Computers & Operations Research 38 (2011) 1610–1619

Contents lists available at ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

Reduction approaches for robust shortest path problems

Daniele Catanzaro *, Martine Labbé, Martha Salazar-Neumann

Graphes et Optimisation Mathématique (G.O.M.), Computer Science Department, Université Libre de Bruxelles (U.L.B.), Boulevard du Triomphe, CP 210/01, B-1050 Brussels, Belgium

ARTICLE INFO

Available online 18 February 2011

Keywords:
Minimax regret optimization
Robust optimization
Shortest path problem
Interval data
Preprocessing
Pegging test

ABSTRACT

We investigate the uncertain versions of two classical combinatorial optimization problems, namely *the Single-Pair Shortest Path Problem* (SP-SPP) and *the Single-Source Shortest Path Problem* (SS-SPP). The former consists of finding a path of minimum length connecting two specific nodes in a finite directed graph G; the latter consists of finding the shortest paths from a fixed node to the remaining nodes of G. When considering the uncertain versions of both problems we assume that cycles may occur in G and that arc lengths are (possibly degenerating) nonnegative intervals. We provide sufficient conditions for a node and an arc to be always or never in an optimal solution of the *Minimax regret Single-Pair Shortest Path Problem* (MSP-SPP). Similarly, we provide sufficient conditions for an arc to be always or never in an optimal solution of the *Minimax regret Single-Source Shortest Path Problem* (MSS-SPP). We exploit such results to develop pegging tests useful to reduce the overall running time necessary to exactly solve both problems.

1. Introduction

A classical problem in combinatorial optimization consists of finding a path of minimum length connecting two specific nodes of a given finite directed weighted graph G. This problem is known in the literature as *the Shortest Path Problem* (SPP) and arises in a wide variety of practical problem settings, both as stand-alone models and as subproblems in more complex problem settings. For example, the SPP arises in the telecommunications and transportation industries whenever one wants to send a message or a vehicle between two geographical locations as quickly or as cheaply as possible. Similarly, the SPP arises in urban traffic planning when drivers are assumed to move along shortest paths from their origins to their destinations [1].

Various versions of the SPP are described in the literature, mainly differing from each other in the type of origin-destination considered. In particular, we can distinguish between: *the Single-Pair Shortest Path Problem* (SP-SPP) in which one aims to find the path of minimum length connecting two specific nodes of G; *the Single-Source Shortest Path Problem* (SS-SPP) in which one aims to find the path of minimum length connecting a fixed node to the remaining nodes of G; and *the All-Pairs Shortest Path Problem* (AP-SPP) in which one aims to find the paths of minimum length connecting every pair of nodes of G. When arc lengths are deterministically known such versions can be solved efficiently, e.g., by using Dijkstra's algorithm if arc lengths are nonnegative, or using Bellman–Ford's algorithm if arc lengths are generic [1]. However, in practical applications the information about arc lengths could be uncertain due to its dependence on unpredictable factors (e.g., network congestions or hardware failures in telecommunications; traffic conditions, accidents, traffic jams or weather conditions in transportation, see Montemanni and Gambardella [2]). For this reason, investigating the SPP under uncertainty becomes indispensable to solve a wide variety of real world problems.

In the recent years several authors have achieved results of fundamental importance in the study of the uncertain versions of the SPP. In particular, Montemanni and Gambardella [3] investigated *the Minimax regret Single-Pair Shortest Path Problem* (MSP-SPP), i.e., the problem of minimizing the maximum deviation from the optimal single-pair shortest path over all possible feasible assignments of arc lengths. The authors provided results about the complexity of the problem under discrete uncertainty, including proof of its $\mathcal{NP}$-hard nature. More recently, Averbakh and Lebedev [4], Zielinski [5], Kasperski and Zielinski [6] investigated the complexity of a version of the MSP-SPP called *the MSP-SPP under interval uncertainty*, i.e., the MSP-SPP in which arc lengths are supposed to be nonnegative intervals. The authors proved that in such a version the problem remains $\mathcal{NP}$-hard even if G is a multi-digraph or is directed, acyclic, and either has a layered structure or is planar and regular of degree three.

Averbakh and Lebedev [4] proved that the problem is polynomially solvable when the number of arcs with uncertain lengths is fixed or is bounded by the logarithm of a polynomial function of the total number of arcs.

From a computational point of view, Karasan et al. [7] first proposed a mixed integer programming formulation for the

* Corresponding author. Tel.: +32 2 650 5628; fax: +32 2 650 5970.
E-mail address: dacatanz@ulb.ac.be (D. Catanzaro).

0305-0548/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2011.01.022

MSP-SPP. Specifically, the authors showed that for each optimal solution p of the MSP-SPP there exists a *scenario* s, i.e., an assignment of possible values to each uncertain parameter of the problem, for which p is an optimal solution of the deterministic problem corresponding to s. Moreover, the authors also showed that the knowledge of those arcs which are never on shortest paths can be used to generate preprocessing procedures that speed-up the resolution of the problem. Karasan et al. applied these strategies to acyclic directed graphs in which arc lengths are nonnegative intervals and showed their efficacy on instances containing up to 420 nodes. Montemanni et al. [8] and Montemanni and Gambardella [9] extended Karasan et al.'s work by presenting exact algorithms able to tackle instances of the MSP-SPP in general directed graphs.

Here, we investigate the MSP-SPP and the Minimax regret Single-Source Shortest Path Problem (MSS-SPP) under interval uncertainty. We extend Karasan et al.'s results by considering finite directed graphs in which cycles may occur and arc lengths may *degenerate*, i.e., be deterministically known. We provide sufficient conditions for a node or an arc to be always or never in an optimal solution of the MSP-SPP. Similarly, we provide sufficient conditions for an arc to be always or never in an optimal solution of the MSS-SPP. From such conditions we develop pegging tests, similar to the ones used by Yokoya et al. for the repeated assignment problem (see Yokoya et al. [10]). We show, by means of numerical results, that our pegging tests are efficient in practice and vastly reduce the overall running time necessary to exactly solve instances of both the MSP-SPP and the MSS-SPP.

2. Notation and problem statement

Let G be a finite directed graph having $V(G)$ and $A(G)$ as node-set and arc-set, respectively. We assume that $|V(G)|=n$, $|A(G)|=m$, and that each node $j \in V(G)$ is reachable from node 1. We call node 1 *the origin* and node *n the destination*. We denote (i,j) as the arc from node i to node j in G. If A' is a nonempty subset of $A(G)$, we denote $G-A'$ as the subgraph of G with edge set $A(G)\backslash A'$, i.e., the subgraph obtained from G by deleting arcs in A'. Similarly, we denote $G+A'$ as the graph obtained from G by adding a set of arcs A'. For sake of notation, if $A'=\{(i,j)\}$ we will write $G-(i,j)$ and $G+(i,j)$ instead of $G-\{(i,j)\}$ and $G+\{(i,j)\}$, respectively. If H_1 and H_2 are two subgraphs of G, we define $H_1 \cup H_2$ as the subgraph having node-set $V(H_1) \cup V(H_2)$ and arc-set $A(H_1) \cup A(H_2)$. We denote $P_1(G)$ as the set of all the paths in G connecting the origin to any other node of G, and $P_{1k}(G)$ as the set of all $(1,k)$-paths in G, i.e., the set of all the paths in G connecting the origin to node k of G. Finally, given a path $p \in P_{1k}(G)$, we denote $A(p)$ as the set of arcs of p. Unless not stated otherwise, we will always assume that arc lengths of G are nonnegative. We consider the following problem.

Problem 1. The Single-Pair Shortest Path Problem (SP-SPP)

Given a finite directed weighted graph G having lengths l_{ij} associated with each arc $(i,j) \in A(G)$, find a shortest $(1,n)$-path in G, i.e., a $(1,n)$-path such that the sum of its arc lengths is minimum.

The SP-SPP can be efficiently solved by means of Dijkstra's algorithm [1]. Moreover, Dijkstra's algorithm also provides an optimal solution to the following problem.

Problem 2. The Single-Source Shortest Path Problem (SS-SPP)

Given a finite directed weighted graph G having lengths l_{ij} associated with each arc $(i,j) \in A(G)$, find a shortest directed $(1,j)$-paths in G, for all $j \in V(G)$, $j \neq 1$.

Suppose now that arc lengths are uncertain and vary arbitrarily inside nonnegative intervals $[\underline{l}_{ij},\overline{l}_{ij}]$, for all $(i,j) \in A(G)$. We model uncertainty by means of the concept of a *scenario* s, i.e., an assignment of possible arc lengths for all $(i,j) \in A(G)$ for which lengths are uncertain. We denote S as the set of all possible scenarios for arc lengths in G, and l^s_{ij} as the length of arc $(i,j) \in A(G)$ under the scenario s. We set the length l^s_p of the path $p \in P_{1k}(G)$ for a fixed scenario $s \in S$ as

$$l^s_p = \sum_{(i,j) \in A(p)} l^s_{ij}.$$

We call the uncertain versions of the SP-SPP and the SS-SPP, the *uncertain single-pair shortest path problem* and the *uncertain single-source shortest path problem*, respectively. Note that, when a specific scenario is fixed, the uncertain version of the problem reduces to its corresponding deterministic version.

A possible way to face a problem under uncertainty consists of minimizing the maximum deviation from the value of an optimal solution over all possible scenarios. This approach is known in the literature as *the minimax regret approach* (see Kouvelis and Yu [3]) and, when used in the context of the uncertain single-pair shortest path problem, it gives rise to the following optimization problem.

Problem 3. The Minimax regret Single-Pair Shortest Path Problem (MSP-SPP)

Given a finite directed graph G whose arc lengths are nonnegative intervals $[\underline{l}_{ij},\overline{l}_{ij}]$, for all $(i,j) \in A(G)$, find the minimax regret shortest directed $(1,n)$-paths in G, i.e.,

$$\min_{p \in P_{1n}(G)} \max_{s \in S} (l^s_p - l^s_{p^{*s}}),$$

where $l^s_{p^{*s}}$ is the length of the shortest path p^{*s} in the scenario s.

The MSP-SPP can be solved, e.g., by using the strategies described in Karasan et al. [7]. In the next sections we shall describe strategies that may speed-up the resolution process. Before that, we introduce some definitions that will prove useful throughout the paper. Similar to Karasan et al. [7] we say that an arc (i,j) is *strong* if for all scenarios $s \in S$ there exists at least one shortest path p from the origin to the destination such that $p \ni (i,j)$; an arc (i,j) is 0-*persistent* if for all scenarios $s \in S$ there does not exist a shortest path p from the origin to the destination such that $p \ni (i,j)$. Moreover, we also introduce the following two definitions: an arc (i,j) is 1-*persistent* if for all scenarios $s \in S$ and all shortest paths p from the origin to the destination it holds that $p \ni (i,j)$; a node v is 0-*persistent* if for all scenarios $s \in S$ there does not exist a shortest path p from the origin to the destination passing through v. We shall present now a number of sufficient conditions to identify 0-persistent arcs and nodes and 1-persistent arcs in generic instances of the MSP-SPP.

3. Persistency in the MSP-SPP

As observed in Karasan et al. [7], given a generic instance of the MSP-SPP, identifying and removing the arcs that are never on a shortest $(1,n)$-path may speed-up the resolution process. Unfortunately, identifying such arcs is hard as in general this task involves solving a $\mathcal{NP}$-complete problem. However, Karasan et al. [7] showed that when dealing with directed layered graphs with small width and non-degenerating arc lengths (i.e., arcs not having deterministically fixed lengths) the decision problem can be efficiently solved. In this section We extend Karasan et al.'s [7] results by considering finite directed graphs in which cycles may occur and arc lengths can *degenerate*. To this end, we denote $\underline{s}$ and $\overline{s}$ as two scenarios for which $l^{\underline{s}}_{ij} = \underline{l}_{ij}$, for all $(i,j) \in A(G)$,

Pegging test 1 - 1-persistent arc finder

1. Find a shortest path p in G from the origin to the destination under the scenario $\bar{s}$.
2. Choose an arc $(k,r) \in A(p)$.
3. Construct a scenario s' for which $l^{s'}_{ij} = \bar{l}_{ij}$ if $(i,j) \in A(p)$, and $l^{s'}_{ij} = \underline{l}_{ij}$ otherwise.
4. Find a shortest path q in $G-(k,r)$ under the scenario s'. If such a path exists, go to step 5, otherwise mark (k,r) as 1-persistent and go to step 7.
5. Compute $l^{s'}_q$ and $l^{s'}_p$.
6. If $l^{s'}_q > l^{s'}_p$ then mark arc (k,r) as 1-persistent.
7. If there are still unvisited arcs in $A(p)$ then go to step 2.

Fig. 1. Pegging test to find 1-persistent arcs in generic instances of the MSP-SPP.

and $l^{\bar{s}}_{ij} = \bar{l}_{ij}$, for all $(i,j) \in A(G)$, respectively. We also denote $P^*_1(G,s)$ and $P^*_{1k}(G,s)$ as the set of all shortest 1-paths in G under a scenario s, and the set of all shortest $(1,k)$-paths in G under a scenario s, respectively.

Let (k,r) be an arc of G such that when the graph $G-(k,r)$ is considered, the destination is always reachable from the origin. For each $s \in S$, let $p^{*s} \in P^*_{1n}(G,s)$. Then, it holds that arc (k,r) is 1-persistent if and only if for all $q \in P_{1n}(G-(k,r))$ and for all $s \in S$, it holds that $l^s_q > l^s_{p^{*s}}$.

The following proposition holds: can also be

Proposition 1. *Let p be the shortest $(1,n)$-path under the scenario $\bar{s}$. Let $(k,r) \in A(p)$, and let s' be a scenario such that $l^{s'}_{ij} = \bar{l}_{ij}$ if $(i,j) \in A(p)$, and $l^{s'}_{ij} = \underline{l}_{ij}$ otherwise. Consider the graph $G-(k,r)$ and suppose that the destination is reachable from the origin. Let $q \in P^*_{1n}(G-(k,r),s')$ be the shortest $(1,n)$-path under the scenario s'. If it holds that $l^{s'}_q > l^{s'}_p$ then arc (k,r) is1-persistent.*

Proof. If it holds that $l^{s'}_q > l^{s'}_p$ then, for all $p' \in P_{1n}(G-(k,r))$, we have that $l^{s'}_{p'} \geq l^{s'}_q > l^{s'}_p$. Hence, for all $p' \in P_{1n}(G-(k,r))$ it holds that

$$\sum_{(i,j)\in A(p')\setminus A(p)} \underline{l}_{ij} = \sum_{(i,j)\in A(p')\setminus A(p)} l^{s'}_{ij} > \sum_{(i,j)\in A(p)\setminus A(p')} l^{s'}_{ij} = \sum_{(i,j)\in A(p)\setminus A(p')} \bar{l}_{ij}.$$

Then, for all $s \in S$, it holds that

$$\sum_{(i,j)\in A(p')\setminus A(p)} l^{s}_{ij} \geq \sum_{(i,j)\in A(p')\setminus A(p)} \underline{l}_{ij} > \sum_{(i,j)\in A(p)\setminus A(p')} \bar{l}_{ij} \geq \sum_{(i,j)\in A(p)\setminus A(p')} l^{s}_{ij},$$

which, in turn, implies that for all $s \in S$ and for all $p' \in P_{1n}(G-(k,r))$, $l^s_{p'} > l^s_p$. Thus, if $p^{*s} \in P^*_{1n}(G,s)$ then $l^s_{p'} > l^s_p \geq l^s_{p^{*s}}$, hence (k,r) is 1-persistent. □

From Proposition 1 we can easily derive a $O(mn^2)$ pegging test (shown in Fig. 1) to find 1-persistent arcs in generic instances of the MSP-SSP. The rationale of such a test can be explained by means of the following example.

Example 1. Consider the graph given in Fig. 2(a). Construct a scenario $\bar{s}$ by setting the lengths of all arcs to their upper bounds. Find the shortest path p from the origin to the destination. Path p is represented with bold lines in Fig. 2(b). Construct now a scenario s' by setting the lengths of arcs $(1,a)$ and (a,n) to their upper bounds and the lengths of the remaining arcs to their respective lower bounds. The length of p under the scenario s' is $l^{s'}_p = 6$. Delete arc $(1,a) \in A(p)$ and find a shortest path q from the origin to the destination (see Fig. 2(c)). The length of path q under the scenario s' is $l^{s'}_q = 7$. Since $l^{s'}_q > l^{s'}_p$ we can conclude that arc $(1,a)$ is 1-persistent. Consider now arc (a,n) in p under the scenario s' and delete it from $A(p)$ (see Fig. 2(d)). After computing the shortest path q from the origin to the destination in $G-(a,n)$ we will find a path having length $l^{s'}_q = 6$. Since $l^{s'}_q = l^{s'}_p$ this time we cannot conclude that arc (a,n) is 1-persistent.

It is worth noting that Proposition 1 provides only a sufficient condition for an arc to be 1-persistent. To show that, consider the following example.

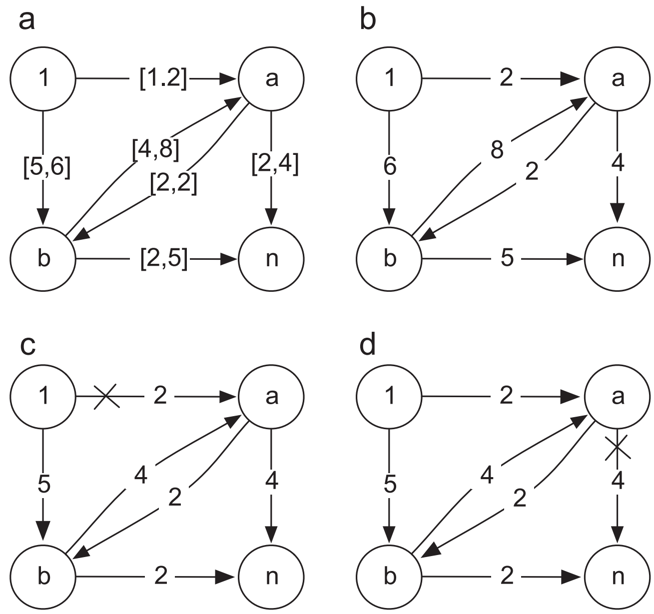

Fig. 2. A graphical example showing how the pegging test 1 works (see text for details).

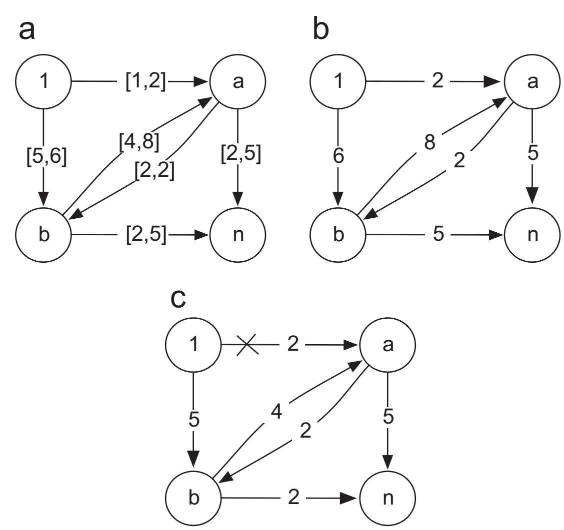

Fig. 3. An example showing that Proposition 1 is only a sufficient condition.

Example 2. Consider the graph G shown in Fig. 3(a). As in the previous example, we first set the lengths of all arcs to their upper bounds and then we compute the shortest $(1,n)$-path p in G. Path p is represented with bold lines in Fig. 3(b). Again, we construct a scenario s' by setting the lengths of arcs $(1,a)$ and (a,n) to their

Pegging test 2 - 0-persistent node finder

1. Find the shortest paths from the origin to each node of G under the scenario $\overline{s}$. Let q be the $(1,n)$-path.
2. Choose a node $k \in V(G) \setminus V(q)$ and consider a shortest $(1,k)$-path p_1 under the scenario $\underline{s}$.
3. Find a shortest path p_2 in G from node k to the destination under the scenario $\underline{s}$.
4. Compute $l^{\underline{s}}_{p_1}$, $l^{\underline{s}}_{p_2}$, and $l^{\overline{s}}_{q}$. If $l^{\underline{s}}_{p_1} + l^{\underline{s}}_{p_2} > l^{\overline{s}}_{q}$ then mark node k as 0-persistent.
5. If there are still unvisited nodes in $V(G)$ then go to step 2.

Fig. 4. Pegging test to find 0-persistent nodes in generic instances of the MSP-SPP.

Pegging test 3 - T-1-persistent arcs finder

1. Find a 1-shortest spanning tree T in G under the scenario $\overline{s}$.
2. Choose an arc $(k,r) \in A(T)$ and consider a path $p \in P_{1r}(G)$ such that $A(p) \subset A(T)$.
3. Construct a scenario s' such that $l^{s'}_{ij} = \bar{l}_{ij}$ if $(i,j) \in A(p)$, and $l^{s'}_{ij} = \underline{l}_{ij}$ otherwise.
4. Find a shortest $(1,r)$-path q in $G-(k,r)$ under the scenario s'.
5. If such a path exists then go to step 6, otherwise mark arc (k,r) as T-1-persistent and go to step 7.
6. Compute $l^{s'}_{q}$ and $l^{s'}_{p}$. If $l^{s'}_{q} > l^{s'}_{p}$ then mark arc (k,r) as T-1-persistent.
7. If there are still unvisited arcs in $A(T)$ then go to step 2.

Fig. 5. Pegging test to find T-1-persistent arcs in generic instances of the MSS-SPP.

upper bounds and the lengths of the remaining arcs to the respective lower bounds. The length of p under the scenario s' is $l^{s'}_{p} = 7$. Delete arc $(1,a) \in A(p)$ and find the shortest $(1,n)$-path q in the graph $G-(1,a)$ under the scenario s'. The length of such a path is $l^{s'}_{q} = 7 = l^{s'}_{p}$ (see Fig. 3(c)) hence we cannot conclude that arc $(1,a)$ is 1-persistent, although, as it is easy to verify, it actually is. The following proposition provides a sufficient condition for a node to be 0-persistent.

Proposition 2. *Let q be the shortest $(1,n)$-path under the scenario $\overline{s}$. Let $k \in V(G) \backslash V(q)$. Consider the shortest paths p_1 and p_2 from the origin to node k and from node k to the destination under the scenario $\underline{s}$, respectively. If it holds that $l^{\underline{s}}_{p_1} + l^{\underline{s}}_{p_2} > l^{\overline{s}}_{q}$ then node k is 0-persistent.*

Proof. Note that all paths $p \in P^*_{1n}(G,\underline{s})$ are such that $l^{\underline{s}}_{p} \le l^{\overline{s}}_{q}$. Hence, if $l^{\underline{s}}_{p_1} + l^{\underline{s}}_{p_2} > l^{\overline{s}}_{q}$, then $p_1 \cup p_2$ is not the shortest $(1,n)$-path under the scenario $\underline{s}$. Since p_1 and p_2 are shortest paths, all $(1,n)$-paths p that contain node k are such that $l^{\underline{s}}_{p} \ge l^{\underline{s}}_{p_1} + l^{\underline{s}}_{p_2}$. Then, for all scenarios $s \in S$ and all paths $q^{*s} \in P^*_{1n}(G,s)$, it holds that

$$l^{s}_{p} \ge l^{\underline{s}}_{p} \ge l^{\underline{s}}_{p_1} + l^{\underline{s}}_{p_2} > l^{\overline{s}}_{q} \ge l^{s}_{q^{*s}}.$$

Thus, there does not exist a shortest $(1,n)$-path that contains node k. Hence, node k is 0-persistent. □

From Proposition 2 we can easily derive a $O(n^3)$ pegging test (shown in Fig. 4) to find 0-persistent nodes in generic instances of the MSP-SSP.

4. Persistency in the minimax regret single-source shortest path problem

The minimax regret approach discussed in Section 2 can also be used to face the uncertain single-source shortest path problem. Specifically, denoted $T^1(G)$ as the set of all the 1-spanning trees in G, i.e., the set of the feasible solutions to the SS-SPP when the source is node 1, $T^*_1(G,s)$ as the set of all 1-shortest path spanning trees of G under a scenario $s \in S$, $A(T)$ as the arc-set of the 1-spanning tree $T \in T^1(G)$, $p_{1k}(T)$ as the $(1,k)$-path such that $A(p_{1k}(T)) \subset A(T)$, l^{s}_{T} as the length of T under the scenario $s \in S$, i.e., the sum of its arc lengths under the scenario s, and

$$l^{s}_{T^{*s}} = \min_{T \in T^1(G)} l^{s}_{T},$$

the minimax regret version of the uncertain single-source shortest path problem can be stated in terms of the following optimization problem.

Problem 4. The Minimax regret Single-Source Shortest Path Problem (MSS-SPP)

Given a finite directed graph G whose arc lengths are non-negative intervals $[\underline{l}_{ij}, \bar{l}_{ij}]$, for all $(i,j) \in A(G)$, find a shortest directed $(1,j)$-path in G, for all $j \in V(G)$, $j \neq 1$, i.e.,

$$\min_{T \in T^1(G)} \max_{s \in S} (l^{s}_{T} - l^{s}_{T^{*s}}).$$

The MSS-SPP can be solved using approaches similar to the ones described in Karasan et al. [7] and Salazar-Neumann [11]. In this section we shall describe a number of strategies that may speed-up its resolution process. To this end we introduce the following definitions. We say that an arc (i,j) is *T*-1-*persistent* if, for all scenarios $s \in S$, (i,j) lies on all 1-shortest path spanning trees. Analogously, we say that an arc (i,j) is *T*-0-*persistent* if, for all scenarios $s \in S$, (i,j) never lies on the 1-shortest path spanning tree. As shown in Salazar-Neumann [11], T-1-persistent arcs are characterized by conditions similar to the ones used to characterize 1-persistent arcs in the MSP-SPP, hence, for the sake of conciseness, we omit them and just provide in Fig. 5 a $O(mn^2)$ pegging test able to find T-1-persistent arcs in generic instances of the MSS-SSP.

The following proposition characterizes *T*-0-persistent arcs.

Proposition 3. *Let $p^{*s} \in P^*_{1k}(G,s)$, for all $s \in S$. Then, arc (k,r) is T-0-persistent if and only if, for all $q \in P^*_{1r}(G,s)$ and $s \in S$, it holds that $l^{s}_{q} < l^{s}_{p^{*s}} + \underline{l}_{kr}$.*

Proof. (Sufficiency). Let (k,r) be a T-0-persistent arc of G. Then, for all $p^{*s} \in P^*_{1k}(G,s)$, $p^{*s} + (k,r)$ is never the $(1,r)$-shortest path in G. Hence, for all $s \in S$, if $q \in P^*_{1r}(G,s)$, we have $l^{s}_{q} < l^{s}_{p^{*s}} + \underline{l}_{kr}$. (Necessity). If for all $s \in S$ and $q \in P^*_{1r}(G,s)$ it holds that $l^{s}_{q} < l^{s}_{p^{*s}} + \underline{l}_{kr}$, then for all $p \in P_{1k}(G)$, $l^{s}_{q} < l^{s}_{p^{*s}} + \underline{l}_{kr} \le l^{s}_{p} + l^{s}_{kr}$. Thus, any $(1,r)$-path in G passing through arc (k,r) is never a $(1,r)$-shortest path, independently from the scenario $s \in S$. Hence, arc (k,r) is T-0-persistent. □

The use of Proposition 3 could prove impractical to find T-0-persistent arcs in generic instances of the MSS-SPP, as it involves checking whether the condition $l^{s}_{q} < l^{s}_{p^{*s}} + \underline{l}_{kr}$ holds for all scenarios

Pegging test 4 - 0 and T-0-persistent arcs finder

1. Delete all arcs of the form $(j,1)$.
2. Find a 1-shortest spanning tree T in G under the scenario $\underline{s}$.
3. If there exists an arc $(k,r) \in A(G) \setminus A(T)$ select it, else stop.
4. Consider a path $p \in P_{1k}(G)$ such that $A(p) \subset A(T)$.
5. Find a shortest $(1,r)$-path q in G under the scenario $\bar{s}$.
6. Compute $l_q^{\bar{s}}$.
7. If $l_q^{\bar{s}} < l_p^{\underline{s}} + \underline{l}_{kr}$ then mark arc (k,r) as T-0-persistent (hence 0-persistent). Delete arc (k,r) and go to step 3.

Fig. 6. Pegging test to find 0 and T-0-persistent arcs in the MSS-SPP.

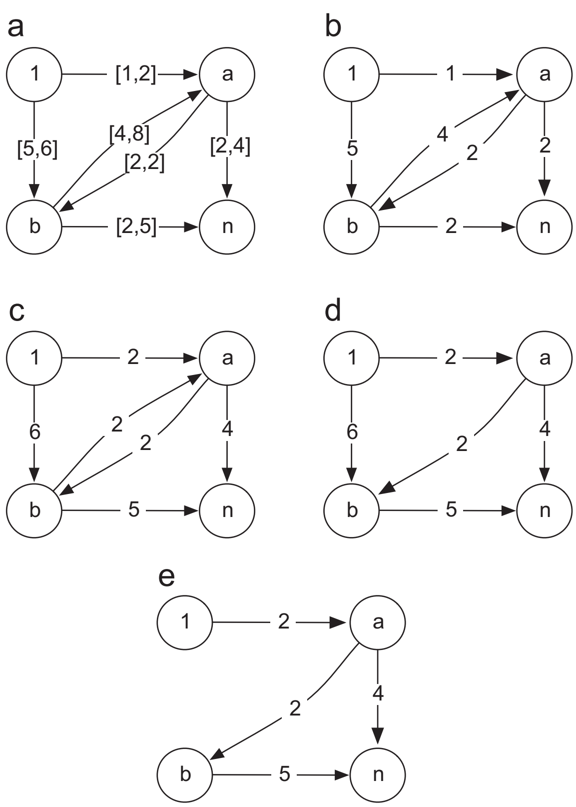

Fig. 7. A graphical example showing how the pegging test 4 works (see text for details).

$s \in S$. To overcome this problem, we provide below a sufficient condition for an arc to be T-0-persistent.

Proposition 4. *Let T be a 1-shortest path spanning tree under a scenario $\underline{s}$ and let $(k,r) \in A(G) \setminus A(T)$. Let $p \in P_{1k}(G)$ be such that $A(p) \subset A(T)$. If for all $q \in P^*_{1r}(G,\bar{s})$ it holds that $l_q^{\bar{s}} < l_p^{\underline{s}} + \underline{l}_{kr}$, then arc (k,r) is T-0-persistent.*

Proof. The statement trivially follows by observing that $l^s_{q^{*s}} \le l_q^{\bar{s}} < l_p^{\underline{s}} + \underline{l}_{kr} \le l^s_{p^{*s}} + \underline{l}_{kr}$. $\square$

From the previous propositions we can derive a $O(mn^2)$ pegging test (shown in Fig. 6) to find T-0-persistent (and hence 0-persistent) arcs in the MSS-SSP. The rationale of such a test can be explained by means of the following example.

Example 3. Consider the graph G given in Fig. 7(a). Construct a scenario $\underline{s}$ by setting the lengths of all arcs to their respective lower bounds. Find the 1-shortest path spanning tree from the origin to all nodes of G. The tree is represented with bold lines in Fig. 7(b). Choose an arc not in the tree, e.g., (b,a), and consider a path $p \in P_{1b}(G)$ such that $A(p) \subset A(T)$. The length of path p under the scenario $\underline{s}$ is $l_p^{\underline{s}} = 3$. Note also that $\underline{l}_{ba} = 4$. Now, consider a shortest $(1,a)$-path q under the scenario $\bar{s}$. The length of such a path is $l_q^{\bar{s}} = 2$ (see Fig. 7(c)). Since $l_q^{\bar{s}} < l_p^{\underline{s}} + \underline{l}_{ba}$ then we can conclude that arc (b,a) is T-0-persistent. Similarly, take a second arc $(1,b) \notin A(T)$ and consider a path $p \in P_{1b}(G)$ such that $A(p) \subset A(T)$. Consider a shortest $(1,b)$-path q under the scenario $\bar{s}$. The length of such a path is $l_q^{\bar{s}} = 4$, while $l_p^{\underline{s}} = 0$ and $\underline{l}_{1b} = 5$ (see Fig. 7(d)). Since $l_q^{\bar{s}} < l_p^{\underline{s}} + \underline{l}_{1b}$ then we can conclude that arc $(1,b)$ is T-0-persistent. Finally, choose arc $(b,n) \notin A(T)$ and consider a path $p \in P_{1b}(G)$ such that $A(p) \subset A(T)$. Consider a shortest $(1,n)$-path q under the scenario $\bar{s}$. The length of such a path is $l_q^{\bar{s}} = 6$, while $l_p^{\underline{s}} = 3$ and $\underline{l}_{bn} = 2$ (see Fig. 7(e)). Since $l_q^{\bar{s}} \ge l_p^{\underline{s}} + \underline{l}_{ba}$, then we cannot conclude that arc (b,n) is T-0-persistent.

5. Numerical results

As shown in Karasan et al. [7] and Salazar-Neumann [11], both the MSP-SPP and the MSS-SPP can be formulated as mixed integer linear programming problems. Such formulations can be improved by means of the pegging tests previously discussed which aim at removing (or fixing the value of) those variables which are known to be never (or always) in the optimal solutions to the respective problems. In this section we investigate the practical efficiency of the pegging tests by running them on a number of randomly generated instances. Specifically, we generate a number of random instances of the MSP-SPP and the MSS-SPP by means of the LEDA-4.2 libraries. The instances contain neither node-loops nor parallel arcs, and include a number of nodes n ranging in the set $\{100,250,500,750,1000\}$. For a fixed n, we define the value $D = n/(m(m-1))$ as the *density* of the graph, and consider four possible density values, namely 0.25, 0.5, 0.75, and 1.0. For a fixed pair (n, D) we generate 10 graphs obtaining an overall number of 200 instances of both problems. We set the interval data as in Karasan et al. [7], i.e., for a fixed arc a of a generic instance and a random case base scenario c_a from a uniform distribution, we generate the lower bound l_a from a uniform distribution in the interval $[(1-d)c_a,(1+d)c_a]$, and the upper bound u_a from a uniform distribution in the interval $[(l_a+1)c_a,(1+d)c_a]$, where *the deviation parameter* d is a real number in (0,1). We consider two possible intervals for the uniform distribution used to compute c_a, namely (1,20) and (1,100). Similarly, we consider three possible values for the deviation parameter, namely 0.3, 0.6, and 0.9. We implement the formulations provided by Karasan et al. [7] and Salazar-Neumann [11] by means of Mosel 2.0 of Xpress-MP, Optimizer version 18, running on a Pentium 4, 3.2 GHz, equipped with 2 GByte RAM and operating system Gentoo release 7 (kernel linux 2.6.17, gcc version 4.0). We implemented the pegging tests 1-4 in ANSI-C and used the LEDA-4.2 libraries to solve both the MSP-SPP and the MSS-SSP. The results so obtained are shown in Figs. 8–15. Specifically, Figs. 8 and 9 show the performances of the pegging test 1; Figs. 10 and 11 show the performances of the pegging test 2; Fig. 12 shows the performances of the pegging test 3; and finally Figs. 13–15 show the performances of the pegging test 4.

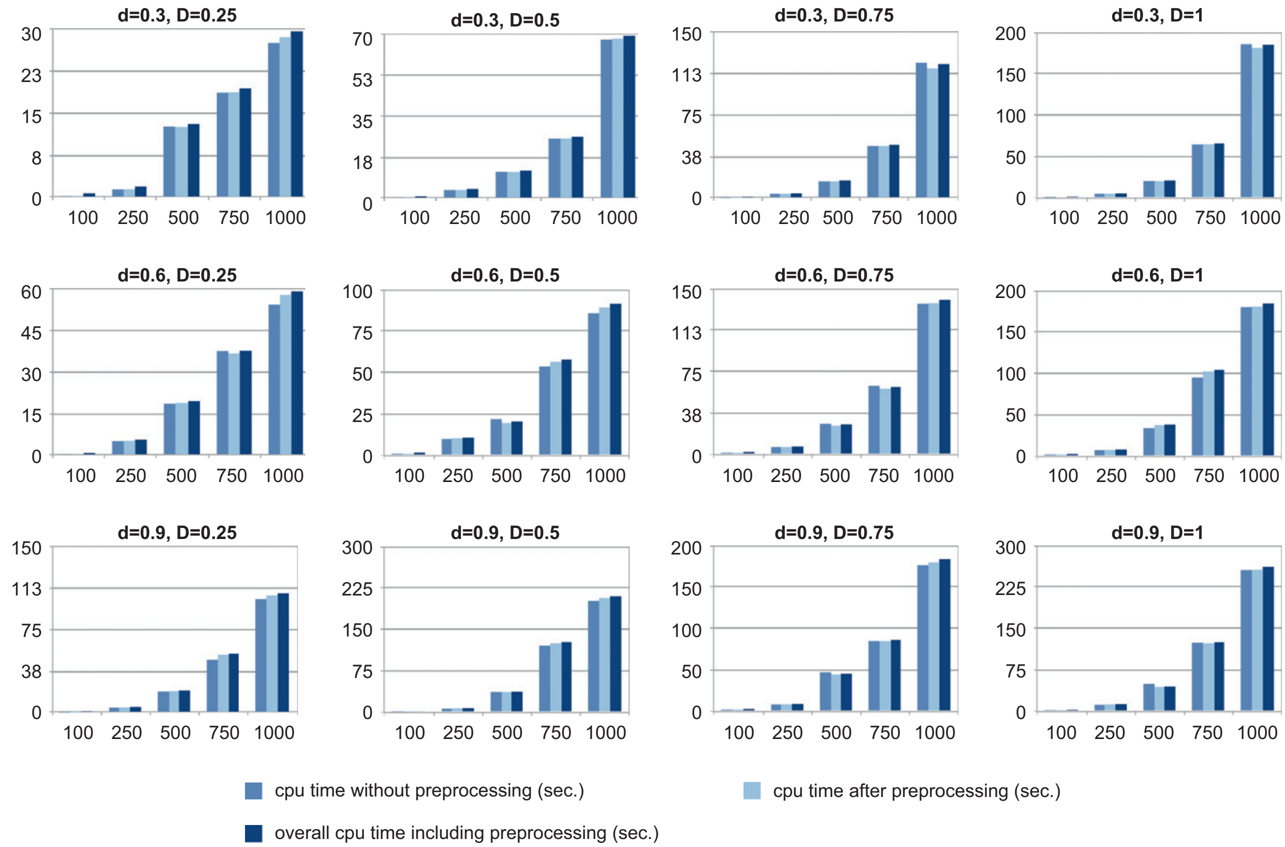

Fig. 8. Performances of the pegging test 1 for $c_a \in (1,20)$.

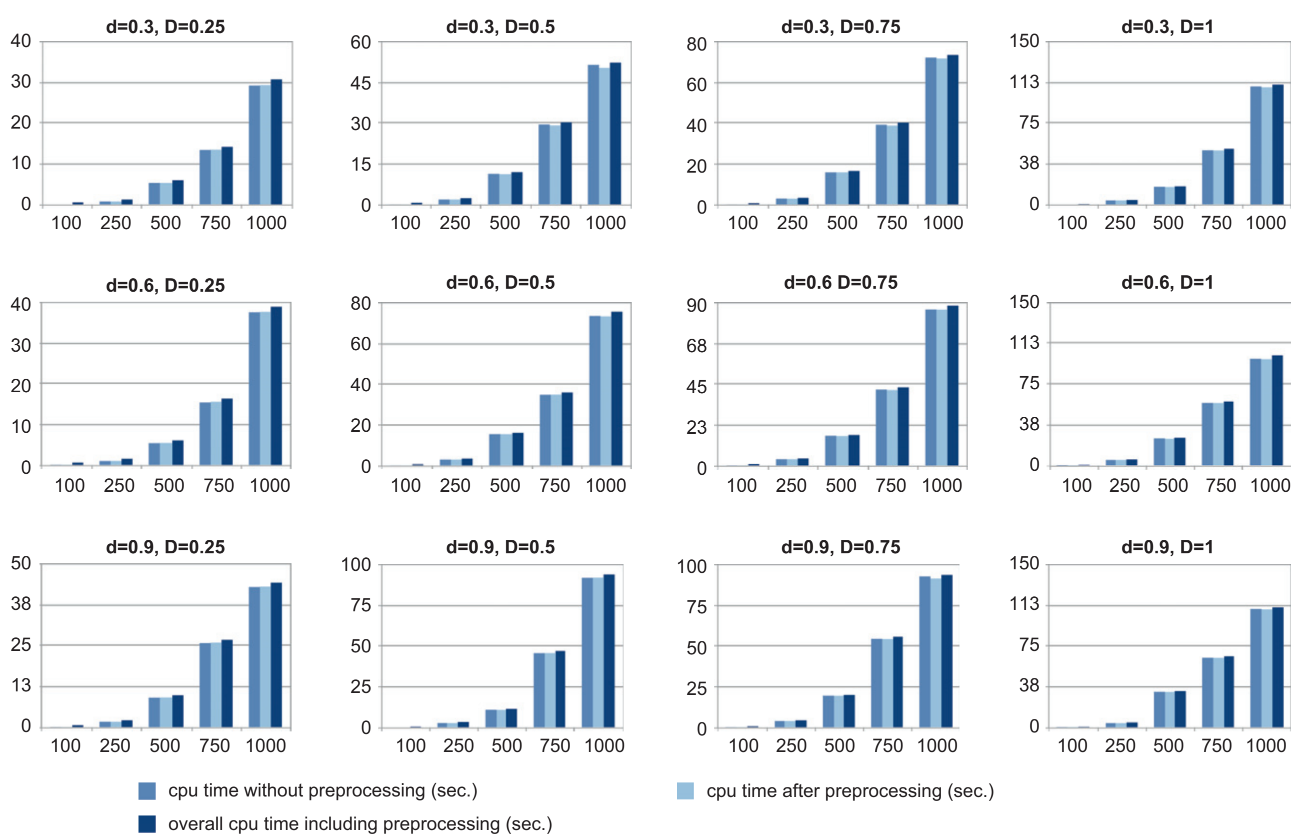

Fig. 9. Performances of the pegging test 1 for $c_a \in (1,100)$.

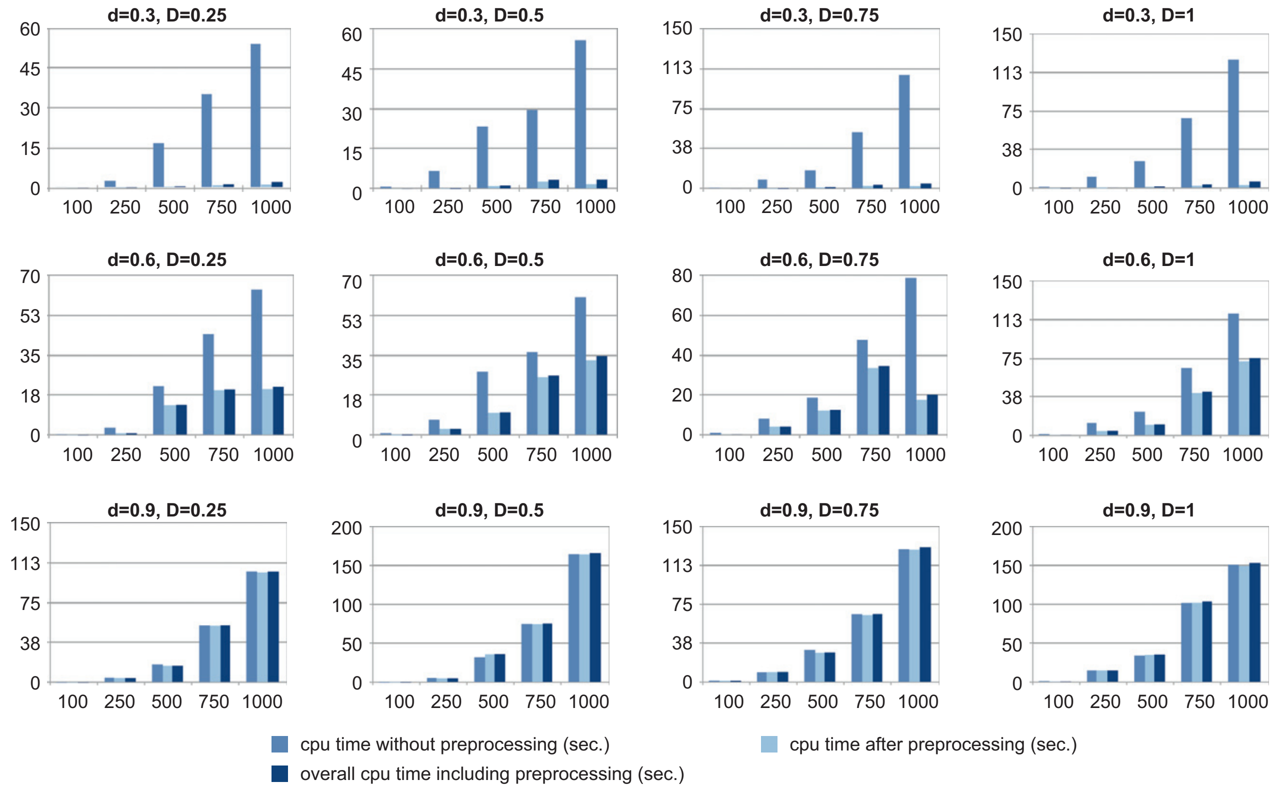

Fig. 10. Performances of the pegging test 2 for $c_a \in (1,20)$.

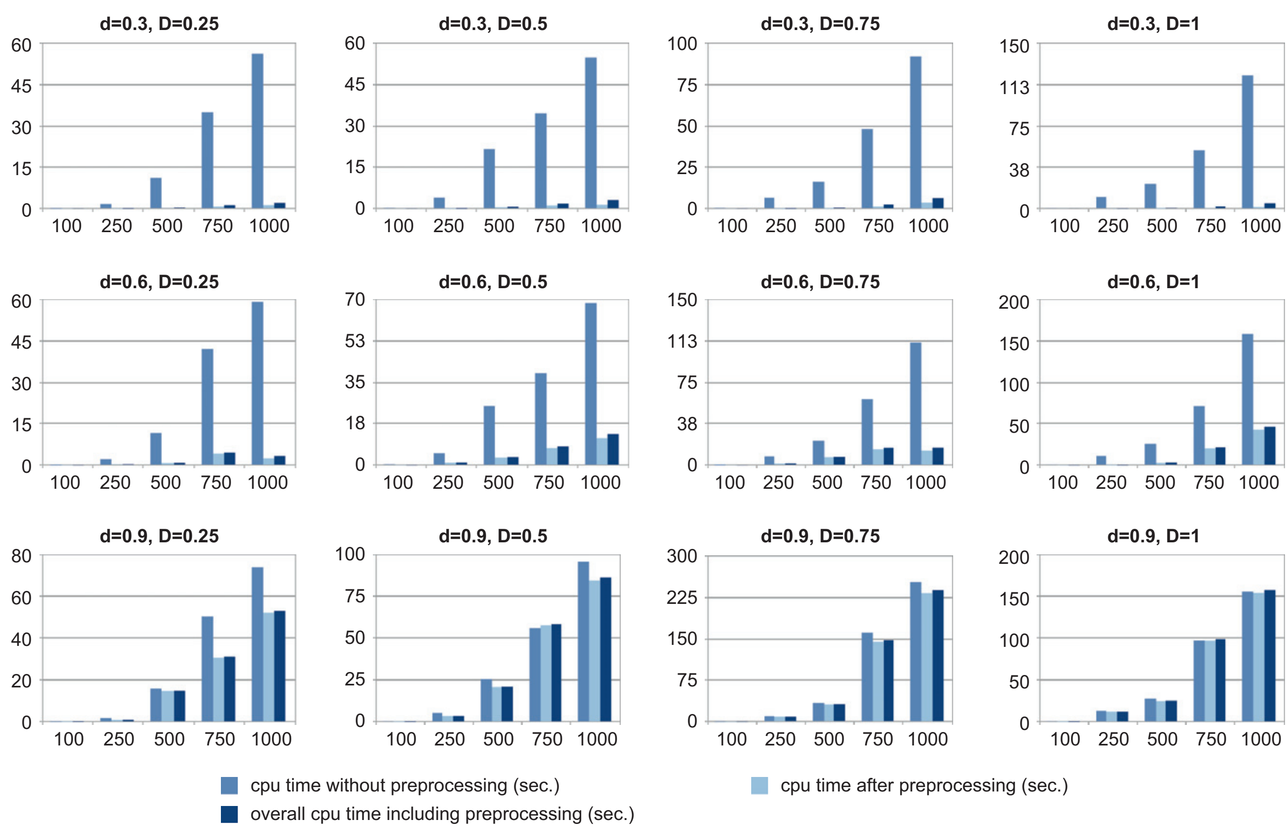

Fig. 11. Performances of the pegging test 2 for $c_a \in (1,100)$.

For a fixed number of nodes, deviation parameter, and density, each graph in Figs. 8–15 shows the mean of: the cpu time taken by Xpress-MP Optimizer to solve exactly an instance of the MSP-SPP, respectively the MSS-SPP, without pegging test (denoted in the legend as "cpu time without preprocessing"); the cpu time taken to run the pegging test (denoted in the legend

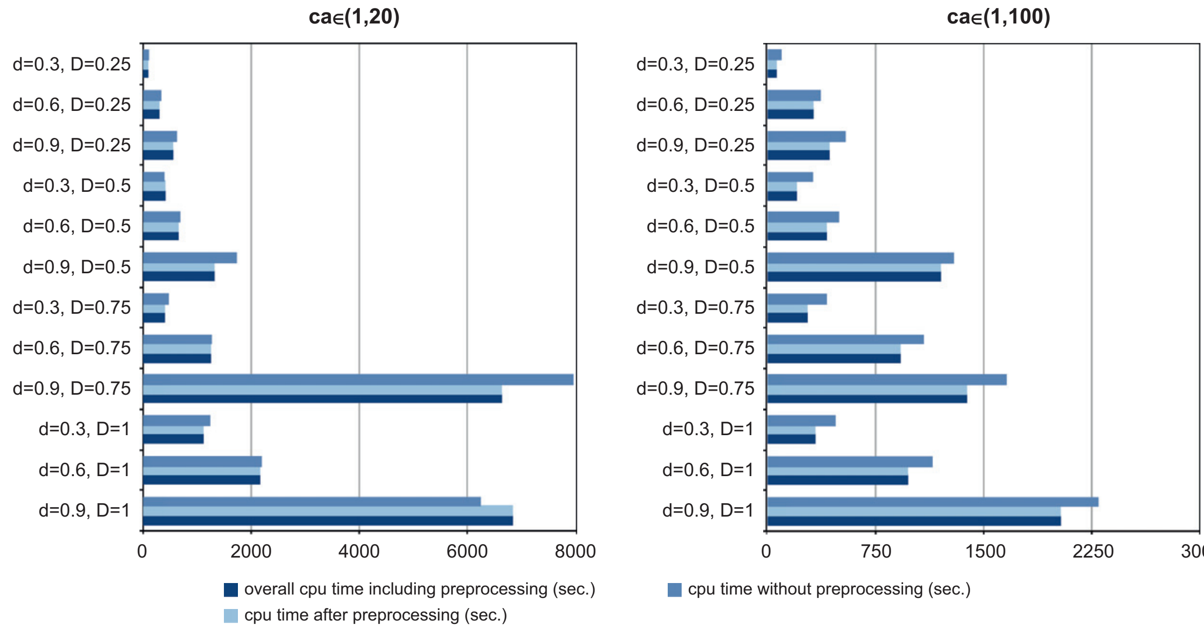

Fig. 12. Performances of the pegging test 3.

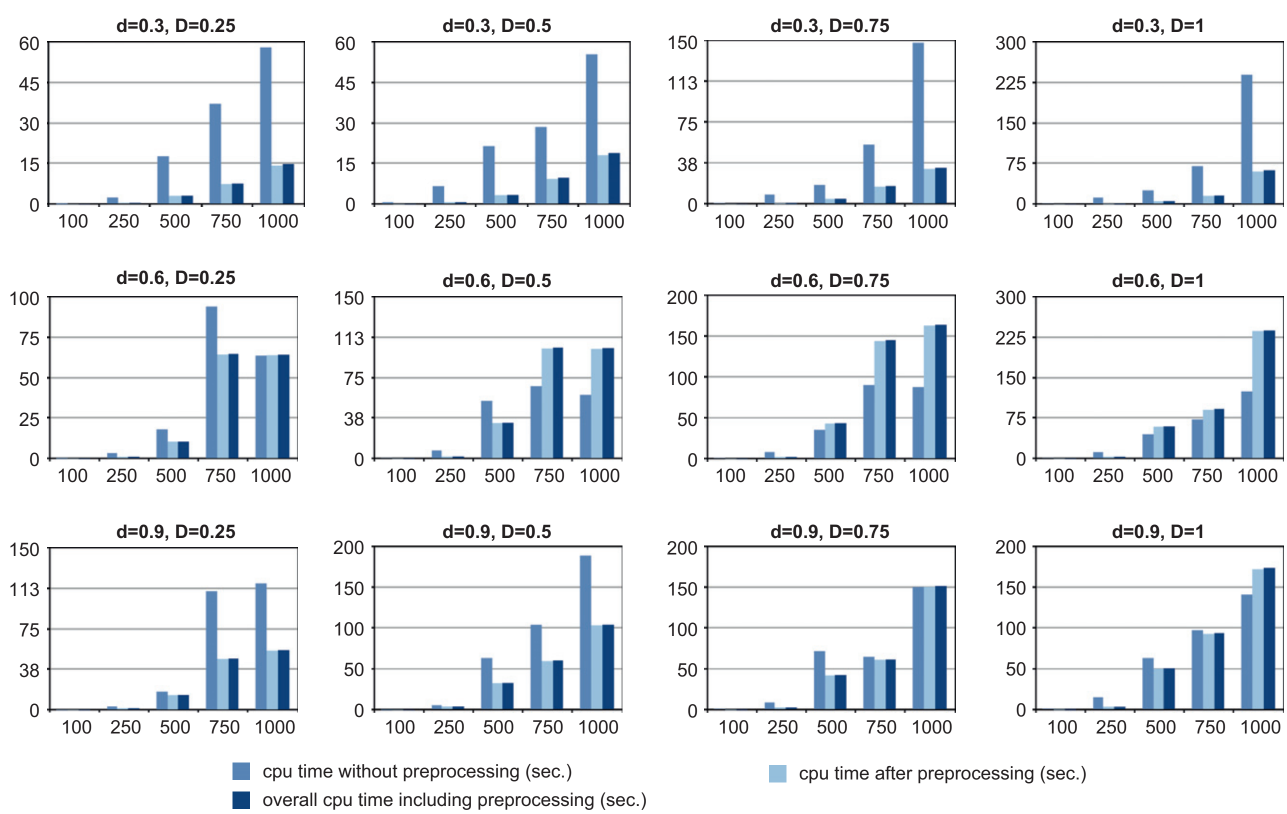

Fig. 13. Performances of the pegging test 4 (0-persistent arcs) for $c_a \in (1,20)$.

as ‘‘preprocessing time’’); the cpu time taken by Xpress-MP Optimizer to solve exactly an instance of the MSP-SPP, respectively the MSS-SPP, after the pegging test (denoted in the legend as ‘‘cpu time after preprocessing’’); and the overall cpu time taken by Xpress-MP Optimizer to solve exactly an instance of the MSP-SPP, respectively the MSS-SPP, including the overhead due to the pegging test (denoted in the legend as ‘‘cpu time including preprocessing’’). All the times are expressed in seconds. Denoted $\mu(cpu_1)$ and $\mu(cpu_2)$ as the average cpu-times with and without pegging test, we call *reduction time* the ratio $\mu(cpu_2)-\mu(cpu_1)/\mu(cpu_2)*100$. When the reduction time is positive the use of a pegging tests proves efficient, otherwise it is inefficient.

As a general trend, Figs. 8 and 9 show that, for the analyzed instances at least, finding 1-persistent arcs proves inefficient. Specifically, apart from sporadic cases, the reduction time is

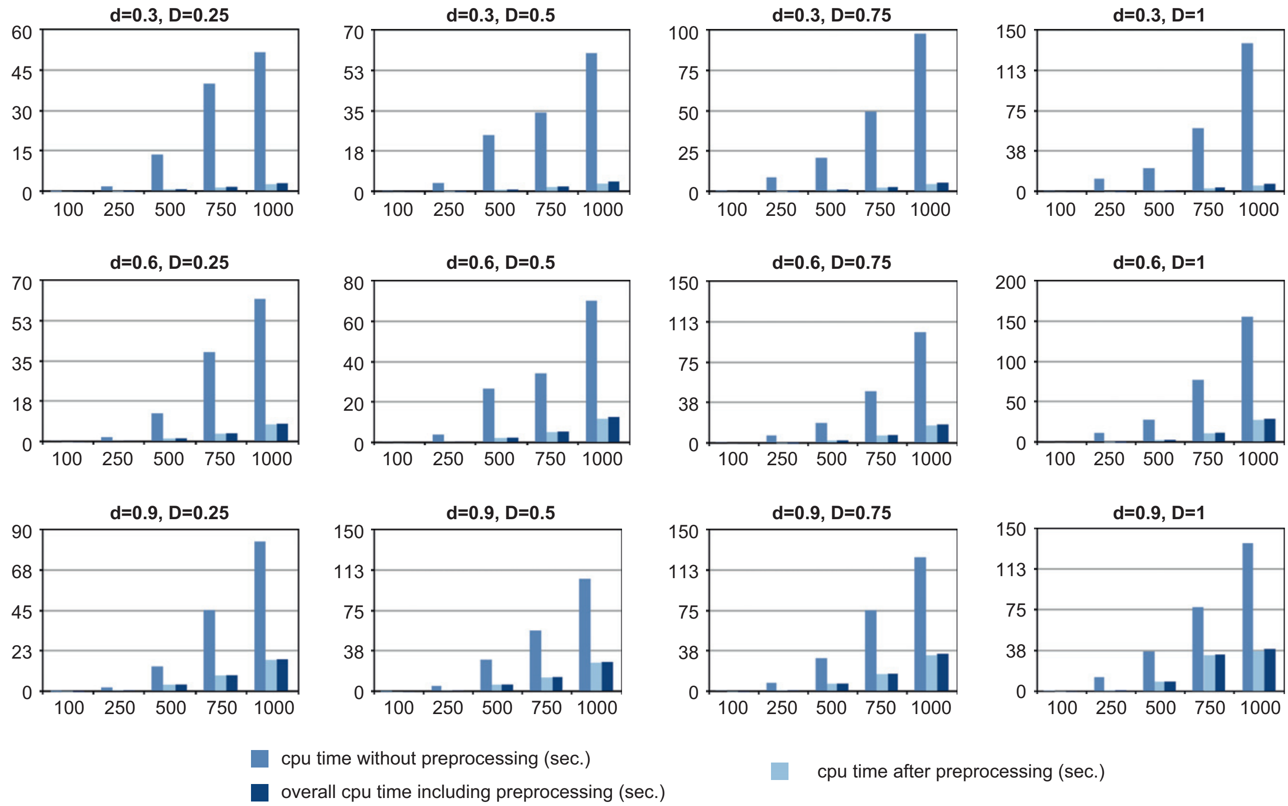

Fig. 14. Performances of the pegging test 4 (0-persistent arcs) for $c_a \in (1,100)$.

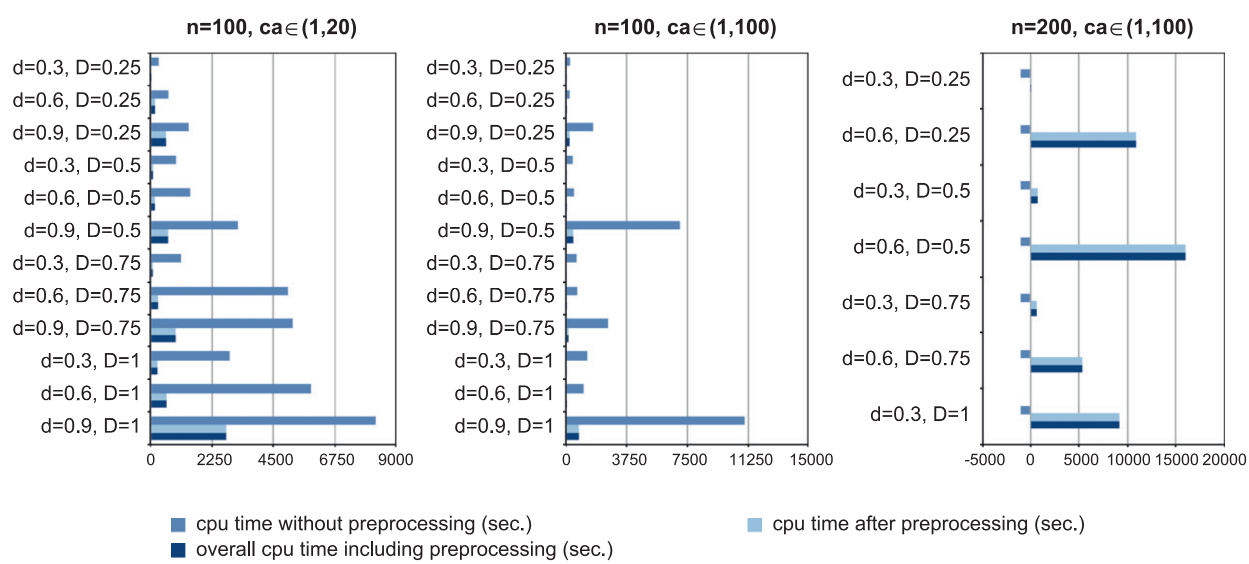

Fig. 15. Performances of the pegging test 4 (T-0-persistent arcs).

always negative, ranging from -1% to -236% when $c_a \in (1,20)$, and from -1% to -495% when $c_a \in (1,100)$. This behavior is justified by the fact that the number of 1-persistent arcs in the considered random instances is in average quite small and the overhead required to find them, although negligible, is not worth the effort. On the contrary, Figs. 10 and 11 show that finding 0-persistent nodes proves very efficient. Specifically, the reduction time is generally positive, ranging from 6% to 97% when $c_a \in (1,20)$, and from 6% to 98% when $c_a \in (1,100)$. Moreover, the reduction time is more marked when $c_a \in (1,100)$. The better performances in this second case are justified by the fact that, for the instances analyzed, the number of 0-persistent nodes is in average much larger than 1-persistent arcs, hence the overhead required to find them is worth the effort. Interestingly, the efficiency of the pegging test 2 seems related to the deviation parameter d. Specifically, when d increases the reduction time tends to decrease, even becoming negative for $d=0.9$. A possible justification for this behavior can be found by observing that for large values of d the intersections between arc length intervals tend to be more frequent, decreasing consequently the number of 0-persistent nodes.

Although finding 1-persistent arcs proves inefficient, Fig. 12 shows that finding T-1-persistent arcs proves efficient. Specifically, apart from sporadic cases, the reduction time is always positive, ranging from 1.38% to 23.55% when $c_a \in (1{,}20)$, and from 7.04% to 35.16% when $c_a \in (1{,}100)$. Moreover, similarly to the case of 0-persistent nodes, the reduction is more marked when $c_a \in (1{,}100)$. Note, however, that the reduction is less prominent than the case of 0-persistent nodes. In fact, when a 0-persistent node is found in a given instance, it is deleted together with all its incident arcs providing, therefore, a more effective preprocessing. Interestingly, we observed that the efficiency of pegging test 3 seems related to the density of the input graphs of the analyzed instances. Specifically, when the density increases the reduction time tends to increase as well as due to the decreasing number of T-1-persistent arcs. In fact, when the density of a graph increases, the number of shortest paths between two nodes increases as well, thus it becomes less likely that a specific arc belongs to all shortest paths between two such nodes. Note finally that Fig. 12 does not include results for instances containing more than 100 nodes. In fact, we experienced a lack of sufficient memory to solve larger instances without a pegging test.

Finally, Figs. 13–15 show that finding 0-persistent arcs in the MSP-SPP and the MSS-SPP, respectively, proves very efficient. Specifically, the reduction time is generally positive for the MSP-SPP, ranging from 3% to 93% when $c_a \in (1{,}20)$, and from 54% to 97% when $c_a \in (1{,}100)$. Similarly, the reduction time for the MSP-SPP ranges from 60.38% to 94.33% when $c_a \in (1{,}20)$, and from 88.44% to 99.61% when $c_a \in (1{,}100)$. Moreover, in both cases the reduction is more marked when $c_a \in (1{,}100)$. Similarly to the case of T-1-persistent arcs, we experienced a lack of sufficient memory to solve exactly larger instances without preprocessing T-0-persistent arcs. This phenomenon is evidenced in the right-most graph of Fig. 15 by means of negative bars. This fact shows once again that, for these instances at least, the pegging test proves extremely useful.

6. Conclusions

We investigated two versions of the Shortest Path Problem (SPP) under interval uncertainty, namely the Minimax regret Single-Pair Shortest Path Problem (MSP-SPP), and the Minimax regret Single-Source Shortest Path Problem (MSS-SPP). The former consists in minimizing the maximum deviation from the optimal single-pair shortest path over all possible feasible assignments of arc lengths; the latter consists in minimizing the maximum deviation from the optimal single-source shortest paths over all possible feasible assignments of arc lengths. We extended the results given by Karasan et al. [7] by considering finite directed graphs in which cycles may occur and arc lengths may *degenerate*, i.e., be deterministically known. We gave sufficient conditions for a node or an arc to be always or never in an optimal solution of the MSP-SPP. Similarly, we gave sufficient conditions for an arc to be always or never in an optimal solution of the MSS-SPP. From such conditions we developed pegging tests, similar to the ones used by Yokoya et al. for the repeated assignment problem (see Yokoya et al. [10]), and we showed, by means of numerical results, that the tests are efficient in practice and vastly reduce the overall running time necessary to exactly solve instances of both the MSP-SPP and the MSS-SPP.

Acknowledgments

The first author acknowledges support from the Belgian National Fund for Scientific Research (F.N.R.S.), of which he is "Chargé de Recherches". Both the first and the second authors also acknowledge support from Communauté Française de Belgique—Actions de Recherche Concertées (ARC) and "Ministerio de Ciencia e Innovación" through the research project MTM2009-14039-C06. Finally, the authors thank the anonymous referees for helpful comments on the previous version of the manuscript.

References

[1] Ahuja RK, Magnanti TL, Orlin JB. Network flows. Upper Saddle River, NJ: Prentice-Hall; 1993.
[2] Montemanni R, Gambardella LM. The robust shortest path problem with interval data via Benders' decomposition. 4OR 2005;3:315–28.
[3] Kouvelis P, Yu G. Robust discrete optimization and its applications. Boston, MA: Kluwer Academic Publishers; 1997.
[4] Averbakh I, Lebedev V. Interval data minmax regret network optimization problems. Discrete Applied Mathematics 2004;138(3):289–301.
[5] Zielinski P. The computational complexity of the relative robust shortest path problem with interval data. European Journal of Operations Research 2004;158:570–6.
[6] Kasperski A, Zielinski P. The robust shortest path problem in series-parallel miltidigraphs with interval data. Operations Research Letters 2006;34:69–76.
[7] Karasan OE, Pinar MC, Yaman H. The robust shortest path problem with interval data. Technical Report, Bilkent University. Ankara, Turkey, http://www.optimization-online.org; 2001.
[8] Montemanni R, Gambardella LM, Donati AV. A branch-and-bound algorithm for the robust shortest path problem with interval data. Operations Research Letters 2004;32:225–32.
[9] Montemanni R, Gambardella LM. An exact algorithm for the robust shortest path problem with interval data. Computers and Operations Research 2004; 31:1667–80.
[10] Yokoya D, Duin CW, Yamada T. A reduction approach to the repeated assignment problem. European Journal of Operations Research 2011;210(2): 185–93, doi:10.1016/j.ejor.2010.10.027.
[11] Salazar-Neumann M. Advances in robust combinatorial optimization and linear programming. PhD thesis, G.O.M., Université Libre de Bruxelles (U.L.B.), Brussels, Belgium; 2010.