Excluded Minors, Network Decomposition, and Multicommodity Flow

Philip Klein* Brown University

Serge A. Plotkin† Stanford University

Satish Rao NEC Research Institute

Abstract

In this paper we show that, given a graph and parameters δ and r, we can find either a $K_{r,r}$ minor or an edge-cut of size $O(mr/\delta)$ whose removal yields components of weak diameter $O(r^2\delta)$; *i.e.*, every pair of nodes in such a component are at distance $O(r^2\delta)$ in the original graph.

Using this lemma, we improve the best known bounds for the min-cut max-flow ratio for multicommodity flows in graphs with forbidden small minors. In general graphs, it was known that the ratio is $O(\log k)$ for the uniform-demand case (the case where there is a unit-demand commodity between every pair of nodes), and that the ratio is $O(\log^2 k)$ for arbitrary demands, where k is the number of commodities. In this paper we show that for graphs excluding any fixed graph as a minor (*e.g.* planar graphs or bounded-genus graphs), the ratio is $O(1)$ for the uniform-demand case and $O(\log k)$ for the arbitrary demand case.

For such graphs, our method yields min-ratio cut approximation algorithms with performance bounds that match the above ratios. Computation of such cuts is a basic step for a variety of approximation algorithms for NP-complete problems.

1 Introduction

The multicommodity flow problem is a natural generalization of the maximum-flow problem where, instead of a single commodity, we have several different commodities. Each commodity has an associated source/sink pair and a demand; the goal is to ship all of the demands from the sources to the corresponding sinks such that the total amount of flow through each edge is at most the edge's capacity. The optimization version of the multicommodity flow problem, called the *concurrent flow problem* [17], is to maximize the *throughput*, which is defined as the value z such that there exists a multicommodity flow shipping a fraction z of each demand. In the case of a single commodity, the maximum throughput is exactly the value of the maximum flow.

Many optimization problems can be approximately solved using special cases of multicommodity flow or concurrent flow. Applications include small area VLSI layout, network routing, and efficient simulations of one interconnection network by another.

An upper bound on the throughput can be derived by considering cuts in the network. For any cut, the throughput times the sum of demands that cross the cut must not exceed the capacity of the cut. Thus the throughput is at most the minimum, taken over all cuts, of the ratio of demand across the cut to capacity of the cut. The celebrated max-flow min-cut theorem for single-commodity flow [7, 8] states that the throughput always achieves this upper bound; in other words, max-flow equals min-cut. Moreover, this theorem yields an algorithm for finding the minimum cut.

Equality between multicommodity max-flow (throughput) and min-cut (minimum cut ratio) does not hold for all multicommodity flow instances; nor is there a fast multicommodity min-cut algorithm in general unless P=NP. Instead, one might try to show that the minimum cut ratio is not much more than the maximum throughput, and that there is an algorithm to find approximately minimum-ratio cuts.

Leighton and Rao [11] were the first to obtain such an approximate min-cut max-throughput result. They

*Research supported by NSF grant CCR-9012357 and NSF PYI award CCR-9157620, together with PYI matching funds from Thinking Machines Corporation and Xerox Corporation. Additional support provided by DARPA contract N00014-91-J-4052 ARPA Order No. 8225, by NEC Research Institute, and by DIMACS.

†Research supported by NSF Research Initiation Award CCR-900-8226, by U.S. Army Research Office Grant DAAL-03-91-G-0102, by a grant from Mitsubishi Electric Laboratories, and by NEC Research Institute.

considered the special case of *uniform demands* (unit-demand commodity between every pair of nodes), and showed that for instances of this form the min-cut/max-flow ratio is bounded by $O(\log n)$, where n is the number of nodes in the network. Clearly, in order that a given multicommodity flow instance be feasible (*i.e.* that there exist flows that satisfy the demands and the capacity constraints), it is necessary for the capacity of each cut to be at least the demand separated by this cut; Leighton and Rao showed that, conversely, it is sufficient that the capacity of each cut exceed the separated demand by an $O(\log n)$ factor.

Their proof also yields an approximation algorithm for the minimum-ratio cut; the algorithm outputs a cut whose ratio is at most $O(\log n)$ times that of the minimum-ratio cut (or, in short, min-cut). This approximation algorithm is the key subroutine of approximation algorithms for a variety of NP-hard graph problems, including finding small-area VLSI layouts [6], graph chordalization, and register sufficiency [10].

An approximate relation between max-flow and min-cut for arbitrary multicommodity flow was discovered by Klein, Rao, Agrawal, and Ravi [10]. They showed that the min-cut/max-flow ratio is bounded by $O(\log C \log D)$, where C is the sum of the capacities and D is the sum of the demands. This ratio was improved to $O(\log^2 k)$ (k is the number of commodities) through the work of Tragoudas [18]; Garg, Vazirani and Yannakakis [9]; and Plotkin and Tardos [15].

In this paper, we show that for planar networks, the above bounds can be improved to $O(\log k)$ for the general-demand case, and to $O(1)$ for uniform-demand case. In fact, we prove that the improved bounds hold for any minor-closed family of graphs that excludes some constant-size minor.

Previous work in this direction concentrated on finding special classes of instances for which min-cut/max-flow equality holds. For example, Seymour [16] considered the capacity-demand graph, the graph obtained from the original network by adding a source-sink edge for each commodity; he showed that if this graph excludes K_5 as a minor, then max-flow equals min-cut. Okamura and Seymour [13] showed that in a planar graph, if all sources and sinks lie on the boundary of a single face, max-flow equals min-cut.

Our results are based on a new method for *graph decomposition*. Various graph decompositions have already proved useful in the context of distributed computation. Examples include routing with small size tables [14, 4], symmetry-breaking and coloring [2], and synchronization of asynchronous networks [5]. Generally, the approach is to remove a small set of edges and thereby decompose the graph into connected regions such that the nodes in each region are "near" each other. The decompositions fall into two categories according to whether the nodes in a region are required to be nearby in the subgraph induced by this region (we say that such a region has small *strong* diameter), or whether it is sufficient that they be nearby in the original graph (in which case the region has small *weak* diameter) [12].

All previously known decomposition theorems hold for arbitrary graphs. The best known decomposition removes at most a constant fraction of the edges and achieves regions with strong diameter $O(\log n)$. We prove that for graphs excluding a fixed minor (*e.g.* planar graphs or bounded-genus graphs), there is a decomposition that removes at most a constant fraction of the edges and achieves regions of $O(1)$ weak diameter. In view of the many applications in distributed computing that have been proposed for decomposition theorems, it is possible that our result will be useful in achieving improved performance for distributed computation in, *e.g.*, planar networks.

2 Preliminaries

An instance of the *multicommodity flow problem* (MFP) consists of an undirected graph $G = (V, E)$, a non-negative capacity $u(vw)$ for every edge $vw \in E$, and a specification of k commodities, numbered 1 through k, where the specification for commodity i consists of a source-sink pair $s_i, t_i \in V$ and a non-negative demand d_i. We will denote the number of nodes by n, and the number of edges by m. For notational convenience, we assume that $m \geq n$, and that the graph G is connected and has no parallel edges. Also, for notational convenience, we arbitrarily direct each edge. If there is an edge directed from v to w, this edge is unique by assumption, and we denote it by vw. We assume that the capacities and the demands are integral, and denote the sum of capacities by C and the sum of demands by D.

A multicommodity flow f consists of functions $f_i : E \to \mathbb{R}^+$ for every commodity i, which represent the *flow* of commodity i on edge vw. If the flow of commodity i on edge vw is oriented in the same direction as edge vw, then $f_i(vw)$ will be positive, otherwise it will be negative; the signs only serve to indicate the direction of the flows. For every commodity i we require the *conservation constraints*:

$$(1) \quad \sum_{wv \in E} f_i(wv) - \sum_{vw \in E} f_i(vw) = 0 \quad \forall v \notin \{s_i, t_i\}.$$

We require also that $\sum_{vw \in E} f_i(vw) = d_i$ for $v = s_i$. We define the value of the total flow on edge vw to be

$f(vw) = \sum_i |f_i(vw)|$, and say that a multicommodity flow f in G is *feasible* if $f(vw) \leq u(vw)$ for all edges vw. (Note that $f(vw)$ is always non-negative.)

We consider the optimization version of this problem, called the *concurrent flow problem*, first defined by Shahrokhi and Matula [17]. In this problem the objective is to compute the maximum possible value z such that there is a feasible multicommodity flow with demands $z \cdot d_i$ for $i = 1, \ldots, k$. We call z the *throughput* of the multicommodity flow and use z^* to denote the optimal (maximum) value of z.

As in single-commodity flows, a solution to a multicommodity flow problem cannot be achieved if there is a cut $(U, \overline{U})$ of the graph for which the demand $D(U)$ that has to cross this cut exceeds $C(U)$, the capacity of this cut. This motivates the following definition of the *minimum cut* for multicommodity flow. This cut is sometimes referred to as the minimum-ratio cut to emphasize the difference from the single-commodity definition.

$$(2) \qquad \mathcal{S}^* = \min_{U \subseteq V} \frac{C(U)}{D(U)}$$

By the above, the min-cut value S^* is at least as big as the max-flow (or throughput) value z^*. Formally, the result in [11] means that for the uniform-demand case, $S^*/z^* = O(\log n)$; combination of the results in [10, 18, 9, 15] mean that, in general, $S^*/z^* = O(\log^2 k)$. We show that, for graphs not containing a fixed graph as a minor, the above bounds can be improved to $O(1)$ for the uniform demand case and $O(\log k)$ for the general demand case.

3 Decomposition and the Min-Cut Max-Flow Relations

We use a characterization of the optimum solution given by linear programming duality as follows. Let $\ell : E \to \mathbb{R}^+$ be a nonnegative *length* function. For nodes $v, w \in V$ let $dist_\ell(v, w)$ denote the length of the shortest path from v to w in G with respect to the length function ℓ. The following theorem is a special case of the linear programming duality theorem.

Theorem 3.1 $1/z^* \leq \sum_{i=1}^k dist_\ell(s_i, t_i) \cdot d_i$, subject to the constraints $\ell \geq 0$ and $\sum_{vw \in E} \ell(vw)u(vw) = 1$.

It follows that in order to bound S^*/z^*, it is sufficient to provide an upper bound on $\sum_{i=1}^k dist_\ell(s_i, t_i)d_i$ for any length function ℓ satisfying the above two constraints. For the sake of simplicity, we wish to work with integral length functions, and we therefore round the length function as follows. For each edge vw, we set $\hat{\ell}(vw) = \lceil \ell(vw)/C \rceil$, where C is the sum of capacities of all the edges in the network. It is clear that $dist_\ell(s_i, t_i) \leq (1/C) dist_{\hat{\ell}}(s_i, t_i)$, and that

$$(3) \qquad \sum_{vw \in E} \hat{\ell}(vw)u(vw) \leq 2C; \quad \hat{\ell} \geq 0.$$

A bound on $\sum_{vw \in E} dist_{\hat{\ell}}(s_i, t_i) \cdot d_i$ for legal integral functions $\hat{\ell}$ (i.e. those satisfying Equation 3), translates directly to a bound on the value of z^* as stated below.

Theorem 3.2 $1/z^* \leq (1/C) \sum_{i=1}^k dist_{\hat{\ell}}(s_i, t_i) d_i$ subject to the constraint $\sum_{vw \in E} \hat{\ell}(vw)u(vw) \leq 2C$.

The method that we (and, implicitly, the previous papers) use to prove our results is to find a graph decomposition where each component has small diameter and the capacity of edges leaving the regions is small.[1]

Definition 3.3 An (α, β)-decomposition of a graph G under distance function $\hat{\ell}$ is a partition of the nodes of G into regions $\{S_1, S_2, \ldots S_k\}$ such that $dist_{\hat{\ell}}(u, v) \leq \alpha$ for any nodes u, v in the same region S_i, and the sum of capacities of all edges with endpoints in different regions is at most β.

We can restate the results in [10] in terms of existence of a certain graph decomposition:

Lemma 3.4 Consider a concurrent multicommodity flow instance with network G. Suppose that for every positive integer q and every integral $\hat{\ell}$ satisfying (3) there exists a $(\sigma q, \sigma' C/q)$-decomposition. Then the instance satisfies $\mathcal{S}^*/z^* = O(\sigma\sigma' \log D)$.

Proof: (Sketch) Given a distance function $\hat{\ell}$, let $q = \lceil 2\sigma' C/(\mathcal{S}^* D) \rceil \leq 4\sigma' C/(\mathcal{S}^* D)$, and construct an $(4\sigma\sigma' C/(\mathcal{S}^* D), \mathcal{S}^* D/2)$-decomposition.

Define D' to be the sum of demands of commodities i for which s_i and t_i are not in the same region. For each region, the sum of capacities of edges leaving the region is at least $\mathcal{S}^*$ times the sum of demands leaving the region, by definition of $\mathcal{S}^*$. Summing over all regions, we get that the sum of capacities of edges between regions is at least $\mathcal{S}^*$ times D'. (Each demand and each capacity is counted twice.) The decomposition has the property that the sum of capacities of edges between regions is at most $\mathcal{S}^* D/2$, so it follows that $\mathcal{S}^* D' \leq \mathcal{S}^* D/2$.

Thus all but $D' \leq D/2$ of the demand is associated with commodities with both endpoints in the same region of the decomposition. For such commodity j, we have $d_{\hat{\ell}}(s_j, t_j) \leq 4\sigma\sigma' C/\mathcal{S}^* D$, and hence

[1] This is a slight generalization of the graph decomposition used in [1, 2, 3], where there is no notion of edge capacity.

these commodities contribute at most $\sigma\sigma' 4C/\mathcal{S}^*$ to $\sum_i dist_{\hat\ell}(s_i,t_i)d_i$.

By recursively applying the argument above for the remaining $D' \leq D/2$ of the demand, we can bound $\sum_i dist_{\hat\ell}(s_i,t_i)d_i$ by $O(\sigma\sigma'(C/\mathcal{S}^*)\log D)$. By Theorem 3.2 we get:

$$\frac{\mathcal{S}^*}{z^*} = O(\sigma\sigma' \log D).$$

∎

By using the fact that every graph has a $(O(q), O(C\log C/q))$-decomposition [3], we infer from this lemma that for the general-demand case $\mathcal{S}^*/z^* \leq O(\log C \log D)$. The following lemma can be proved using an argument similar to the one used to prove Lemma 3.4 together with the techniques of [11].

Lemma 3.5 Under the same hypothesis as in Lemma 3.4, if the concurrent flow instance is a uniform-demand instance then $\mathcal{S}^*/z^* = O(\sigma\sigma')$.

Combining the results in [15] that imply that $O(\log D)$ in the above bounds can be replaced with $O(\log k)$ together with Lemmas 3.5 and 3.4, and with the decomposition Theorem 4.6, proved in the next section, we get:

Theorem 3.6 For any concurrent flow problem in a graph not containing $K_{r,r}$ as a minor, $\mathcal{S}^*/z^* = O(r^3 \log k)$, where k is the number of commodities. For any uniform-demand concurrent flow in such a graph, $\mathcal{S}^*/z^* = O(r^3)$.

4 Proof of the main decomposition theorem

In this section we prove the main technical theorem, which implies that in any graph we can either exhibit a small clique as a minor or find a small subset of edges whose removal separates the graph into connected components such that any two nodes in the same connected component are relatively close in the original graph. We prove the theorem for a graph with unit length and unit capacity edges. We can apply this theorem to a graph with integral lengths $\hat\ell(\cdot)$ and capacities $\hat u(\cdot)$ by replacing each edge vw with $u(vw)$ parallel edges and then replacing each resulting edge with a path of length $\hat\ell(vw)$. This modified graph has $\sum_{vw} \hat\ell(vw)u(vw)$ edges. (Note that this sum is at most $2C$ in the concurrent flow application.)

Given a breadth-first search tree $\mathcal{T}$ of a graph G, the jth *level* of $\mathcal{T}$ is the set of nodes whose distance in G from the root of $\mathcal{T}$ is i. If a node belongs to the jth level of $\mathcal{T}$, we say the node's level is j. For a node v in $\mathcal{T}$, the *ancestor-path* of v in $\mathcal{T}$ is the path from v to the root of $\mathcal{T}$.

For a graph G_1 and a distance parameter δ, we describe a series of at most $r+1$ subgraphs of G_1. Assume G_1 is connected. Construct a breadth-first search tree $\mathcal{T}_1$ of G_1 rooted at an arbitrary node. If $\mathcal{T}_1$ has fewer than δ levels, the series stops. Otherwise, select a set of δ consecutive levels of $\mathcal{T}_1$, and let G_2 be a connected component of the graph induced on these levels. Repeat this procedure on G_2, obtaining a breadth-first search tree $\mathcal{T}_2$, selecting δ consecutive levels of $\mathcal{T}_2$, and choosing a connected component G_3 of the induced subgraph. Continue this construction, obtaining G_4, G_5, and so on, up to at most G_{r+1}.

The key property of the subgraphs that are produced in the above process is that each G_{i+1} is a connected component induced by the nodes in δ consecutive levels of $\mathcal{T}_i$.

Definition 4.1 Let H and G be graphs. Suppose that for each node v of H, G contains a connected subgraph $\mathcal{A}(v)$, and for every edge uv of H there is an edge $\mathcal{E}(uw)$ in G between $\mathcal{A}(u)$ and $\mathcal{A}(v)$. If the $\mathcal{A}(v)$'s are disjoint, we call $\cup_v \mathcal{A}(v)$ an H-minor of G. In this case, we refer to the $\mathcal{A}(v)$'s as *supernodes*, and to the $\mathcal{E}(uv)$'s as *superedges*.

We use $K_{r,s}$ to denote the complete bipartite graph with r nodes $g_1,\ldots,g_r$ on one side of the bipartition, and s nodes $b_1,b_2,\ldots,b_s$ on the other side. We refer to the r nodes g_i as *green* nodes and to the s nodes b_i as *blue* nodes. Correspondingly, in discussing a $K_{r,s}$-minor of G, we refer to the supernodes $\mathcal{A}(v)$ as green supernodes or blue supernodes, according to whether v is a green or blue node of $K_{r,s}$.

The following is the property of the graphs G_1, G_2, etc., which construction we have described above:

Theorem 4.2 If G_1 excludes $K_{r,r}$ as a minor, the last graph G_i in this series has the property that any two of its nodes are within distance $O(r^2 d)$ of each other in the original graph G_1.

If the process terminates before producing G_{r+1}, the claim is trivially true because in this case the breadth-first search tree of the last graph G_i has fewer than δ levels, so every pair of nodes have distance at most 2δ in G_i, and hence in G_1 as well. We therefore assume that the process continues until G_{r+1} is produced. In the remainder of the paper, when we refer to the "distance" between two nodes, we mean distance in G_1 measured in edges unless otherwise stated.

To prove Theorem 4.2, we assume that there are two nodes in G_{r+1} that are at distance at least $(r-1)(4(r+1)\delta+1)$ in G_1, and we show inductively that G_{r-s} contains $K_{r,s}$ as a minor. The induction shows that G_2

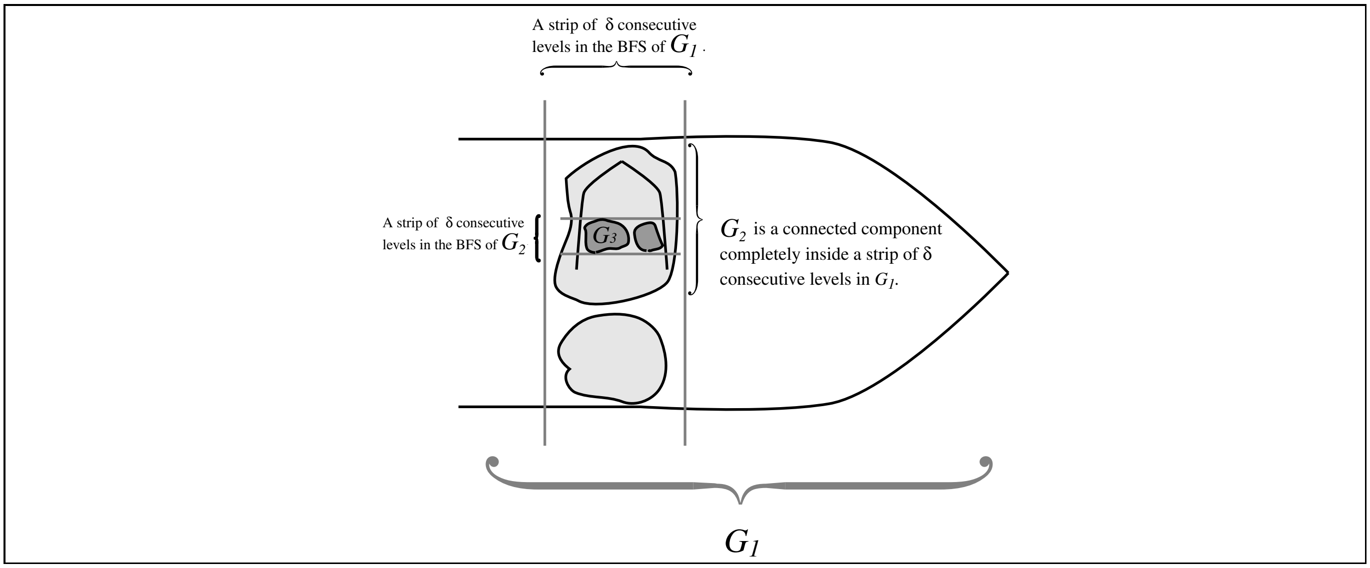

Figure 1: Construction of graphs G_2, G_3, etc.

contains $K_{r,r-1}$ as a minor; one additional step shows that G_1 contains $K_{r,r}$.

Before presenting the induction proof, we state several helpful lemmas and definitions.

Lemma 4.3 Assume there exist two nodes in G_{r+1} that are a distance of at least $(r-1)(4(r+1)\delta+1)$ from each other. Then there exist nodes $v_1, v_2, \ldots, v_r$ such that the distance between any two of them is at least $4(r+1)\delta+1$.

The lemma uses the fact that G_{r+1} is connected.

Definition 4.4 Consider a $K_{r,b}$-minor in G such that starting at each green supernode $\mathcal{A}(g)$ there is a path P_g. We say that the paths $\{P_g\}$ are *green tails* if each path P_g is disjoint from the other paths and from all supernodes except $\mathcal{A}(g)$. We will refer to P_g's ending node (outside of $\mathcal{A}(g)$) as the *tip of the tail*. Note that several nodes of P_g may belong to $\mathcal{A}(g)$, not just P_g's first node.

To aid in finding tails that are disjoint from previous superedges and supernodes, we use a span of δ consecutive levels as a *moat*. This is illustrated in Figure 2, where we see three graphs constructed in three consecutive steps of the decomposition process: G_i, G_{i+1}, and G_{i+2}. The usefulness of moats is captured in the following lemma.

Lemma 4.5 Let v and w be two nodes in G_{i+1} such that w lies on v's ancestor-path in T_i. Then the levels of v and w in $\mathcal{T}_{i+1}$ differ by less than δ.

Proof: Recall that all the nodes in G_{i+1} are in some δ consecutive levels of the $\mathcal{T}_i$ tree. Hence an ancestor-path from v to the root of $\mathcal{T}_i$ consists of two subpaths P_1P_2, where P_1 goes from v to the last node of the ancestor-path belonging to one of these δ levels, and P_2 continues to the root of $\mathcal{T}_i$. Since w is in G_{i+1}, it belongs to one of these levels, and hence lies in P_1.

By definition of an ancestor-path, each successive node of P_1 lies on a lower-numbered level, and hence its length is less than δ. Therefore the level in T_{i+1} of any node on P_1 is within $\delta-1$ of the level of v. ∎

We now commence the induction proof. We show by reverse induction on b that, for $b = r, r-1, \ldots, 1$ the following conditions hold.

1. There is a $K_{r,r-b}$-minor in G_{b+1} with green tails in G_b that are ancestor-paths in the tree $\mathcal{T}_b$ and that have length exactly 4δ.
2. Let h_j be the *middle node* of the j^{th} tail (*i.e.* the $2\delta+1^{st}$ node in the tail). Then each pair of middle nodes h_j are at distance more than $4b\delta$ from each other in G_1.

First we establish the basis of the induction. We assumed that G_{r+1} contained two nodes at distance at least $(r-1)(4(r+1)\delta+1)$ in G_1. Hence, by Lemma 4.3, G_{r+1} contains nodes $v_1, v_2, \ldots, v_r$ at distance at least $4(r+1)\delta+1$ from one another. We use these nodes to construct a $K_{r,0}$ minor. For every v_j, initialize the green supernode $\mathcal{A}(g_j)$ to contain the single node v_j. Note that $K_{r,0}$ has no edges so we need not construct any superedges. We do need to construct tails. For each v_j, the tail is the ancestor-path in $\mathcal{T}_r$ of length 4δ starting at v_j. The fact that the distance between any two nodes v_i, v_j is more than $4(r+1)\delta$ implies that these paths exist and are disjoint. For the j^{th} path, let h_j be

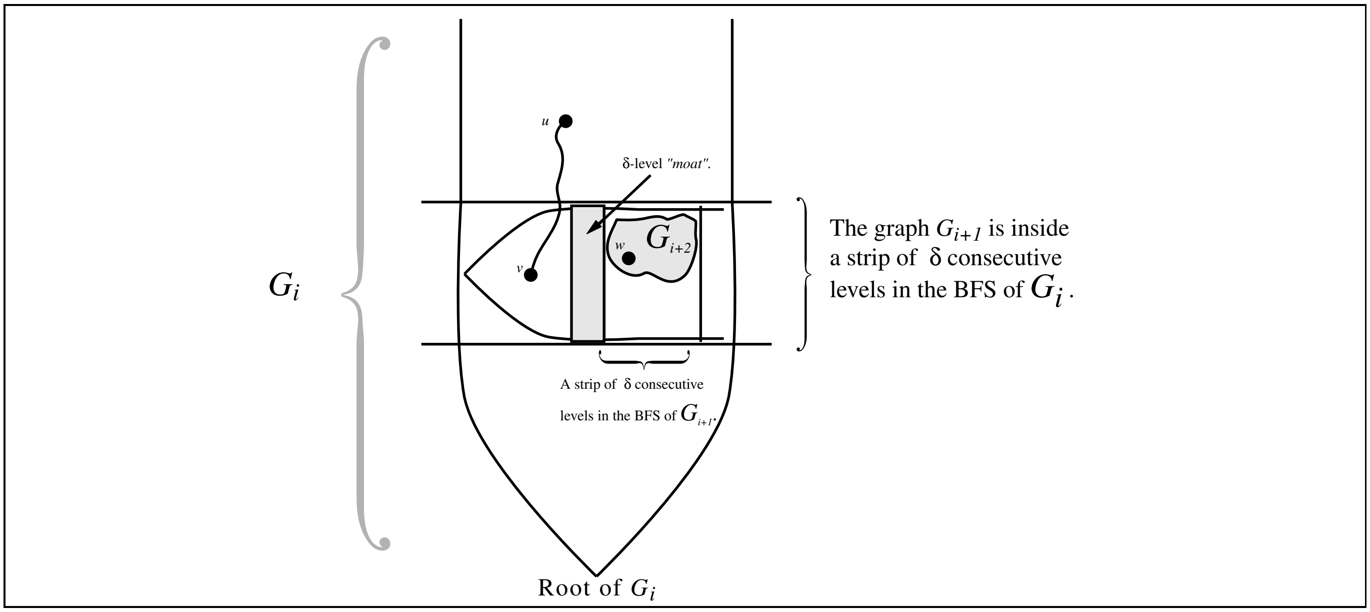

Figure 2: The "moat" argument.

the $2\delta + 1^{st}$ node on this path (the *middle node*). Note that h_j is at a distance exactly 2δ in the path from the first node. Since each pair v_i, v_j of distinct green supernodes are far apart, and since h_i is close to v_i and h_j is close to v_j, we conclude that the middle nodes are far apart. Thus the induction conditions are satisfied.

Now we address the inductive step. Assuming the conditions 1 and 2 hold for $b = i+1$, we show they hold for $b = i$, where $i \geq 1$. As illustrated in Figure 3, we have a $K_{r,r-i-1}$-minor in G_{i+2} with green tails Q_j of length 4δ that go up the tree T_{i+1}. The middle nodes h_j of the tails are at a distance more than $4(i+1)\delta$ from each other. Note that each middle node is separated in $\mathcal{T}_{i+1}$ from G_{i+2} by a moat of at least δ levels of $\mathcal{T}_{i+1}$. Moreover, the tips of the tails (denoted by $\hat{h}_1, \hat{h}_2$, etc. in the Figure), are separated from the middle nodes by a moat of at least δ levels.

The inductive step is depicted in Figure 4. Let P_j denote the ancestor-path in $\mathcal{T}_i$ of length 4δ from h_j, and let u_j be the $2\delta + 1^{st}$ node in P_j. The paths $\{P_j\}$ are disjoint since, by the inductive hypothesis, the distance between any two old middle nodes $h_j, h_{j'}$ is more than $4(i+1)\delta$. Similarly, any two new middle nodes $u_j, u_{j'}$ are a distance more than $4i\delta$ from each other.

We now construct a new blue supernode $\mathcal{A}(b_{i+1})$ as illustrated in Figure 4. Let $\mathcal{A}(b_{i+1})$ consist of the ancestor-paths in $\mathcal{T}_{i+1}$ from the tips of the old tails up to the root of $\mathcal{T}_{i+1}$. Clearly these paths form a connected subgraph since each contains the root.

For each j, augment the green supernode $\mathcal{A}(g_j)$ to obtain $\mathcal{A}'(g_j)$ by adding the nodes of the old tail Q_j up to but not including its tip. The last edge of Q_j, which goes from the augmented supernode $\mathcal{A}'(g_j)$ to the tip of the old tail, is a new superedge. Thus we have constructed a $K_{r,r-i}$-minor in G_{i+1}. We have also constructed ancestor-paths P_j of the right length from the green supernodes. It remains only to show that these paths are proper tails, *i.e.* that they do not intersect the supernodes (except that P_j may intersect $\mathcal{A}'(g_j)$).

We use the "moat" argument. By Lemma 4.5, for any node w of G_{i+1} that lies on P_j, the level of w in $\mathcal{T}_{i+1}$ cannot differ from the level of h_j by more than $\delta - 1$. Since the new blue supernode $\mathcal{A}(w_{i+1})$ consists only of nodes whose levels differ from the levels of the middle nodes $\{h_j\}$ by at least δ, the paths P_j do not intersect it. Similarly, all of the old (unaugmented) supernodes are contained in G_{i+2}, and hence cannot have any nodes in common with P_j.

We must also show that P_j does not intersect the augmented green supernodes. In particular, for each augmented supernode $A'(g_{j'})$ where $j' \neq j$, we must show that P_j does not intersect the added nodes $\mathcal{A}'(g_{j'}) - \mathcal{A}(g_{j'})$. The added nodes all belonged to the old tail $Q_{j'}$. Suppose that a node w belongs to both P_j and $Q_{j'}$. By Lemma 4.5, w is within δ levels of the middle node h_j in the tree $\mathcal{T}_{i+1}$. Hence w is within 2δ levels of $h_{j'}$. It follows that w is a distance at most 2δ from $h_{j'}$. Since every node of P_j that is in G_{i+1} is a distance at most δ from h_j, it follows that h_j and $h_{j'}$ are at distance at most 3δ apart, contradicting the induction hypothesis. We conclude that the new ancestor-paths P_j are true tails. This completes the induction step.

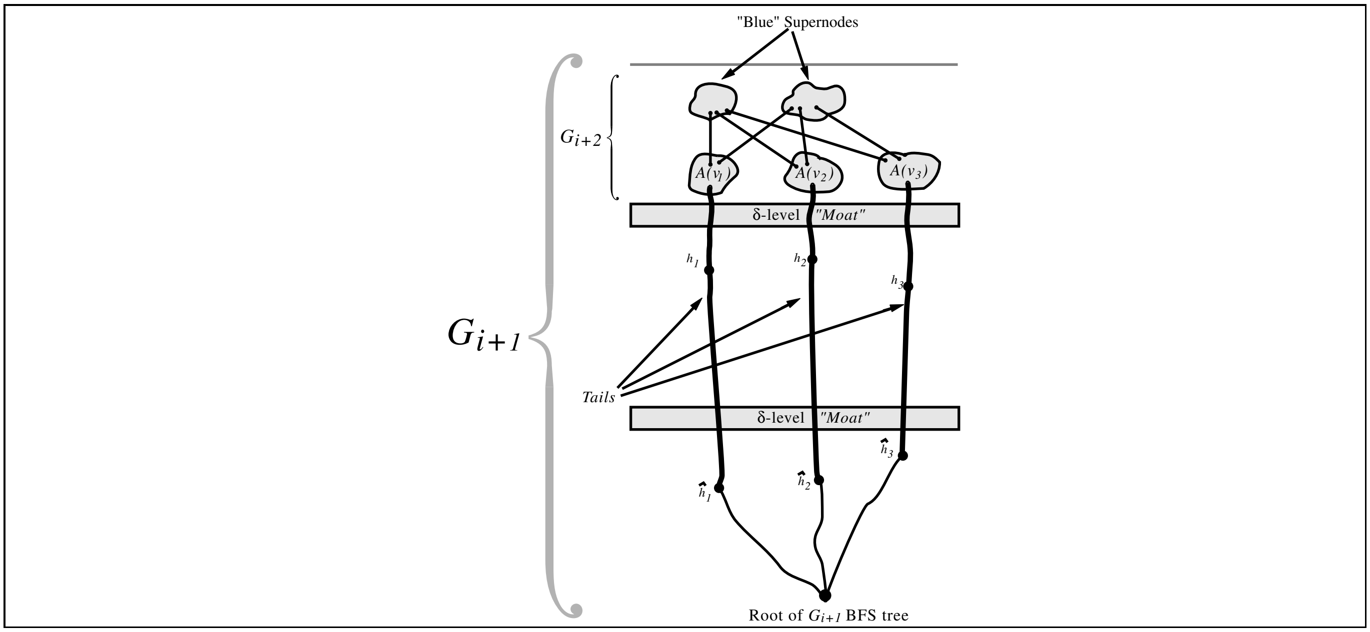

Figure 3: The inductive assumptions.

We have shown that G_2 contains a $K_{r,r-1}$-minor with green tails in G_1 such that the middle nodes on these tails are a distance more than 4δ from each other. Just as in the induction step, we can construct an r^{th} blue supernode in G_1 and augment the old green supernodes, thereby obtaining a $K_{r,r}$-minor. This completes the proof of Theorem 4.2. ∎

Following is a direct application of Theorem 4.2:

Theorem 4.6 Given a graph with m edges and parameter α, we can find (in polynomial time) either a $K_{r,r}$ minor, or a set of mr/α edges whose deletion will decompose G into connected components where the distance (in the original graph) between any two nodes in the same component is $O(r^2\alpha)$.

Proof: Given a breadth-first search tree of the graph, we divide the nodes into α classes. For $k = 0, 1, \ldots, \alpha - 1$, the k^{th} class consists of nodes at levels $k \pmod{\alpha}$. Similarly, the edges spanning different levels of the breadth-first search tree can be divided up into classes; each such edge is placed in a class corresponding whichever endpoint is nearer the root. These edge-classes are disjoint, so one of them contains at most a fraction $1/\alpha$ of the edges.

Remove the edges of such an edge-class, find breadth-first search trees of the components of the resulting graph, and continue for a total of r stages. In each stage, at most m/α edges are removed, for a total of rm/α edges. By Theorem 4.2, either a $K_{r,r}$ minor can be constructed, or each resulting connected component has the desired property. ∎

Acknowledgments

We are grateful to Éva Tardos for many helpful discussions.

References

[1] B. Awerbuch, B. Berger, L. Cowen, and D. Peleg. Fast constructions of sparse neighborhood covers. In *Proc. 10th Annual ACM Symposium on Principles of Distributed Computing*, 1992.

[2] B. Awerbuch, A. V. Goldberg, M. Luby, and S. A. Plotkin. Network Decomposition and Locality in Distributed Computation. In *Proc. 30th IEEE Annual Symposium on Foundations of Computer Science*, pages 364–369, 1989.

[3] B. Awerbuch and D. Peleg. Routing with polynomial communication-space trade-off. *SIAM J. Disc. Math*, 5(2):151–162, 1992.

[4] Baruch Awerbuch, Amotz Bar-Noy, Nati Linial, and David Peleg. Improved routing strategies with succinct tables. *J. Alg.*, 11:307–341, 1990.

[5] Baruch Awerbuch and David Peleg. Network synchronization with polylogarithmic overhead. In *Proc. 31st IEEE Annual Symposium on Foundations of Computer Science*, pages 514–522, 1990.

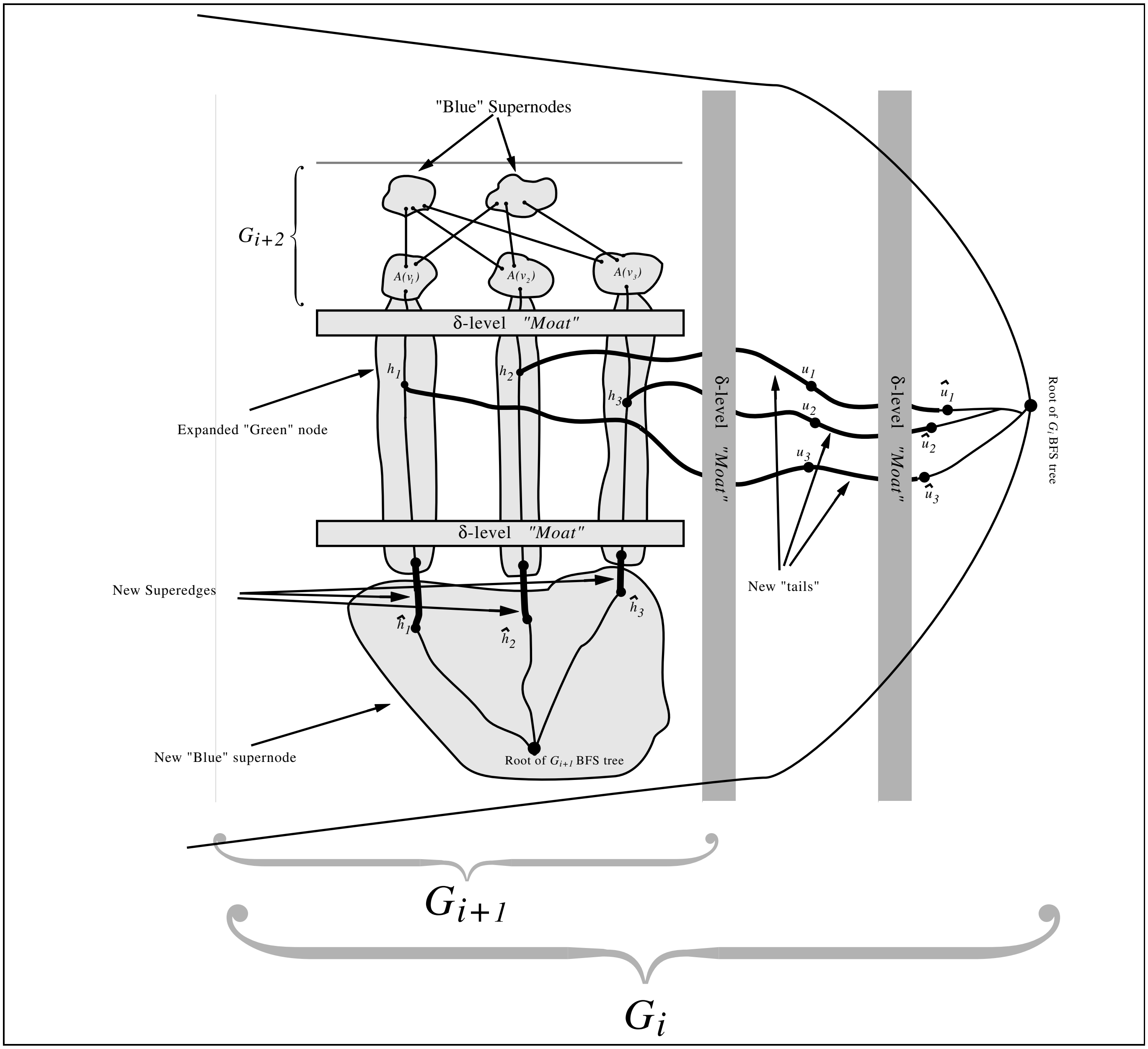

Figure 4: The inductive step.

[6] Sandeep N. Bhatt and Tom Leighton. A framework for solving VLSI layout problems. *J. Comp. and Syst. Sci.*, 28(2):300–343, April 1984.

[7] P. Elias, A. Feinstein, and C.E. Shannon. A note on the maximum flow through a network. *IRS Trans. Information Theory*, 2:117–119, 1956.

[8] L. R. Ford, Jr. and D. R. Fulkerson. Maximal Flow Through a Network. *Canadian Journal of Math.*, 8:399–404, 1956.

[9] N. Garg, V. V. Vazirani, and M. Yannakakis. Approximate max-flow min-(multi)cut theorems and their applications. In *Proc. 23th ACM Symposium on the Theory of Computing*, May 1993.

[10] P. N. Klein, S. Rao, A. Agrawal, and R. Ravi. An approximate max-flow min-cut relation for multicommodity flow, with applications. Submitted to *Combinatorica* (1992). Preliminary version appeared as "Approximation through multicommodity flow," In *Proc. 31th IEEE Annual Symposium on Foundations of Computer Science*, pages 726–727, 1990.

[11] T. Leighton and S. Rao. An approximate max-flow min-cut theorem for uniform multicommodity flow problems with applications to approximation algorithms. In *Proc. 29th IEEE Annual Symposium on Foundations of Computer Science*, pages 422–431, 1988.

[12] N. Linial and M. Saks. Decomposing graphs into regions of small diameter. In *Proc. 2nd ACM-SIAM Symposium on Discrete Algorithms*, pages 320–330. ACM/SIAM, January 1991.

[13] H. Okamura and P.D. Seymour. Multicommodity flows in planar graphs. *J. Combinatorial Theory (B)*, 31:75–81, 1981.

[14] D. Peleg and E. Upfal. A tradeoff between size and efficiency for routing tables. *J. ACM*, 36:510–530, 1989.

[15] S. Plotkin and É. Tardos. Improved bounds on the max-flow min-cut ratio for multicommodity flows. In *Proc. 23th ACM Symposium on the Theory of Computing*, May 1993.

[16] P.D. Seymour. Matroids and multicommodity flows. *European Journal of Combinatorics*, 2:257–290, 1981.

[17] F. Shahrokhi and D. Matula. The maximum concurrent flow problem. *J. Assoc. Comput. Mach.*, 37:318–334, 1990.

[18] S. Tragoudas. *VLSI partitioning approximation algorithms based on multicommodity flow and other techniques.* PhD thesis, University of Texas at Dallas, 1991.