Approximating Multicast Congestion*

Santosh Vempala[1] and Berthold Vöcking[2]

[1] Dept. of Mathematics, MIT
Email: vempala@math.mit.edu
[2] International Computer Science Institute
Email: voecking@icsi.berkeley.edu

Abstract. We present a randomized algorithm for approximating multicast congestion (a generalization of path congestion) to within $O(\log n)$ times the best possible. Our main tools are a linear programming relaxation and iterated randomized rounding.

1 The Problem

Let $G = (V, E)$ represent a communication network with n vertices. A *multicast* request is specified as a subset of vertices, called terminals, that should be connected. In the multicast congestion problem, we are given several multicast requests, $S_1, \ldots, S_m \subseteq V$. A solution to the problem is a set of m trees, where the i^{th} tree spans the terminals of the i^{th} multicast request. The *congestion* of an edge in a solution is the number of multicast trees that use the edge. The problem is to find a set of multicast trees that minimize the maximum congestion (over all the edges).

The multicast congestion problem generalizes two well-known optimization problems. The special case where each request consists of just two terminals is the standard routing problem of finding *integral paths with minimum congestion.* The latter problem (which itself is a generalization of the problem of finding edge disjoint paths) is in Karp's original list of NP-hard combinatorial problems [5], and hence our general problem is also NP-hard. On the other hand, integral path congestion can be approximated within a factor of $O(\log n)$ [6]. Our main result is an algorithm for approximating the multicast congestion to within a factor

* Santosh Vempala was supported by a Miller fellowship at U.C.Berkeley and an NSF CAREER award. Berthold Vöcking was supported by a grant of the "Gemeinsames Hochschulsonderprogramm III von Bund und Ländern" through the DAAD.

of $O(\log n)$, thus placing it in the same level of approximability as the special case.

The second problem that multicast congestion generalizes is the minimum Steiner tree problem which is the special case of a single multicast with a slightly different objective function. The goal is to minimize the sum rather than the maximum of the congestion over all edges. Finding a minimum Steiner tree is max-SNP hard, i.e. there is a constant $\epsilon > 0$ such that it is NP-hard to find a $(1 + \epsilon)$ approximation [1] (and thus our problem is also max-SNP hard). The precise constant to which the Steiner tree can be approximated is an open problem. The best known upper bound is 1.598 [3].

Our algorithm relies on a linear programming relaxation of the problem. The relaxation is motivated by the following observation: in any integral solution to the problem, there is at least one edge across any cut that separates two terminals belonging to the same multicast request. This can be viewed as a multicommodity flow for each multicast, with the objective of minimizing the net congestion. The LP formulation is described in detail in section 2.

A solution to the LP can be viewed as a separate fractional solution for each multicast. Unfortunately, the fractional solution cannot be decomposed into a set of trees of fractional weights, which would allow to adapt the simple rounding algorithm used for the standard routing problem [6], i.e., for each multicast, choose one of the fractional trees at random with respect to the fractional weights of the tree. Instead we decompose the fractional solution for each multicast into a set of paths. These paths have the property that the total flow from each multicast terminal to other terminals in the same multicast is at least 1. We apply a variant of randomized rounding to these paths. A single iteration of randomized rounding, however, does not suffice, and we need to iterate randomized rounding on a subproblem till we find a solution. The rounding algorithm is presented in section 3.

To prove the approximation guarantee, we make two observations. The first is that in each phase (i.e. one iteration) the expected congestion is no more than twice the fractional congestion (i.e. the congestion of the LP). Second, the total number of phases is at most $\log k$, where k is the maximum number of terminals in a single multicast, which is, of course, bounded by n. Since the fractional congestion in each phase is a lower bound on the integral congestion, the expected congestion of an edge over the entire algorithm is at most $O(\log k)$ times the optimal congestion. Further, the congestion is concentrated around this expectation and a straightforward application of Chernoff bounds leads to an $O(\log n)$ approximation (the details are in section 4).

2 The Relaxation

Let x_{te} be a 0-1 variable indicating whether edge e is chosen for the t^{th} multicast. Consider the following integer program (IP).

$$
\begin{aligned}
&\min z \\
\sum_{e\in\delta(S)} x_{te} &\geq 1 \quad \forall t, \forall S \subset V \text{ s.t. } S \cap S_t \neq \phi, (V \setminus S) \cap S_t \neq \phi \qquad (1)\\
\sum_t x_{te} &\leq z \quad \forall e \in E \\
x_{te} &\in \{0,1\} \quad \forall t, \forall e \in E \qquad (2)
\end{aligned}
$$

The constraints (1) ensure that any solution to the integer program connects the vertices in each multicast. Thus, this program corresponds exactly to our problem. An LP relaxation is obtained by relaxing the constraints (2) to simply $0 \leq x_{te} \leq 1$. Let us refer to this relaxation as LP1. Although the program has exponentially many constraints, it can be solved in polynomial (in n) time by designing a separation oracle ([2]) that checks whether the flow across every relevant cut is at least 1.

In the case of the standard routing problem, i.e., two terminals in each multicast, a fractional solution of LP1 can be decomposed into several paths of fractional weight so that the sum of the weights of the fractional paths for each multicast is one, and the sum of the weights of the paths crossing an edge e corresponds to the weight of that edge, i.e., $\sum_t x_{te}$. Therefore, choosing one of the fractional paths for each multicast at random (with respect to the weights) results in an integral solution in which the expected number of paths that cross an edge corresponds to the weight of that edge. Unfortunately, in the case of more than two terminals, the fractional solution cannot be be decomposed into a collection of fractional trees that add up to $\sum_t x_{te}$, for each edge e. For this reason, it is unclear how to obtain an integral solution of bounded cost from LP1.

Instead we consider a different relaxation LP2 which is at least as strong as the one above (i.e. the feasible region of the second relaxation is contained in the feasible region of the first relaxation). The relaxation is described in terms of a multicommodity flow between pairs of terminals of each multicast. In the relaxation, the variable $f_t(i,j)$ denotes the flow between the terminals i and j in the t^{th} multicast, and $x_{te}(i,j)$ denotes the flow on edge e of commodity (i,j) in the t^{th} multicast.

$$\min z$$

$$x_{te}(i,j) \text{ is a flow of value } f_t(i,j) \quad \forall t, \forall i < j \in S_t \tag{3}$$

$$\sum_{i \in S \cap S_t, j \in (V \setminus S) \cap S_t} f_t(i,j) \geq 1 \quad \forall t, \forall S \subset V \tag{4}$$

$$\sum_{t,i,j} x_{te}(i,j) \leq z \quad \forall e \in E$$

$$0 \leq x_{te}(i,j) \leq 1 \tag{5}$$

Theorem 1. $\min LP1 \leq \min LP2 \leq \min IP$

Proof. Any solution to the multicast congestion problem (IP) can be mapped to a solution of LP2 with same objective value as follows. For each multicast, connect all nodes in the integral tree by a cycle so that each edge in the tree is included in the cycle exactly twice. Each pair of terminals that are neighbors in the cycle (i.e., terminals that are connected by a subpath of the cycle that does not include any other terminal) exchange a flow of weight $\frac{1}{2}$. Figure 1 gives an example of this transformation. This construction defines a multicommodity flow along the edges of the tree so that the total flow along each edge is 1 and the total flow across any cut in the graph that separates two or more terminals is at least 1. Hence, the constraints of LP2 can be satisfied while preserving the objective value.

Any solution of LP2 can be mapped to a solution of LP1 with the same objective value by simply adding up all the flows on an edge, separately for each multicast. □

Theorem 2. *The relaxation LP2 can be solved in polynomial time.*

Proof. We design a separation oracle for LP2. Given an assignment to the variables of LP2, for each multicast S_t, we can construct a complete graph $G_t = (S_t, E_t)$ where the weight of an edge (i,j) is $f_t(i,j)$. Then constraints (4) simply require that every cut in this graph has weight at least 1. This can be easily checked by finding the minimum cut in the graph. The other constraint sets can be checked in polynomial time by examination. □

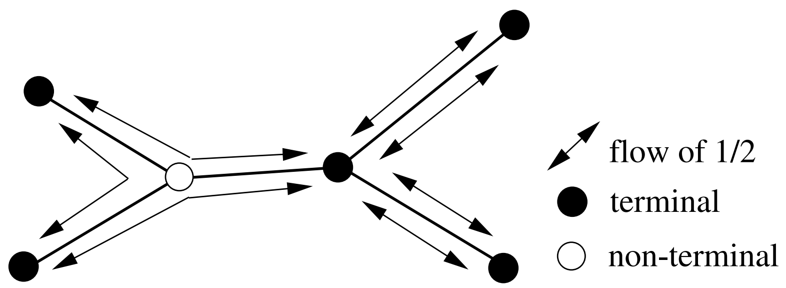

Fig. 1. Transformation of an IP solution to a solution of LP2.

3 Iterated Randomized Rounding

A solution to LP2 can be separated into fractional solutions for each multicast. We will now describe how to round these fractional solutions into trees, one for each multicast. The rounding is completely independent for each multicast and is achieved by the following procedure applied separately to the fractional solution for each multicast (i.e. the values of $x_{te}(i,j)$, separately for each t).

1. Decompose the fractional solution into flow paths.
2. Choose one path randomly out of each multicast terminal with probability equal to the value of the flow on the path (randomized rounding).
3. If the multicast terminals are all connected, then stop and output the solution.
4. Otherwise, contract the vertices corresponding to connected components, and form a new multicast problem with the contracted vertices as the new multicast terminals.
5. The original fractional solution is still a solution for the new multicast problem. Repeat the above steps till all multicast terminals are connected.

Lemma 1. *There are at most* $\log k$ *iterations, where* k *is the maximum number of terminals in a multicast.*

Proof. In each iteration, each multicast terminal is connected to at least one other terminal. Thus, the number of components drops by at least a factor of two in each iteration. $\square$

4 The Approximation Guarantee

Let X_{te}^{ℓ} be the random variable indicating whether an active edge e is selected for the t^{th} multicast in the ℓ^{th} phase of the rounding. Define $x_{te} = \sum_{i,j} x_{te}(i,j)$, i.e., the total flow through edge e of the t^{th} multicast.

Lemma 2. $E[X_{te}^{i}] \leq 2 \cdot x_{te}$.

Proof. The flow through the edge e for each commodity (i,j) is divided among a set of paths. The random variable X_{te}^{ℓ} is 1 if one of these paths is selected and zero otherwise. The expected value of X_{te}^{ℓ} is the sum of the probabilities of these paths being selected. Since the probability of a path being selected by any fixed one of its two end points is exactly the flow on it, the sum of the probabilities of all paths through e (for the t^{th} multicast) is $2 \cdot x_{te}$. □

Theorem 3. *With high probability, the congestion of the solution found by the algorithm is less than $O(\log k \cdot OPT + \log n)$.*

Proof. Fix an edge e. Let $L(e)$ denote the number of multicasts including e. By definition, $L(e) = \sum_t \sum_\ell X_{te}^{\ell}$. From Lemma 2, we can conclude that $E[L(e)] \leq 2 \cdot x_{te} \leq 2 \cdot \log kOPT$. We will apply a Chernoff bound to show that it is unlikely that $L(e)$ deviates significantly from its expectation.

For each multicast t, the random variables X_{te}^{ℓ} are independent from random variables for other multicasts. Random variables for the same multicast, however, may be dependent, but these variables are negatively correlated, because an edge e that is chosen in round ℓ for a multicast will not be chosen in round $\ell' > \ell$ for the same multicast. As a consequence, we can apply a Chernoff bound (see e.g. [4]). Set $C \geq \max\{2eE[L(e)], 2 \cdot (\alpha + 2) \cdot \log n\}$ with $\alpha > 0$ denoting an arbitrary constant. Then

$$\text{Prob}[L(e) \geq C] \;\leq\; 2^{-C/2} \;\leq\; n^{-\alpha-2} \;.$$

Summing this probability over all edges yields that the congestion does not exceed $C = O(\log k \cdot OPT + \log n)$, with probability $1 - n^{-\alpha}$. □

5 Open questions

An obvious open question is whether the approximation factor can be improved. It is worth noting that path congestion is conjectured to be approximable to within a factor of 2. In fact, the worst known gap for the

LP relaxation is a factor of 2. However, the best known approximation even for that special case is $O(\log n)$.

The path congestion problem can actually be approximated to within a constant times the optimum *plus* $O(\log n)$ (which is also a multiplicative $O(\log n)$ when the congestion is a constant, but better nevertheless). A less ambitious open problem is whether the tree congestion can also be approximated to within a constant plus $O(\log n)$.

References

1. M. Bern, and P. Plassmann, *The Steiner problem with edge lengths 1 and 2.* Information Processing Letters 32:171-176, 1989.
2. M. Grötschel, L. Lovász and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization.* Springer-Verlag, 1987.
3. S. Hougardy and H. J. Promel, *A 1.598 approximation algorithm for the steiner problem in graphs.* Proc. 10th Ann. ACM-SIAM Symp. on Discrete Algorithms, ACM-SIAM, 448-453, 1999.
4. T. Hagerup and C. Rüb, *A guided tour of Chernoff bounds.* Information Processing Letters 33(6):305-308, 1990.
5. R. M. Karp, *Reducibility among combinatorial problems.* Complexity of Computer Computations, R. E. Miller, J. W. Thatcher, Eds., Plenum Press, New York, 85-103, 1972.
6. P. Raghavan and C.D. Thompson *Randomized rounding: a technique for provably good algorithms and algorithmic proofs.* Combinatorica, 7(4):365-374, 1987.