Available online at www.sciencedirect.com

ScienceDirect

Discrete Applied Mathematics 155 (2007) 572–578

DISCRETE APPLIED MATHEMATICS

www.elsevier.com/locate/dam

Note

Flow trees for vertex-capacitated networks

Refael Hassin[a], Asaf Levin[b]

[a]Department of Statistics and Operations Research, Tel-Aviv University, Tel-Aviv 69978, Israel
[b]Department of Statistics, The Hebrew University, Jerusalem 91905, Israel

Received 30 August 2004; accepted 4 August 2006
Available online 17 November 2006

Abstract

Given a graph $G=(V,E)$ with a cost function $c(S)\geqslant 0\,\forall S\subseteq V$, we want to represent all possible min-cut values between pairs of vertices i and j. We consider also the special case with an additive cost c where there are vertex capacities $c(v)\geqslant 0\ \forall v\in V$, and for a subset $S\subseteq V$, $c(S)=\sum_{v\in S}c(v)$. We consider two variants of cuts: in the first one, *separation*, $\{i\}$ and $\{j\}$ are feasible cuts that disconnect i and j. In the second variant, *vertex-cut*, a cut-set that disconnects i from j does not include i or j. We consider both variants for undirected and directed graphs. We prove that there is a flow-tree for separations in undirected graphs. We also show that a compact representation does not exist for vertex-cuts in undirected graphs, even with additive costs. For directed graphs, a compact representation of the cut-values does not exist even with additive costs, for neither the separation nor the vertex-cut cases.

Keywords: Cut-tree; Flow-tree; Compact representation

1. Introduction

Let $G=(V,E)$ be an undirected graph with edge capacities $c_e\geqslant 0\ e\in E$. For $s,t\in V$, an $s-t$ *edge-cut* is a partition (S,T) of V such that $s\in S$ and $t\in T$. The capacity $c(S,T)$ of the cut is $\sum_{e:|S\cap e|=1}c_e$. The ALL-PAIR MINIMUM CUT PROBLEM is to compute a minimum capacity $s-t$ cut for every $s,t\in V$. If G is a tree, the problem is particularly simple. In this case, the minimum $s-t$ cut value is the smallest capacity over the edges of the unique $s-t$ path, and a minimum $s-t$ edge-cut is induced when this edge is removed from the tree. Given G, an edge-capacitated tree T with vertex set V is called a *flow-tree* if for every $s,t\in V$, the value of a minimum $s-t$ cut in G and T is equal. The reason for this terminology is that by the max-flow min-cut theorem, the maximum $s-t$ flow is equal to the minimum $s-t$ cut value. Hence a flow tree T provides a *compact representation* of all the maximum $s-t$ flows in G. If not only the minimum cut values of G and T are the same but also the minimum cuts in T are minimum cuts in G, then T is called a *cut-tree*. A celebrated theorem by Gomory and Hu [4] states that a cut-tree always exists. We note that a cut-tree is a flow-tree, but in general flow-trees which are not cut-trees also exist. A consequence of this theorem is that there is a set of $|V|-1$ cuts that contains a minimum $s-t$ cut for every $s,t\in V$. This is a surprising result since the number of $s-t$ cut problems is a quadratic function of $|V|$. Moreover, Gomory and Hu also presented an algorithm that constructs a cut-tree by computing $|V|-1$ minimum $s-t$ cuts.

E-mail addresses: hassin@post.tau.ac.il (R. Hassin), levinas@mscc.huji.ac.il (A. Levin).

0166-218X/\$ - see front matter
doi:10.1016/j.dam.2006.08.012

Hassin [9] and Cheng and Hu [2] considered a general symmetric cost function $c(S,T)$ over the partitions of V. In this generalization no graph is defined, and the minimum $s-t$ cut is simply the minimum cost partition such that $s \in S$ and $t \in T$. They provided constructions of a flow-tree for this extension.

Hassin [9–11], Hartvigsen [7,8], and Einstein and Hassin [3] also extended Gomory–Hu's result differently, by considering a large number of problems of a common type. The problems are defined over a space of solutions, share the same objective function to be optimized, and differ in their set of constraints. They extended the notions of cut and flow-trees to general data-structures that compactly represent the optimal solutions.

Consider an undirected graph $G=(V,E)$ with a non-negative cost function $c(S)\ \forall S \subseteq V$. An $s-t$ *vertex-cut* is a set of vertices $S \subseteq V$ such that s and t are in distinct connected components of the graph induced by $V \backslash S$ in G (in particular, $s,t \in V\backslash S$). Note that if $(s,t)\in E$, then there is no $s-t$ vertex-cut. The minimum $s-t$ vertex-cut problem is to find a minimum cost $s-t$ vertex-cut. An $s-t$ *separation* is a set of vertices $S \subseteq V$ such that at least one of the following conditions holds: $s \in S$ or $t \in S$ or S is an $s-t$ vertex-cut. The minimum $s-t$ separation problem is to find a minimum cost $s-t$ separation. An *additive cost c* is the special case where there are vertex capacities $c(v)\ \forall v \in V$, and for a subset $S \subseteq V$, $c(S)=\sum_{v\in S} c(v)$. For separations and additive costs, Granot and Hassin [5] proved that a flow-tree exists. Gusfield and Naor [6] claimed that for this case there is also a cut-tree. However, Benczúr [1] showed that such a cut-tree does not exist.

We note that there is a significant difference between the two problems. In separations, for every three vertices i, j, k, an $i-j$ separation is either an $i-k$ separation or a $j-k$ separation. However, this is not true for vertex-cuts. For example, $\{k\}$ may be an $i-j$ vertex-cut but is never a $k-i$ nor a $k-j$ vertex cut.

For the vertex-cut case with additive costs Benczúr [1] suggested an algorithm that constructs a flow-tree over an extended set of $2n$ vertices. However, the construction is incorrect, and we prove that a compact representation for the all-pair vertex-cut problem with additive costs is not helpful, since it might have $\Theta(n^2)$ distinct optimal values. Next, we present a flow-tree construction for separations with general costs. Finally we show that when the graph is directed, neither separations nor vertex-cuts have a flow-tree even with additive costs because there might be $\Theta(n^2)$ distinct optimal values.

2. The number of minimum vertex-cuts

Benczúr [1] suggested the following construction to obtain a flow-tree with $2n$ vertices. First apply the standard transformation of a vertex capacitated graph $G=(V,E)$ to an arc capacitated directed graph $D=(\tilde{V},A)$, (i.e., replace each edge of E by two anti-parallel directed arcs with infinite capacity, then replace each vertex $v \in V$ by two vertices v' and v'' so that the arcs entering v now enter v' and those leaving v now leave v''. Also add the arc (v',v'') with capacity $c(v)$). Now assign a value $c(S)$ to every cut $S \subset \tilde{V}$ that is equal to the minimum of the two directed edge cuts $(S, \tilde{V}\backslash S)$ and $(\tilde{V}\backslash S, S)$ associated with it. There is an exception to this rule, that whenever one of the cut sides contains a single vertex its value is set to be infinity. In this construction any finite capacity cut corresponds to a subset of V, which is a vertex-cut with the same capacity as the edge-cut. This produces a symmetric function, and according to Hassin [9] and Cheng and Hu [2] there exists a flow-tree.

The mistake in these arguments arises since a set $\{i, j\}$ results in a finite cost set. But this set is not an $i-k$ vertex-cut, because it contains i. Next, we show an example in which there are $\Theta(n^2)$ distinct optimal values of $s-t$ vertex-cuts.

Example 2.1. Let $G=(V_1\cup V_2, E)$ be an undirected graph such that $|V_1|=|V_2|$, and the vertex capacities are defined as follows: $c(v)>0\ \forall v\in V_2$ and $c(v)=0\ \forall v \in V_1$, such that for every $u,v,u',v' \in V_2$, $c(u)+c(v) \neq c(u')+c(v')$ unless $\{u,v\}=\{u',v'\}$. E is defined as follows: consider the edge set of a complete bipartite graph with sides V_1 and V_2, and delete from it the edge set of a perfect matching $\{(v,m(v))|v\in V_1, m(v)\in V_2\}$ to obtain E. In this example there are at least $(n/4)(n/2-1)$ distinct values of a minimum $s-t$ vertex-cut.

Proof. Consider $u,v \in V_1$. A minimum $u-v$ vertex-cut is $S_{u,v}=V\backslash\{u,v,m(u),m(v)\}$. This is so since for every $w \in V_2$ such that $w \neq m(u), m(v)$, w must be in the vertex-cut because otherwise, (u,w,v) is a path connecting u and v. Note that $S_{u,v}$ is a minimum cost set that contains all vertices $w \in V_2\backslash\{m(u),m(v)\}$. The cost of $S_{u,v}$ is $c_{u,v}=\sum_{w\in V_2\backslash\{m(u),m(v)\}} c(w)$. Therefore, for every $\{u,v\}\neq\{u',v'\}$, $c_{u,v}\neq c_{u',v'}$. Therefore, in this example there are at least $\binom{\frac{n}{2}}{2}$ distinct values of minimum $s-t$ vertex-cuts. $\square$

Benczúr [1] proved that a cut-tree does not exist for the all-pair minimum vertex-cut problem with additive costs. Example 2.1 strengthens this result, since a cut-tree is also a flow-tree, and a flow-tree does not exist by Example 2.1.

3. A flow-tree for general cost separations

Let c be a cost function defined for every subset $S \subseteq V$. For a pair of vertices $s, t \in V$, the minimum capacity $s-t$ separation is an $s-t$ separation S such that $c(S)$ is minimized. In this section we extend Granot and Hassin's [5] result, and construct a flow-tree that encodes the minimum cost separations with respect to c. We note that a construction similar to Benczúr's construction of a flow-tree for separations (where we do not use the exception rule), will result in a flow-tree with $2n$ vertices. However, we get a better result in this section. We first prove that such a flow-tree exists. Then, we construct it with exactly $n-1$ calls to a procedure that computes a minimum separation between a given pair of vertices.

Let (u_i, v_i) $i = 1, \ldots, \binom{n}{2}$ be a numbering of the pairs of vertices in G. Denote by $X = \{X_1, X_2, \ldots\}$ the set of all non-empty subsets of V. Let A be the $\binom{n}{2} \times |X|$ binary matrix such that $a_{ij} = 1$ if and only if X_j is a $u_i - v_i$ separation. A set of rows S of the matrix A is called *dependent* if there exists $S' \subseteq S$ such that $\sum_{i \in S'} a_{ij} = 0 \pmod 2$ $\forall j$, and otherwise S is *independent*. Let $r(A)$, the *rank* of A, be the maximum cardinality of an independent subset of the rows of A.

Theorem 3.1 (*Hassin [9]*). *The number of distinct solutions is at most r(A).*

Remark 3.2. The existence of a flow-tree does not follow from Theorem 3.1, even for additive costs.

Proof. To show that the existence of a flow-tree is not a consequence of Theorem 3.1, we present an example in which $r(A) = n$. This proves the claim, since for this example $r(A) = n$ and in a flow-tree the number of distinct solutions is at most $n - 1$.

Consider the graph $G = (V, E)$ such that $V = \{1, 2, 3\}$ and $E = \{(1, 2), (2, 3)\}$. There are three pairs of vertices $\{1, 2\}$, $\{2, 3\}$ and $\{1, 3\}$ that correspond to three rows of A. Consider the three columns of A that correspond to the three separations $\{1\}$, $\{2\}$, and $\{3\}$. The following is the sub-matrix of A that corresponds to these three separations:

$$\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}.$$

This sub-matrix is non-singular over GF_2, and therefore, $r(A) = 3 = n$. $\square$

For a vertex $v \in V$, denote by c_v the minimum cost of a set S_v that contains v. For $u, v \in V$ let $c_{u,v}$ be the minimum cost of a $u - v$ separation, and let $S_{u,v}$ be a $u - v$ separation such that $c(S_{u,v}) = c_{u,v}$. Let N be the network that consists of the complete graph with vertex set V, and capacity $c_{u,v}$ associated with an edge (u, v) of the graph.

The following theorem extends Theorem 1 in [5], its proof is very similar and it is given here for completeness.

Theorem 3.3. *For each simple cycle in N, the minimum of the edge capacities on the cycle is attained by at least two edges therein.*

Proof. The proof is by induction on the number $m \geqslant 3$ of edges in the cycle. If $m = 3$ and the claim is false, then there is a labeling i, j, k of the cycle's vertices for which $c_{i,j} \geqslant c_{j,k} > c_{i,k}$. Clearly, $j \notin S_{i,k}$, since otherwise $c_{i,k} \geqslant c_j \geqslant c_{i,j}$, and this is a contradiction. If $i \notin S_{i,k}$, let I be the connected component of the subgraph induced by $V \setminus S_{i,k}$ that contains i. Otherwise, let $I = \emptyset$. If $j \in I$, then $S_{i,k}$ is a $j - k$ separation, whence $c_{j,k} \leqslant c_{i,k}$; whereas if $j \notin I$, then $S_{i,k}$ is an $i - j$ separation, so $c_{i,j} \leqslant c_{i,k}$. Both cases lead to a contradiction and therefore, the claim holds for $m = 3$.

Suppose that the claim is true for all integers strictly less than m, $m > 3$, and consider m. Label the vertices of the cycle $1, 2, \ldots, m$ so that $c_{1,m} = \min\{c_{1,2}, c_{2,3}, \ldots, c_{m-1,m}, c_{1,m}\}$. By applying the induction hypothesis to the cycles $(1, 3, 4, \ldots, m)$ and $(1, 2, 3)$, we see that

$$c_{1,m} = \min\{c_{1,3}, c_{3,4}, \ldots, c_{m-1,m}\} \geqslant \min\{c_{1,2}, c_{2,3}, c_{3,4}, \ldots, c_{m-1,m}\}.$$

Therefore, the claim holds for m. $\square$

Corollary 3.4. *A maximum (total) capacity spanning tree* $T_N = (V, E_T)$ of N *is a flow-tree for the all-pair minimum separation problem.*

Proof. Consider i and j such that $(i, j) \notin E_T$. By Theorem 3.3 the cycle in N, that consists of the edge (i, j) and the path in T_N between i and j, satisfies that the minimum of the edge capacities on the cycle is attained by at least two edges therein. Therefore, $c_{i,j}$ is at least the minimum of the edge capacities along the $i - j$ path in T_N. Since (i, j) does not belong to a maximum spanning tree of N, we conclude that $c_{i,j}$ is at most the minimum of the edge capacities along the $i - j$ path in T_N. Hence (i, j) has a capacity equals to the minimum capacity of an edge along the path from i to j in T_N. Therefore, T_N is a flow-tree. $\square$

We present an algorithm that computes such a flow-tree and uses at most $n - 1$ computations of minimum cost $s - t$ separations. The construction is motivated by the one used by Cheng and Hu [2].

3.1. Ancestor tree algorithm

We construct an *ancestor tree* T. T is a binary tree, rooted at *root*. (The outdegree of every internal vertex of T is 2, except for the root that has just one outgoing edge.) Its internal vertices correspond to minimum cost separations. Its edges and leaves correspond to subsets of V. We do not distinguish between the vertices and edges of T and the separations and vertex sets to which they correspond. An edge corresponds to a subset of vertices that are disconnected from the rest of G by the separation at its tail vertex. A leaf corresponds to the intersection of the edges along the path from *root* to this leaf. Each leaf contains a vertex which is denoted as its *seed*. Let T_i be the ancestor tree obtained after $i - 1$ iterations. T_i has i leaves. T_1 consists of a single internal vertex *root* with cost $c_{\text{root}} = -\infty$, one out-going edge that corresponds to V, and one leaf. We arbitrarily set one vertex from V to be the seed of this leaf.

In each iteration we pick a leaf S corresponding to at least two vertices, and compute a minimum separation $S_{p,q}$ between the seed, $p \in S$, and an arbitrary vertex, $q \in S\backslash\{p\}$. If $p \notin S_{p,q}$, denote by P the set of vertices in the connected component that contains p in $V\backslash S_{p,q}$. Let $\hat{S}$ be the father of S in T_i.

While adding $S_{p,q}$ to T_i to form T_{i+1}, we change the structure of T_i. Consider the $root - \hat{S}$ directed path in T_i. Let x be the last vertex on this path such that $c_x \leqslant c_{p,q}$, and let y be its son (y corresponds to a partition or, if $x = \hat{S}$, the leaf S.)

To form T_{i+1} we replace the edge (x, y) from T_i by the path $(x, S_{p,q})$, $(S_{p,q}, y)$ in T_{i+1}. We also add an edge outgoing from $S_{p,q}$ to a new leaf with seed q. The edge $(x, S_{p,q})$ in T_{i+1} corresponds to the same subset as the edge (x, y) from T_i.

If $p \notin S_{p,q}$ then the edge $(S_{p,q}, y)$ in T_{i+1} corresponds to P and the edge to the new leaf corresponds to $V\backslash P$. Otherwise, $(S_{p,q}, y)$ corresponds to $\{p\}$ and the edge to the new leaf corresponds to $V\backslash\{p\}$.

Lemma 3.5. *If* $p \in S_{p,q}$ *then* $x = \hat{S}$.

Proof. Note that for creating $\hat{S}$ we have computed (earlier) an optimal separation between p and some other vertex t. $c_{p,t}$ is at most the cost of any set that contains p (because such a set is a feasible $p - t$ separation). Therefore $c(\hat{S}) = c_{p,t} \leqslant c_{p,q}$, and $x = \hat{S}$. $\square$

The construction uses $n - 1$ minimum separation computations. During the construction we note that for each separation S in T_i, the edges with tail S partition V, and therefore the leaves of T_i partition V.

Lemma 3.6. *The cost of an internal vertex of* T_i *is at most the cost of its non-leaf sons.*

Proof. For $i = 1$ the claim is trivial, and if we assume the property for T_i, then the claim for T_{i+1} follows by the assumption and our choice of x and y. $\square$

When we create the edge $(S_{p,q}, y)$ corresponding to P, a leaf S' in the subtree of T_i rooted at y is changed to $S' \cap P$. The first part of the following lemma shows that each of these seeds remains in its leaf.

Let $G^{(i)}_{\text{seeds}}$ be the graph over the seeds of T_i, with an edge between seeds s, s' if $S_{s,s'}$ is in T_i.

Lemma 3.7.

(i) *The seeds in T_i remain in the same leaves also in T_{i+1}.*
(ii) *For each separation $S_{r,s}$ in T_i, the edges with tail at $S_{r,s}$ are separated by $S_{r,s}$, and $S_{r,s}$ is the lowest common ancestor of the leaves containing r and S.*
(iii) *The subgraph of $G^{(i)}_{\text{seeds}}$ induced by the seeds contained in the leaves of a rooted subtree of T_i is connected.*

Proof. The claim trivially holds for $i = 1$.

We assume the property for T_i and prove it for T_{i+1}. If $x = \hat{S}$ then (i) trivially holds and (ii) and (iii) follow in a straightforward manner from the induction assumption and the change of the tree in this case. Therefore, by Lemma 3.5, we may assume that $p \notin S_{p,q}$.

(i) Assume by contradiction that $y \neq S$, and there is a seed S in the subtree induced by y, such that p and S are separated by $S_{p,q}$ (i.e., we assume that $s \notin P$). By the induction assumption, p and S are connected via some path of seeds in the induced subgraph of $G^{(i)}_{\text{seeds}}$ over the seeds contained in the subtree of T_i rooted at y. We argue that there is a pair of seeds v, v' in this path that are separated by $S_{p,q}$. This is so because $p \in P$ and $s \notin P$. Let v be the last vertex on this path (while traversing it from p to S) such that $v \in P$, and let v' be the next vertex. Then, v and v' are disconnected by $S_{p,q}$ and we computed the minimum separation $S_{v,v'}$.

By part (ii) of the induction assumption, v and v' belong to different sons of $S_{v,v'}$, and $S_{v,v'}$ is the lowest common ancestor of the leaves that contain v and v' in T_i (and hence the path between $S_{v,v'}$ to the leaf that contains v is disjoint to the path from $S_{v,v'}$ to the leaf that contains v'). Since $S_{v,v'}$ belongs to the subtree of T_i rooted at y, we conclude by Lemma 3.6 that $c_y \leqslant c_{v,v'}$. Since by the definition of y, $c_{p,q} < c_y$, we conclude that $c_{p,q} < c_{v,v'}$. Therefore, $S_{p,q}$ contradicts the optimality of $S_{v,v'}$.

(ii) By the inductive hypothesis the claim holds for T_i. By (i), the seeds of T_i are in the same leaves also in T_{i+1}, so that the claim holds for $S_{r,s} \neq S_{p,q}$. It also holds for $S_{p,q}$ because we add $S_{p,q}$ along the path in T_i from the root to S, $p, q \in S$, and p, q belong to the leaves in the subtree of T_{i+1} rooted at $S_{p,q}$.

(iii) Consider a rooted tree T' of T_{i+1}. If the new seed q is not in T', then T' is also a rooted subtree of T_{i-1}. By (i), the seed set of the leaves of T' does not change from T_i to T_{i+1}, and the claim follows by the induction assumption.

Suppose that T' includes q, then since q is in a son of $S_{p,q}$ in T_i, T' contains p. By the induction assumption, the seed set of T' besides q is connected, and we connect q to p. Therefore, the subgraph of $G^{(i+1)}_{\text{seeds}}$ induced by the seeds in T' is connected. □

Lemma 3.8. *Let u and w be seeds in T_i, and let $S_{x,y}$ be the lowest common ancestor of their leaves. Then, $c_{u,w} = c_{x,y}$.*

Proof. By part (iii) of Lemma 3.7, there is a sequence of seeds $(u=v_0, v_1, \ldots, v_k, v_{k+1}=w)$ such that $S_{u,v_1}, S_{v_1,v_2}, \ldots, S_{v_{k-1},v_k}, S_{v_k,w}$ are internal vertices in the subtree of T_i rooted at $S_{x,y}$. By Lemma 3.6, $c_{x,y} \leqslant \min\{c_{v_0,v_1}, c_{v_1,v_2}, \ldots, c_{v_{k-1},v_k}, c_{v_k,v_{k+1}}\}$. By part (ii) of Lemma 3.7, $S_{x,y}$ is a $u-w$ separation, and therefore there is $0 \leqslant l \leqslant k$ such that $S_{x,y}$ is a $v_l - v_{l+1}$ separation. Therefore, also $c_{x,y} \geqslant \min\{c_{u,v_1}, c_{v_1,v_2}, \ldots, c_{v_{k-1},v_k}, c_{v_k,w}\}$. Since $S_{x,y}$ is a $u-w$ separation, $c_{u,w} \leqslant c_{x,y}$. Consider the separation $S_{u,w}$. There is j such that $j = 0, 1, \ldots, k$ and $S_{u,w}$ is a $v_j - v_{j+1}$ separation. Therefore also $c_{u,w} \geqslant c_{x,y} = \min\{c_{u,v_1}, c_{v_1,v_2}, \ldots, c_{v_{k-1},v_k}, c_{v_k}, v_w\}$,and the claim follows. □

Theorem 3.9. *At the end of the construction each leaf corresponds to a single vertex, and the minimum cost separation between a pair of vertices of G is the separation at the lowest common ancestor of the two corresponding leaves in T.*

Proof. Since each leaf corresponds to the intersection of sets along the path from the root to the leaf, and the edges outgoing from each internal vertex partition V, we conclude that the leaves of T_i partition V. Therefore, each leaf of T_n corresponds to a single vertex. The claim follows by Lemma 3.8. □

Remark 3.10. Using Corollary 3.4, we can transform an ancestor tree (with n leaves) to a regular (with n vertices) flow-tree in $O(n^2)$ time by the following: first compute from the ancestor tree $c_{i,j}$ for every pair $i, j \in V$, and then compute a maximum spanning tree in the resulting network.

4. The number of minimum separations and vertex-cuts in directed graphs

In a directed graph $G=(V, E)$ and $i, j \in V$, an $i \to j$ *separation* is a subset $S \subseteq V$, such that the induced subgraph of G over $V \backslash S$ does not contain a directed path from i to j. Note that $S=\{i\}$ and $S=\{j\}$ are $i \to j$ separations. An $i \to j$ *vertex* $-$ *cut* is an $i \to j$ separation S such that $i, j \notin S$.

For edge-cuts in directed graphs, Jelinek [12] and Mayeda [13] presented an example with $\Theta(|V|^2)$ distinct minimum cut values. We note that the regular transformation from an edge-cut to a vertex-cut (i.e., replace each edge by a path of two edges with a new middle vertex, such that this vertex has the capacity of the original edge, and the capacities of the old vertices are ∞) increases the number of vertices in Jelinek's example to $\Theta(|V|^2)$. Therefore, the transformed example is not suitable for our purpose.

Theorem 4.1. *There are directed graphs with $\Theta(|V|^2)$ distinct values of minimum vertex-cuts.*

Proof. Replacing each undirected edge by two anti-parallel directed edges in Example 2.1, we obtain an instance with n vertices and $\Theta(|V|^2)$ distinct values of minimum vertex-cuts. $\square$

Theorem 4.2. *There are directed graphs with $\Theta(|V|^2)$ distinct values of minimum separations.*

Proof. Let k be an integer. Consider a directed graph $G=(V_1 \cup V_2 \cup V_3, E_{12} \cup E_{23})$ defined as follows:

$$V_i = \left\{v_1^i, v_2^i, \ldots, v_k^i\right\}, \quad i=1,2,3,$$
$$E_{12} = \{(v_p^1, v_q^2) | p \leqslant q\},$$
$$E_{23} = \{(v_p^2, v_q^3) | p \leqslant q\}.$$

Define the cost c as follows:

$$c(v_p^i) = \begin{cases} \infty & \text{if } i=1, \\ 2^p & \text{if } i=2, \\ \infty & \text{if } i=3. \end{cases}$$

The minimum cost $v_p^1 \to v_q^3$ separation for $p \leqslant q$ is $\{v_p^2, v_{p+1}^2, \ldots, v_q^2\}$ with cost $c_{p,q} = 2^{q+1} - 2^p$. Therefore, for every $p<q$, $p'<q'$ such that $(p,q) \neq (p',q')$, $c_{p,q} \neq c_{p',q'}$. Therefore, there are $\Theta(|V|^2)$ distinct values of the minimum separations in the graph. $\square$

Remark 4.3. The $\Theta(|V|^2)$ minimum separations in the proof of Theorem 4.2 are also minimum vertex cuts. Thus this example provides an alternative proof for Theorem 4.1.

5. Concluding remarks

We have shown that in undirected graphs the minimum separation values can be represented compactly in a flow-tree, whereas a compact representation of the minimum vertex-cut values is impossible. We have also shown that in directed graphs a compact representation of either vertex-cut values or separation values is impossible.

It is open whether there is (in a vertex-capacitated network) a compact representation (that uses $o(n^2)$ memory) of the minimum separations (and not only their values).

The minimum k-cut problem is to find a minimum weight edge-set that separates k specified vertices. Hassin [10] provided an algorithm that uses only $2\binom{n-1}{k-1}$ minimum k-cut computations to compute a compact representation of all the minimum k-cut solutions. Hartvigsen [8], improved this result, and provided an algorithm that uses only $\binom{n-1}{k-1}$ minimum k-cut computations in order to compute a compact representation of all the minimum k-cut solutions. Define a *k-cut separation* to be a vertex set S that separates k specified vertices and may contain some of the specified vertices. An open question is whether there is in a vertex-capacitated network a compact representation of the minimum k-cut separations.

References

[1] A.A. Benczúr, Counterexamples for directed and node capacitated cut-trees, SIAM J. Comput. 24 (1995) 505–510.

[2] C.K. Cheng, T.C. Hu, Ancestor trees for arbitrary multi-terminal cut functions, Ann. Oper. Res. 33 (1991) 199–213.

[3] O. Einstein, R. Hassin, The number of solutions sufficient for solving a family of problems, Math. Oper. Res. 30 (2005) 880–896.

[4] R.E. Gomory, T.C. Hu, Multi-terminal network flows, SIAM J. Appl. Math. 9 (1961) 551–570.

[5] F. Granot, R. Hassin, Multi-terminal maximum flows in node capacitated networks, Disc. Appl. Math. 13 (1986) 157–163.

[6] D. Gusfield, D. Naor, Efficient algorithms for generalized cut-trees, Networks 21 (1991) 505–520.

[7] D. Hartvigsen, Generalizing the all-pairs min cut problem, Discrete Math. 147 (1995) 151–169.

[8] D. Hartvigsen, Compact representations of cuts, SIAM J. Disc. Math. 14 (2001) 49–66.

[9] R. Hassin, Solution bases of multiterminal cut problems, Math. Oper. Res. 13 (1988) 535–542.

[10] R. Hassin, An algorithm for computing maximum solution bases, Oper. Res. Lett. 9 (1990) 315–318.

[11] R. Hassin, Multiterminal xcut problems, Ann. Oper. Res. 33 (1991) 215–225.

[12] F. Jelinek, On the maximum number of different entries in the terminal capacity matrix of oriented communication nets, IEEE Trans. Circuit Theory 10 (1963) 307–308.

[13] W. Mayeda, On oriented communication nets, IRE Trans. Circuit Theory CT-9 (1962) 261–267.