; X \leftarrow X \cup \{q\}$ 
    EndIf
     $v_i \leftarrow v_{i+1}$ 
  EndWhile
  If  $((p$  is loopless) and  $(d(p) \leq \Delta_{\text{delay}})$  and  $(h(p) \leq \Delta_{\text{jitter}}))$  Then
    If  $(c(p) = M_c)$  Then /* Dominance test */
      If  $(h(p) = m_h)$  Then  $\mathcal{P}_X \leftarrow \mathcal{P}_X \cup \{p\};$ 
      Else
        If  $(h(p) < m_h)$  Then
           $\mathcal{P}_X \leftarrow \{p\}; m_h \leftarrow h(p)$ 
        EndIf
      EndIf
    Else
      If  $(h(p) < m_h)$  Then
         $\mathcal{P}_N \leftarrow \mathcal{P}_N \cup \mathcal{P}_X; \mathcal{P}_X \leftarrow \{p\}; M_c \leftarrow c(p); m_h \leftarrow h(p);$ 
        If  $(c(p) > \hat{c})$  Then  $\text{continue} \leftarrow \text{False}$ 
      EndIf
    EndIf
  EndIf
EndWhile

```

An analogous algorithm (suitably adapted from Algorithms 2 and 3) could also be sketched by ranking paths according to h instead of c , that is, according to the loopless paths number of arcs.

It should also be remarked that, even if the initial network is connected, some of its paths may not satisfy (2), (3) or (4). Therefore it is possible that no loopless path in $(\mathcal{N}, \mathcal{A})$ is feasible and that the subset of \mathcal{P}_N we want to compute is empty, cases in which the problem has no solution.

4 Application

In order to show the applicability and performance of the algorithm we now present a specific model of application to a routing problem of video traffic in a ATM (Asynchronous Transfer Mode) type network. Firstly we will describe the specific model for this application and then results from extensive computational experiments will be presented and discussed. We also show computational results of using the developed algorithm on a network obtained from real geographic coordinates.

4.1 Application model

We begin by presenting the simulation of the video traffic routing problem in undirected communication networks with n nodes and $m = 4n$ arcs. Each node of the network corresponds to a point randomly chosen in a rectangular grid with dimension 400×240 and a mesh size unit of 10 Km. Each node is adjacent to at least 2 and at most 10 other nodes (recall that the average node degree of the network is 4). Besides, the generated networks contain at least an Hamiltonian path, in order to assure that $\mathcal{P}_{si} \neq \emptyset$ and $\mathcal{P}_{it} \neq \emptyset$ for any $i \in \mathcal{N}$.

The video traffic routing problem was simulated using a flow specification $(\sigma_k, r_k, S_{\max}^k)$, where:

- $\sigma_k = 10 S_{\max}$ denotes the token bucket size or maximum burst size (in bits),
- $r_k = r = 1.5 \times 10^6$ bit/s is the token generation rate of the leaky bucket (stochastic model associated with the nodes),
- $S_{\max}^k = S_{\max}$ is the maximum packet size of the flow k (in bits),

and $S_{\max} = 53 \times 8$ bit which is the size of an ATM cell.

As in [16] it was assumed that each node in the communication network is modeled as a queueing system which uses Weighted Fair Queueing (WFQ) service discipline, allowing us to formulate the problem as follows, concerning the delay and jitter constraints. In order to try to obtain the lowest call blocking probability, the cost function we used was $c(p) = \sum_{(i,j) \in p} c_{ij}$, where $c_{uv} = \frac{1}{b_{uv}}$ for any $(u, v) \in \mathcal{A}$. Thus, the values associated with each arc (i, j) , where i and j correspond, respectively, to points (x_i, y_i) and (x_j, y_j) in the initial grid, are:

- its available bandwidth, in Mb/s, that is a random value denoted by $b_{ij} \in \{0.52, 2.52, \dots, 150.52\}$, which corresponds to a link capacity of 155.52 Mb/s;
- its cost, given by $c_{ij} = \frac{1}{b_{ij}}$;
- its delay, in ms, given by $d_{ij} = \left(\frac{S_{\max}^k}{r_k} + \frac{S_{\max}}{R_{ij}} \right) + \frac{\ell_{ij}}{2c/3} = \left(\frac{53 \times 8}{1.5} + \frac{53 \times 8}{155.52} \right) \times 10^{-3} + \frac{\ell_{ij}}{200}$, where $\ell_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ represents the Euclidean distance, in Km, between (x_i, y_i) and (x_j, y_j) , $c = 300$ Km/ms is the speed of light and for any $(u, v) \in \mathcal{A}$, $R_{uv} = 155.52 \times 10^6$ bit/s is the bandwidth capacity of arc (u, v) .

Concerning the problem's constraints, we considered that every path should have, at least a bandwidth capacity of $r_k = 1.5$ Mb/s (transmission rate required by the video traffic), that is, $\Delta_{\text{bandwidth}} = 1.5$, and that the delay upper bound Δ_{delay} varied in $\{10, 15, \dots, 60\}$ ms. Since we are assuming a WFQ service discipline, the jitter constraint can be expressed in terms of the number of arcs, namely $\Delta_{\text{jitter}} = ma(s, t) + \Delta_{\text{arcs}}$, where $ma(s, t)$ denotes the minimum number of arcs of a feasible loopless path from s to t and $\Delta_{\text{arcs}} \in \{2, 4\}$.

Firstly the bandwidth values were randomly obtained in $\{0.52, 2.52, \dots, 150.52\}$. Later, other methods were used for generating these values in order to study the number of non-dominated solutions in different problems.

Therefore, we have considered two types of networks concerning the bandwidth values. The first type consists in partitioning $\{0.52, 2.52, \dots, 150.52\}$ in classes, assigning to each one a predefined percentage of b_{ij} values. In the experiments we performed, five classes with equal amplitude and three different distributions have been considered. So, the set $\{0.52, 2.52, \dots, 150.52\}$ was partitioned in

$$\{0.52 + 2k : k = 0, \dots, 75\} = \bigcup_{i=0}^4 I_i,$$

where

$$I_i = \{0.52 + 2k : k = 15i, \dots, 15(i+1) - 1\}, \quad i = 0, 1, 2, 3, \quad \text{and} \quad I_4 = \{0.52 + 2k : k = 60, \dots, 75\}.$$

The used bandwidth distributions were those presented in Table 1. It should be noticed that using Distribution 1

	I_0	I_1	I_2	I_3	I_4
Dist 1	20%	20%	20%	20%	20%
Dist 2	50%	20%	15%	10%	5%
Dist 3	41%	6%	6%	6%	41%

Table 1: Bandwidth distributions

corresponds simply to generate values uniformly in $\{0.52, 2.52, \dots, 150.52\}$, which was the initial procedure.

In the second type of experiments the bandwidth values were generated considering the following spatial partition of the original grid according to x :

$$\{0, 10, \dots, 400\} \times \{0, 10, \dots, 240\} = (\{0, \dots, 120\} \cup \{130, \dots, 250\} \cup \{260, \dots, 400\}) \times \{0, \dots, 240\}.$$

Thus, given the arc (i, j) , where i and j correspond, respectively, to points (x_i, y_i) and (x_j, y_j) in the initial grid:

- if $x_i, x_j \in \{0, 10, \dots, 120\}$, then $b_{ij} \in \{0.52 + 2k : k = 0, \dots, 24\}$;
- if $x_i, x_j \in \{130, 140, \dots, 250\}$, then $b_{ij} \in \{0.52 + 2k : k = 25, \dots, 49\}$;
- if $x_i, x_j \in \{260, 270, \dots, 400\}$, then $b_{ij} \in \{0.52 + 2k : k = 50, \dots, 75\}$;
- otherwise $b_{ij} \in \{0.52 + 2k : k = 0, \dots, 75\}$.

Another type of network was considered, which was obtained by using geographic coordinates of 1088 US cities of the following states:

Alabama, Arizona, Arkansas, California, Colorado, Connecticut, Delaware, District of Columbia, Florida, Georgia, Idaho, Illinois, Indiana, Iowa, Kansas, Kentucky, Louisiana, Maine, Maryland, Massachusetts, Michigan, Minnesota, Mississippi, Missouri, Montana, Nebraska, Nevada, New Hampshire, New Jersey, New Mexico, New York, North Carolina, North Dakota, Ohio, Oklahoma, Oregon, Pennsylvania, Rhode Island, South Carolina, South Dakota, Tennessee, Texas, Utah, Vermont, Virginia, Washington, West Virginia, Wisconsin, Wyoming.

These data can be reached on the Internet, at the url www.realestate3d.com/gps/latlong.htm. Considering those cities as nodes of the network $m = 8n$ arcs between them were randomly generated (assuring the network is connected). The bandwidth values were randomly generated in $\{0.52, 2.52, \dots, 150.52\}$ and the additional constraints considered for this problem were the previous ones.

The obtained results are summarized in the following subsection.

4.2 Computational tests and results

In the performed tests undirected networks with $n \in \{500, 1000, \dots, 3000\}$ nodes have been used, the number of arcs being $m = 4n$. For each value of n we have generated networks using 10 distinct seeds and for each of them the algorithm has been tested by considering $\frac{n^2}{25000}$ origin-destination node pairs. Table 2 shows the number of origin-destination node pairs for each network and the total number of problems solved for each network's size.

n	Number of s - t pairs for each seed	Total number of problems solved
500	10	100
1000	40	400
1500	90	900
2000	160	1600
2500	250	2500
3000	360	3600

Table 2: Number of video traffic routing problems solved.

As for the US coordinates network $m = 5n$ ($n = 1088$) arcs were used and 20 origin-destination node pairs were considered for each one of the 10 seeds chosen.

The code was written in C language and the computational results were obtained in a AMD Athlon server at 1.5 GHz, with 256 Mbytes of RAM and running over Linux.

As referred to, several values of n and Δ_{delay} have been considered, and for each of these pairs the average values corresponding to certain features of the problem were calculated. The plots and tables presented bellow show the variation rate of the average values of some of these features, depending on the parameters n and Δ_{delay} . The jitter upper bound was set to $\Delta_{\text{jitter}} = ma(s, t) + \Delta_{\text{arcs}}$, with $\Delta_{\text{arcs}} = 2$, except while studying the number of non-dominated solutions of the problem for the randomly generated networks, situations in which it was also considered the case $\Delta_{\text{arcs}} = 4$. In the case of the US city network the values $\Delta_{\text{arcs}} \in \{5, 6\}$ were also used for testing purposes.

The relative number of problems with one feasible ideal solution or 2, 3 or 4 non-dominated solutions found by the algorithm for the various empirical statistical distributions of available bandwidth values, are represented in Figures 2 to 4, as a function of the number of network nodes and of the delay constraint. In each figure two sets of curves, one for $\Delta_{\text{arcs}} = 2$ and the other for $\Delta_{\text{arcs}} = 4$, are shown. Also the relative number of problems with one feasible ideal solution or 2, 3 or 4 non-dominated solutions found by the algorithm for the spatial partition mentioned in the previous subsection are displayed in Figure 5.

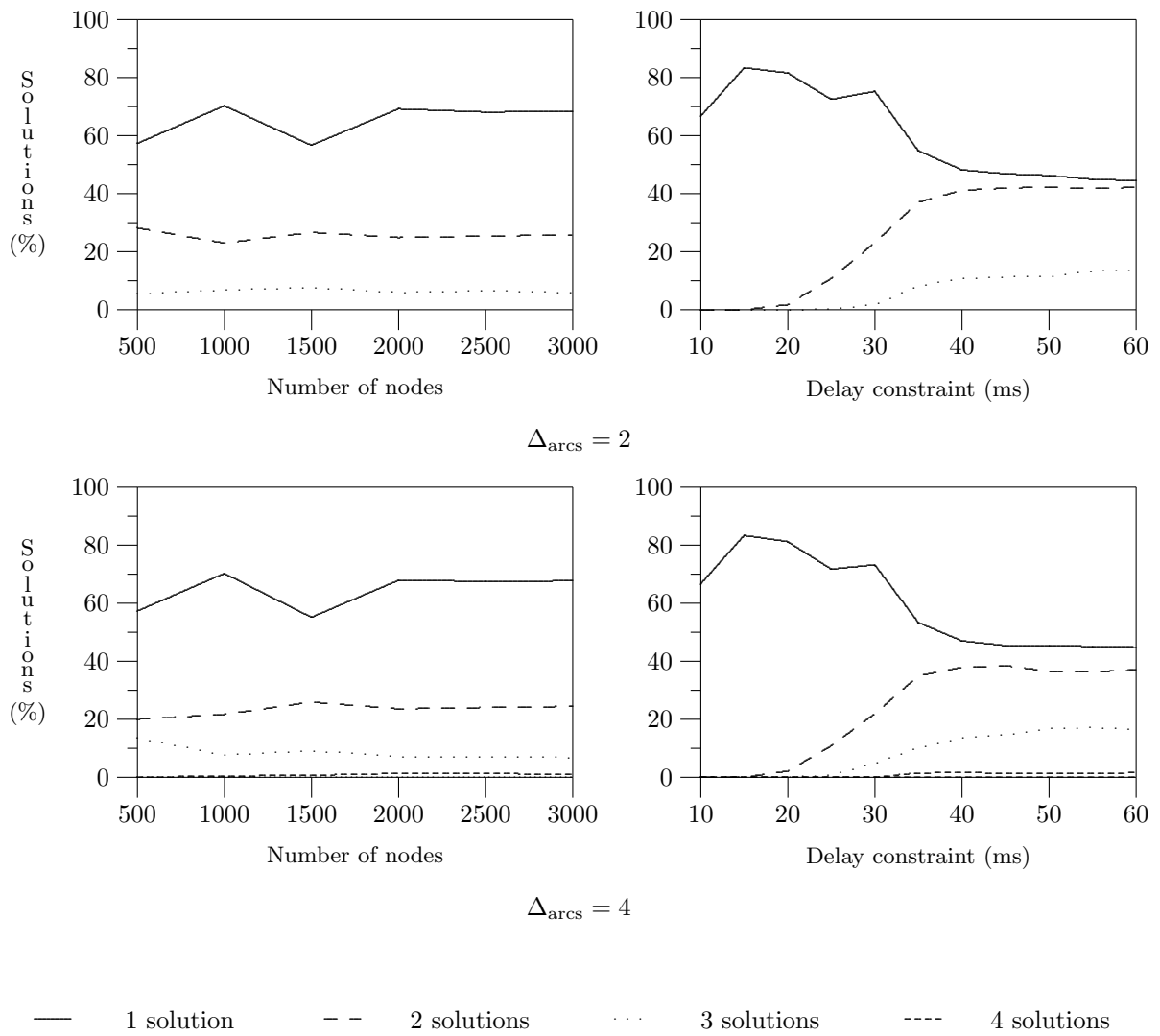


Figure 2: Percentage of problems with 1, 2, 3 and 4 non-dominated solutions – Distribution 1.

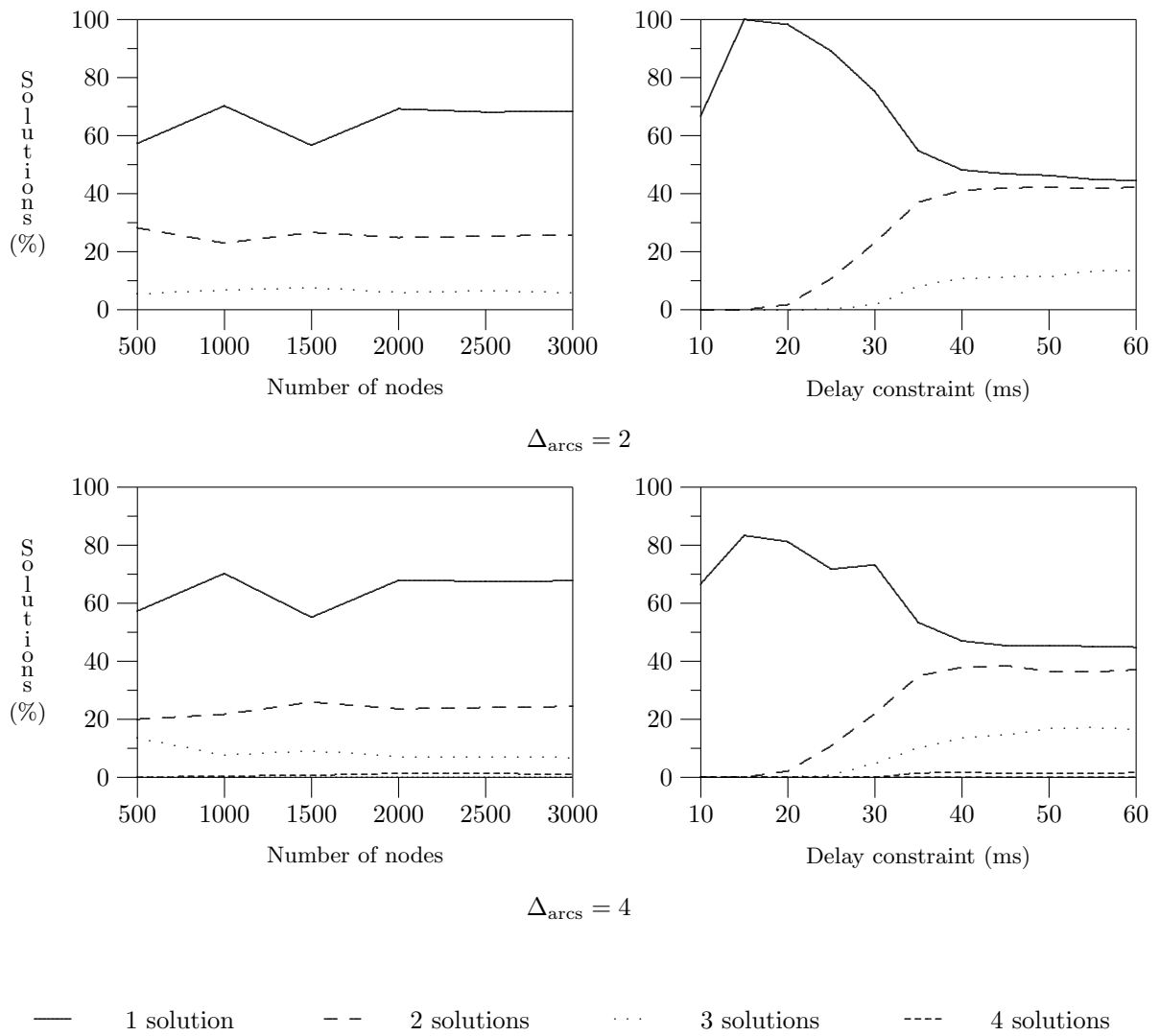


Figure 3: Percentage of problems with 1, 2, 3 and 4 non-dominated solutions – Distribution 2.

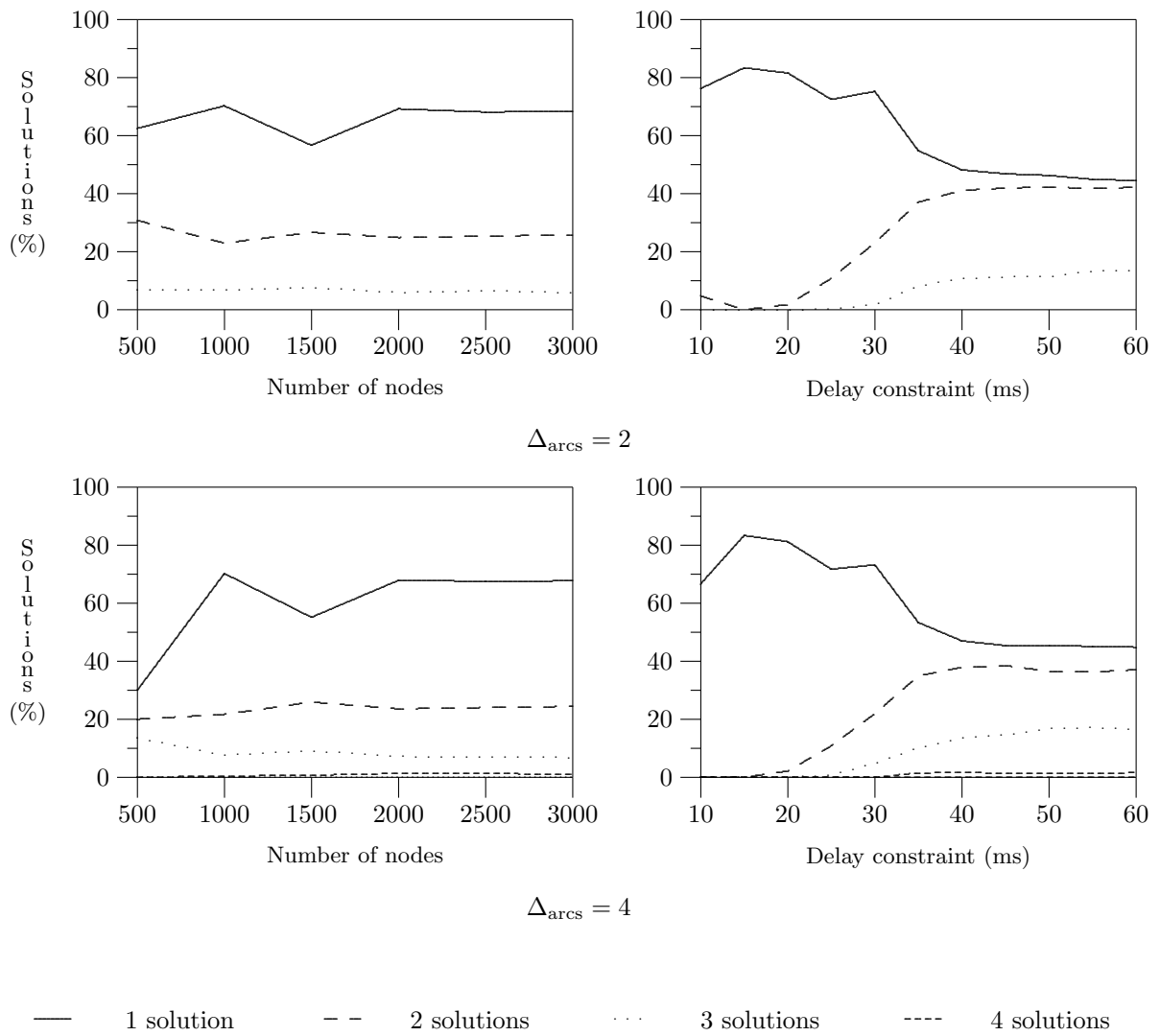


Figure 4: Percentage of problems with 1, 2, 3 and 4 non-dominated solutions – Distribution 3.

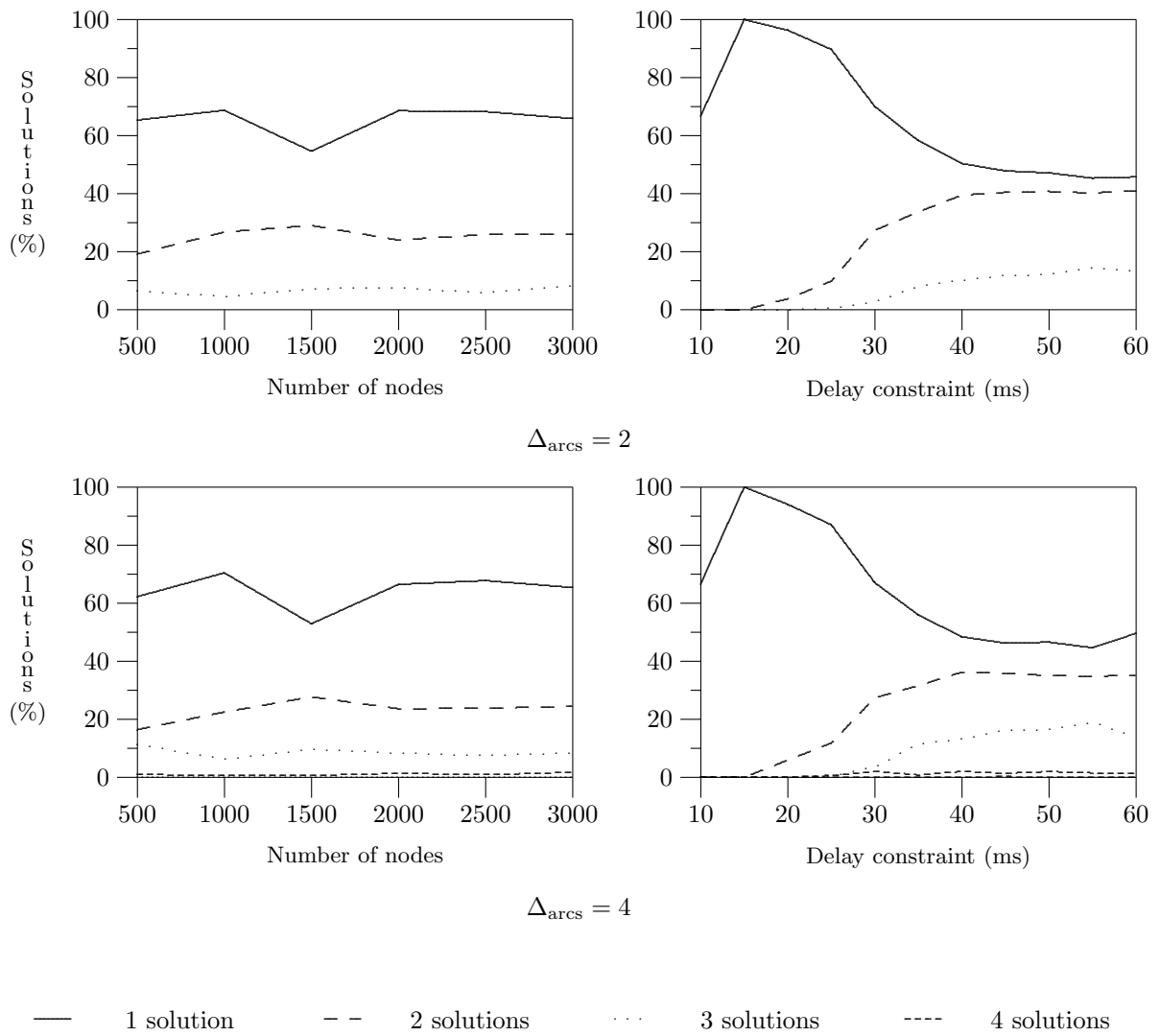


Figure 5: Percentage of problems with 1, 2, 3 and 4 non-dominated solutions – Vertical partition.

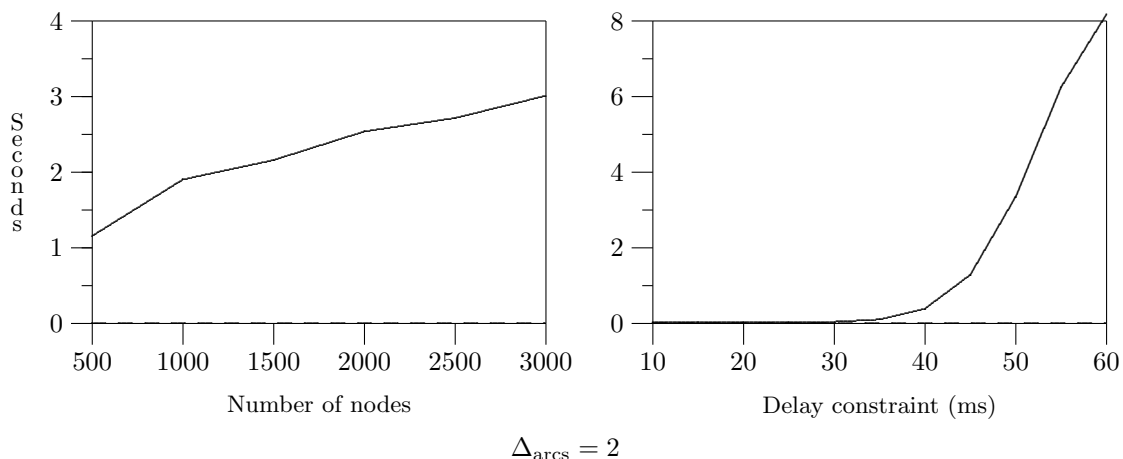


Figure 6: Running times of video traffic.

From these results it is apparent that the increase in the delay constraint (allowing paths with increased length, as shown in Figure 7 where the minimum, the average and the maximum number of arcs of the non-dominated solutions, are presented) tends to provoke an increase in the number of problems with greater number of non-dominated solutions, reflecting an increased conflict between the objective functions, for the feasible paths. The increase in the jitter bound parameter Δ_{arcs} from 2 to 4 has even a more relevant impact in the increase in the number of non-dominated solutions by allowing the occurrence of problems with 4 non-dominated solutions. It may be also concluded that the profile of the problem non-dominated solutions does not vary significantly with the empirical statistical or spatial distribution of the available bandwidths in the arcs so that we may consider, with some degree of confidence, that the previous conclusions are representative of the typical features of this type of problems. Also note that there are some problems with no feasible solution, a situation which, when occurring in practice, would naturally require the relaxation of some constraints.

The computational efficiency of the algorithm may be illustrated through Figure 6 where running times are represented as a function of the number of nodes (values averaged over all delay constraints) and of the delay constraint (values averaged over all considered number of nodes).

The plots in Figures 8 and 9 show the analogous results obtained with the US network. In general the results for this network follow the same trends obtained for the randomly generated networks. Taking into account the major conditions considered for generating the random networks and the characteristics of the US network these results are not surprising. As for the experiments with higher Δ_{arcs} values, namely $\Delta_{\text{arcs}} = 6$ (see Figure 8), the number of non-dominated solutions tends to increase as an expected result of the increase in the upper bound in the jitter constraint. However, it should be noticed that increasing Δ_{arcs} implies a substantial increase in the number of feasible paths, hence leading in some cases to a situation where the program exceeds the maximum space of memory allowed, and thus the computation of non-dominated solutions is not completed.

Note that this algorithmic approach while giving all the non-dominated solutions enables to solve problems of great dimension in a few seconds. We considered the number of nodes varying from hundreds to thousands which may be of practical interest in applications to WANs (Wide Area Networks) where the number of nodes (representing routers or switches) may be very high. By contrast the heuristics such as Pornavalai *et al.* [16] have only been tested with some success with networks having up to 500 nodes.

Overall it may be concluded that although the objective functions are not strongly conflicting (as a result of the significant correlation between cost and number of arcs) there is a significant percentage of problems with 2, 3 and even 4 non-dominated solutions that are calculated exactly in short processing times and with modest memory requirements. In many situations we seek only one solution for each node pair, which could be selected from the non-dominated solutions set by some practical preference aggregation rule, privileging more or less the cost or the number of links. Concerning the above mentioned heuristic methods none of the main features of the proposed approach is possible since they are approximate methods where there is not the guarantee of

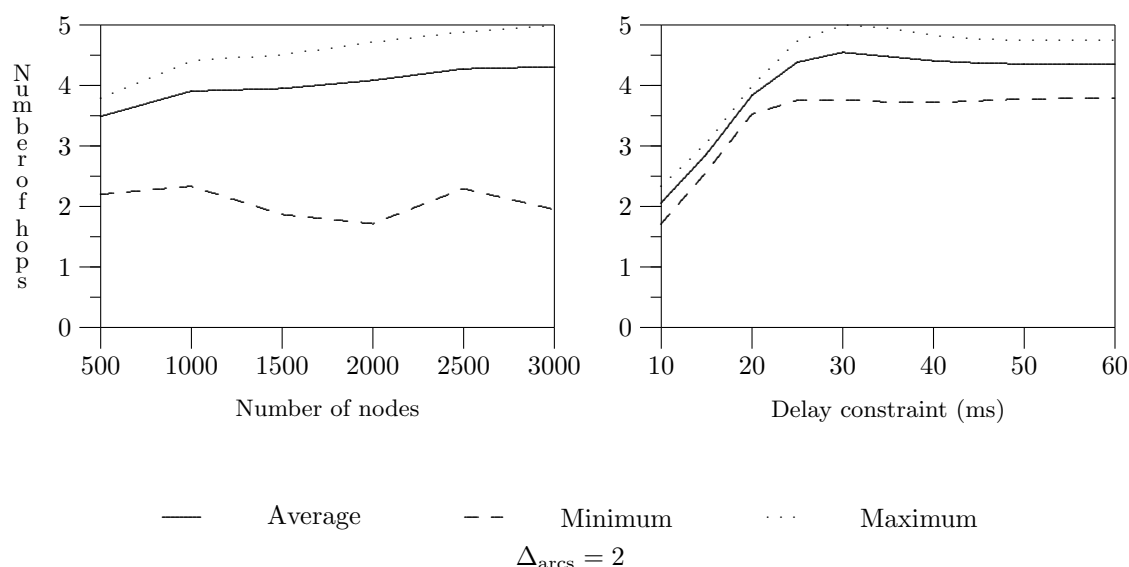


Figure 7: Number of arcs of video traffic.

finding an acceptable solution in a reasonable time, or even less that it is an non-dominated solution.

5 Conclusions

Routing problems in communication networks supporting multiple services, namely multimedia applications, involve the selection of paths satisfying multiple constraints of a technical nature, designated as QoS (Quality of Service) requirements and seeking simultaneously to “optimise” the chosen objective functions. Typical objective functions are the number of arcs and the cost of accepting a call in each arc, related with the bandwidth available in each link. As for the constraints on the paths, in the case of multimedia applications, these are typically the minimum bandwidth required by the call and the maximum allowed delay and jitter. Although traditional models in this area were single-objective, in many situations it is important to consider different, eventually conflicting objectives.

Having in mind to explore the multicriterion nature of this type of problem we developed in this paper a bicriterion model dedicated to calculating the whole set of non-dominated paths for traffic flows associated with multimedia type services in multiservice networks. In this context the major issue dealt with was the adaptation of a ranking type approach for a bicriterion shortest path problem including additional constraints. For this purpose an exact algorithm was presented based on the bicriterion shortest path algorithm by Clímaco and Martins [8] and on the MPS algorithm [11, 12].

This model was applied to a specific routing problem of video traffic in a high-speed network and extensive computational results were obtained and discussed.

Overall we may conclude that although the objective functions are not strongly conflicting there is a significant percentage of problems with 2, 3 and even 4 non-dominated solutions that are calculated exactly in short processing times and with modest memory requirements. This makes the model attractive and flexible for the resolution of the routing problem in real time for up to a few hundred nodes or in short time periods (as for example in the context of certain type of dynamic routing methods in public switched networks) for up to thousands of nodes, having in mind the characteristics of the used computer system.

Acknowledgments

The authors thank the anonymous referees for the helpful comments on the initial draft of this paper.

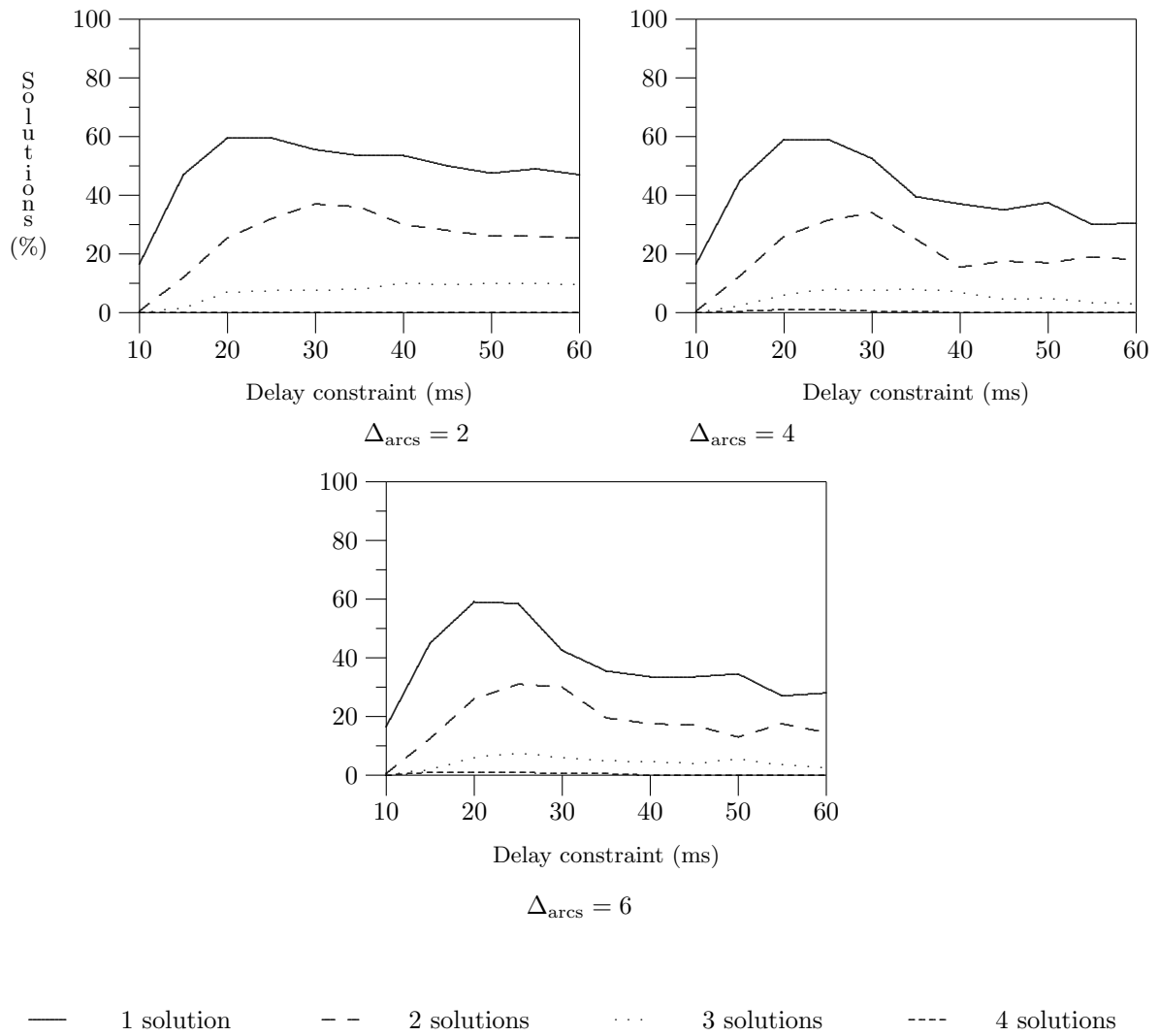


Figure 8: Percentage of problems with 1, 2, 3 and 4 non-dominated solutions in the USA video traffic network.

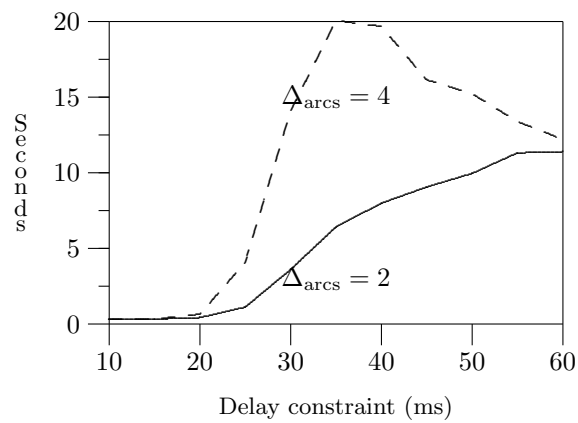


Figure 9: Average running times of the USA video traffic network.

References

- [1] J. A. Azevedo, M. E. O. S. Costa, J. J. E. R. S. Madeira, and E. Q. V. Martins. An algorithm for the ranking of shortest paths. *European Journal of Operational Research*, 69:97–106, 1993.
- [2] J. A. Azevedo, J. J. E. R. S. Madeira, E. Q. V. Martins, and F. M. A. Pires. A shortest paths ranking algorithm, 1990. Proceedings of the Annual Conference AIRO'90, Models and Methods for Decision Support, Operational Research Society of Italy, 1001–1011.
- [3] J. A. Azevedo, J. J. E. R. S. Madeira, E. Q. V. Martins, and F. M. A. Pires. A computational improvement for a shortest paths ranking algorithm. *European Journal of Operational Research*, 73:188–191, 1994.
- [4] 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.
- [5] V. P. Kompella, J. C. Pasquale, and G. C. Polyzos. Multicast routing for multimedia communications. *IEEE/ACM Transactions on Networking* 1, 3:286–292, 1993.
- [6] W. C. Lee, M. G. Hluchyj, and P. A. Humblet. Routing subject to quality of service constraints in integrated communication networks. *IEEE Network*, pages 46–55, July/August 1995.
- [7] J. C. N. Clímaco and E. Q. V. Martins. On the determination of the nondominated paths in a multiobjective network problem. Proceedings of V Symposium über Operations Research, Köln, (1980), in *Methods in Operations Research*, 40, (Anton Hain, Königstein, 1981), 255–258.
- [8] J. C. N. Clímaco and E. Q. V. Martins. A bicriterion shortest path algorithm. *European Journal of Operational Research*, 11:399–404, 1982.
- [9] E. Q. V. Martins. An algorithm for ranking paths that may contain cycles. *European Journal of Operational Research*, 18:123–130, 1984.
- [10] E. Q. V. Martins. On a multicriteria shortest path problem. *European Journal of Operational Research*, 16:236–245, 1984.
- [11] E. Q. V. Martins, M. M. B. Pascoal, and J. L. E. Santos. A new algorithm for ranking loopless paths. *Research Report, CISUC*, 1997. (<http://www.mat.uc.pt/~marta/Publicacoes/mps.ps.gz>).
- [12] E. Q. V. Martins, M. M. B. Pascoal, and J. L. E. Santos. Deviation algorithms for ranking shortest paths. *The International Journal of Foundations of Computer Science*, 10(3):247–263, 1999. (<http://www.mat.uc.pt/~marta/Publicacoes/deviation.ps.gz>).
- [13] E. Q. V. Martins, M. M. B. Pascoal, and J. L. E. Santos. A new improvement for a k shortest paths algorithm. *Investigação Operacional*, 21(1):47–60, 2001. (http://www.mat.uc.pt/~marta/Publicacoes/IMP_MS.ps.gz).
- [14] E. Q. V. Martins and J. L. E. Santos. A new shortest paths ranking algorithm. *Investigação Operacional*, 20(1):47–62, 2000. (<http://www.mat.uc.pt/~eqvm/cientificos/investigacao/Artigos/K.ps.gz>).
- [15] M. Mueller-Hannemann and K. Weihe. Pareto shortest paths is often feasible in practice. In *Proceedings of the 5th International WAE'01, Lecture Notes in Computer Science*, volume 2141, pages 185–198. Aarhus, Denmark, 2001.
- [16] C. Pornavalai, G. Chakraborty, and N. Shiratori. Routing with multiple qos requirements for supporting multimedia applications. *Telecommunication Systems*, 9:357–373, 1998.
- [17] J. L. E. Santos. Uma abordagem ao problema do trajecto óptimo multiobjectivo. *Investigação Operacional*, 19(2):211–226, 1999.

- [18] A. Skriver. A classification of bicriterion shortest path (bsp) algorithms. *Asia-Pacific Journal of Operational Research*, 17(2):192–212, 2000.
(<http://www.imf.au.dk/cgi-bin/w3-mysql/publications/genericpublication.html?publ=242>).