Carnegie Mellon University
Research Showcase

Computer Science Department | School of Computer Science

1-1-2003

Improved Results for Directed Multicut

Anupam Gupta
Bell Labs, anupamg@cs.cmu.edu

Follow this and additional works at: http://repository.cmu.edu/compsci

Recommended Citation
Gupta, Anupam, "Improved Results for Directed Multicut" (2003). *Computer Science Department.* Paper 847.
http://repository.cmu.edu/compsci/847

This Conference Proceeding is brought to you for free and open access by the School of Computer Science at Research Showcase. It has been accepted for inclusion in Computer Science Department by an authorized administrator of Research Showcase. For more information, please contact research-showcase@andrew.cmu.edu.

Improved Results for Directed Multicut

Anupam Gupta
Lucent Bell Labs
600 Mountain Avenue
Murray Hill NJ 07974
anupamg@research.bell-labs.com

Abstract

We give a simple algorithm for the MINIMUM DIRECTED MULTICUT problem, and show that it gives an $O(\sqrt{n})$-approximation. This improves on the previous approximation guarantee of $O(\sqrt{n \log k})$ of Cheriyan, Karloff and Rabani [1], which was obtained by a more sophisticated algorithm.

1 Introduction

Assume we are given a directed network $G = (V, A)$ with positive edge capacities $u_e : A \to \mathbb{Z}_{\geq 0}$, and with k source-sink pairs $\{(s_i, t_i)\}_{i=1}^k$, with $s_i, t_i \in V$ for all i. A *directed multicut* is a set of arcs $M \subseteq A$ such that for any (directed) path P from some s_i to its corresponding t_i, $P \cap M \neq \phi$. The MINIMUM DIRECTED MULTICUT problem is to find the multicut $M \subseteq A$ with the least total capacity $u(M)$, where $u(M) = \sum_{e \in M} u_e$.

This problem, being an important tool for designing divide-and-conquer algorithms for NP-hard problems, has a long and illustrious history. The undirected case is better understood: we point the interested reader to the survey by Shmoys [4] for many details and references. However, the directed variant of the problem appears to be much harder, and is NP-hard even for $k = 2$ [2], a case that can be solved efficiently for the undirected variant [3].

The first non-trivial approximation algorithm for directed multicut, an $O(\sqrt{n \log n})$ approximation algorithm, was given by Cheriyan et al. [1]. Central to their result is an algorithm which, given a network with $u_e \geq 1$ for all $e \in A$, outputs a multicut M with capacity $O(F^2 \log n)$, where F is the maximum multiflow in G with terminals $\{(s_i, t_i)\}_i$ (defined in the next section).

They also gave a much simpler algorithm which outputs a cut of capacity at most $O(F^3)$. In this note, we show that a variant of this latter algorithm gives us the following results.

THEOREM 1.1. *Given a directed multicommodity flow network G_0 with $u_e \geq 1$ for all $e \in A$, we can efficiently find a multicut M with $c(M) = O(F^2)$, where F is the maximum multiflow in G with terminals $\{(s_i, t_i)\}_i$.*

THEOREM 1.2. *We can efficiently find a directed multicut with cost within $O(\sqrt{n})$ of optimal.*

The proofs of these theorems, along with the algorithms to effectively find these cuts, are given in the following two sections.

2 Relating Cuts and Flows

Note that the following integer linear program is a reformulation of the minimum multicut problem.

$$\text{(IP1)} \qquad \begin{aligned} \min \textstyle\sum_e u_e x_e & \\ \text{s.t.} \quad x(P) \geq 1 & \quad \forall s_i\text{-}t_i \text{ paths } P, \forall i \\ x_e \in \{0,1\} & \end{aligned}$$

Relaxing the integrality constraints to $x_e \geq 0$ gives us a linear program (LP1) that can be solved in polynomial time. We interpret the variable x_e as the "length" of an arc e, and $\sum_{e \in S} u_e x_e$ to be the "volume" of a set of arcs S.

It is easily seen that the linear programming dual of (LP1) is the following, which is a formulation of the so-called MAXIMUM MULTIFLOW problem on G with terminals $\{(s_i, t_i)\}_i$.

$$\text{(LP2)} \qquad \begin{aligned} \max \textstyle\sum_P f(P) & \\ \text{s.t.} \quad \textstyle\sum_{P : P \ni e} f(P) \leq u_e & \quad \forall e \in A \\ f(P) \geq 0 & \end{aligned}$$

Let F be the optimal value of (LP2), and hence value of the maximum multiflow in G with terminals $\{(s_i, t_i)\}_i$. By linear programming duality, the minimum multicut has value at least F; we now proceed to find a cut of value $O(F^2)$.

Algorithm I: The algorithm maintains a current graph G, initially the input graph G_0. As long as there is a source-sink pair such that G has a directed path from s_i to t_i, we find a good cut separating s_i from t_i as described below, remove these edges to get the new G, and continue.

To find the cut, we look at the subnetwork $H_i = G[s_i, t_i]$, where $G[x, y]$ denotes the subgraph of G induced by edges e which lie on some directed path from x to y.

Since x is a solution to (LP1) and H_i is a subnetwork of G_0, the distance from s_i to t_i in H_i is at least 1. Let $F_i = \sum_{e \in H_i} u_e x_e$. Let us look at level-cuts in H_i, i.e., cuts that are obtained by deleting all points (i.e., all edges that these points lie on) in H_i at some distance r from s_i. Furthermore, we restrict our attention to those cuts with $r \in [\frac{1}{3}, \frac{2}{3}]$, i.e., those "far" from both s_i and t_i, and find the smallest such cut C_i. A simple averaging argument shows that this cut in H_i has capacity at most $F_i/(\frac{2}{3} - \frac{1}{3}) = 3F_i$.

To finish, we must show that the sum of the cuts in the various stages does not exceed $O(F^2)$. For the rest of the discussion, we assume that all edges have capacity $u_e = 1$. This assumption can be discharged by replacing every edge e with $\lceil u_e \rceil$ parallel edges, which changes F by at most a factor of 2; furthermore, this assumption is only for simplicity – the proof can be done without this assumption.

Proof of Theorem 1.1: Let us associate two counters, the left counter $\mathsf{A}_l(e)$, and the right counter $\mathsf{A}_r(e)$, with each edge e in the graph, both initially set to 0. We also define a potential function $\Phi = \sum_e x(e)(\mathsf{A}_l(e) + \mathsf{A}_r(e))$. When making a cut in some H_i, we increment counters for all the edges in H_i (and no other edges) thus: If an edge $e \in H_i$ lies on the left of the cut, we increment $\mathsf{A}_l(e)$; if it lies to the right, we increment $\mathsf{A}_r(e)$. (In the event that the edge itself is cut, we can increment either of the counters.) Since the cut value is $O(F_i)$, and $\sum_{e \in H_i} x(e) = F_i$, the value of Φ goes up by exactly F_i. Hence it suffices to show that the final value of Φ is $O(F^2)$.

For this, we show that both $\mathsf{A}_l(e), \mathsf{A}_r(e) \leq O(F)$, i.e., an edge can lie in some H_i only $O(F)$ times. We will show this for A_l; the proof for A_r is identical. Consider an iteration when e lies in H_i and $\mathsf{A}_l(e)$ is incremented. The definition of H_i ensures that e lies on some s_i-t_i path. Let this path $P_i(e)$ be called the *witness* for e in C_i, and let $Q_i(e)$ be those edges in $P_i(e)$ that lie in or to the right of the cut C_i. Note that the fact that the cut C_i is at distance at most $\frac{2}{3}$ from s_i implies that the edges on $Q_i(e)$ have $\sum_{e' \in Q_i(e)} x_{e'} \geq \frac{1}{3}$.

Let us consider a subsequent cut C_j where $\mathsf{A}_l(e)$ is incremented, and look at the corresponding $Q_j(e)$, the portion of the witness path $P_j(e)$ for e in C_j lying in or to the right of C_j. We claim that $Q_i(e)$ and $Q_j(e)$ cannot share any edges. Indeed, if e' is an edge in $Q_i(e) \cup Q_j(e)$, then there exists a path from e to e' *after* C_i has been deleted, and hence a path between s_i and t_i. But this contradicts the fact that C_i is an s_i-t_i cut, and proves our claim. Hence, for every cut C_i, the edges in $Q_i(e)$ are disjoint. Furthermore, $x(Q_i(e)) \geq \frac{1}{3}$ for all i, and $\sum x(Q_i(e)) \leq F$, the sum taken over all i where $\mathsf{A}_l(e)$ is incremented. Thus $\mathsf{A}_l(e) \leq F/\frac{1}{3} = 3F$. A similar argument shows $\mathsf{A}_r(e) \leq 3F$, and hence $\Phi \leq 6F^2$, proving the theorem. ■

3 An approximation algorithm

Since we do not have any restrictions on the capacities of edges in Theorem 1.2, the algorithm is slightly different:

Algorithm II: Consider all edges with $x_e \geq 1/\sqrt{n}$, and cut them (which corresponds to raising x_e to 1). Now run the previous algorithm on the remaining graph to detach the remaining terminal pairs.

THEOREM 3.1. *The cut found by the above algorithm is within $O(\sqrt{n})$ of optimum.*

Proof. The cost of the edges cut in the first step is at most $F\sqrt{n}$, since each cut edge has x_e raised from $\geq 1/\sqrt{n}$ to 1.

Let us now bound the capacity of the edges cut in the second step. We use three simple facts. The first fact extends one used before: for each iteration i where $\mathsf{A}_l(e)$ is incremented, the length of $Q_i(e)$ in length at least $\frac{1}{3}$. Since all edges surviving the first step have length less than $1/\sqrt{n}$, there must be at least $\frac{1}{3}\sqrt{n}$ edges on $Q_i(e)$.

Secondly, let $h(P)$ be the set of vertices at the heads of edges in a directed path P. Hence there are at least $\frac{1}{3}\sqrt{n}$ vertices in each $h(Q_i(e))$.

Finally, for any subsequent cut C_j where $\mathsf{A}_l(e)$ is raised, $h(Q_j(e)) \cap h(Q_i(e)) = \phi$. Indeed, if there is a vertex v in the intersection, then there would be a path from e to v that survived the deletion of C_i, giving a contradiction. Hence the sets $h(Q_i(e))$ are disjoint for all iterations i where $\mathsf{A}_l(e)$ is incremented, and since each such set has at least $\frac{1}{3}\sqrt{n}$ vertices, $\mathsf{A}_l(e) \leq 3\sqrt{n}$. Similarly, $\mathsf{A}_r(e) \leq 3\sqrt{n}$, and thus Φ and the total cut capacity by $O(F\sqrt{n})$.

References

[1] Joseph Cheriyan, Howard Karloff, and Yuval Rabani. Approximating directed multicuts. In *Proceedings of the 42nd Annual IEEE Symposium on Foundations of Computer Science*, pages 320–328, 2001.

[2] Naveen Garg, Vijay V. Vazirani, and Mihalis Yannakakis. Multiway cuts in directed and node-weighted graphs. In *Proceedings of the 21st International Colloquium on Automata, Languages and Programming*, volume 820 of *Lecture Notes in Computer Science*, pages 487–498, 1994.

[3] T. C. Hu. *Combinatorial algorithms*. Addison-Wesley Publishing Company Advanced Book Program, Reading, MA, 1982.

[4] David B. Shmoys. Cut problems and their application to divide-and-conquer. In Dorit S. Hochbaum, editor, *Approximation Algorithms for NP-hard Problems*, pages 192–235. PWS Publishing, 1997.