Approximation algorithms for finding a k-connected subgraph of a graph with minimum weight

Tamás Hajba

Submitted to HEJ. Manuscript no.: ANM-001130-A

This version is for internal use only!

Abstract

It is a frequent problem in network design to find a subgraph of a graph with minimum cost that satisfies certain connectivity requirements. We give a review of the different classes of the problem and the best known approximation algorithms.

1 Introduction

An important problem in network design is to design networks that are resilient to failures. Suppose we are given a graph $G = (V, E)$, where each edge e is associated with a cost $c(e)$. The goal is to find a spanning subgraph G' of G that satisfies certain connectivity requirements, at minimum cost of the edges used. Such a possible requirement is that the subgraph G' should be k edge-connected or k node-connected. A graph is k *edge-connected* if the deletion of any $k-1$ edges leaves it connected. A graph is k *node-connected*, if it has at least $k+1$ nodes and the deletion of any $k-1$ nodes leaves it connected. So a k edge-connected network continues to allow communication between functioning sites even after as many as $k-1$ links failed.

Unfortunately both the minimum weight k edge-connected subgraph problem and the minimum weight k node-connected subgraph problems are NP-complete. One possible way to get "good" feasible solution to an NP-complete problem is to use approximation algorithms. An α-*approximation algorithm* is a polynomial algorithm that always produces a solution whose value is at most α times the optimum value.

The paper organized as follows: in section 2 and 3 we investigate the minimum weight k edge-connected subgraph problem and the minimum weight k node-connected subgraph problem. In section 4 a generalization of the minimum weight k edge-connected subgraph problem is considered.

2 Edge-connectivity problems

Let $G = (V, E)$ be an undirected graph and for every $e \in E$ let $c(e) \geq 0$ be a nonnegative weight. Consider the problem of finding a minimum weight spanning subgraph $H = (V, E_H)$ that is k edge-connected. Khuller and Vishkin [**?**] gave a method that yields a 2 factor approximation algorithm for this problem. Their algorithm runs as follows: replace each edge

$e = (u, v)$ with two directed edges (u, v) and (v, u) with each edge having weight $c(e)$. Call this graph G^D. Then choose an arbitrary vertex r of G^D and find a minimum weight subgraph H^D of G^D that contains k edge-disjoint paths from r to every other vertex of G^D. As it was shown in [**?**] this subgraph can be found in polynomial time. If at least one of the directed edges (u, v) or (v, u) is picked in H^D , then add (u, v) to E_H.

Lemma 1 *The graph $H = (V, E_H)$ is k edge-connected.*

Proof: Suppose that we can remove from H $k - 1$ edges such that the remaining graph is not connected. Then there exists a vertex v that is in a different component than r. It is clear that r can not have k edge-disjoint paths to v in G^D. Thus, H is k edge-connected. ■

Theorem 1 *The weight of E_H is at most twice the weight of the optimal solution.*

Proof: Let G_{opt} be an optimal solution to the minimum weight k edge-connected problem. Consider all the anti-parallel edges corresponding to edges in G_{opt}. We get a directed subgraph in G^D with weight $2c(G_{opt})$. This subgraph also has the property that it contains k edge-disjoint paths from r to any vertex v. As the algorithm found the minimum weight subgraph satisfying this property it follows that $c(E_H) \leq 2c(G_{opt})$. ■

If all the weights are equal to one then there are better approximation algorithms. For this case Cheriyan and Thurimella [**?**] presented an algorithm that achieves a performance ratio $2 + 2/(k + 1)$.

3 Vertex connectivity problems

Let $G = (V, E)$ be an undirected graph and for every $e \in E$ let $c(e) \geq 0$ be a nonnegative weight. Consider the problem of finding a minimum weight spanning subgraph $H = (V, E_H)$ that is k node-connected. For the unweighted case the best known algorithm is due to Cheriyan and Thurimella [**?**]: their algorithm achieves a performance ratio of $1 + 1/k$. For the general problem no constant factor approximation algorithms are known. The best known algorithm is the algorithm of Ravi and Williamson [**?**] that achieves a factor of $2H(k)$ where $H(k) = 1 + 1/2 + \ldots, 1/k$. For the case of finding a 2 vertex-connected graph with minimum weight an approximation algorithm achieving ratio $2 + 1/n$ was given by Khuller and Raghavachari [**?**]. Their algorithm uses similar technique to the one used in the previous section. The idea is as follows: Create a new directed graph G^D as follows: replace each edge $e = (u, v)$ with two directed edges (u, v) and (v, u) with each edge having weight $c(e)$. Let $e = (x, y)$ be the lowest edge in G. Add a new vertex r to G^D and the directed edges (r, x) and (r, y) of weight 0 to G^D. Let H^D be a subgraph in G^D with minimum weight that contains 2 vertex-disjoint paths from r to each vertex v in G^D. Frank and Tardos [**?**] showed that H^D can be found in polynomial time. If at least one of the directed edges (u, v) or (v, u) is picked in H^D , then add (u, v) to E_H. It can be easily shown that the graph $H = (V, E_H \cup \{e\})$ is 2 node-connected.

Theorem 2 *The weight of $E_H \cup \{e\}$ is at most $(2 + \frac{1}{n})$ times the optimal solution.*

Proof: Let G_{opt} be a minimum weight 2 vertex-connected subgraph of G. Consider the anti-parallel edges corresponding to edges in G_{opt}. We get a directed subgraph in G^D with weight $2c(G_{opt})$. From x and y there are 2 vertex-disjoint directed paths to every vertex v and so there

are 2 vertex-disjoint directed paths from r to every other vertex v. Since the algorithm of Frank and Tardos finds the minimum weight subgraph of G^D having 2 vertex-disjoint paths from r to every vertex v we get that $c(E_H) \leq 2c(G_{opt})$. Since every 2 vertex-connected graph has at least n edges, the minimum weight edge of G is at most $\frac{1}{n}c(G_{opt})$. So $c(E_H \cup \{e\}) \leq 2 + \frac{1}{n}$. ∎

4 Generalized Steiner-problem

We are given a graph with non-negative weight function $c : E \to \mathbf{Q}_+$ on the edges. We are also given a function $f : 2^V \to \mathbf{N}$. Find a minimum weight set of edges $E' \subset E$ such that for every subset $S \subset V$ at least $f(S)$ edges of E' have exactly one endpoint in S.This problem can be formulated as an integer problem:

$$\begin{aligned} \min \sum_{e \in E} c_e x_e \\ x(\delta(S)) \geq f(S) \text{ for each } S \subset V, \\ x_e \in \{0,1\} \text{ for each } e \in E. \end{aligned} \tag{1}$$

A special case of this problem is the so called generalized Steiner problem: find a minimum weight subgraph of G that for every pair of nodes v and w contains at least $r(v,w)$ edge-disjoint paths from v to w. If $r \equiv k$, than the problem reduces to the minimum weight k edge-connected subgraph problem.
A function $f : 2^V \to \{0,1\}$ is called *uncrossable* if $f(V) = 0$ and if $f(A) = f(B) = 1$ for any two sets A and B, then either $f(A \cup B) = f(A \cap B) = 1$ or $f(A - B) = f(B - A) = 1$.
A function $f : 2^V \to \mathbf{N}$ is called *weakly supermodular* if $f(V) = 0$ and for any two sets A and B we have
$f(A) + f(B) \leq \max\{f(A - B) + f(B - A), f(A \cup B) + f(A \cap B)\}$.

For uncrossable functions f Williamson [**?**] et al. give a factor 2 approximation algorithm. The algorithm uses the primal-dual method for approximation algorithms. The dual of the linear programming relaxation of (1) is:

$$\begin{aligned} \max \sum_{\in V} h(S) y(S) \\ \sum_{S : e \in \delta(S)} y(S) \leq c(e) \text{ for each } e \in E \\ y(S) \geq 0 \text{ for each } S \subset V. \end{aligned} \tag{2}$$

Algorithm Uncrossable begins with the infeasible primal solution $F = \emptyset$ and the feasible dual solution $y_S = 0$ for all S. As long as there exists a violated set (that is a set S such that $h(S) = 1$ and $\delta_F(S) = 0$), the algorithm iteratively performs a primal-dual improvement step. In each iteration, the algorithm first identifies the minimal (with respect to inclusion) violated sets for F; we will call these sets active. Then the algorithm increases the value of the dual solution by uniformly raising the variables y_S corresponding to the active sets until some edge e becomes tight, i.e. ,$c(e) = \sum_{S : e \in \delta(S)} y(S)$. Edge e is then added to F. When F becomes feasible, the algorithm executes an edge-deletion stage, eliminating unnecessary edges from F. We denote the resulting set of edges by F'. Since at the end of the algorithm y is a feasible dual solution, the following theorem implies that algorithm uncrossable is a 2-approximation algorithm.

Theorem 3 *If F' and y are the set of edges and the dual variables, respectively, returned by the algorithm, then $\sum_{e \in F'} c(e) \leq 2 \sum_S y(S)$.* ■

Williamson et al. [**?**] provided a $2H(f_{max})$-approximation algorithm for the case when the function f is weakly supermodular. We call this algorithm weakly supermodular. The algorithm works in phases, and ensures that after phase p we have a set of edges F_p that satisfies $f_p(S) = f(S) - f_{max} + p$. Hence after f_{max} phases the set of edges $F_{f_{max}}$ is a feasible solution to the problem. To augment the set of edges F_{p-1} to F_p we have to add least one edge to each cut $\delta(S)$ for which $|\delta_{F_{p-1}}(S)| < f_p(S)$. Let $h_p(S) = max\{f_p(S) - |\delta_{F_{p-1}}(S), 0$. It can be prooved that the function h is uncrossable. So augmenting F_{p_1} to F_p we have to solve the IP program with function h_p. To solve this we can invoke the algorithm uncrossable defined above. Thus the algorithm consists of f_{max} invocation of the Uncrossable algorithm.

Theorem 4 *The algorithm weakly supermodular is a $2H(f_{max})$ approximation algorithm for any weakly supermodular function f, where $H(f_{max}) = 1 + 1/2 + \ldots + 1/f_{max}$.* ■

4.1 A 2-approximation algorithm

Recently Jain [**?**] presented a 2-approximation algorithm for the problem (1) with weakly supermodular functions. The relaxation of the problem is:

$$\begin{aligned} \min \sum_{e \in E} c_e x_e \\ x(\delta(S)) \geq f(S) \text{ for each } S \subset V, \\ 0 \leq x_e \leq 1 \text{ for each } e \in E. \end{aligned} \tag{3}$$

We assume that we are given an oracle that decides from a vector x whether x is a feasible solution of (3) or it finds a constraint that is not satisfied. The algorithm works as follows:
1. Solve the (3) LP program optimally (this can be done by the ellipsoid method).
2. Make an optimal extreme point solution from an optimal solution.
3. Fix those edges to 1 that have value at least 1/2 in the optimal extreme point solution.
4. Delete these edges and solve the remaining problem iteratively.

Theorem 5 (Jain) *In any extreme point solution of problem (3) there is at least one edge with $x_e \geq 1/2$.* ■

According to this theorem after at most $|E|$ iteration we get a feasible solution of (3).
Let X^* be an extreme point solution of problem (3) and let $E_{\frac{1}{2}+}$ be the edges with $X^*(e) \geq 1/2$. Let $G_{res} = G - E_{\frac{1}{2}+}$. After fixing the values of the edges of $E_{\frac{1}{2}+}$ to 1 we have to solve the following modified problem:

$$\begin{aligned} \min \sum_{e \in E(G_{res})} c_e x_e \\ x(\delta_{G_{res}}(S)) \geq f(S) - \sum_{e \in E_{\frac{1}{2}+} \cap \delta_{G(S)}} 1 \text{ for each } S \subset V, \\ 0 \leq x_e \leq 1 \text{ for each } e \in E(G_{res}). \end{aligned} \tag{4}$$

The function in problem (4) is weakly supermodular so we can do the iteration again.

Theorem 6 *Let z^* and z^*_{res} be the optimal values of (3) and (4) respectively. If E_{res} is a solution of problem (4) with weight at most $2z^*_{res}$ then $E_{res} \cup E_{\frac{1}{2}+}$ is a feasible solution of (3)with weight at most $2z^*$.* ■

The theorem shows that the final solution is at most twice the optimum so the algorithm is a 2-approximation.

References

[1] S. Khuller and R. Thurimella. Biconnectivity approximations and graph carvings. *Journal of the ACM*, 41(2): 214-235, 1994.

[2] J. Edmomds. Matroid intersection. *Annals of the Discrete Mathematics*, 4:185-204, 1979.

[3] J. Cheriyan and R. Thurimella. Approximating Minimum-Size k-Connected Spanning subgraphs via Matching. *37th Annual Symposium on Foundations of Computer Science*, 292-301, 1996

[4] R. Ravi, D. Williamson. An approximation algorithm for minimum-cost vertex-connectivity problems. *Proc. 6th Annual ACM-SIAM Symposium on Discrete Algorithms*, 332-341, 1995.

[5] S. Khuller and B. Raghavachari. Improved approximation algorithms for uniform connectivity problems. *Journal of Algorithms* 21, 433-450 (1996)

[6] A. Frank and E. Tardos. An application of submodular flows. *Linear Algebra an its Applications*, 114/115:320=348, 1989.

[7] M. X. Goemans, D. P. Williamson. A general approximation technique For constrained forest problems. *Proc. 3rd Annual ACM-SIAM Symp. on Discrete Algorithms*, 307-316, 1992.

[8] M. Goemans, A. Goldberg, S. Plotkin, D. Shmoys, E. Tardos, D. Williamson. Improved approximation algorithms for network design problems. *Proc. of the 5th SODA*, 223-232 1994.

[9] K. Jain. A factor 2 approximation algorithm for the generalized Steiner network problem.