Computers & Operations Research 39 (2012) 2888–2898

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

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

# Efficient determination of the $k$ most vital edges for the minimum spanning tree problem

Cristina Bazgan [a,b,*], Sonia Toubaline [a], Daniel Vanderpooten [a]

[a] Université Paris-Dauphine, LAMSADE Place du Maréchal de Lattre de Tassigny, 75775 Paris Cedex 16, France
[b] Institut Universitaire de France, France

ARTICLE INFO

Available online 28 February 2012

*Keywords:*
Most vital edges
Minimum spanning tree
Exact algorithms
Mixed integer program

ABSTRACT

We study in this paper the problem of finding in a graph a subset of $k$ edges whose deletion causes the largest increase in the weight of a minimum spanning tree. We propose for this problem an explicit enumeration algorithm whose complexity, when compared to the current best algorithm, is better for general $k$ but very slightly worse for fixed $k$. More interestingly, unlike in the previous algorithms, we can easily adapt our algorithm so as to transform it into an implicit enumeration algorithm based on a branch and bound scheme. We also propose a mixed integer programming formulation for this problem. Computational results show a clear superiority of the implicit enumeration algorithm both over the explicit enumeration algorithm and the mixed integer program.



## 1. Introduction

In many applications involving the use of communication or transportation networks, we often need to identify critical infrastructures. By critical infrastructure we mean a set of links whose damage causes the largest perturbation within the network. Modeling this network as a weighted graph, identifying critical infrastructures amounts to finding a subset of edges whose removal from the graph causes the largest increase in the total weight. In the literature this problem is referred to as the $k$ most vital edges problem. In this paper, we are interested in determining a subset of edges of the graph whose deletion causes the largest increase in the weight of a minimum spanning tree (MST). This problem is referred to as $k$ Most Vital Edges MST.

The problem of finding the $k$ most vital edges of a graph has been investigated for various problems including shortest path [1,9,15], maximum flow [22,18,23], 1-median and 1-center [2]. For the minimum spanning tree problem defined on a graph $G$ with $n$ vertices and $m$ edges, Frederickson and Solis-Oba [6] showed that, for general $k$, $k$ Most Vital Edges MST is $NP$-hard and proposed an $O(\log k)$-approximation algorithm. The problem remains $NP$-hard even for complete graphs with weights 0 or 1 and 3-approximable for graphs with weights 0 or 1 [3]. For a fixed $k$ the problem is obviously polynomial. The case $k=1$ has been largely studied in the literature [7,8,20]. Hsu et al. [7] gave two algorithms that run in $O(m \log m)$ and $O(n^2)$. Iwano and Katoh [8] proposed an algorithm in $O(m\alpha(m,n))$ using Tarjan's result [21], where $\alpha$ is the inverse Ackermann function. Pettie [16] improved the results of Tarjan [21] and Dixon et al. [5], giving rise to the current best deterministic algorithm in $O(m \log \alpha(m,n))$. For general $k$, several exact algorithms based on an explicit enumeration of possible solutions have been proposed [11,12,19]. The best one [11] runs in time $O(n^k\alpha((k+1)(n-1),n))$ and was achieved by reducing $G$ to a sparse graph. Using Pettie's result [16], the running time of the later algorithm becomes $O(n^k \log \alpha((k+1)(n-1),n))$.

In this paper, we propose a new efficient algorithm also based on an explicit enumeration of all possible solutions for $k$ Most Vital Edges MST. Its complexity $O(n^k \log \alpha(2(n-1),n))$ for fixed $k$ is theoretically very slightly worse than the complexity of the algorithm proposed by Liang [11] using Pettie's result [16]. However, given the fact that $\alpha(m,n)$ is always less than 4 in practice, the complexity of these two algorithms can be deemed as equivalent. Moreover, the complexity of our algorithm is better than that of Liang's algorithm for general $k$. More interestingly, unlike any other algorithm, our algorithm has two specific useful features. First, it can also determine an optimal solution for $i$ Most Vital Edges MST, for each $1 \le i \le k$, with the same time complexity. Second, it can be easily adapted to establish an implicit enumeration algorithm based on a branch and bound procedure. We also present in this paper a mixed integer programming formulation to solve $k$ Most Vital Edges MST. We implement and test all these proposed algorithms using, for the implicit enumeration algorithm, different branching and evaluation strategies. The results show that the implicit enumeration algorithm is much faster than the explicit enumeration algorithm as well as the resolution of the mixed integer program. Moreover, the implicit enumeration algorithm can handle significantly larger instances due to a better use of memory space. Finally, we also propose an $\varepsilon$-approximate algorithm.

* Corresponding author at: Université Paris-Dauphine, LAMSADE Place du Maréchal de Lattre de Tassigny, 75775 Paris Cedex 16, France.
*E-mail addresses:* bazgan@lamsade.dauphine.fr (C. Bazgan), toubaline@lamsade.dauphine.fr (S. Toubaline), vdp@lamsade.dauphine.fr (D. Vanderpooten).

0305-0548/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2012.02.023

The rest of the paper is organized as follows. In Section 2, we introduce notations and some results related to our problem. In Section 3, we present a new explicit enumeration algorithm that solves $k$ Most Vital Edges MST. In Section 4, we propose another exact algorithm based on an implicit enumeration scheme. In Section 5, we present a mixed integer programming formulation for $k$ Most Vital Edges MST. Computational results are presented in Section 6. We also include in these experiments an $\epsilon$-approximate version of our implicit enumeration scheme. Conclusions are provided in Section 7.

## 2. Basic concepts and preliminary results

Let $G=(V,E)$ be a weighted undirected connected graph with $|V|=n$, $|E|=m$, where $w(e)\geq 0$ is the integer weight of each edge $e\in E$. We denote by $G-E'$ the graph obtained from $G$ by removing the subset of edges $E'\subseteq E$. $k$ Most Vital Edges MST consists of finding a subset of edges $S^*\subseteq E$ with $|S^*|=k$ that maximizes the weight of a MST in the graph $G-S^*$. We assume that $G$ is at least $(k+1)$ edge-connected, since otherwise any selection of $k$ edges including the edges of a minimum unweighted cut is a trivial solution. Therefore, we assume $k\leq\lambda(G)-1$, where $\lambda(G)$ is the edge-connectivity of $G$. Also, without loss of generality, we suppose in the following that all weights are different (by introducing, if necessary, an arbitrary total order on edges with the same weight). This non-restrictive assumption implies the uniqueness of minimum spanning trees or forests. For a non-necessarily connected graph, a minimum spanning forest (MSF) is the union of minimum spanning trees for each of its connected components. In this paper, a tree or a forest is considered as a graph but also, for convenience, as a subset of edges. For a set of edges $F$, $w(F)$ represents the sum of the weights of the edges in $F$.

We denote by $T_0$ the MST of $G$. Remark that an optimal solution of $k$ Most Vital Edges MST must contain at least one edge of $T_0$. For $i\geq 1$, let $T_i$ be the MSF of the graph $G_i=G-\bigcup_{j=0}^{i-1}T_j$. We use in the following the graph $U_k^G=(V,\bigcup_{j=0}^{k}T_j)$, which has the following interesting property.

**Lemma 1** (*Liang and Shen [12]*). *For any $S\subseteq E$, $|S|\leq k$, any edge of the MST of graph $G-S$ belongs to $U_k^G$.*

By Lemma 1, solving $k$ Most Vital Edges MST on $G$ reduces to solving the same problem on the sparser graph $U_k^G$ whose number of edges is at most $(k+1)(n-1)$.

Considering $T$ a MST of a graph, the replacement edge $r(e)$ for an edge $e\in T$ is defined as the edge $e'\neq e$ of minimum weight which connects the two disconnected components of $T\backslash\{e\}$. The sensitivity of a minimum spanning tree $T$, i.e. the allowable variation for each edge weight so that $T$ remains a minimum spanning tree, can be computed in $O(m\log\alpha(m,n))$ [16]. In particular, for edges in $T$, this algorithm provides replacement edges. As a consequence, we get the following result.

**Lemma 2.** *1 Most Vital Edges MST defined on a graph with n vertices and m edges is solvable in $O(m\log\alpha(m,n))$.*

**Proof.** Let $T^*$ be the minimum spanning tree in a given graph. We calculate the replacement edges $r(e)$ for all edges $e\in T^*$. The most vital edge is the edge $e^*$ such that $w(r(e^*))-w(e^*)=\max_{e\in T^*} w(r(e))-w(e)$. □

Actually, replacement edges belong to a specific subset of edges as shown by the following result.

**Lemma 3.** *For each edge $e\in T_i$, we have $r(e)\in T_{i+1}$ for $i=0,\ldots,k-1$.*

**Proof.** Given a graph $G$, Liang [11] shows that for each edge $e\in T_0$, $r(e)\in T_1$. Applying this to graph $G_i$, for which $T_i$ is the MSF, we get the result. □

## 3. An explicit enumeration algorithm for finding the $k$ most vital edges

We propose an algorithm that constructs a search tree of depth $k-1$ in a breadth-first mode. The branching scheme is similar to the standard scheme used for enumerating the $k$ best solutions [14,10]. In this scheme, the subset of solutions is partitioned by excluding a subset of edges and including another subset of edges. More precisely in our case, at the $i$th level of the search tree, $i=0,\ldots,k-1$, a node $s$ is characterized by:

- $mv(s)$: a subset of $i$ edges that are excluded. These edges correspond to a tentative partial selection of the $k$ most vital edges.
- $\tilde{U}(s)=U_{k-|mv(s)|}^{G'(s)}$, where $G'(s)=(V,E\backslash mv(s))$. Hence, we have $\tilde{U}(s)=(V,\bigcup_{i=0}^{k-|mv(s)|}T_i(s))$, where $T_i(s)$ is the MSF in $G'(s)-\bigcup_{j=0}^{i-1}T_j(s)$.
- $mst(s)$: a subset of edges that must be included and are forbidden to deletion. These edges belonging to $T_0(s)$, will necessarily belong to any MST associated with any descendant of $s$. Depending on the position of $s$ in the search tree, the cardinality of $mst(s)$ varies from 0 to $n-2$.

Denote by $N_i$, for $i=0,\ldots,k-1$, the set of nodes of the search tree at the $i$th level. We describe in the following the exact algorithm (Section 3.1) and exemplify its use on an illustrative example (Section 3.2).

### 3.1. Description of the algorithm

We first construct the graph $U_k^G$. Let $a$ be the root of the search tree with $mv(a)=mst(a)=\emptyset$, $\tilde{U}(a)=U_k^G$, $w(T_0(a))=w(T_0)$, and $N_0=\{a\}$.

For a level $i$, $0\leq i\leq k-2$, we compute for each node $s\in N_i$ and each edge $e\in T_0(s)$, the replacement edges $r(e)$ in $T_1(s)$. Node $s$ gives rise to $n-1-|mst(s)|$ children in $N_{i+1}$. Each such child $d$, corresponding to an edge $e_j$ in $T_0(s)\backslash mst(s)=\{e_1,\ldots,e_{n-1-|mst(s)|}\}$, is characterized by:

- $mv(d)=mv(s)\cup\{e_j\}$.
- $mst(d)=mst(s)\cup(\cup_{\ell=1}^{j-1}\{e_\ell\})$.
- $\tilde{U}(d)$ is updated from $\tilde{U}(s)$ as follows (using Lemma 3):
  - $T_0(d)=T_0(s)\cup\{r(e_j)\}\backslash\{e_j\}$ and hence $w(T_0(d))=w(T_0(s))-w(e_j)+w(r(e_j))$.
  - For $j=1,\ldots,k-|mv(d)|$, $T_j(d)$ is obtained from $T_j(s)$ by deleting the replacement edge $e_{rep}$ of the edge deleted from $T_{j-1}(s)$ and replacing it by its replacement edge $r(e_{rep})\in T_{j+1}(s)$. If for a level $i$ and an edge $e_{rep}$, then the replacement edge $r(e_{rep})$ does not exist, $T_j(d)=T_j(s)\backslash\{e_{rep}\}$ and $T_\ell(d)=T_\ell(s)$ for $\ell=j+1,\ldots,k-|mv(d)|$. If for a level $i$, $T_i(s)=\emptyset$, then $T_\ell(d)=\emptyset$ for $\ell=i,\ldots,k-|mv(d)|$.

At level $k-1$, for each node $s\in N_{k-1}$ and for all edges $e\in T_0(s)\backslash mst(s)$, we find $r(e)$ in $T_1(s)$ and determine a node $s^*$ that verifies $\max_{s\in N_{k-1}}\max_{e\in T_0(s)\backslash mst(s)}(w(T_0(s))-w(e)+w(r(e)))$. An optimal solution is the subset $mv(s^*)\cup\{e^*\}$, where $e^*=\arg\max_{e\in T_0(s^*)\backslash mst(s^*)} w(T_0(s^*))-w(e)+w(r(e))$. The largest weight of a MST in the partial graph obtained by deleting this subset is $w(T_0(s^*))-w(e^*)+w(r(e^*))$.