# Approximate max-flow min-(multi)cut theorems and their applications

*Naveen Garg*
*Vijay V. Vazirani*
Department of Computer Science and Engg.
Indian Institute of Technology, Delhi

*Mihalis Yannakakis*
AT & T Bell Laboratories, Murray Hill, NJ 07974

**Abstract**

Consider the multicommodity flow problem in which the object is to maximize the sum of commodities routed. We prove the following approximate max-flow min-multicut theorem:

$$\frac{\text{min multicut}}{O(\log k)} \leq \text{max flow} \leq \text{min multicut},$$

where $k$ is the number of commodities. Our proof is constructive; it enables us to find a multicut within $O(\log k)$ of the max flow (and hence also the optimal multicut). In addition, the proof technique provides a unified framework in which one can also analyse the case of flows with specified demands, of Leighton-Rao and Klein et.al., and thereby obtain an improved bound for the latter problem.

## 1 Introduction

Much of flow theory, and the theory of cuts in graphs, is built around a single theorem - the celebrated max-flow min-cut theorem of Ford and Fulkerson [FF], and Elias, Feinstein and Shannon [EFS]. The power of this theorem lies in that it relates two fundamental graph-theoretic entities via the potent mechanism of a min-max relation.

The importance of this theorem has led researchers to seek its generalization to the case of *multicommodity flow*. In this setting, each commodity has its own source and sink, and the object is to maximize the sum of the flows subject to capacity and flow conservation requirements. The notion of a *multicut* generalizes that of a cut, and is defined as a set of edges whose removal disconnects each source from its corresponding sink. Clearly, maximum multicommodity flow is bounded by minimum multicut; the question is whether equality holds. This can be established for some special cases, e.g. if there are only two commodities [Hu]; however, one can construct very simple examples to show that equality does not hold in general. Consider a tree of height one with three leaves. Each pair of leaf vertices form the source-sink pair for a commodity. All edges have unit capacities. The max flow in this graph is $\frac{3}{2}$, whereas minimum multicut is 2.

Why does the theorem hold for a single commodity, and why does the generalization fail? To seek an explanation, let us consider the LP formulation of the maximum multicommodity flow problem. As shown in Section 5, the dual of this is the LP relaxation of the minimum multicut problem, i.e. the optimal *integral* solution to the dual is the minimum multicut. In general, the vertices of the dual polyhedron are not integral. However, for the case of a single commodity, they are integral (see [GV] for an exact characterization), and the max-flow min-cut theorem is simply a consequence of the LP duality theorem. For the multicommodity case, the LP duality theorem shows only that maximum flow is equal to the minimum fractional (i.e. relaxed) multicut.

In this situation, the best one can hope for is an approximate max-flow min-cut theorem. In ground-breaking work, Leighton and Rao [LR] gave the first such theorem. Let us consider a second formulation of the multicommodity flow problem that has also been studied widely in the past. In this formulation, a demand, $D_i$, is specified for each commodity, $i$. The object is to determine the maximum number, $f$, such that $fD_i$ amount of each commodity $i$ can be routed simultaneously, subject to capacity and conservation constraints. (Equivalently, the object is to determine the minimum number, $u$, such that if the capacity of each edge is multiplied by $u$, then all the demands can be simultaneously satisfied. Clearly, at optimality $f = \frac{1}{u}$.) The analogue of a minimum cut in this case is a sparsest cut: one that minimizes the ratio of capacity of the cut to the demand across the cut. Let $\alpha$ be this minimum. Clearly, $f \leq \alpha$, and once again equality does not hold in general.

Leighton and Rao considered a special case of the above-stated formulation, called uniform multicommodity flow, in which there is a commodity corresponding to each pair of vertices, and all the demands are unity. They proved the following approximate max-flow min-cut theorem:

$$\frac{\alpha}{O(\log n)} \leq f \leq \alpha,$$

where $n$ is the number of vertices in the graph. Subsequently, Klein, Agrawal, Ravi and Rao [KARR] managed to attack the arbitrary demands problem, and proved:

$$\frac{\alpha}{O(\log C \log D)} \leq f \leq \alpha,$$

where $C$ is the sum of capacities of all edges and $D$ is the sum of all demands. However, one restriction they impose is that all capacities and demands be integral. Both papers also give polynomial time algorithms for finding an approximation to the sparsest cut; the factors being $O(\log n)$ and $O(\log C \log D)$ respectively.

In this paper, we address the first version of the multicommodity flow problem, henceforth referred to as the *maximum multicommodity flow* problem, and prove the following approximate max-flow min-multicut theorem:

$$\frac{M}{O(\log k)} \leq f \leq M,$$

where $f$ is the max flow, $M$ is the minimum multicut, and $k$ is the number of commodities. We also show that our theorem is tight up to a constant factor, and we give a polynomial time algorithm for finding a multicut within $O(\log k)$ of the optimal fractional, and therefore also integral multicut.

Our general approach is along the lines of Leighton and Rao [LR] and Klein et al [KARR]: We consider the LP-relaxation of the minimum multicut problem, and use its optimal solution to define a graph with distance labels on the edges. Starting from a source or a sink, we grow a region in this graph until we find a cut of small enough capacity separating the root from its mate. The region is removed and the process is repeated. Our method differs in several respects from the previous ones: it dispenses with the discretization of the edge distances and employs a technique that leads to quicker termination of the region growth process and thus a better bound on the capacity of the cut. These techniques are encapsulated in the two region growing lemmas of Section 4, which use the idea of packing cuts to grow regions. These lemmas are useful in the demands version of the multicommodity flow problem as well. Using these lemmas, we improve the Klein et.al. result to:

$$\frac{\alpha}{O(\log k \log D)} \leq f \leq \alpha,$$

thus making the bound independent of capacities; this also enables us to dispense with the restriction that capacities be integral. The region growing lemmas also establish a unified framework in which simpler proofs of the theorems of Leighton and Rao and Klein et.al. can be given, by dispensing with tokenizing distances.

The multicut problem finds numerous applications, e.g. in circuit partitioning problems. It was first stated by Hu in 1963 [Hu]. For $k = 1$, the problem coincides of course with the ordinary min cut problem. For $k = 2$, it can be also solved in polynomial time by two applications of a max flow algorithm [YKCP]. The problem was proven NP-hard and MAX SNP-hard for any $k \geq 3$ by Dahlhaus, Johnson, Papadimitriou, Seymour and Yannakakis [DJPSY]. As a consequence of the MAX SNP-hardness, there is no polynomial time approximation scheme for multicut for $k \geq 3$ (assuming $P \neq NP$) [ALMSS].

Dahlhaus et.al. [DJPSY] studied the *multiway cut* problem: Given a set of "terminal" vertices $S$, find a minimum weight set of edges that disconnects every terminal from every other terminal; i.e., this is the special case of the multicut problem where there is one commodity for every pair of vertices from the subset $S$. They gave a factor-of-two approximation algorithm for this case, and showed that it can be used to approximate the general multicut problem within a factor of 2 with a running time that has $k$ in the exponent; thus the running time is polynomial only for fixed $k$. Klein et.al. [KARR] used their approximation algorithm for sparsest cut to give an $O(\log C \log^2 k)$ approximation algorithm for multicut.

Our improvement for multicut in turn gives us an $O(\log n)$ approximation algorithm for the problem of deleting the minimum number of clauses to make a 2CNF≡ formula satisfiable. The previous bound, due to Klein et.al., for this problem was $O(\log^3 n)$. This also yields an $O(\log n)$ approximation for the problem of deleting the minimum number of edges to make a graph bipartite.

## 2 LP formulations for max multicommodity flow

Given an undirected graph $G = (V, E)$, a capacity function $c : E \rightarrow \Re^+$, and $k$ pairs of vertices (not necessarily distinct) $\{s_i, t_i\}$ $1 \leq i \leq k$, we associate a commodity, $i$, with the pair $\{s_i, t_i\}$ and designate $s_i$ as the source and $t_i$ as the sink for commodity $i$. A *multicommodity flow* is a way of simultaneously routing commodities from their sources to sinks, subject to capacity and conservation constraints.

The assumption that each commodity has a single source and a single sink can be made without loss of generality. The more general case where a commodity $i$ may have a set $S_i$ of sources and a set $T_i$ of sinks can be easily reduced to this one by adding a new source $s_i$ with edges to the vertices in $S_i$ and a new sink $t_i$ with edges to $T_i$.

A multicommodity flow in which the sum of the flows over all the commodities is maximized will be called a *max (multicommodity) flow.* A *multicut* is defined as a set of edges whose removal disconnects each $\{s_i, t_i\}$ pair. The weight of the multicut is the sum of the capacities of the edges in it. The MULTICUT problem is to find a multicut of minimum weight.

We say that two vertices *share index* $i$ if they form the source-sink pair for commodity $i$.

Assume that there exist edges $(t_i, s_i)$ in $G$, $1 \le i \le k$. These edges are special; the only flow allowable on edge $(t_i, s_i)$ is commodity $i$ flowing from $t_i$ to $s_i$. There are no capacity restrictions on these edges. This allows us to view maxflow as a circulation in which the sum of the flows in the edges $(t_i, s_i)$, $1 \le i \le k$, is to be maximized. Let $f^l_{ij}$ denote the flow of commodity $l$ in edge $(i, j)$. The LP formulation of the problem is as follows

$$\begin{array}{llllll}
\text{maximize} & \sum_{i=1}^k f^i_{t_i s_i} & & & & \\
\text{subject to} & \sum_{(j,i)\in E} f^l_{ji} - \sum_{(i,j)\in E} f^l_{ij} & \le & 0 & \forall i \in V & \forall l \in [1..k] \\
 & \sum_{l=1}^k f^l_{ij} + \sum_{l=1}^k f^l_{ji} & \le & c_{ij} & \forall (i,j) \in E - \cup_{i=1}^k \{(t_i, s_i)\} & \\
 & f^l_{ij} & \ge & 0 & \forall (i,j) \in E & \forall l \in [1..k]
\end{array}$$

The first set of inequalities say that the total flow into vertex $i$, of each commodity, is at most the total flow out of it, of that commodity. Note that if these inequalities hold for each vertex $i \in V$, then in fact they must all hold with equality, thereby implying flow conservation at each node. This is so because a deficit in the flow balance at one node must imply a surplus at some other. The second set of inequalities are capacity constraints on the edges; the total flow over all commodities summed in both directions is at most the capacity of the edge.

The dual of this LP is

$$\begin{array}{llllll}
\text{minimize} & \sum_{(i,j)\in E} d_{ij} c_{ij} & & & & \\
\text{subject to} & d_{ij} & \ge & p^l_i - p^l_j & \forall (i,j) \in E - \cup_{i=1}^k \{(t_i, s_i)\} & \forall l \in [1..k] \\
 & p^i_{s_i} - p^i_{t_i} & \ge & 1 & \forall l \in [1..k] & \\
 & p^l_i & \ge & 0 & \forall i \in V & \forall l \in [1..k] \\
 & d_{ij} & \ge & 0 & \forall (i,j) \in E - \cup_{i=1}^k \{(t_i, s_i)\} &
\end{array}$$

The variable $d_{ij}$ can be viewed as a distance label on the edge $(i, j)$ and $p^l_i$ as the potential corresponding to commodity $l$ on vertex $i$. Thus the dual problem is an assignment of potentials to vertices and distance labels to edges so that the potential difference (for each commodity) across each edge is no more than the distance label of that edge. Further, the potential difference between the source and the sink for each commodity should be at least 1. These two conditions together imply that the distance between each $s_i, t_i$ under the distance label assignment $d_{ij}$ is at least 1. The following LP (with, however, exponentially many constraints) expresses this in a much simpler manner.

$$\begin{array}{lllll}
\text{minimize} & \sum_{e\in E} d_e c_e & & & \\
\text{subject to} & \sum_{e\in E} d_e q^j_i(e) & \ge & 1 & \forall q^j_i \\
 & d_e & \ge & 0 & \forall e \in E
\end{array}$$

where $q^j_i$ denotes the $j^{th}$ path in $G$ (under some arbitrary numbering) from $s_i$ to $t_i$ and $q^j_i(e)$ is the characteristic function of this path, i.e. $q^j_i(e) = 1$ if $e \in q^j_i$, 0 otherwise.

Clearly, the distance labels of a feasible solution to the first LP give a feasible solution to the second LP with the same objective function. Conversely, given a feasible solution to the second LP compute potentials on the vertices for each commodity as follows

> $p_i^l$ = length of the shortest path from vertex $i$ to the sink for commodity $l$, under distance labels $d_e$.

It can be checked that these potentials together with the distance labels $d_e$ are a feasible solution to the first LP with the same objective function. The two formulations of the dual are hence equivalent.

The dual program can now be viewed as an assignment of non-negative distance labels, $d_e$, to edges $e \in E$, so as to minimize $\sum_{e \in E} d_e c_e$, subject to the constraint that each $\{s_i, t_i\}$ pair be at least a unit distance apart. An integral solution to the dual problem corresponds to a multicut; the edges with $d_e = 1$ form a multicut. Hence, the dual is the LP relaxation of the MULTICUT problem.

## 3 Overview of the algorithm

In this section we will give a high level description of our algorithm, justifying the steps taken on intuitive grounds.

Our goal is to pick a set of edges of small capacity whose removal separates all $s_i, t_i$ pairs; the total capacity of edges picked should be within a small factor of the max flow (our factor is $O(\log k)$). Clearly, such edges will be bottlenecks for the max flow, so, one possibility is to find a max flow using an LP subroutine, and start with the set of saturated edges. A better possibility is to find an optimal solution to the dual LP, and consider the set of edges having positive distance labels. Notice that by complementary slackness, $d_e > 0 \Rightarrow \sum_{l=1}^{k} f_{ij}^l + \sum_{l=1}^{k} f_{ji}^l = c_e$, where $e = (i,j)$, i.e. $e$ must be saturated in every max flow. Moreover, the edges $D = \{e | d_e > 0\}$ constitute a multicut.

The entire set $D$ may have a very large capacity; we wish to pick a small capacity subset that is still a multicut. The optimal dual solution is the most cost effective way of picking a fractional multicut. So, this provides the clue that edges with large distance labels should be more important than edges with small distance labels for our purpose.

Instead, our algorithm indirectly gives preference to edges having large distance labels. We start by defining the length of edge $e$ in $G$ as $d_e$. We then find disjoint sets, such that for each set $S$, $C_{\nabla(S)} \leq \epsilon . wt(S)$, where $C_{\nabla(S)}$ is the capacity of the cut $(S, \overline{S})$, $\epsilon$ is an appropriately chosen parameter, and $wt(S)$ is roughly $\sum c_e d_e$, where the sum is over all edges having at least one endpoint in $S$. No set contains both source and sink of any pair, and for each commodity, either the source or the sink is in some set. Under these conditions, the union of the cuts of the sets is a multicut, and has capacity bounded by $2\epsilon F$, where $F$ is the value of the maximum flow.

The sets are formed by a fairly general procedure (rather than by exploiting any special structure of $G$). Each set is formed by growing out radially, w.r.t. the $d_e$'s, from one of the sources. This process is formally described in the next section. The reason for adopting radial growth is that this maximizes the weight of the set for a given bound on the pairwise distance between vertices in the set. The idea of packing cuts is used for growing sets, and for book-keeping. This yields simpler proofs, as well as a more precise bound on $\epsilon$.

# 4 Two crucial lemmas

In this section we shall prove two region growing lemmas that will be central to our multicut algorithm. We shall prove these in sufficient generality so that they can be applied to the other versions of the multicommodity flow problem as well.

Given a graph $G = (V, E)$, a capacity function $c : E \rightarrow \Re^+$, and distance labels $d : E \rightarrow \Re^+$ on the edges, define $B = \sum_{e \in E} d_e c_e$. A subset of vertices $V' \subseteq V$ is provided to the region-growing algorithm as the set of candidate roots from which regions will be grown; in our case, $V'$ is the set of sources and sinks. We associate a variable $y_S$ with each subset $S \subset V$; initially $y_S = 0$ for all $S$. The cut associated with a set $S$, denoted by $\bigtriangledown(S)$, is the set of edges with exactly one end point in $S$. The capacity of the cut, $C_{\bigtriangledown(S)}$, is $\sum_{e \in \bigtriangledown(S)} c_e$.

## 4.1 Growing a region

A region is grown in a radial manner starting from a root vertex, $r$. The order in which vertices are included in the region is the same as the order in which Dijkstra's algorithm finds shortest paths to vertices from $r$. We begin by picking a vertex, $r \in V'$, and assign it a weight $wt(r) = B/q$, where $q = |V'|$.

At any point in the algorithm we identify a set, $A$, as the active set and raise its variable $y_A$. Initially, the active set is $\{r\}$. Define the weight enclosed by the set $A$ as

$$wt(A) = \sum_{S \subseteq A} y_S C_{\bigtriangledown(S)} + wt(r)$$

An important requirement is that the $y_S$'s must form a packing i.e. $\forall e \in E$: $\sum_{S: e \in \bigtriangledown(S)} y_S \leq d_e$. Thus, if while raising $y_A$ we find that $\sum_{e \in \bigtriangledown(S)} y_S = d_e$ for some edge $e = (u, v) \in \bigtriangledown(A)$, $u \in A$, we make the set $A \cup \{v\}$ active, i.e. $A \leftarrow A \cup \{v\}$, and start increasing the variable corresponding to it. We keep growing the active set in this manner, one vertex at a time, until

$$C_{\bigtriangledown(A)} \leq \epsilon \cdot wt(A) \tag{1}$$

is satisfied, where $\epsilon$ is a constant that will be set appropriately while applying the lemma. Let $\mathcal{R}$ denote the active set for which condition 1 is satisfied.

Define the *radius* of $A$, $rad(A) = \sum_{S \subseteq A} y_S$.

**Lemma 4.1** $rad(\mathcal{R}) < \frac{\ln(q+1)}{\epsilon}$.

*Proof*: Let $S_1, S_2, \ldots, S_l$ denote the successive sets for which the variable $y_S > 0$. It is easy to check that these sets are nested, i.e. if $i < j$ then $S_i \subset S_j$. In what follows we denote the value of the variable $y_{S_i}$ by $y_i$ and $C_{\bigtriangledown(S_i)}$ by $C_i$.

Since while raising the variable $y_{S_i}$ from 0 to $y_i$ condition 1 was not satisfied

$$C_i > \epsilon \cdot wt(S_i)$$

From our definition of the weight enclosed by a set it follows that

$$\begin{aligned} wt(S_i) &= wt(S_{i-1}) + y_i C_i \\ &> wt(S_{i-1}) + \epsilon y_i wt(S_i) \end{aligned} \tag{2}$$

Thus,

$$wt(S_i) > \frac{wt(S_{i-1})}{(1-\epsilon y_i)}$$

Hence,

$$\begin{aligned} wt(S_l) &> \frac{wt(r)}{(1-\epsilon y_1)(1-\epsilon y_2)\ldots(1-\epsilon y_l)} \\ &= \frac{B}{q(1-\epsilon y_1)(1-\epsilon y_2)\ldots(1-\epsilon y_l)} \end{aligned}$$

Since the $y_S$'s form a packing ($\forall e \in E$: $\sum_{S:e\in\nabla(S)} y_S \le d_e$) it follows that

$$\sum_{i=1}^{l} y_i C_i = \sum_{i=1}^{l}\left(y_i \sum_{e\in\nabla(S_i)} c_e\right) = \sum_{e\in E}\left(c_e \sum_{S:e\in\nabla(S)} y_S\right) \le \sum_{e\in E} c_e d_e = B$$

and hence,

$$wt(S_l) = \sum_{i=1}^{l} y_i C_i + wt(r) \le B + \frac{B}{q} = B(1+\frac{1}{q})$$

Therefore,

$$\frac{B}{q(1-\epsilon y_1)(1-\epsilon y_2)\ldots(1-\epsilon y_l)} < wt(S_l) \le B(1+\frac{1}{q})$$

which implies,

$$\frac{1}{(1-\epsilon y_1)(1-\epsilon y_2)\ldots(1-\epsilon y_l)} < q+1$$

Taking natural logs we get

$$\sum_{i=1}^{l} \ln(1-\epsilon y_i)^{-1} < \ln(q+1)$$

From equation 2 it follows that $\epsilon y_i \le 1$, $1 \le i \le l$. Since $\ln(1-x)^{-1} \ge x$ for $|x| \le 1$,

$$\epsilon \sum_{i=1}^{l} y_i < \ln(q+1)$$

Thus, $rad(\mathcal{R}) = \sum_{i=1}^{l} y_i < \frac{\ln(q+1)}{\epsilon}$. ∎

Let $dist_d(u,v)$ denote the shortest path distance between $u$ and $v$ under the distance label assignment $d$. Consider vertex $v \in \mathcal{R}$. Let $S_i$ be the first set containing $v$, i.e. $v \in S_i - S_{i-1}$. Then from the manner in which we grow the region it follows that

$$dist_d(r,v) = rad(S_{i-1}) \tag{3}$$

**Corollary 4.2** *$\forall u,v \in \mathcal{R}$, $dist_d(r,u) \le rad(\mathcal{R})$ and $dist_d(u,v) \le 2rad(\mathcal{R})$.*

### 4.2 Growing disjoint regions

Having grown a region rooted at an arbitrary vertex in $V'$, remove all vertices contained in the region, and grow another region starting from a new root picked from $V'$. Continue in this manner until the residual graph contains no vertex of $V'$. Let $\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_p$ denote the regions formed. It is easy to see that these are disjoint. Clearly, $p \leq |V'| = q$.

Let $M = \bigtriangledown(\mathcal{R}_1) \cup \bigtriangledown(\mathcal{R}_2) \cup \ldots \bigtriangledown(\mathcal{R}_p)$.

**Lemma 4.3** $\sum_{e \in M} c_e \leq 2\epsilon B$.

*Proof*: Let $G_i = (V_i, E_i)$ be the graph obtained by deleting vertices contained in $\cup_{j=1}^{i-1} \mathcal{R}_j$ $(G_1 = G)$. Further, let $C^i_{\bigtriangledown(S)}$ denote the capacity of the cut $\bigtriangledown(S)$ in $G_i$. Note that $\sum_{e \in M} c_e = \sum_{i=1}^{p} C^i_{\bigtriangledown(\mathcal{R}_i)}$.

Each region $\mathcal{R}_i$ satisfies condition 1. Therefore, $C^i_{\bigtriangledown(\mathcal{R}_i)} \leq \epsilon.wt(\mathcal{R}_i)$, $1 \leq i \leq p$. Hence,

$$\begin{aligned}
\sum_{i=1}^{p} C^i_{\bigtriangledown(\mathcal{R}_i)} &\leq \epsilon \sum_{i=1}^{p} wt(\mathcal{R}_i) \\
&= \epsilon \left( \sum_{i=1}^{p} \sum_{S \subseteq \mathcal{R}_i} y_S C^i_{\bigtriangledown(S)} + \sum_{i=1}^{p} wt(r_i) \right)
\end{aligned}$$

where $r_i$ is the root of region $\mathcal{R}_i$. Since the $y_S$'s form a packing,

$$\sum_{i=1}^{p} \sum_{S \subseteq \mathcal{R}_i} y_S C^i_{\bigtriangledown(S)} \leq \sum_{e \in E} d_e c_e = B$$

Also,

$$\sum_{i=1}^{p} wt(r_i) = \frac{B}{q} p \leq B$$

Thus, $\sum_{i=1}^{p} C^i_{\bigtriangledown(\mathcal{R}_i)} \leq 2\epsilon B$ and hence $\sum_{e \in M} c_e \leq 2\epsilon B$. ∎

When growing region $\mathcal{R}_j$ in the graph $G_j$, we have that $\sum_{e \in E_j} d_e c_e \leq \sum_{e \in E} d_e c_e = B$. However, the proof of Lemma 4.1 goes through without any modifications. As regards equation 3, note that $rad(S_{i-1})$ is now the shortest path distance between $r$ and $v$ in the graph $G_j$; the shortest distance between these vertices in $G$ might be even smaller. Hence Corollary 4.2 still holds.

**Corollary 4.4** $\sum_{i=1}^{p} C_{\bigtriangledown(\mathcal{R}_i)} \leq 4\epsilon B$.

*Proof*: An edge in $M$ occurs in at most two cuts $\bigtriangledown(\mathcal{R}_i)$, $1 \leq i \leq p$. Thus,

$$\sum_{i=1}^{p} C_{\bigtriangledown(\mathcal{R}_i)} \leq 2 \sum_{e \in M} c_e \leq 4\epsilon B$$

∎

The time complexity of growing disjoint regions is $O(m + n \log n)$ as our algorithm is essentially the same as Dijkstra's algorithm for shortest paths.

# 5 Approximate max flow min multicut theorem

Clearly, the maxflow, $F$, is less than the weight of the minimum multicut, $M$, i.e. $F \leq M$.

The main result of this section is an algorithm that finds a multicut of weight at most $F \cdot O(\log k)$. We state this as a theorem for later reference.

**Theorem 5.1 (Approximating the minimum multicut)** *Consider an instance of the* MULTICUT *problem specified by a graph $G = (V, E)$, a capacity function $c : E \rightarrow \Re^+$ and $k$ pairs of vertices. One can, in polynomial time, find a multicut separating the specified pairs of vertices having weight within a factor $O(\log k)$ of the maximum flow over these pairs.*

Since $M$ is the minimum multicut, $M \leq F \cdot O(\log k)$. Thus the ratio of the optimal integral solution to the optimal fractional solution of the dual program is at most $O(\log k)$.

**Corollary 5.2 (Approximate max-flow min-multicut theorem)**

$$F \leq M \leq F \cdot O(\log k)$$

Further this bound on the ratio of the minimum multicut and maxflow is tight as shown in Theorem 5.4.

For planar graphs, Tardos and Vazirani [TV] obtain a constant factor approximation for the minimum multicut. Garg, Vazirani and Yannakakis [GVY] approximate the minimum multicut on trees to within twice the optimal. They also give a factor $\frac{1}{2}$ approximation algorithm for maximum integral multicommodity flow on trees, and they show that even for planar graphs, the integrality gap for flow is unbounded thereby ruling out LP duality based methods for approximating maximum integral multicommodity flow. Both these results also establish approximate max-flow min-multicut theorems.

## 5.1 Finding the multicut

First solve the dual LP to obtain a set of distance labels, $d_e$, $e \in E$. Next, grow regions as in section 4. The constant $\epsilon$ and the set $V'$ are chosen to ensure that $\bigtriangledown(\mathcal{R}_1) \cup \bigtriangledown(\mathcal{R}_2) \cup \ldots \bigtriangledown(\mathcal{R}_p)$ is a multicut.

The vertices of $V$ that are the source for some commodity form the set $V'$, i.e. $V' = \cup_{i=1}^{k}\{s_i\}$. Thus, $s_i \in \cup_{j=1}^{p}\mathcal{R}_j$, $1 \leq i \leq k$. Now if we can choose $\epsilon$ so that no two vertices in $\mathcal{R}_i$, $1 \leq i \leq p$, share an index, we shall be done.

**Lemma 5.3** *If $\epsilon = 2\ln(k+1)$ then no two vertices in $\mathcal{R}_i$ share an index.*

*Proof*: Note that $q = |V'| \leq k$. Therefore, if $\epsilon = 2\ln(k+1)$ then by Lemma 4.1, $rad(\mathcal{R}_i) < \frac{1}{2}$. Hence by Corollary 4.2 the distance between any two vertices in $\mathcal{R}_i$ is less than 1. Since the assignment of distance labels to edges is such that the distance between any two vertices sharing an index is at least 1, no two vertices in $\mathcal{R}_i$ share an index. ■

Substituting this choice of $\epsilon$ into Lemma 4.3 we find that the multicut obtained has weight at most $4B\ln(k+1)$. Since $B = \sum_{e \in E} d_e c_e = F$, the maxflow, the weight of the multicut is within a factor $4\ln(k+1)$ of the maxflow.

### 5.2 A tight example

**Theorem 5.4** *$\forall k, n$, $k \leq n$, there exists an $n$ vertex graph, $G$, and $\Omega(k^2)$ pairs of vertices in $G$ such that the ratio between the minimum multicut and the max flow is $\Omega(\log k)$.*

*Proof*: As in [LR], we use an expander graph. Let $G = (V, E)$ be a $k$-vertex, bounded degree expander graph (each vertex has degree at most $d$, for an appropriate constant $d$). Every vertex has at most $\frac{k}{2}$ vertices within distance $\log_d(\frac{k}{2})$. Thus, $G$ has $\Omega(k^2)$ pairs of vertices that are a distance $\log_d(\frac{k}{2})$ or more apart. Let these be the pairs for the MULTICUT instance. All edges of the graph have unit capacity, and hence the total capacity of the edges is $O(k)$. Since each flow path is $\Omega(\log k)$ long, the maximum flow is $O\left(\frac{k}{\log k}\right)$. The optimum multicut can be viewed as a partition of the vertex set of $G$. Since $G$ is an expander, any set, $S$, in the partition has $\Omega(|S|)$ edges running across it, provided $|S| \leq \frac{k}{2}$. Any subgraph with more than $\frac{k}{2}$ vertices has pairs that are $\log_d(\frac{k}{2})$ apart, and hence contains a pair of vertices that share an index. Thus, no set in the partition induced by the multicut has more than $\frac{k}{2}$ vertices, and so each set $S$ in the partition has $\Omega(|S|)$ edges running across it. Hence the number of edges in the multicut is $\Omega(k)$. This yields a ratio of $\Omega(\log k)$ between the weight of the minimum multicut and the maxflow.

Note that splitting an edge by adding vertices on the edge does not change the max flow or the minimum multicut. We modify $G$ into an $n$-vertex graph by adding an appropriate number of vertices on the edges of $G$. ∎

We remark that in the case of the multiway cut problem [DJPSY], i.e., the special case of the multicut problem where the given set of source-sink pairs for the commodities consists of all pairs of vertices from a given subset $S$ of terminals, the gap between min cut and max flow is much smaller, it is at most $2 - \frac{2}{k}$ [Cu]. Of course, if $S$ is the whole set of vertices (i.e., there is one commodity for every pair of vertices), the problem is trivial and there is no gap: max-flow = min-cut = total capacity of the graph.

## 6 Multicommodity flow with specified demands

We next consider the case when along with the source and sink for commodity $i$, $1 \leq i \leq k$, we are also specified a demand, $dem(i)$, for the commodity. A multicommodity flow is *feasible* if it meets the demand for each commodity.

As in section 4 we define the cut associated with $S$, denoted by $\nabla(S)$, as the set of edges with exactly one end point in $S$. The capacity of the cut, $C_{\nabla(S)}$, is $\sum_{e \in \nabla(S)} c_e$. The set $S$ separates commodity $i$ iff exactly one of $\{s_i, t_i\}$ is in $S$. The demand across the cut, $D_{\nabla(S)}$, is the sum of the demands of all commodities separated by set $S$. Clearly, the following condition is necessary for the existence of a feasible multicommodity flow:

*Cut condition*: $\forall S \subseteq V$, $C_{\nabla(S)} \geq D_{\nabla(S)}$

However, this condition is not sufficient. Extensive work has been done on characterizing graphs, with distinguished sources and sinks, for which the cut condition is both necessary and sufficient. Again, no complete characterization is known.

Klein et.al. view this problem as follows: they wish to find the minimum factor, $u$, by which the capacity of the edges should be raised to ensure a feasible flow. Equivalently, they wish to find the maximum factor, $f$, such that it is possible to route simultaneously an amount $f \cdot dem(i)$ of each commodity while satisfying the capacity constraints; at optimality $f = \frac{1}{u}$.

Let $q_i^j$ denote a path in $G$ from $s_i$ to $t_i$ and $q_i^j(e)$ be the characteristic function of this path, i.e. $q_i^j(e) = 1$ if $e \in q_i^j$, 0 otherwise. Then an LP formulation for this problem is

$$\begin{array}{llcll} \text{maximize} & f & & & \\ \text{subject to} & \sum_{i,j} f_i^j q_i^j(e) & \leq & c_e & \forall e \in E \\ & \sum_j f_i^j & \geq & f.dem(i) & 1 \leq i \leq k \\ & f_i^j & \geq & 0 & \forall q_i^j \end{array}$$

where $f_i^j$ is the flow along the path $q_i^j$. Thus, multicommodity flow is feasible if and only if $f$ is at least 1.

We can formulate the dual LP which now calls for an assignment of nonnegative distance labels, $d_e$, to edges $e \in E$ so as to minimize $\sum_{e \in E} d_e c_e$, subject to the constraint $\sum_{i=1}^{k} dist_d(s_i, t_i) dem(i) \geq 1$, where $dist_d(u, v)$ is the shortest path distance between $u$ and $v$ under this assignment of distance labels. At optimality, we have $\sum_{e \in E} d_e c_e = f$.

How does $f$ relate to the structure of the graph? The cut condition motivates the following definition. Define the *sparsest cut* as the cut that minimizes the ratio $\frac{C_{\nabla(S)}}{D_{\nabla(S)}}$. Let,

$$\alpha = \min_{S \subseteq V} \frac{C_{\nabla(S)}}{D_{\nabla(S)}}$$

Clearly, $f \leq \alpha$. Klein et.al. show that the "throughput" $f$ is at least $\frac{\alpha}{O(\log C \log D)}$, where $C$ is the sum of all capacities and $D$ is the total demand. This was later improved to $\frac{\alpha}{O(\log n \log D)}$ by Tragoudas [Trag].

We improve this result by providing a tighter bound on $f$.

**Theorem 6.1**

$$\frac{\alpha}{O(\log k \log D)} \leq f \leq \alpha$$

*Thus, for a multicommodity flow to be feasible it is necessary that the sparsest cut ratio, $\alpha$, be at least 1 and sufficient that it be $O(\log k \log D)$.*

Plotkin and Tardos [PT] give a method of scaling demands so that the $\log D$ factor in Theorem 6.1 can be replaced by $\log k$. Hence they improve the bound in Theorem 6.1 to $O(\log^2 k)$.

For uniform multicommodity flow on bounded degree expanders, the ratio between $f$ and $\alpha$ is $O(\log n)$ ($n$ is the number of vertices) [LR]. We can, as in Theorem 5.4, modify this, by adding new vertices on edges, to get a graph on $n$ vertices and $k$ commodities ($k \leq \frac{n(n-1)}{2}$) for which $f = \frac{\alpha}{O(\log k)}$. It is an open problem to bridge the gap between the $O(\log^2 k)$ bound of [PT] and this $O(\log k)$ example.

For planar graphs, Klein, Plotkin and Rao [KPR] show that multicommodity flow is feasible if the sparsest cut ratio, $\alpha$, is logarithmic.

### 6.1 Proof of Theorem 6.1

First solve the dual LP to obtain a set of distance labels, $d_e$, $e \in E$. We need to bound the optimal "throughput", $f$, in terms of $\alpha$. By LP duality we have that $f$ is equal to $B = \sum_{e \in E} d_e c_e$.

The proof proceeds by finding short paths, $P(s_i, t_i)$, for each commodity, and furthermore ensuring that

$$\sum_{i=1}^{k} dem(i) length_d(P(s_i, t_i)) \leq \frac{f}{\alpha} O(\log k \log D) \tag{4}$$

Since $dist_d(s_i, t_i) \leq length_d(P(s_i, t_i))$ and $1 \leq \sum_{i=1}^{k} dem(i) dist_d(s_i, t_i)$, Theorem 6.1 follows.

As in Klein et.al., the algorithm for finding paths proceeds in phases. Let the *residual demand* after phase $i$, $D_i$, be the total demand over commodities which have not been assigned any path in phases 1 to $i$. In the $(i+1)^{\text{st}}$ phase we assign paths to some of these commodities so that the residual demand is halved, i.e. $D_{i+1} \leq \frac{D_i}{2}$. Thus the total number of phases is at most $\log D$. Note that the Lemmas of Section 4 provide a way of growing regions with small radius and small cuts. The small cuts imply that most of the demand is included within the regions. Thus we can get short paths for a good fraction of the demand.

Our implementation of a phase involves growing regions as in Section 4. The vertices that are sources for some commodity form the set $V'$. Thus, $q = |V'| \leq k$. Each commodity, both of whose endpoints (source and sink) lie within a region, is assigned a path within that region. In this way, we ensure that the length of the path is at most twice the radius of the region. Since we wish to halve the residual demand we need to choose $\epsilon$ so that most of the commodities have both their endpoints included in the same region. Contrast this with Lemma 5.3 where $\epsilon$ was chosen so that no commodity had its endpoints in the same region.

Since, $\forall S \subseteq V$: $\frac{C_{\nabla(S)}}{D_{\nabla(S)}} \geq \alpha$, it follows that $D_{\nabla(S)} \leq \frac{1}{\alpha} C_{\nabla(S)}$. Thus, the residual demand after phase $i$ is

$$D_{i+1} \leq \sum_{j=1}^{p} D_{\nabla(\mathcal{R}_j)} \leq \frac{1}{\alpha} \sum_{j=1}^{p} C_{\nabla(\mathcal{R}_j)}$$

From Corollary 4.4 we get

$$D_{i+1} \leq \frac{4\epsilon B}{\alpha} = \frac{4\epsilon f}{\alpha}$$

We choose $\epsilon = \frac{\alpha D_i}{8f}$. Thus, $D_{i+1} \leq \frac{D_i}{2}$.

Since $q \leq k$ it follows from Lemma 4.1 and Corollary 4.2 that any two vertices in $\mathcal{R}_i$ are at most $\frac{2\ln(k+1)}{\epsilon}$ apart. Thus any commodity both of whose endpoints lie in a region can be assigned a path of length at most $\frac{2\ln(k+1)}{\epsilon} = \frac{16 f \ln(k+1)}{\alpha D_i}$.

Thus the contribution of each phase to the left hand side of equation 4 is at most $\frac{16 f \ln(k+1)}{\alpha}$. Hence the total contribution of all phases is at most $\frac{16 f \ln(k+1) \log D}{\alpha} = \frac{f}{\alpha} O(\log k \log D)$.

# 7 Uniform multicommodity flow

For uniform multicommodity flow Leighton and Rao prove that the "throughput", $f$, is at least $\frac{\alpha}{O(\log n)}$ ($\alpha$ is the sparsest cut ratio). The actual constant they obtain is 30 [private communication from Satish Rao]. In this section we shall use our region-growing lemmas to give a proof that is cleaner than the original by dispensing with discretization (the rest of the ideas remain essentially the same). This proof also improves the constant to $8 \ln 2$. The uniform multicommodity flow problem is a special case of the multicommodity flow problem of section 6 where there is one commodity for each unordered pair of vertices and the demand for each commodity is the same, say 1 unit (or equivalently, for each ordered pair of vertices $u_i, u_j$ we wish to route half unit of flow from $u_i$ to $u_j$). Thus the demand across a cut $\bigtriangledown(S)$ is $D_{\bigtriangledown(S)} = |S| \left|\overline{S}\right|$, and the sparsest cut is the cut that minimizes the ratio $\frac{C_{\bigtriangledown(S)}}{D_{\bigtriangledown(S)}} = \frac{C_{\bigtriangledown(S)}}{|S|\left|\overline{S}\right|}$. Let

$$\alpha = \min_{S \subseteq V} \frac{C_{\bigtriangledown(S)}}{|S| \left|\overline{S}\right|}$$

The throughput $f$ is bounded from above by $\alpha$.

The dual program calls again for an assignment of non negative distance labels, $d_e$, to edges $e \in E$, so that $\sum_{e \in E} d_e c_e$ is minimized, subject to the constraint that $\sum_{v,w \in V} dist_d(v,w) \geq 1$, where the sum extends over all unordered pairs of vertices. At optimality we have

$$\sum_{e \in E} d_e c_e = f$$

**Theorem 7.1 (Leighton and Rao)**

$$\frac{\alpha}{O(\log n)} \leq f \leq \alpha$$

*Thus in any $n$-vertex graph one can always find a uniform multicommodity flow of $\frac{\alpha}{O(\log n)}$ units.*

Leighton and Rao also provide an example to show that the bound on $f$ is essentially tight.

## 7.1 Proof of Theorem 7.1

We follow the structure of the Leighton-Rao proof, except that we shall use the lemmas of section 4. First solve the dual LP to obtain a set of distance labels $d_e$. By LP duality, we have that $f$ is equal to $B = \sum_{e \in E} d_e c_e$. Since $\sum_{v,w \in V} dist_d(v,w) \geq 1$, it suffices to show that $\sum_{v,w \in V} dist_d(v,w) \leq \frac{f}{\alpha} \cdot O(\log n)$. Notice that the following two facts hold.

1. For a vertex $r$ in $G$, since $dist_d(v,w) \leq dist_d(r,v) + dist_d(r,w)$, we have

$$\sum_{v,w \in V} dist_d(v,w) \leq n \sum_{v \in V} dist_d(r,v)$$

2. A closely related quantity to the sparsest cut is the *flux* or the *minimum edge expansion ratio* of the graph. It is defined as

$$\beta = \min_{S \subseteq V} \frac{C_{\nabla(S)}}{\min(|S|, |\overline{S}|)}$$

It is easy to see that $\frac{\alpha n}{2} \leq \beta \leq \alpha n$

These observations imply that if $\sum_{v \in V} dist_d(r, v) \leq \frac{\Delta}{\beta}$, for some quantity $\Delta$, then $\sum_{v,w \in V} dist_d(v, w) \leq \frac{2\Delta}{\alpha}$. Thus it suffices to show that there is a vertex $r$ such that $\sum_{v \in V} dist_d(r, v) \leq \frac{f}{\beta} \cdot O(\log n)$.

We wish to choose vertex $r$ so that the sum of the distances of all other vertices from it is small. Note that the regions grown in section 4 have bounded radius (Lemma 4.1). Thus if we can grow a region with a large number of vertices, we can pick $r$ to be the root of this region.

Grow regions as in section 4 with $V' = V$. These regions partition the vertex set. We choose $\epsilon$ to ensure that one of the regions has at least $\frac{n}{2}$ vertices.

Assume all regions have less than $\frac{n}{2}$ vertices. Since $\forall S \subseteq V$: $\frac{C_{\nabla(S)}}{\min(|S|,|\overline{S}|)} \geq \beta$, we have $C_{\nabla(\mathcal{R}_i)} \geq \beta\,|\mathcal{R}_i|$ and therefore

$$\sum_{i=1}^{p} C_{\nabla(\mathcal{R}_i)} \geq \beta \sum_{i=1}^{p} |\mathcal{R}_i| = \beta n$$

By Corollary 4.4 we have

$$\sum_{i=1}^{p} C_{\nabla(\mathcal{R}_i)} \leq 4\epsilon B = 4\epsilon f$$

Thus,

$$\beta n \leq \sum_{i=1}^{p} C_{\nabla(\mathcal{R}_i)} \leq 4\epsilon f$$

Choosing $\epsilon < \frac{\beta n}{4f}$ violates the above inequality. Hence with this choice of $\epsilon$ one of the regions, say $\mathcal{R}*$, has at least $\frac{n}{2}$ vertices. Let $r$ be the root of region $\mathcal{R}*$.

Next, we argue that $r$ has the desired property. Reset all the variables $y_S$ to zero, and grow a region from root $r$ as in section 4. However, now we do not check for condition 1 and stop only when all vertices are included in the region. Let $\{r\} = S_1 \subset S_2 \subset \cdots S_l = \mathcal{R} \subset \cdots S_t = V$ denote the sets with $y_S > 0$. Let $S_l = \mathcal{R}$ be the first set in this chain that contains all the vertices of $\mathcal{R}*$. The set $\mathcal{R}$ may be a proper superset of $\mathcal{R}*$, because some nodes may have been deleted from the graph in the previous stage by the time we grew the region around $r$. In any case however, $rad(\mathcal{R}) \leq rad(\mathcal{R}*)$. Since $q = |V'| = n$, by Lemma 4.1 $rad(\mathcal{R}) \leq \frac{\ln(n+1)}{\epsilon} = \frac{4f\ln(n+1)}{\beta n}$.

If $S_i$ is the smallest set containing vertex $v$ then $dist_d(r, v) = \sum_{S \subset S_i} y_S$. Therefore, the total distance of all vertices from $r$ is

$$\sum_{v \in V} dist_d(r, v) = \sum_{S} y_S(n - |S|) = \sum_{S \subset \mathcal{R}} y_S(n - |S|) + \sum_{S \supseteq \mathcal{R}} y_S(n - |S|)$$

Now,

$$\sum_{S \subset \mathcal{R}} y_S(n - |S|) \leq n \sum_{S \subset \mathcal{R}} y_S \leq n.rad(\mathcal{R}) \leq \frac{4f\ln(n+1)}{\beta}$$

Since $|\mathcal{R}| \geq \frac{n}{2}$, all supersets of $\mathcal{R}$ have at least $\frac{n}{2}$ vertices. Hence, for these sets $\frac{C_{\nabla(S)}}{n-|S|} \geq \beta$ and so,

$$\sum_{S \supseteq \mathcal{R}} y_S(n - |S|) \leq \frac{1}{\beta} \sum_{S \supseteq \mathcal{R}} y_S C_{\nabla(S)} \leq \frac{1}{\beta} \sum_{e \in E} d_e c_e = \frac{f}{\beta}$$

Therefore,

$$\sum_{v \in V} dist_d(r, v) \leq \frac{f}{\beta}(4 \ln(n+1) + 1)$$

It follows that the throughput $f$ satisfies

$$f \geq \frac{\alpha}{8 \ln(n+1) + 2} = \frac{\alpha}{O(\log n)}$$

# 8 Applications

Several graph problems can be viewed as edge deletion problems; we wish to find a minimum weight set of edges whose removal yields a graph with a desired structure $\pi$ [Ya]. Klein et.al. [KARR] propose a method for approximating such a problem when the property $\pi$ can be specified as a 2CNF$\equiv$ formula so that deleting edges from the graph corresponds to deleting clauses in the formula. In particular, they show how to model the minimum edge deletion graph bipartization problem - deleting a minimum weight set of edges so that the resulting graph is bipartite. The 2CNF$\equiv$ deletion problem is defined as follows.

> A 2CNF$\equiv$ formula, $F$, is a weighted set of clauses of the form $P \equiv Q$ where $P$, $Q$ are literals. Find a minimum weight set of clauses the deletion of which makes the formula satisfiable.

Klein et. al. showed that this problem can be reduced to the minimum multicut problem as follows. Construct a graph $G(F)$ whose vertex set is the set of literals in $F$. For each clause of the kind $P \equiv Q$ include two edges $(P, Q)$ and $(\overline{P}, \overline{Q})$ of capacity equal to the weight of the clause $P \equiv Q$. Then, it is not hard to see that the 2CNF$\equiv$ formula, $F$, is satisfiable iff no connected component of the graph $G(F)$ contains both a literal and its complement [JLL]. Let $M$ be a minimum weight set of edges whose removal separates the pairs of complementary literals in $G(F)$, and let $W$ be the minimum weight set of clauses whose deletion makes $F$ satisfiable. Then we have:

**Lemma 8.1** *$wt(W) \leq wt(M) \leq 2wt(W)$.*

*Proof*: The minimum multicut, $M$, in $G(F)$ corresponds to a set of clauses (of weight at most $wt(M)$) whose deletion makes the formula satisfiable. Hence, $wt(W) \leq wt(M)$.

Each clause of $F$ corresponds to two edges in $G(F)$. Thus the set $W$ corresponds to a multicut in $G(F)$ of weight at most $2wt(W)$. Therefore, $wt(M) \leq 2wt(W)$. ∎

Finding the set of edges, $M$, is exactly the MULTICUT problem on the graph $G(F)$ with every pair of complementary literals forming a source-sink pair. Thus, the number of pairs, $k$, is equal to the number of variables in the formula, $n$, and hence by Theorem 5.1 we can approximate $M$ to within a factor $O(\log n)$. Using Lemma 8.1 we get

**Theorem 8.2** *Given a 2CNF≡ formula, one can in polynomial time find a set of clauses of weight at most a factor $O(\log n)$ of the minimum weight set of clauses whose deletion makes the formula satisfiable.*

**Corollary 8.3** *The edge-deletion graph bipartization problem can be approximated within a factor $O(\log n)$ in polynomial time.*

We leave open the question whether these problems can be approximated within some constant factor. We know they are both MAX SNP-hard [PY] and hence do not have a polynomial time approximation scheme unless $P = NP$ [ALMSS].

**Acknowledgement** We wish to thank Phil Klein for simplifying the formulation in Section 6, which made the presentation more uniform.

## References

[ALMSS] S. Arora, C. Lund, R. Motwani, M. Sudan and M. Szegedy (1992), "Proof verification and hardness of approximation problems", *Proceedings of* $32^{nd}$ *IEEE Symposium on Foundations of Computer Science* (1992), pp. 14-23.

[Cu] W. H. Cunningham (1991), "The optimal multiterminal cut problem", *DIMACS Series in Disc. Math. and Theor. Comput. Sci.* 5 (1991), pp. 105-120.

[DJPSY] E. Dahlhaus, D.S. Johnson, C.H. Papadimitriou, P.D. Seymour and M. Yannakakis (1992), "The complexity of multiway cuts", *Proceedings,* $24^{th}$ *ACM Symposium on Theory of Computing* (1992).

[EFS] P. Elias, A. Feinstein and C.E. Shannon (1956), "A note on the maximum flow through a network", *IRE Transactions on Information Theory IT* 2 (1956), pp. 117-119.

[FF] L.R. Ford, Jr. and D.R. Fulkerson (1956), "Maximal flow through a network", *Canadian Journal of Mathematics* 8 (1956).

[GV] N. Garg and V. V. Vazirani, "A characterization of the vertices and edges of the $s-t$ cut polyhedron, with algorithmic applications," *Proc. Integer Programming and Combinatorial Optimization* (1993).

[GVY] N. Garg, V. V. Vazirani and M. Yannakakis, "Primal-dual approximation algorithms for integral flow and multicut in trees, with applications to matching and set cover", *Proc. 20th Intl. Coll. on Automata, Languages and Programming* (1992), pp. 64-75.

[JLL] N. D. Jones, Y. E. Lien and W. T. Lasser, "New problems complete for nondeterministic logspace," *Math. Systems Theory* 10 (1976), pp. 1-17.

[Hu] T.C. Hu (1963), "Multicommodity network flows", *Operations Research* 11 (1963), pp. 344-360.

[KARR] P. Klein, A. Agrawal, R. Ravi and S. Rao (1990), "Approximation through multicommodity flow", *Proceedings, $31^{st}$ Symposium on Foundations of Computer Science* (1990), pp. 726-737.

[KPR] P. Klein, S. Plotkin and S. Rao, "Ecluded minors, network decomposition, and multicommodity flow," *proc. $25^{th}$ ACM Symposium on Theory of Computing* (1992), pp. 682-690.

[KST] P. Klein, C. Stein, and E. Tardos (1990), "Leighton-Rao might be practical: faster approximation algorithms for concurrent flow with uniform capacities", *Proceedings, $22^{nd}$ ACM Symposium on Theory of Computing* (1990), pp. 310-321.

[LR] F.T. Leighton and S. Rao (1988), "An approximate max-flow min-cut theorem for uniform multicommodity flow problems with application to approximation algorithms", *Proceedings, $29^{th}$ Symposium on Foundations of Computer Science* (1988), pp. 422-431.

[PY] C. H. Papadimitriou and M. Yannakakis (1991), "Optimization, approximation and complexity classes", *Journal of Computer and System Sciences* 43 (1991), pp. 425-440.

[PT] S. Plotkin and E. Tardos, "Improved bounds on the max-flow min-cut ratio for multicommodity flows," *Proc. $25^{th}$ ACM Symposium on Theory of Computing* (1992), pp. 691-697.

[TV] E. Tardos and V. V. Vazirani, "Improved bounds for the max-flow min-multicut ratio for planar and $K_{r,r}$-free graphs," *Information Processing Letters* (1993).

[Trag] S. Tragoudas (1991) "VLSI partitioning approximation algorithms based on multicommodity flows and other techniques." Ph.D. thesis, University of Texas at Dallas.

[Ya] M. Yannakakis (1981), "Edge-Deletion problems", *SIAM Journal of Computing* 10 (1981), pp. 297-309.

[YKCP] M. Yannakakis, P. C. Kanellakis, S. C. Cosmadakis, and C. H. Papadimitriou, "Cutting and partitioning a graph after a fixed pattern", *Proceedings, $10^{th}$ Intl. Coll. on Automata, Languages and Programming* (1983), pp. 712-722.