ISTITUTO DI ANALISI DEI SISTEMI ED INFORMATICA
CONSIGLIO NAZIONALE DELLE RICERCHE

M. Conforti, A. Galluccio, G. Proietti

AUGMENTATION PROBLEMS AND NETWORK MATRICES

R. 583 Gennaio 2003

Michele Conforti – Dipartimento di Matematica Pura ed Applicata, Università di Padova, Via Belzoni 7, 35131 Padova, Italy. E-mail: conforti@math.unipd.it.

Anna Galluccio – Istituto di Analisi dei Sistemi ed Informatica del CNR, Viale Manzoni 30, 00185 Roma, Italy. Email: galluccio@iasi.rm.cnr.it.

Guido Proietti – Dipartimento di Informatica, Università di L'Aquila, Via Vetoio, 67010 L'Aquila, Italy, and Istituto di Analisi dei Sistemi ed Informatica del CNR, Viale Manzoni 30, 00185 Roma, Italy. Email: proietti@di.univaq.it.

This work has been partially supported by the CNR-Agenzia 2000 Program, under Grants No. CNRC00CAB8 and CNRG003EF8, and by the Research Project REAL-WINE, partially funded by the Italian Ministry of Education, University and Research.

ISSN: 1128–3378

Collana dei Rapporti dell'Istituto di Analisi dei Sistemi ed Informatica, CNR

viale Manzoni 30, 00185 ROMA, Italy

tel. ++39-06-77161
fax ++39-06-7716461
email: iasi@iasi.rm.cnr.it
URL: http://www.iasi.rm.cnr.it

Abstract

We study the following NP-hard graph augmentation problem: Given a weighted graph G and a connected spanning subgraph H of G, find a minimum weight set of edges of G to be added to H so that H becomes 2-edge-connected. We provide a formulation of the problem as a set covering problem and we exhibit instances in which the Linear Programming Relaxation of our formulation always gives an integer solution. This yields instances of the problem that can be solved in polynomial time, and suggests an interesting application for solving efficiently a survivability problem on communication networks. Finally, we present a 2-approximation algorithm for integer programs with constraint set defined by graphic matrices.

1. Introduction

Several network design problems have been formulated as integer programming problems and in many cases mathematical programming techniques revealed very effective to understand and to solve these problems. These techniques had a leading role in designing efficient approximation algorithms for network problems but they were also crucial in enlightening the structural properties of these problems. In particular, they help to understand where the hardness of a problem lies by characterizing those instances which are polynomial-time solvable.

Here, we consider the problem of strengthening the edge-connectivity of an existing network to fulfill increasing communication reliability requirements. Such a strengthening is realized through the addition of a set of new links, in such a way that the original degree of (either edge or node)-connectivity of the network is increased. Generally, these additional links must be selected from a set of *potential* links, where each potential link has a certain cost, based on some standard measure (e.g., the set-up cost). Therefore, besides from increasing the connectivity degree, one wants also to minimize some objective function of these link costs.

From a theoretical point of view (and for the edge-connectivity case, which is of interest for this paper), this gives rise to a family of *augmentation problems* on graphs [1]. Among them, one of the most basic problems can be formulated as follows: Given an undirected, 2-edge-connected and real weighted graph $G = (V, F)$, and given a connected spanning subgraph $H = (V, E)$, find a minimum-weight subset of $F - E$, say $\textsc{Aug}_2(G, H)$, such that $H' = (V, E \cup \textsc{Aug}_2(G, H))$ is 2-edge-connected.

This problem is NP-hard [4], and most of the research in the past focused on the design of approximation algorithms for solving it. To this respect, efficient approximation algorithms are known, with performance ratio 2 [4, 10]. For the unweighted case, improving the approximation ratio below 2 has been a long standing open problem. Just recently, Nagamochi [12] developed a $(1.875 + \epsilon)$-approximation algorithm, for any constant $\epsilon > 0$, afterwards improved to 3/2 by Even *et al.* [2]. Analogous versions of augmentation problems for node-connectivity and for directed graphs have been widely studied, and we refer the interested reader to the following comprehensive papers [3, 11].

The problem of characterizing polynomially solvable cases of the 2-edge-connectivity augmentation problem has been also investigated. The polynomiality of the problem can be achieved by giving additional constraints on the structure of the pair (G, H). First, Eswaran and Tarjan [1] proved that $\textsc{Aug}_2(G, H)$ can be found in polynomial time if G is complete and all edges in G have weight 1, i.e., all potential links between sites may be activated at the same cost. Afterwards, Watanabe and Nakamura extended this result to any desired edge-connectivity value [15], and faster algorithms in this case have been proposed in [5]. From another perspective, the structure of H can be equally relevant in order to have polynomial time algorithms. For example, $\textsc{Aug}_2(G, H)$ can be found in polynomial time if G is any weighted graph and H is a spanning tree of G which can be rooted in a node $r \in V$ in such a way that for every edge $uv \in F - E$, either u is an ancestor of v, or v is an ancestor of u in the rooted tree [6].

In this paper we move one significant step towards the characterization of the intrinsic complexity of the 2-edge-connectivity augmentation problem. In particular, we formulate the problem as a *set covering* problem on special matrices defined by the pair (G, H), and we provide sufficient conditions on (G, H) so that the linear programming relaxation of our formulation always gives an integer solution. The result has several consequences. First of all, it yields instances of the problem that can be solved efficiently. As we will show in the paper, these particular instances have not only theoretical but also practical interest, since they model a classic survivability

problem in many-to-one communication networks (i.e., networks where data flow from multiple sources towards a single sink). Furthermore, it allows us to build a 2-approximation algorithm to solve integer linear programs on a special class of $(0,1)$-matrices, called *graphic matrices*.

The paper is organized as follows: in Section 2, we recall some basic results and definitions from integer programming and graph theory. In Section 3, 4 and 5, we describe the formulation of the edge-connectivity augmentation problem as a set covering problem, and we prove sufficient and easy-to-check conditions that make it solvable in polynomial time. Then, in Section 6, we explain in detail how to transform the edge-connectivity augmentation problem into a min-cost circulation problem in a directed graph and, as a consequence, we provide an effective polynomial time algorithm to solve the 2-edge-connectivity augmentation problem in a large class of graphs. After, in Section 7 we show the practical impact of our achievement. Finally, the last section is devoted to the generalization of our result to integer programming with constraint set defined by graphic matrices.

2. Basic definitions

Let A be a $(0,1)$-matrix with n rows and m columns and let $w \in \mathbb{R}^m$ and $b \in \mathbb{R}^n$. An *integer programming (IP)* problem is the following:

$$\begin{array}{l} \min w^T x \\ s.t.\ Ax \geq b \\ \quad x \quad integral. \end{array} \tag{1}$$

Being $\mathcal{P} = \{x \in \mathbb{R}^m : Ax \geq b, x \geq 0\}$ a polyhedron, an IP problem consists in finding an integral vector $x^* \in \mathcal{P}$ that minimizes the linear function $w^T x$ over the integral vectors of $\mathcal{P}$. This problem is NP-hard, in general, but some instances can be solved as *linear programming (LP)* problems, for all objective functions. For example, when the polyhedron $\mathcal{P}$ is integral, i.e., all of its extreme points have integer coordinates, the integrality constraint in (1) can be replaced by $x \geq 0$, and the optimum of (1) can be found by any polynomial time algorithm developed for linear programming.

A $(0, \pm 1)$-matrix A is *totally unimodular* (TU, for short) if $det(A') = 0, \pm 1$ for every square submatrix A' of A. A classical result of Hoffman a Kruskal [7] shows that $\{x \in \mathbb{R}^m : Ax \geq b, x \geq 0\}$ is an integral polyhedron for every integral vector b if and only if A is a TU matrix.

The case when the elements of A, b and x in (1) are restricted to be 0 or 1, is usually known as the *set covering problem* and it is one of the most studied IP problems since many combinatorial optimization problems can be formulated in this form. The integrality of the set covering polyhedron $\{x \in \mathbb{R}^m : Ax \geq 1, x \geq 0\}$ is guaranteed by the weaker property of A of being *balanced*. A $(0,1)$-matrix M is balanced if and only if it contains no square submatrix of odd order whose row and column sums are all 2. Since the determinant of a square matrix of odd order whose row and column sums are all 2 is ± 2, it follows that every TU matrix is balanced. This containment in general is known to be strict.

We complete the section with some graph theory definitions. Let $G = (V, F)$ be an undirected graph with node set V and edge set F. G is said to be *weighted* if it is given a real function $w : F \mapsto \mathbb{R}$. Given two vertices $v_0, v_k \in V$, a (simple) *v_0v_k-path* in G is a sequence $\langle v_0, e_1, v_1, \ldots, e_k, v_k \rangle$, where v_i's are distinct vertices in V, and each e_i is an edge of F with endvertices v_{i-1} and v_i. A graph is *connected* if every pair of vertices is joined by a path. A graph G is *2-edge-connected* if the removal of any edge from G leaves G connected. Given a

node $v \in V$, the *star* of v in G, denoted $\delta_G(v)$, is the set of edges $\{e_1, e_2, \ldots, e_k\}$ of F having v as endnode. We denote by $deg_G(v)$ the cardinality of the star of v in G.

If $\vec{G} = (V, A)$ is a *directed* graph (*digraph*, for short), then we denote by $\delta^+(v)$ ($\delta^-(v)$) the set of arcs in A leaving (entering, respectively) a node $v \in V$. Given two vertices $v_0, v_k \in V$, a v_0v_k-path in $\vec{G}$ is a sequence $\langle v_0, a_1, v_1, \ldots, a_k, v_k\rangle$, where v_i's are distinct vertices in V, and each a_i is either an arc $v_{i-1}v_i$ or an arc v_iv_{i-1} of A. A v_0v_k-path in $\vec{G}$ is *directed* if every arc a_i along the path is oriented from v_{i-1} to v_i.

3. Formulating the augmentation problem

Given an undirected, 2-edge-connected and real weighted graph $G = (V, F)$, and given a connected spanning subgraph $H = (V, E)$ of G, finding a 2-*edge-connectivity augmentation* of H with respect to G means to select a minimum weight set of edges in $F - E$, denoted as $\textsc{Aug}_2(G, H)$, such that the graph $H' = (V, E \cup \textsc{Aug}_2(G, H))$ is 2-edge-connected. Notice that if H is connected, then, without loss of generality, we can assume that H is a tree. Indeed, each 2-edge-connected component of H can be contracted into a single *vertex*. This transforms the graph G into a multigraph $\mathcal{G}$, and the graph H into a tree T, whose edges are the bridges of H. It is then easy to see that finding $\textsc{Aug}_2(G, H)$ is equivalent to finding $\textsc{Aug}_2(\mathcal{G}, T)$. Based on that, we will restrict ourselves to the problem of finding 2-edge-connectivity augmentation of trees. Furthermore, since edges in $E \cap F$ are not feasible for augmenting the edge-connectivity of H, we will assume for the sake of simplicity that $E \cap F = \emptyset$.

Let then be given a real weighted graph $G = (V, F)$ and a tree $T = (V, E)$ defined over the same vertex set. Let $E = \{e_1, \ldots, e_n\}$, and $F = \{f_1, \ldots, f_m\}$, with $E \cap F = \emptyset$. The *graphic matrix* associated with the pair (G, T) is the $(0, 1)$-matrix M whose rows and columns are indexed by the edges in E and F, respectively, and whose elements are defined as follows:

$$m_{ij} = \begin{cases} 1 & \text{if } e_i \text{ belongs to the unique subpath of } T \\ & \text{between the endnodes of } f_j; \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

We formulate the 2-edge-connectivity augmentation problem as the following integer program:

$$\min \left\{ w^T x : Mx \geq \mathbf{1}, x \in \{0, 1\}^m \right\}. \tag{3}$$

Indeed, if we let w to be the vector of weights of the edges of G, it is easy to see that an optimal vector for (3) is the incidence vector of a minimum weight subset $F' \subseteq F$ such that $T' = (V, E \cup F')$ is 2-edge-connected.

The problem (3) is a *set covering* problem. This problem is NP-hard even if M is a graphic matrix, but if M is balanced, and hence, totally unimodular then all the basic feasible solutions (in the linear programming sense) of the constrains system $\{Mx \geq \mathbf{1}, x \geq 0\}$ are integer-valued and the problem becomes polynomial-time solvable.

In the following we explore which graphic matrices are balanced and we show that balancedness and totally unimodularity coincide in the class of graphic matrices. Then we design a combinatorial polynomial algorithm to solve the connectivity augmentation problem in the case of balanced graphic matrices and we provide some relevant applications.

4. Network matrices

A $(0, \pm 1)$-matrix N is a *network matrix* (see Schrijver [14]) if there exists a digraph $\vec{D} = (V, F)$ and a directed tree $\vec{T} = (V, E)$ on the same node set (with $E \cap F = \emptyset$) such that the rows of N are indexed by the edges of $\vec{T}$ and the columns of N are indexed by the arcs of $\vec{D}$. For $f_j = uv \in F$, the elements of N are defined as follows:

$$n_{ij} = \begin{cases} +1 & \text{if the unique } uv\text{-path in } T \text{ contains } e_i \text{ as a forward arc;} \\ -1 & \text{if the unique } uv\text{-path in } T \text{ contains } e_i \text{ as a backward arc;} \\ 0 & \text{if the unique } uv\text{-path in } T \text{ does not contain } e_i. \end{cases} \tag{4}$$

We summarize some known facts on TU and network matrices and prove them in a comprehensive and elementary way (see Schrijver [14] and Nemhauser, Wolsey [13] for more details).

Lemma 4.1. *Let A be a $(0, \pm 1)$-matrix having at most one $+1$ and one -1 in every column. Then A is TU.*

Proof. Assume not and let A' be a minimal square submatrix of A that is not TU. Assume A' contains a column with one nonzero entry: Then, by the cofactor expansion formula, A' contains a smaller matrix A'' where $det(A') = \pm det(A'')$ and the minimality of A' is contradicted. So by our assumption, A' contains exactly two nonzero entries in each column and they are $+1$ and -1. Therefore by adding all the rows the null vector is obtained, showing that A' is singular, and therefore $det(A') = 0$, a contradiction. ∎

Let $A_{\vec{T}}$ be the node-arc incidence matrix of a directed tree $\vec{T} = (V, E)$ and let $A_{\vec{D}}$ be the node-arc incidence matrix of a directed graph $\vec{D} = (V, F)$ on the same node set. Finally, let N be the network matrix associated with the pair $(\vec{D}, \vec{T})$. Let $f_i = uv$ be an arc of $\vec{D}$, and let N^{f_i} be the column of N indexed by f_i. Since N^{f_i} is the incidence vector of the uv-path in $\vec{T}$ with forward edges having $+$ sign and backward edges having $-$ sign, then $A_{\vec{T}} N^{f_i}$ is a vector indexed on V whose j-th component is defined as follows:

$$(A_{\vec{T}} N^{f_i})_j = \begin{cases} +1 & \text{if } j = u; \\ -1 & \text{if } j = v; \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

This shows that $A_{\vec{T}} N = A_{\vec{D}}$.

The proof of Lemma 4.1 shows that both $A_{\vec{T}}$ and $A_{\vec{D}}$ are not full rank matrices. Let $\tilde{A}_{\vec{T}}$ and $\tilde{A}_{\vec{D}}$ arise from $A_{\vec{T}}$ and $A_{\vec{D}}$ by deleting one row (corresponding to the same node). By using the fact that a spanning tree always has a node of degree 1, it is easy to prove that the columns of $\tilde{A}_{\vec{T}}$ are independent and therefore $\tilde{A}_{\vec{T}}$ is a square nonsingular matrix where $rank(A_{\vec{T}}) = rank(\tilde{A}_{\vec{T}})$. So the system $A_{\vec{T}} N = A_{\vec{D}}$ implies that $N = \tilde{A}_{\vec{T}}^{-1} \tilde{A}_{\vec{D}}$.

Lemma 4.2. *Let N be a network matrix. Then N is TU.*

Proof. Assume N is associated with the pair $\vec{D} = (V, F), \vec{T} = (V, E)$. Let N' be a submatrix obtained by removing columns corresponding to $F' \subseteq F$ and rows corresponding to $E' \subseteq E$. Let $\vec{D}'$ be obtained from $\vec{D}$ by deleting the edges in F' and $\vec{T}'$ be obtained from $\vec{T}$ by contracting the edges in E' (i.e., identifying their endnodes and removing the corresponding loops). It is immediate to see that N' is the network matrix associated with the pair $\vec{D}', \vec{T}'$. So network matrices are closed under taking submatrices and therefore it suffices to show that $det(N') =$

$0, \pm 1$. By Lemma 4.1, both $A_{\vec{T}}$ and $A_{\vec{D}}$ are TU. So $det(\tilde{A}_{\vec{T'}}^{-1}) = \pm 1$ since $\tilde{A}_{\vec{T'}}$ is nonsingular. Since $N' = \tilde{A}_{\vec{T'}}^{-1}\tilde{A}_{\vec{D'}}$, then $det(N') = \pm det(\tilde{A}_{\vec{D'}}) = 0, \pm 1$. ∎

5. Graphic matrices and network matrices

It follows from the definitions of graphic and network matrices that a graphic matrix M associated with an undirected graph G and a tree T is also a network matrix if and only if there is an orientation $\vec{G}$ of G and $\vec{T}$ of T such that for each arc uv of $\vec{G}$ the uv-path in $\vec{T}$ is a directed path from v to w (i.e. all the edges are forward edges). We call these orientations *good*.

Note that if $(\vec{G}, \vec{T})$ is a good orientation of G, T, then also the orientations obtained from $\vec{G}$, $\vec{T}$ by reversing the directions of all the arcs of $\vec{G}$ and $\vec{T}$ are good. This shows that, in order to find a good orientation if one exists, one can orient an edge of G or T arbitrarily. Note now that the orientation of an edge e of T forces the orientation of all the edges of G that contain e in the subpath of T between their endnodes. Furthermore the orientation of an edge f of G forces the orientations of every edge of T that is contained in the subpath of T between the endnodes of f. It is immediate now to see that this rule provides a fast algorithm to find a good orientation of G, T if one exists.

We now provide a characterization of the graphic matrices that are also network matrices, i.e., which pairs (G, T) admit a good orientation. A node v of T is a *junction* if $deg_T(v) \geq 3$. For any junction v with $\delta_T(v) = \{e_1, e_2, \ldots, e_k\}$, let G_v be the graph whose node set $v_{e_1}, v_{e_2}, \ldots, v_{e_k}$ represents edges $e_1, e_2, \ldots, e_k$ and $v_{e_i}v_{e_j} \in E(G_v)$ if and only if e_i, e_j belong to an xy-subpath identified by an edge $f = xy$ of G.

Theorem 5.1. *A graphic matrix M associated with G and T is a network matrix if and only if for every junction v of T, the graph G_v is bipartite.*

Proof. (Necessity) Assume G_v is not bipartite for some $v \in V$: Then G_v contains an odd chordless cycle $v_{e_1}, v_{e_2}, \ldots, v_{e_j}$. This shows that T contains distinct edges $e_1, e_2, \ldots, e_j$, j odd, having v as endnodes and G contains distinct edges $f_1, f_2, \ldots, f_j$ such that for $1 \leq i \leq j-1$, e_i, e_{i+1} are consecutive edges of the subpath of T between the endnodes of f_i and e_j, e_1 are consecutive edges of the subpath of T between the endnodes of f_j. Let B be the square submatrix of M whose rows and columns are indexed by $e_1, e_2, \ldots, e_j$ and $f_1, f_2, \ldots, f_j$. Now B is a square matrix of odd order j and by the above argument, B has two 1's per row and column. By the cofactor expansion formula, it is easy to see that $det(B) = \pm 2$. So M is not TU and by Lemma 4.2, M is not a network matrix.

(Sufficiency) Let (G, T) be a pair satisfying the hypothesis of the theorem. he proof is by induction on the number of junctions of T.

Assume first that T has only one junction v, and let E_1, E_2 be a partition of the edges in $\delta_T(v)$ corresponding to the bipartition of the nodes of G_v. Orient the edges in $\delta_T(v)$ so that $\delta_T^-(v) = E_1$, $\delta_T^+(v) = E_2$. This orientation forces the orientation of all the edges of G and all the other edges of T in some subpath between endnodes of edges of G. Since every such subpath contains an edge of E_1 and one edge of E_2, the orientation thus obtained is good.

So T contains at least two junctions. Let V_1, V_2 be a partition of V so that only one edge e of T is in $\delta_T(V_1)$ and the subtrees T_1, T_2, obtained from T by contracting V_2, V_1 into single nodes v_2, v_1 and removing loops, both contain less junctions than T. Let G_1, G_2, obtained from G by contracting V_2, V_1 into single nodes v_2, v_1. Note that both pairs (G_1, T_1) and (G_2, T_2) satisfy the hypothesis of the theorem. So, by the induction hypothesis, both pairs (G_1, T_1) and (G_2, T_2)

admit good orientations and we can assume without loss of generality that e is oriented the same way in T_1 and T_2. This forces that edges of G that are common to G_1 and G_2 to get the same orientation. It is now immediate to combine the two orientations to obtain a good orientation of (G, T). ∎

From the proof of the necessity part and from Lemma 4.2 we can deduce the following:

Corollary 5.2. *Let M be a graphic matrix. Then M is balanced if and only if M is a network matrix.*

The results of this section can also be generalized. In fact, the totally unimodularity of a graphic matrix A implies a more general result (by the Hoffmann-Kruskal theorem), namely that the optimal solution of the following linear program

$$\min\{w^T x : Ax \geq b, c \geq x \geq 0\} \tag{6}$$

is integral for any b and c integral vectors. So, depending on the values of b and c, it is possible to identify interesting network problems that can be solved in polynomial time when A satisfies conditions of Theorem 5.1. So, in the next section we describe in detail an algorithm that solves the problem (3), and in the final section we provide two applications.

6. Augmentation as circulation

We now show that the set covering problem with constraint matrices that are network matrices can be solved by flow techniques. Consider the linear programming relaxation

$$\min\{w^T x : Mx \geq \mathbf{1},\ x \geq 0\} \tag{7}$$

of the set covering problem and rewrite it in standard form by adding a vector s of slack variables:

$$\min\{w^T x : Mx - Is = \mathbf{1},\ x, s \geq 0\}. \tag{8}$$

Now, suppose that M is a network matrix, associated with the pair (G, T). Then by Lemma 4.2 M is a TU matrix and by the Hoffman-Kruskal's theorem every basic feasible solution of the above linear program is an integral vector. So an optimal solution of the above LP solves the set covering problem.

Furthermore if M is a network matrix, the pair (G, T) admits a good orientation $(\vec{G}, \vec{T})$. Hence, $M = \tilde{A}_{\vec{T}}^{-1} \tilde{A}_{\vec{G}}$ and we obtain the following equivalent linear program:

$$\min\{w^T x : \tilde{A}_{\vec{G}} x - \tilde{A}_{\vec{T}} s = \tilde{A}_{\vec{T}} \mathbf{1},\ x, s \geq 0\}. \tag{9}$$

Note that the matrix $-A_{\vec{T}}$ is the incidence matrix of the directed tree $\vec{T}'$ obtained from $\vec{T}$ by reversing the orientations of all its arcs. By substituting the vector s with $s' = s + 1$, the linear program (9) can be written as the *min-cost circulation problem*:

$$\min\{w^T x : \tilde{A}_{\vec{G}} x + \tilde{A}_{\vec{T}'} s' = \mathbf{0},\ x \geq 0, s' \geq 1\}, \tag{10}$$

and an integral optimal solution can be found by flow techniques, see e.g. [13].

The equivalence between the above set covering problem and the circulation problem can be seen directly as follows. In the sequel, for any set S and vector $p \in \mathbb{R}^{|S|}$ and for any $U \subseteq S$, we use $p(U)$ to abbreviate $\sum_{i \in U} p_i$. The equality constraints of the above linear system are:

$$x(\delta^+_{\vec{G}}(v)) - x(\delta^-_{\vec{G}}(v)) + s'(\delta^+_{\vec{T}'}(v)) - s'(\delta^-_{\vec{T}'}(v)) = 0 \quad \forall v \in V, \tag{11}$$

For edge e_j of $\vec{T}$, let $S_j \subset V$ be such that $\delta^+_{\vec{T}'}(S_j) = e_j$, where $\delta^+_{\vec{T}'}(S_j) = \bigcup_{v \in S_j} \delta^+_{\vec{T}'}(v)$. If $x \geq 0, s' \geq 1$ is a vector satisfying (11), then

$$x(\delta^+_{\vec{G}}(S_j)) - x(\delta^-_{\vec{G}}(S_j)) + s'(\delta^+_{\vec{T}'}(S_j)) - s'(\delta^-_{\vec{T}'}(S_j)) = 0.$$

Since $s'(\delta^+_{\vec{T}'}(S_j)) \geq 1$ and $\delta^-_{\vec{T}'}(S_j) = \emptyset$, we have that $x(\delta^-_{\vec{G}}(S_j)) \geq 1$ and the constraint of (7) associated with e_j is satisfied.

On the other hand, assume that x is a solution of (7) and let

$$s'_{e_i} = \sum_{f_j = uv \text{ contains } e_i \text{ in its } uv\text{-path}} x_{f_j}.$$

Then, it is straightforward to check that if x is an augmenting set, then $s'_{e_i} \geq 1$, for each $e_i \in E$ and x, s' is a flow vector in a good orientation of (G, T).

7. Practical impact

The results of previous sections have some interesting practical applications.

One arises in networks in which the communication occurs in a *many-to-one* fashion. In these networks, we have a set of *source* nodes which transmit messages to a single destination node, named the *sink*. Messages are routed from the sources to the sink through a set of intermediate nodes, which serve as *routers*. As a classic example of networks using a many-to-one communication protocol, we mention *sensor networks*, where data of interest (e.g., temperature, pollution index, etc.) are gathered at different locations (sources), and are transmitted to a single destination point (sink), where they can be stored and analyzed.

Given the inherent hierarchical structure, the typical topology of a many-to-one communication network is a directed rooted tree, with the sink as the root, the sources as the leaves, the routers as the internal nodes, and all the arcs oriented towards the root. In this tree, an arc uv represent the communication channel between node u and v. Hence, depending on the boundary conditions, such a link will afford a given amount of traffic, named its *load*. In the simplest situation, each link handles an amount of traffic which is proportional to the number of source nodes using it as a bridge. Therefore, the communication tree can be modelled as a tree $T(r) = (V, E)$ rooted at the source node r, where with each $e \in E$ is associated an integer $b(e)$ representing the link load.

Under normal working conditions, the tree links guarantee the data flow from the sources to the sink. However, networks usually undergo link failures, and therefore one might be interested in increasing the network reliability by creating alternative paths from the leaves towards the root. This can be accomplished by adding new links to the tree. Given the underlying hierarchical structure, we can assume that these links join *related* nodes in $T(r)$, i.e., nodes which belong to a given root-leaf path (this can be motivated, for instance, by constraints over the terrain where the network is deployed). Each of these additional links f is characterized by two main features: (1) a weight $w(f)$, which summarizes the cost to route a message through that link, and (2) a link capacity, say $c(f)$, which denotes the maximum amount of traffic load that can be routed through f. Let $G = (V, F)$ be the corresponding capacitated weighted graph.

10.

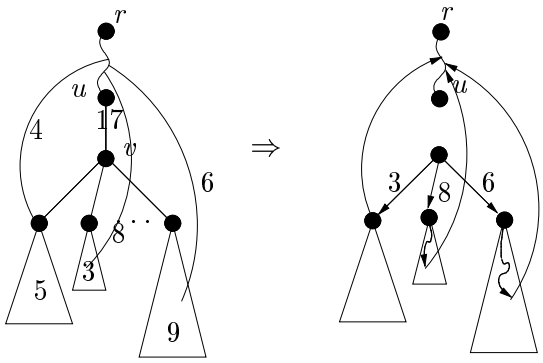

Figure 1: An edge uv in $T(r)$ is removed, and messages are redirected through replacement edges. On the left, values in triangles denote the number of sources in the subtree, the value on edge uv denote the load, while values on replacement edges denote capacities; on the right, values on tree edges denote the number of redirected messages.

Then, in the depicted scenario, it makes sense to define the following problem: Given the couple $(G, T(r))$, find a minimum-cost set of edges in G whose addition to T guarantees that the traffic load $b(e)$, previously carried on e, can be re-routed on the additional edges, for every $e \in E$. Indeed, after the failure of any $e = uv$ of $T(r)$, an optimal solution for this problem allows us to maintain (with a minimum set-up cost) the traffic between the leaves and the root, by means of the following simple changes to the communication protocol: let $f_1, \ldots, f_k$ be the set of additional edges associated with e (notice that $c(f_1) + \ldots c(f_k) \geq b(e)$); then, collect in v all the messages sent from the leaves below v in $T(r)$, and redistribute them among the replacement edges (according to their capacity) by letting the messages descend $T(r)$ to reach the starting nodes of $f_1, \ldots, f_k$, from where they will be routed above u (see Figure 1).

From a linear programming point of view, the above problem can be formulated as follows:

$$\min\{w^T x : Ax \geq b, c \geq x \geq 0\} \tag{12}$$

where w, b and c are defined as above, and A is the graphic matrix associated with G and T, considering this latter as undirected. Since $T(r)$ is a tree and edges in G joins related nodes of $T(r)$, it is easy to see that for each non-leaf node $v \in V, v \neq r$, G_v is a forest (more precisely, it is a star plus possibly some isolated nodes). Thus, it is bipartite, and our result applies. Therefore, this problem is solvable in polynomial time.

8. Further extensions

A further important application of Theorem 5.1 concerns the existence of a factor 2 approximation algorithm for solving the following IP problem, called *graphic IP problem*:

$$\min\{w^T x : Ax \geq b, x \geq 0, x \textit{ integral}\} \qquad A \textit{ graphic.} \tag{13}$$

This problem is NP-hard (it is easy to see that the IP formulation of the general k-edge-connectivity augmentation problem reduces to a graphic IP problem). The algorithm starts from a pair (G, T) whose weight vector for edges of G is w and whose graphic matrix is A, and

constructs a directed pair $(\vec{G}', \vec{T})$ whose weight vector for edges of G' is $\tilde{w}$ and whose graphic matrix A' is also a network matrix. By Corollary 5.2, the linear program $\min\{\tilde{w}^T x : A'x \geq b, x \geq 0\}$ has an integer optimal solution x^*, and the algorithm modifies x^* to obtain a feasible solution of the original graphic IP problem whose weight is no more than twice the optimum.

The algorithm reminds the procedure developed by Khuller and Thurimella [10] to obtain a 2-approximation algorithm for the 2-edge-connectivity augmentation problem and it works as follows:

2-APPROX ALGORITHM FOR GRAPHIC IP PROBLEM

Input: A graphic matrix A associated with $G = (V, F)$ and $T = (V, E)$ and a weight vector w associated with the edges of G.

1. Select an arbitrary node r of V and root T at r. Orient all edges of T away from the root. Denote by $\vec{T}$ this orientation;
2. Construct $\vec{G}'$ as follows. For each edge $e = uv \in F$:
   - 2a. if the uv-path, or the vu-path, is directed in $\vec{T}$, then add uv or vu (respectively) to $\vec{G}'$, by maintaining its weight;
   - 2b. otherwise let t be the least common ancestor of u and v in T. Add arcs tu and tv to $\vec{G}'$, and set $\tilde{w}(tu) = \tilde{w}(tv) = w(uv)$.
3. Clearly, $(\vec{G}', \vec{T})$ is a good orientation and the graphic matrix A' associated with $\vec{G}'$ and $\vec{T}$ is TU (by Corollary 5.2). Find an optimal solution x^* of the linear program $\min\{\tilde{w}^T x : A'x \geq b, x \geq 0\}$.
4. Construct a solution $\hat{x}^*$ of (13) as follows:

   if $x^*_e > 0$ and $e \in F$ then $\hat{x}^*_e = x^*_e$;
   if $x^*_f > 0$ and $f \in E(G') \setminus F$, then there exists an edge $e = uv \in F$ such that $f = tu$ and $f' = tv$. Then set $\hat{x}^*_e = \max\{x^*_f, x^*_{f'}\}$;
   otherwise $\hat{x}^*_e = 0$.

Output: A feasible solution $\hat{x}^*$ of the graphic IP problem.

The feasibility of the solution output by the algorithm is easy to check. To prove that the approximation factor is 2 we show that:

Theorem 8.1. *The solution $\hat{x}^*$ output of the algorithm is less than twice the optimal solution x_{OPT} of the graphic IP problem.*

Proof. Let x_{OPT} be an optimal solution of (13). Take the edges of G indexed by the nonzero components of x_{OPT}, say H, and consider the arcs of $\vec{G}'$ "generated" by H using step 2b of the algorithm. If tu and tv are the arcs generated by $e \in H$, then we set $\bar{x}_{tu} = \bar{x}_{tv} = (x_{\mathrm{OPT}})_e$. It is not difficult to see that $\bar{x}$ satisfies the constraint set $A'\bar{x} \geq b, \bar{x} \geq 0$, and it induces a solution of weight less than or equal to $2w(x_{\mathrm{OPT}})$.

Hence, $w(\hat{x}^*) \leq w(x^*) \leq w(\bar{x}) \leq 2w(x_{\mathrm{OPT}})$, as claimed. ∎

Results of similar flavour can be found in [9], where a 2-approximation algorithm for the IP problem (1) with bounded variables is provided over constraints with no more than two variables in each constraint (see [8] for a survey on the topic).

12.

References

[1] K.P. Eswaran and R.E. Tarjan, Augmentation problems, *SIAM Journal on Computing*, **5**(4) (1976) 653–665.

[2] G. Even, J. Feldman, G. Kortsarz and Z. Nutov, A 3/2-approximation algorithm for augmenting the edge-connectivity of a graph from 1 to 2 using a subset of a given edge set, *4th Int. Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX 2001)*, Vol. 2129 of Lecture Notes in Computer Science, Springer, 90–101.

[3] A. Frank, Augmenting graphs to meet edge-connectivity requirements, *SIAM Journal on Discrete Mathematics*, **5**(1) (1992) 25–53.

[4] G.N. Frederickson and J. Jájá, Approximation algorithm for several graph augmentation problems, *SIAM Journal on Computing*, **10**(2) (1981) 270–283.

[5] H.N. Gabow, Application of a poset representation to edge-connectivity and graph rigidity, *Proc. 32nd Ann. IEEE Symp. on Foundations of Computer Science (FOCS'91)*, IEEE Computer Society, 812–821.

[6] A. Galluccio and G. Proietti, Polynomial time algorithms for edge-connectivity augmentation problems, *Algorithmica*, **36**(4) (2003) 361–374.

[7] A.J. Hoffman and J. B. Kruskal, Integral boundary points of convex polyhedra, in: H. W. Kuhn and A. W. Tucker (eds.) *Linear inequalities and related systems*, Princeton University Press, New Jersey (1956), 223-246.

[8] D. S. Hochbaum, Approximating covering and packing problems: set cover, vertex cover, independent, and related problems, in *Approximation Algorithms for NP-Hard Problems*, Dorit S. Hochbaum Eds., PWS Publishing Company, Boston, MA, 1996.

[9] D. S. Hochbaum, N. Megiddo, J. S. Naor and A. Tamir, Tight bounds and 2-approximation algorithms for integer programs with two variables per inequality, *Math. Programming*, **62** (1993) 69–83.

[10] S. Khuller and R. Thurimella, Approximation algorithms for graph augmentation, *Journal of Algorithms*, **14**(2) (1993) 214–225.

[11] S. Khuller, Approximation algorithms for finding highly connected subgraphs, in *Approximation Algorithms for NP-Hard Problems*, Dorit S. Hochbaum Eds., PWS Publishing Company, Boston, MA, 1996.

[12] H. Nagamochi, An approximation for finding a smallest 2-edge-connected subgraph containing a specified spanning tree, *Discrete Applied Mathematics*, **126**(1) (2003) 83–113.

[13] G. L. Nemhauser and L. A. Wolsey, *Integer and Combinatorial Optimization*, J. Wiley & Sons, (1986).

[14] A. Schrijver, *Theory of linear and integer programming*, J. Wiley & Sons, (1986).

[15] T. Watanabe and A. Nakamura, Edge-connectivity augmentation problems, *Journal of Computer and System Sciences*, **35**(1) (1987) 96–144.