Multicommodity Flow, Well-linked Terminals, and Routing Problems

Chandra Chekuri*
Lucent Bell Labs
600 Mountain Avenue
Murray Hill, NJ 07974.
chekuri@research.bell-labs.com

Sanjeev Khanna†
Dept. of Comp. & Inf. Sci.
University of Pennsylvania
Philadelphia, PA 19104
sanjeev@cis.upenn.edu

F. Bruce Shepherd*
Lucent Bell Labs
600 Mountain Avenue
Murray Hill, NJ 07974.
bshep@research.bell-labs.com

ABSTRACT

We study multicommodity routing problems in both *edge* and *node* capacitated undirected graphs. The input to each problem is a capacitated graph $G = (V, E)$ and a set $\mathcal{T}$ of node pairs. In the simplest setting, the goal is to route a unit of flow for as many pairs as possible subject to the edge (node) capacity constraints. If the flow for a routed pair is required to be along a single path, it is the well-studied disjoint paths problem. If we allow fractional routings of the flow, it is known as the all-or-nothing flow problem. The nodes in $\mathcal{T}$ are referred to as terminals.

In recent work [8, 9], the authors obtained the first poly-logarithmic approximation algorithms for some *edge* routing problems. A key idea in these algorithms is to decompose an instance into a collection of instances in which the terminals are *well-linked*. Informally speaking, a set of nodes is well-linked in a graph if it does not have small separators. A decomposition into well-linked instances was previously achieved in [8] via Räcke's hierarchical graph decomposition for oblivious routing [32]. In this paper, we design a simple new decomposition algorithm that is based on computing sparse cuts in a graph. Our new algorithm improves the earlier results for edge routing problems. Another important advantage of the algorithm is that it also applies to *node-capacitated* problems. We note that for oblivious routing with node capacities, an $\Omega(\sqrt{n})$ lower bound is known on the congestion [18], and hence the oblivious routing approach cannot yield poly-logarithmic bounds for well-linked decompositions. Using the new decomposition, we obtain a poly-logarithmic approximation for the node capacitated all-or-nothing flow problem in general graphs and node-disjoint path problem in planar graphs with $O(1)$ congestion. We also show that the flow-cut gap for product multicommodity flows in node capacitated planar graphs is $O(1)$, improving upon the $O(\log n)$ bound from [28].

*Supported in part by an ONR basic research grant N00014-04-M-0042 to Lucent Bell Labs.

†Supported in part by an Alfred P. Sloan Research Fellowship and by an NSF Career Award CCR-0093117.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems

General Terms

Algorithms, Theory.

Keywords

All-or-Nothing Flow, Disjoint Paths, Flow-Cut Gaps, Multicommodity Flow, Network Routing.

1. INTRODUCTION

In this paper, we consider several fundamental multicommodity routing problems in both *edge* and *node* capacitated undirected graphs. In each of these problems we are given a capacitated graph $G = (V, E)$ with n nodes and m edges. We are also supplied with a set of k node-pairs $s_1t_1, \ldots, s_kt_k$, each with a positive integer demand d_i and a positive weight w_i. We call the s_i's and t_i's *terminals* and we assume without loss of generality that they are distinct. The objective is to find a largest (or maximum w-weight) *routable* subset S of $\{1, 2, \ldots, k\}$, that is, the pairs in S can be simultaneously satisfied obeying the capacity of the graph. We call such a set, *routable*. What distinguishes the various problems are the different constraints (edge capacities, integer routings) we place on the feasibility of a routing.

The most fundamental problem is the *disjoint path problem*. Here each pair has $d_i = 1$ and a set S is routable if G contains edge (node) disjoint paths P_i, $i \in S$, such that P_i joins s_i and t_i. When pairs have different integer demands d_i and each pair has to be routed on a *single* path we have the *unsplittable flow problem*: a set S is routable if G contains paths P_i, $i \in S$, such that P_i joins s_i and t_i and d_i flow can be routed on each P_i without violating the edge (node) capacities. In the *all-or-nothing flow problem*, a set S is routable if there exists a multicommodity flow in G such that, for each $i \in S$, d_i flow is routed for pair s_it_i.

The above problems are strongly NP-hard even in planar graphs [20, 13, 30, 27]. For each of them, the strongest upper bound comes from the natural multicommodity flow relaxation. We focus on the integrality gap of this relaxation and the resulting approximation algorithms. We restrict attention to the unit demand version of the problems (that is $d_i = 1$ for $1 \le i \le k$), namely, the disjoint path problem and the unit demand all-or-nothing flow problem. A grouping and scaling technique of Kolliopoulos and Stein [25, 10] allow us to translate, within constant factors, integrality gaps obtained for unit demand instances to arbitrary demand instances.

The best approximation ratio known for disjoint path problems is polynomial in n and m: for edge-disjoint paths (EDP) the ratio is $O(\min\{n^{2/3}, \sqrt{m}\})$ [7] and for node-disjoint paths (NDP) the ratio is $O(\sqrt{n})$ [24, 25]. These results also establish the same upper bounds on the integrality gap of the LP. The weakness of these bounds is balanced by known integrality gaps for the LP; both EDP and NDP have an $\Omega(\sqrt{n})$ integrality gap even in planar graphs [16]. Furthermore, the unsplittable node flow problem with $d_i \in \{1, 2\}$ is NP-hard to approximate to within an $\Omega(n^{1/2-\epsilon})$ factor [17]. Recently, Andrews and Zhang obtained an $\Omega((\log n)^{1/3-\epsilon})$-hardness of approximation for edge disjoint paths and all-or-nothing flow [4], improving upon the APX-hardness known before [16]. Chuzhoy and Khanna have further improved this to an $\Omega((\log n)^{1/2-\epsilon})$-hardness for both edge disjoint paths and all-or-nothing flow [11]. These recent hardness results also hold for the corresponding node problems. Motivated by these negative results, the following relaxed version of the disjoint path problem is of natural interest: what is the integrality gap of the LP if *two* pairs can use an edge (node)? More generally, what is the integrality gap of the LP if $O(1)$ pairs can use an edge (node)? The maximum factor by which the capacity of any edge (node) is violated by the routing is referred to as the *congestion* of the routing. Raghavan and Thompson showed via randomized rounding [33] that an $O(1)$ approximation can be achieved even in directed graphs if $O(\log n/(\log \log n))$ congestion is allowed. With congestion B, an approximation ratio of $O(n^{1/B})$ is achievable [37]. However, until recently, no super constant integrality gap was known for the LP even with congestion restricted to 2. The very recent work of [11] shows that the integrality gap of the LP is at least $(\log n)^{\Omega(1/c)}$ for any constant congestion c. Moreover, [11] also shows that the integrality gap of the LP is superconstant even when the allowed congestion is superconstant. We note here that much better approximation ratios are known for special classes of graphs such as trees, bounded degree expanders, grids and grid-like graphs [16, 26, 22, 23]. In a different direction, if restrictions are placed on both the graph and the pairs, known integral and half-integral results give good approximations, see [15, 36] for surveys of such results.

In a recent work [9], the authors obtained a poly-logarithmic upper bound on the integrality gap for the edge disjoint path problem in *planar* graphs with congestion 2. A key tool used in [9] is a decomposition of the given instance into *well-linked* instances. The notion of well-linked instances can be formulated either based on a cut or flow requirement. Roughly speaking, an instance is cut-well-linked if the terminal set $X = \{s_1, t_1, s_2, t_2, \ldots, s_k, t_k\}$ cannot be separated by small edge cuts. Note that this is a surprisingly strong property since the given instance is interested in only routing the k specified pairs, while well-linkedness of the terminal set requires satisfying the cut condition for routability of *any* matching on X. We note that the notion of a well-linked set (based on the cut condition) is closely related to treewidth and plays an important role in the Graph Minors Project of Robertson and Seymour. Decomposition into well-linked instances to compute approximate solutions to routing problems was first used in the work of the authors on the (edge capacitated) all-or-nothing flow problem [8]; it was used to obtain a poly-logarithmic integrality gap and approximation. The all-or-nothing flow problem was introduced in [10]. The decomposition used in [8] relied on the powerful hierarchical graph decomposition that Räcke developed for oblivious routing [32] and subsequent improvement in the parameters of the decomposition in [19].

In this paper we make several new contributions to the above line of work, in the process clarifying and improving existing results, as well as obtaining new results for node capacitated problems that could not be obtained by ideas in previous work. Before we explain our results in detail, we briefly comment on the difficulty of working with node capacitated problems. First, as shown in [18], oblivious routing for node-capacitated graphs requires a congestion of $\Omega(\sqrt{n})$ even in planar graphs. This rules out a poly-logarithmic bound for a well-linked decomposition if we use the approach from [8]. Further, as observed also in [18], the maximum degree plays an important role in node problems. In the context of routing problems this can be seen by the fact that for edge problems we can transform the given graph to a bounded degree graph (even preserving planarity), while there is no such reduction for node capacitated problems. We obtain well-linked decompositions for node capacitated problems using a different approach. However the ratios we obtain are better for the edge problems when compared to their node counterparts mainly because of the degree reducing transformation mentioned above.

Results: Our main result is a new algorithm that produces a "well-linked decomposition" from a given multicommodity flow in edge-capacitated or node-capacitated undirected graphs. We also make the notion of a well-linked decomposition precise, and in particular, we introduce a weighted (or fractional) version of well-linkedness that facilitates the transformation of the original multicommodity flow. Producing a well-linked set from a fractionally well-linked set is less straightforward in the node case. We also distinguish between *flow-well-linked* and *cut-well-linked* decompositions. These two notions are approximately equivalent on any graph G, separated only by the product multicommodity max flow-min cut gap in G. However, one or the other is preferable based on the application in question. The statement of the decomposition theorem requires technical definitions that we give later.

Using our decomposition algorithm and several other ideas we obtain the following results. In the following we use k to refer to the number of pairs in the given routing problem instance. All approximation bounds stated below are derived by measuring our performance against an optimal LP solution with no congestion. We note that the value of an optimum solution to the flow LP (for either edge or node problems) increases only by a factor of B when a congestion of B is allowed. It then follows that whenever we obtain an α-approximation using a congestion of B, it is also an $(B\alpha)$-approximation with respect to the optimal solution that is allowed a congestion of B. In this paper B will be a small constant and hence we do not explicitly state its impact on the approximation ratio. When working with basic solutions to the LP, we can restrict our attention to the case when $k \le m$ and hence in the bounds below, $\log k$ is $O(\log n)$.

- For the edge capacitated all-or-nothing flow problem, we obtain an $O(\log^2 k)$ approximation in general graphs and an $O(\log k)$ approximation in planar graphs. These improve upon the earlier ratios of $O(\log^3 n \log\log n)$ and $O(\log^2 n \log\log n)$ respectively, obtained in [8].
- For the edge disjoint path problem in planar graphs, we obtain an $O(\log k)$ approximation with congestion 2. This improves the $O(\log^2 n \log\log n)$ approximation from [9].
- For the node-capacitated all-or-nothing flow problem, we obtain a first poly-logarithmic approximation with congestion $(1+\epsilon)$ for any fixed $\epsilon > 0$. We obtain a ratio of $O(\log^4 k \log n)$ for general graphs and $O(\log^2 k \log n)$ for planar graphs.
- For the node disjoint path problem in planar graphs, we obtain an $O(\log^2 k \log n)$ approximation with congestion 4.
- We show that the maxflow-mincut gap for product multicommodity flows in node capacitated planar graphs is $O(1)$. Though an analogous result for the edge capacitated problem on planar graphs has been known for some time [21], only an $O(\log n)$ bound was known for the node problem [28].

Our result on the flow-cut gap in node capacitated planar graphs gives an alternate $O(1)$ approximation, albeit with a much larger constant, for the minimum quotient node cut problem in planar graphs [3]. Although our proof is similar in outline to that for the edge capacitated problem that we presented in [9], there is a crucial technical difference. In the edge capacitated problem, we can use parallel edges to reduce to unit capacity instances. However, in the node capacitated case, there is no such reduction which preserves planarity. This was the technical difficulty in generalizing the $O(1)$ gap proof of Klein, Plotkin and Rao [21] to the node problem (see [3] for some additional discussion). Thus we need to work directly with node capacities. Our proof for the edge problem in [9] differs from the earlier maxflow-mincut gap proofs [28, 29, 6, 21] that rely on the metric induced by the dual of the flow problem. Instead of producing a cut from the dual of an optimum flow, our proof produces a flow from a given cut condition using the existence of large routing structures. This proof has sufficient flexibility to generalize to non uniform node capacities without first reducing to the unit capacity case.

1.1 Preliminaries

Input Instances: We work with a given capacitated graph $G = (V, E, c)$ where c is an integer capacity function on edges (nodes) if we are considering an edge (node) routing problem. We let $n = |V|$ and $m = |E|$. Throughout, for any graph G and proper node subset $S \subseteq V$, we denote by $\delta_G(S)$, or simply $\delta(S)$ if G is clear from the context, the set of edges of G with exactly one endpoint in S. We denote by $N(S)$ the set of nodes in $V \setminus S$ that are adjacent to some node in S. An instance of a routing problem consists of a capacitated graph $G = (V, E, c)$ and a collections of pairs $\mathcal{T} = \{s_1t_1, s_2t_2, \ldots, s_kt_k\}$. Nodes in $\mathcal{T}$ are referred to as terminals. We replace each occurrence of a terminal v that participates in p pairs by new terminals $v_1, v_2, ..., v_p$ that are each connected to v by a unit capacity edge. There is a clear 1-1 correspondence between routings in the original and the modified instances. Thus, without loss of generality, we can assume that each terminal has degree 1 and the pairs in $\mathcal{T}$ form a *perfect matching* on the terminals. As we will see shortly, we can restrict attention to instances in which all capacities are polynomially bounded; this in turn allows us to reduce any routing problem to a unit capacity instance.

Multicommodity Flow LP Formulation: For the given instance with $\mathcal{T} = \{s_1t_1, s_2t_2, \ldots, s_kt_k\}$, we let $\mathcal{P}_i$ denote the paths joining s_i and t_i in G and let $\mathcal{P} = \cup_i \mathcal{P}_i$. The following multicommodity flow relaxation is used to obtain an upper bound on the number of pairs from $\mathcal{T}$ that can be routed in G. For each path $P \in \mathcal{P}$ we have a variable $f(P)$ which is the amount of flow sent on P. We let x_i denote the total flow sent on paths for pair i. We let $\bar{f}$ denote the flow vector with a component for each path P, and we denote by $|\bar{f}|$ the value $\sum_i x_i$. Then the LP relaxation for edge problems is the following.

$$\begin{aligned}
&\max \sum_{i=1}^{k} x_i \quad \text{s.t} \\
&x_i - \sum_{P \in \mathcal{P}_i} f(P) \;=\; 0 \qquad 1 \le i \le k \\
&\sum_{P: e \in P} f(P) \;\le\; c(e) \quad \forall e \in E \\
&x_i, f(P) \;\in\; [0,1] \quad 1 \le i \le k, P \in \mathcal{P}
\end{aligned}$$

For node problems, we replace the capacity constraint by $\sum_{P: v \in P} f(P) \le c(v) \quad \forall v \in V$. We let OPT denote the optimum solution value to the above relaxation. Call a path P *fractionally routed* if $f(P) \in (0,1)$, otherwise $f(P) \in \{0,1\}$ and P is *integrally routed.* If the total flow routed on integrally routed paths is more than OPT/2, then we already obtain a 2-approximation for all versions of the routing problems considered here. Thus the interesting and difficult case is when the fractionally routed paths contribute almost all the value of OPT. From standard polyhedral theory the number of fractionally routed paths in a basic solution to the edge (node) LP above is at most m (n). Therefore we can assume that $c(e) \le m$ for all edges and $c(v) \le n$ for all nodes. By making parallel copies of edges, in the following, we assume that G has only unit capacity edges. For node problems we can transform G and assume that the node capacities are 1, however the transformation does not preserve planarity. Finally, as shown by the lemma below, we can work with multicommodity flows where the flow on each path is $\Omega(1/\log n)$ large.

Lemma 1.1. *Let $\bar{f}$ be a feasible solution to a multicommodity flow instance. Then there is a solution $\bar{f}'$ for the same instance such that (i) $|\bar{f}'| = \Omega(|\bar{f}|)$, and (ii) there is a flow decomposition for $\bar{f}'$ such that the flow on each flow-path is at least $\frac{c}{\log n}$ for some absolute constant c. By duplicating terminals we can assume that the flow for each pair is $\Theta(1/\log n)$ and that the number of pairs is $\Theta(|f| \log n)$.*

Proof Sketch: Consider any pair s_it_i with non-zero flow f_i. Let $\ell = \lceil f_i \log n \rceil$. We create ℓ new terminal pairs $s_i^1t_i^1, s_i^2t_i^2, \ldots, s_i^\ell t_i^\ell$ by connecting $s_1^1, \ldots, s_i^\ell$ to s_i and $t_1^1, \ldots, t_i^\ell$ to t_i via single edges. We set the flow for $s_i^j t_i^j$ to be $1/\log n$ if $j < \ell$ and for $s_i^\ell t_i^\ell$ we set it to $f_i - (\ell-1)/\log n$. Note that this flow is feasible and has exactly the same value as the original flow. This ensures that the flow for each pair is in $[0, 1/\log n]$. Further, we can recover a feasible routing of the same value for the original instance from a feasible routing for the modified instance.

Now we do a flow decomposition for the new pairs and do randomized rounding such that the flow $f(P)$ on a flow path P is rounded up to $1/\log n$ with probability $f(P)\log n$ and to 0 otherwise. In expectation the total flow remains the same. By standard Chernoff-Hoeffding bounds we can argue that, for a sufficiently large but fixed constant c, the probability that the flow on any edge exceeds c, is small. Thus we can scale down the flow on each path to $1/(c \log n)$ and not violate any capacities. Thus, the total flow is $\Omega(|f|)$ and the flow for each pair is precisely $1/(c\log n)$ and uses a single path. Hence the number of pairs is $\Theta(|f|\log n)$. ■

Product Multicommodity Flow, Concurrent Flow and Sparse Cuts: A multicommodity flow instance in a capacitated graph $G = (V, E, c)$ is given by a demand vector d that assigns to each pair uv of nodes a non-negative value. A *product multicommodity flow* is a special case where d is induced by a weight function $w : V \to \mathcal{R}^+$ on the nodes of V: for uv, $d(uv) = w(u)w(v)$. Given a multicommodity flow instance, there are two quantities of interest to us. The maximum *concurrent flow* for the given instance is the largest λ such that λd can be feasibly routed in G. The *sparsity* of a cut is the ratio of the capacity of the cut to the demand separated by the cut, and a sparsest cut is one that achieves the minimum sparsity among all cuts. It is easy to check that the minimum sparsity is an upper bound on the maximum concurrent flow, however the former could be strictly larger. The *maxflow-mincut gap* is the worst case ratio between these two quantities. For product multicommodity flows, this gap is at most $O(\log k)$ in general graphs for both the edge and node cases where k is the number of commodities (non-zero $d(uv)$ entries) [28]. In planar graphs and graphs excluding fixed minors, for the edge case, the gap is $O(1)$ [21]. Given a weight function on the nodes $w : V \to \mathcal{R}^+$, the minimum *quotient cut* is a cut that minimizes the ratio of the capacity of the cut to the node weight of the smaller side in the partition produced by the cut. The sparsity of the product multicommodity flow instance induced by w is within a factor of 2 of the minimum quotient cut value for w. Approximation algorithms for computing sparsest cuts and minimum quotient cuts can be obtained via flow-cut gaps [28, 21, 29, 6] or via other methods that can give better bounds than the flow-cut gap [5, 31, 3].

1.2 Crossbars and Grid Minors

A graph H is an *h-crossbar* with congestion c if it contains a subset $I \subseteq V(H)$ of size h, called the *interface*, such that any matching on I can be integrally routed in H with congestion c. We will be specifically interested in crossbars defined by grid graphs. An *$r \times c$ grid* is a graph $G_{r,c}$ with rc nodes $\{(i,j) : i = 1, 2, \ldots r, j = 1, 2, \ldots c\}$ and with edge set $(\cup_{i=1}^r R_i) \cup (\cup_{j=1}^c C_j)$, where R_i and C_j denote the *row i edges* and *column j edges* respectively. These latter sets are defined as follows: $R_i = \{(i,j)(i,j+1) : j = 1, 2, \ldots, c-1\}$ and $C_j = \{(i,j)(i+1,j) : i = 1, 2, \ldots, r-1\}$. We call the nodes in row 1 the *interface* of the grid. If $r = c$ then we also call $G_{r,c}$ an *r-grid* or grid of *order* r. It can be verified that a h-grid with row 1 as the interface is an h-crossbar with congestion 1 for edge problems and with congestion 2 for node problems.

Our routing algorithms use *grid minors* as opposed to grids. A *minor* in a graph G is a pair (H, Φ) where H is a graph, and Φ is a mapping from $V(H)$ to subsets of $V(G)$, such that (i) $\Phi(u) \cap \Phi(v) = \emptyset$ if $u \neq v$, (ii) $\Phi(u)$ induces a connected subgraph in G for each u, and (iii) $(u,v) \in E(H)$ only if there is an edge between nodes of $\Phi(u)$ and $\Phi(v)$ in G. Due to the correspondence above, we may speak of edges in H as being edges of G as well; if the context is clear, we denote by H_v the set of nodes $\Phi(v)$. If the graph H above is an h-grid, then we refer to it as an *h-grid-minor*. It can be verified that if G contains an h-grid-minor then it contains an h-crossbar with congestion 2 for both edge and node problems.

Finally, for an edge problem, we can transform in polynomial-time, any given instance with unit capacity edges into an instance with maximum degree 4. This reduction also preserves planarity. The transformation is done by replacing each node v of degree $d(v)$ by a grid of order $h := d(v) + 2$. This is referred to as the *grid expansion* and we refer the reader, for instance, to [9] for details.

1.3 Flow and Cut Well-linked Sets

We work with two notions of *well-linked sets*, one based on flows and another based on cuts. In our algorithms, we need to work with fractional versions of these concepts; we define these terms for both edge and node problems.

Given a non-negative weight function $\vec{\pi} : X \to \mathcal{R}^+$ on a set X of nodes, we say that X is *$\vec{\pi}$-flow-linked* in G if there is a feasible multicommodity flow (node or edge-capacitated accordingly) for the problem with demand $\vec{\pi}(u)\vec{\pi}(v)/\vec{\pi}(X)$ between every unordered pair of terminals $u, v \in X$. Note that this is a product multicommodity flow with $w(u) = \vec{\pi}(u)/\sqrt{\vec{\pi}(X)}$. For an edge problem, a set X is *$\vec{\pi}$-cut-linked* in G if $|\delta(S)| \geq \vec{\pi}(S \cap X)$ for all S such that $\vec{\pi}(S \cap X) \leq \vec{\pi}(X)/2$. For a node problem, a set X is *$\vec{\pi}$-node-cut-linked* (or $\vec{\pi}$-cut-linked if the context is clear) in G if $|N(S)| \geq \vec{\pi}(S \cap X)$ for all S such that $\vec{\pi}(S \cap X) \leq \vec{\pi}(X)/2$, where $N(S)$ is the set of nodes in $V \setminus S$ that are adjacent to a node in S. When the context is clear, we do not explicitly include a reference to the edge or node problems. It can be checked easily that if a set X is $\vec{\pi}$-flow-linked in G, then it is $\vec{\pi}/2$-cut-linked. The converse relationship is weaker; if X is $\vec{\pi}$-cut-linked, then it is $\vec{\pi}/\beta$-flow-linked where β is the worst-case mincut-maxflow gap on product multicommodity flow instances on G.

It is sometimes useful to work with "uniformly" linked instances. To this end, if the function $\vec{\pi}(u) = \alpha$ for all $u \in X$, we say that X is *α-flow(or cut)-linked*. Note that 1-cut-linked sets have been well-studied and are also called *well-linked*. As a conversational device, we abuse terminology and refer to a set of terminals as being *"well-linked"* if it is c-flow-linked or c-cut-linked for some global constant c (which we do not specify ahead of time).

The importance of such linked instances is that they form "fractional" crossbars with low congestion for their terminals. This is captured by the following.

PROPOSITION 1.2. *Given a subset $X \subseteq V$ of nodes that is α-flow-linked in G, any matching on X can be fractionally routed with congestion $2/\alpha$.*

Throughout the rest of the paper, when working with well-linked sets, we will implicitly assume that the weight of any node is at most 1.

2. MULTICOMMODITY FLOWS TO SETS OF WELL-LINKED TERMINALS

Using Proposition 1.2, if a set of terminals X is α-flow-linked, then $\Omega(\alpha|X|)$ flow can be routed for any matching on X. The goal of this section is to present algorithms that establish a converse relationship: given an *arbitrary* multicommodity flow vector $\bar{f}$ for a routing problem, we can recover a collection of well-linked terminal sets of weight $\Omega(|\bar{f}|/\text{polylog}(n))$ from them.

The transformation from multicommodity flows to sets of well-linked terminals proceeds in two steps. The first step of the transformation is to partition the input graph into a collection of node-disjoint subgraphs such that each subgraph contains a $\vec{\pi}$-flow-linked or a $\vec{\pi}$-cut-linked set of terminals. The second step of the transformation uses a clustering procedure that maps a $\vec{\pi}$-flow-linked set of terminals to a (smaller) subset of terminals that is 1/2-flow-linked. In what follows, we first present algorithms for converting arbitrary flows to $\vec{\pi}$-flow-linked terminals, and then show how to recover $\Omega(1)$-flow-linked terminals from them.

2.1 Multicommodity Flows to $\vec{\pi}$-flow-linked Terminals

We first present a transformation for creating a collection of flow-linked sets. We discuss another similar decomposition for cut-linked sets at the end of this section.

THEOREM 2.1. *Let* OPT *be a solution to the LP for a given instance $(G, \mathcal{T})$ of EDP (NDP) in a graph G. Let $\beta(G) \geq 1$ be an upper bound on the worst case mincut-maxflow gap for edge (node) product multicommodity flow problems in G. Then there is a partition of G into node-disjoint induced subgraphs $G_1, G_2, \ldots, G_\ell$ and weight function $\vec{\pi} : V(G_i) \to \mathcal{R}^+$ with the following properties. Let $\mathcal{T}_i$ be the induced pairs of $\mathcal{T}$ in G_i and let X_i be the set of terminals of $\mathcal{T}_i$.*

1. *$\vec{\pi}_i(u) = \vec{\pi}_i(v)$ for $uv \in \mathcal{T}_i$.*
2. *X_i is $\vec{\pi}_i$-flow-linked in G_i.*
3. *$\sum_{i=1}^{\ell} \vec{\pi}_i(X_i) = \Omega(\text{OPT}/(\beta(G)\log \text{OPT}))$.*

Moreover, such a partition is computable in polynomial time if there is a polynomial time algorithm for computing a cut of value $\beta(G)$ times the maxflow.

PROOF. We give a proof only for the edge case. A similar proof works for the node case by considering node cuts instead of edge cuts. Recall that all source and sink nodes are distinct, and let the set of all source and sink nodes be denoted by X. We start with a multicommodity flow $\bar{f}$ for X in G with total flow value $\gamma(G) = \text{OPT}$. We view the flow for each pair as being decomposed into flow paths. Given a node-induced subgraph $H = (V(H), E(H))$ of G, we let $\gamma(H)$ be the total flow induced in H by the original flow $\bar{f}$. In other words $\gamma(H)$ counts flow only on flow paths from the original flow path decomposition that are completely contained in H. We stress that the initial flow path decomposition determines $\gamma(H)$ for all subgraphs H. Given a node $u \in X \cap V(H)$ we let $\gamma(u, H)$ denote the flow in H for u. By definition, $\gamma(H) = \frac{1}{2}\sum_{u \in V(H)} \gamma(u, H)$.

We use β as a shortcut for $\beta(G)$. We now describe a recursive partitioning scheme for constructing the sets X_i and graphs G_i. Let OPT denote the value of starting flow and hence it is fixed in the procedure below.

Decomposition Algorithm:

Input: subgraph H.

Output: node-induced subgraph partition of H into $H_1, H_2, \ldots$ with associated weight functions $\vec{\pi}_1, \vec{\pi}_2, \ldots$.

1. Suppose $0 < \gamma(H) \leq \beta \log \text{OPT}$. Let uv be some pair with positive flow in H. Define $\vec{\pi}$ on $V(H)$ by $\vec{\pi}(u) = \vec{\pi}(v) = 1$ and $\vec{\pi}(y) = 0$ for $y \neq u, v$. Stop and output H along with $\vec{\pi}$.
2. Suppose that $\gamma(H) > \beta \log \text{OPT}$. Construct an instance of a product multicommodity flow problem on G with marginals $w(u) = \gamma(u, H)/\sqrt{\gamma(H)}$ for $u \in V$. Let λ be the maximum concurrent flow for this instance.
   (a) If $\lambda \geq 1/(10\beta \log \text{OPT})$, stop the recursive procedure. Let $\vec{\pi}(u) = \gamma(u, H)/(10\beta \log \text{OPT})$. Output H and $\vec{\pi}$.
   (b) Else, find a cut S such that $|\delta(S)| \leq \beta\lambda w(S) w(V \setminus S)$. Recurse on the induced graphs $H[S]$ and $H[V \setminus S]$.

We next prove that the algorithm above has the desired properties. The first two properties in Theorem 2.1 are easy to check as they are enforced in the two terminating steps (Steps 1 and 2a).

We now prove the last and key property. The partitioning procedure naturally defines a recursion tree. The leaves of the tree are the graphs where we stop the recursion either because the flow is sufficiently small or the concurrent flow for the product multicommodity flow that we set up is large enough. Let $G_1, G_2, \ldots, G_\ell$ be the subgraphs produced by the decomposition. From the terminating conditions it follows that $\vec{\pi}(G_i) \geq \gamma(G_i)/(10\beta \log \text{OPT})$. Thus, it is sufficient to prove that $\sum_{i=1}^{\ell} \gamma(G_i) \geq \gamma(G)/2$. We instead show that the flow *lost* in all the recursive steps is at most $\gamma(G)/2$. The flow lost is upper bounded by the total number of edges that are cut in the partitioning process.

LEMMA 2.2. *In the recursive step, the number of edges cut is at most $\frac{1}{10 \log \text{OPT}} \sum_{u \in S} \gamma(u, H)$.*

PROOF. We have that $|\delta(S)| \leq \beta\lambda w(S) w(V \setminus S)$. From the definition of w, $w(S)w(V \setminus S) \leq \sum_{u \in S} \gamma(u, H)$. Therefore $|\delta(S)| \leq \beta\lambda \sum_{u \in S} \gamma(u, H) \leq \frac{1}{10 \log \text{OPT}} \sum_{u \in S} \gamma(u, H)$. ■

Let $\epsilon = (\sum_{u \in S} \gamma(u, H))/2\gamma(H)$. Since the total flow $\gamma(H)$ is at least half of $\sum_{u \in S} \gamma(u, H) + \sum_{u \in V-S} \gamma(u, H)$, without loss of generality $\epsilon \leq 1/2$, otherwise we can work with $V \setminus S$. Let $L(a)$ denote the number of edges cut in the decomposition process when starting with a graph G with total flow a. Thus we can write a recursion for the total number of edges cut as

$$L(a) \leq L(\epsilon a) + L((1-\epsilon)a) + \epsilon a/(5 \log \text{OPT}).$$

For $a \leq \text{OPT}$, it can be checked that $L(a) \leq a \log a/(2 \log \text{OPT})$ satisfies the above recursion. Therefore $L(\text{OPT}) \leq \text{OPT}/2$ and hence the total lost flow is at most $\text{OPT}/2$. ■

For edge problems, $\beta(G) = O(\log k)$ for general graphs [28] and $\beta(G) = O(1)$ for planar graphs [21]. For node problems $\beta(G) = O(\log k)$ for general graphs [28] and for planar graphs this paper proves that the gap is $O(1)$ (see Theorem 4.4). A similar decomposition also works to obtain cut-linked sets, based on the approximation ratio for the minimum quotient cut.

THEOREM 2.3. *Let* OPT *be a solution to the LP for a given instance $(G, \mathcal{T})$ of EDP (NDP) in a graph G. Let $\beta(G) \geq 1$ be an upper bound on the approximation ratio of a polynomial time algorithm for the minimum quotient edge (node) cut problem in G. Then there is a polynomial time algorithm that partitions G into node-disjoint induced subgraphs $G_1, G_2, \ldots, G_\ell$ and to each G_i assigns a weight function $\vec{\pi} : V(G_i) \to \mathcal{R}^+$ with the following properties. Let $\mathcal{T}_i$ be the induced pairs of $\mathcal{T}$ in G_i and let X_i be the set of terminals of $\mathcal{T}_i$.*

1. *$\vec{\pi}_i(u) = \vec{\pi}_i(v)$ for $uv \in \mathcal{T}_i$.*
2. *X_i is $\vec{\pi}_i$-cut-linked in G_i.*
3. *$\sum_{i=1}^{\ell} \vec{\pi}_i(X_i) = \Omega(\text{OPT}/(\beta(G) \log \text{OPT}))$.*

For general graphs $\beta(G) = O(\log k)$ [28] which has been improved to $O(\sqrt{\log n})$ for the edge case [5] and extended to the node case in [1, 14]. For planar graphs $\beta(G) = O(1)$ for both edge [34, 31] and node case [3].

REMARK 2.4. In order to apply Theorems 2.1 and 2.3 to node problems we need an extra technical condition: for $i = 1, \ldots, \ell$, $\min_{u \in G_i, \vec{\pi}_i(u) > 0} \vec{\pi}_i(u) \geq \frac{c}{\beta(G) \log n \log \text{OPT}}$ for some absolute constant c. One checks that this can be achieved as long as we start with a multicommodity flow where the flow on each path is large enough. Such a flow exists by Lemma 1.1. Thus we obtain an α-node-linked instance with $\alpha = \Theta(\frac{1}{\beta(G) \log n \log \text{OPT}})$ and the total number of terminals in the decomposition is $\Omega(\log n \cdot \text{OPT})$.

2.2 $\vec{\pi}$-Flow-linked to Well-linked Terminals

The main idea underlying this transformation is to consolidate terminals into connected clusters such that each cluster has $\Omega(1)$ units of flow originating from it. We can then choose one of the terminals in a cluster as its representative. This idea was used in [8] for edge problems. We establish an analogous result for node problems in this subsection.

We use the clustering scheme of [8] where we choose an arbitrary rooted spanning tree of the graph and partition the tree into edge-disjoint subgraphs of $\vec{\pi}$-weight $\Theta(1)$ each in a bottom-up manner. We obtain the following theorem:

THEOREM 2.5 ([8]). *If X is $\vec{\pi}$-flow-linked in G, then there is a subset $X' \subset X$ such that X' is $\frac{1}{2}$-flow-linked in G and $|X'| = \Omega(\vec{\pi}(X))$.*

For the node case, we again work with a spanning tree or a spanning forest. But the clustering scheme for node problems turns out to be somewhat more difficult than for the edge problems in that the degree of the nodes in the spanning forest plays an important role. In the edge case, the partitioning of the tree implicitly uses the reduction to the bounded degree case. We need the following two lemmas that establish existence of low degree spanning forests where each tree contains sufficiently large weight. For node problems, we do not normally start with a $\vec{\pi}$-flow-linked set, but rather we use a uniformly linked set, as is guaranteed by Remark 2.4. Thus in the following we assume that we have α-node-flow-linked sets.

LEMMA 2.6. *If X is α-node-flow-linked in G, then there is a tree T in G of maximum degree $O(\frac{1}{\alpha} + \log n)$ that spans X.*

PROOF. Let $k = |X|$. Let M be any matching on X. M induces a multicommodity flow instance with $d(u, v) = 1$ if $uv \in M$. Since X is α-node-flow-linked, there is a feasible solution to this multicommodity flow instance in G with node congestion $O(1/\alpha)$. By standard randomized rounding, M can be integrally routed with node congestion $O(\frac{1}{\alpha} + \log n)$. Let $v_1, v_2, \ldots, v_k$ be the nodes of X. Let M_1 be the matching on X given by the pairs $v_1v_2, v_3v_4, \ldots, v_{k-1}v_k$ and let M_2 be the matching on X given by the pairs $v_2v_3, v_4v_5, \ldots, v_kv_1$. Both M_1 and M_2 can be routed integrally with node congestion $O(\frac{1}{\alpha} + \log n)$ and hence the union of these two routings gives a subgraph of G in which X is connected and degree is $O(\frac{1}{\alpha} + \log n)$. ∎

In the above proof we can replace $\log n$ by $\log n / \log \log n$. An alternate clustering that avoids the dependence on n can be obtained by the following result.

LEMMA 2.7. *If X is α-node-flow-linked in G, then for any $h \geq 2$ there is a forest F in G of maximum degree $O(\frac{1}{\alpha} \log h)$ such that each tree in F spans at least h nodes from X.*

PROOF. If X is α-node-flow-linked, given any two sets Y, Z such that $|Y| = |Z| \leq |X|/2$, there exist paths from Y to Z such that each node in $Y \cup Z$ is an end point of exactly one of these paths and such that the number of paths through any node is $O(1/\alpha)$. We use this fact to find the required F. Start with a partition of X into equal sized sets Y, Z. By connecting Y to Z we obtain connected components that contain at least 2 nodes of X. We pick representative nodes from each of the components and connect them up again. After i iterations, each connected component has at least 2^i nodes from X. Hence, after $\lceil \log 1/\alpha \rceil$ iterations, each connected component has at least $1/\alpha$ nodes from X. In each iteration the number of paths using a node can increase by $O(1/\alpha)$, hence the degree at the end of $\lceil \log 1/\alpha \rceil$ iterations is $O(\frac{1}{\alpha} \log \alpha)$. ∎

THEOREM 2.8. *For $\alpha \leq 1$, if X is α-node-flow-linked in G then there is a subset $X' \subset X$ such that X' is $\frac{1}{2}$-node-flow-linked in G and $|X'| = \Omega(\alpha\gamma|X|)$ where $\gamma = \max\{(\frac{1}{\alpha} + \log n)^{-1}, \alpha / \log \frac{1}{\alpha}\}$.*

PROOF. Let F be a forest spanning X with maximum degree Δ such that each connected component of F contains at least $\lceil 2/\alpha \rceil$ nodes from X. We can breakup F into node disjoint subtrees $T_1, T_2, \ldots, T_\ell$ such that each T_i contains between $\lceil 2/\alpha \rceil$ and $\lceil 2\Delta/\alpha \rceil$ nodes from X. Hence $\ell = \Omega(\alpha|X|/\Delta)$. From each T_i we pick one arbitrary node of $X \cap T_i$ to form our set X'. By construction $|X'| = \ell = \Omega(\alpha|X|/\Delta)$. With each $u \in X'$ we associate the tree $T(u)$ from which u was chosen. From $T(u)$ we pick a set of $\lceil 2/\alpha \rceil$ nodes $X(u) \subseteq X$ arbitrarily. Note that for $u, v \in X', u \neq v$, $X(u) \cap X(v) = \emptyset$. We claim that X' is $\frac{1}{2}$-node-flow-linked. It is sufficient to show that we can route any matching M' on

X' with congestion 2. Let $uv \in M'$. The node u sends one unit of flow on $T(u)$ to $X(u)$ by sending $1/(\lceil 2/\alpha \rceil)$ amount to each $w \in X(u)$. We then create an arbitrary bipartite matching M between $X(u)$ and $X(v)$ for each such $uv \in M'$. Thus M is a partial matching on X. By Proposition 1.2, M can be routed using congestion $2/\alpha$ in G, so we can route $1/(\lceil 2/\alpha \rceil)$-flow for M using congestion 1. Composing this routing of M with the distribution of flow on the $T(u)$'s described above, we see that M' can be routed with congestion 2. ■

3. EDGE CAPACITATED PROBLEMS

The following theorem appears in [8].[1]

THEOREM 3.1 ([8]). *For $\alpha \leq 1$, if X is α-edge-flow-linked in G then for any matching M on X, there is a polynomial time algorithm to route one unit of flow for $\Omega(\alpha|M|)$ pairs in M.*

Using Theorem 2.1 and the above we obtain the following.

THEOREM 3.2. *There is an $O(\log^2 k)$ approximation algorithm for the edge capacitated all-or-nothing flow problem in undirected graphs and an $O(\log k)$ approximation in undirected planar graphs.*

Using results from [9] and Theorem 2.1 we obtain the following.

THEOREM 3.3. *There is an $O(\log k)$ approximation algorithm for the edge-disjoint-path problem in planar graphs with congestion 2.*

4. NODE CAPACITATED PROBLEMS

4.1 All-or-Nothing Node Flow

THEOREM 4.1. *For any fixed $\epsilon \in (0, 1]$, given any matching M on an α-flow-linked set X, we can fractionally route $\Omega(\epsilon\alpha\gamma|X|)$ pairs from M with node congestion $(1 + \epsilon)$; here $\gamma = \max\{(\frac{1}{\alpha} + \log n)^{-1}, \alpha/\log\frac{1}{\alpha}\}$. In particular, if X is $\Omega(1/(\log^2 k \log n))$-flow-linked, then $\Omega(|X|/\log^4 k \log^2 n)$ pairs from M can be routed.*

PROOF. We follow the proof idea of Theorem 2.8. The main difference is that instead of choosing the set X' arbitrarily from the node disjoint subtrees (clusters) $T_1, T_2, \ldots, T_\ell$, we use the matching M to guide its selection. As before let d, be the maximum degree of nodes in the trees. Start with an arbitrary pair $uv \in M$. We mark the trees $T(u)$ and $T(v)$ and remove all pairs from M that contain nodes in $T(u)$ or $T(v)$. We add the pair to M' which is initially empty. While there are pairs left in M we pick another pair and repeat the procedure above. It is easy to see that each pair that we add to M' kills at most $2d/\alpha$ pairs in M. Hence $|M'| = \Omega(\alpha|M|/d)$. Call a pair $uv \in M'$ *unsplit* if $T(u) = T(v)$, otherwise, call it *split.* If the number of unsplit pairs is at least half the number of pairs in M' we can route them by node disjoint paths and we are done. Otherwise, the set of terminals of split pairs lie in different subtrees, and we can route them by distributing their flow through the cluster. This results in an overall congestion of 2, one for distributing the flow inside the cluster, and another for routing the flow among clusters. We can obtain congestion $(1+\epsilon)$ by using node disjoint trees of size at least $\lceil 2/(\epsilon\alpha) \rceil$ instead of $\lceil 2/\alpha \rceil$. ■

[1]The clustering scheme presented in [8] for congestion 1 routing needs a further strengthening to obtain the result. Full details will appear in the journal version of [8].

THEOREM 4.2. *There is an $O(\log^4 k \log n)$ approximation algorithm for the all-or-nothing node flow problem with congestion $(1 + \epsilon)$. The ratio improves to $O(\log^2 k \log n)$ for planar graphs.*

Proof Sketch: Starting from an arbitrary multicommodity flow, Theorem 2.1 guarantees $\vec{\pi}$-flow-linked terminal sets. To apply Theorem 4.1, we need the refinement of Theorem 2.1 stated in Remark 2.4 from which we obtain a decomposition into instances in which terminals are $\Theta(1/(\log^2 k \log n))$-flow-linked. Let $(G_i, \mathcal{T}_i)$ be an instance in the decomposition with X_i the terminal set for $\mathcal{T}_i$. Applying Theorem 4.1 to $(G_i, \mathcal{T}_i)$ we can route $\Omega(|X_i|/(\log^4 k \log^2 n)$ pairs from $\mathcal{T}_i$ in G_i. From Remark 2.4, the total number of terminals in the decomposition, that is $\sum_i |X_i|$ is $\Omega(\log n\text{OPT})$. Hence the total number of pairs that are routed is $\Omega(\text{OPT}/(\log^4 k \log n))$. To obtain the ratio for planar graphs we need use the improved flow-cut gap that we show in this paper (see Theorem 4.4). ■

4.2 Node Disjoint Paths in Planar Graphs

In this section we prove the following.

THEOREM 4.3. *There is an $O(\log^2 k \log n)$ approximation algorithm for the node-disjoint path problem in planar graphs with congestion 4.*

The overall framework is similar to that in [9] for the edge case. There are however two subtelities in the node case that we discuss briefly. First, as we mentioned earlier (see Remark 2.4), in the node case, we need to preprocess the LP solution using Lemma 1.1 before using the decomposition procedure. Lemma 1.1 produces a new instance and a corresponding LP solution of value $\Theta(\text{OPT})$ and with $\Theta(\text{OPT} \log n)$ pairs; OPT refers to the value of the LP solution to the original instance. The second issue is whether to use the flow-linked decomposition from Theorem 2.1 or the cut-linked decomposition from Theorem 2.3. The reason to distinguish between the two is that in node capacitated planar graphs, the previously known bound on the flow-cut gap is $O(\log n)$ [28] while the approximation ratio for finding minimum quotient cuts is $O(1)$ [3]. In this paper we are able to improve the flow-cut gap to $O(1)$, however, as we do not provide all the details for this improvement, we use the decomposition guaranteed by Theorem 2.3.

Now we describe details of the algorithm. In the first phase of the algorithm, we solve the flow LP for the given instance and preprocess it as desribed in Lemma 1.1. Then we apply the decomposition algorithm given in Theorem 2.3 to to obtain a collection of cut-linked instances. In the decomposition we use an $O(1)$-approximation algorithm for minimum quotient node cuts [3]. From Theorem 2.3 and Remark 2.4, with $\alpha = \Theta(1/(\log n \log k))$ and $\beta(G) = O(1)$, it follows that we find a collection of α-node-cut-linked instances such that there remain $\Omega(\text{OPT} \log n)$ pairs amongst these instances. In the second phase of the algorithm, we show that given any α-node-cut-linked instance with say k' pairs, we may route $\Omega(\alpha^2 k')$ pairs by node-disjoint paths with constant congestion.

We now describe this second phase. It is based on taking any α-node-cut-linked instance (for any $\alpha \le 1$) and producing a large minor which we can then use to route the terminals. We note that if $\alpha = \Omega(1)$, we can then actually route *any* constant fraction of the pairs. This gives us the following result.

THEOREM 4.4. *The flow-cut gap for product multicommodity flow in node capacitated planar graphs is $O(1)$.*

We now return to the proofs. The following is the main technical result we need to prove Theorem 4.3. It applies to planar graphs with unit node capacities. A capacitated version of this result is needed to obtain Theorem 4.4 and to generalize Theorem 4.3 to the capacitated case; we defer the details to the full version.

THEOREM 4.5. *Let $\alpha \le 1$ and X be a α-node-cut-linked set in a planar graph G. Then there is a grid minor H in G of size $h = \Omega(\alpha|X|)$ such that h nodes of X can route to the interface of H with node-disjoint paths. Moreover, we may compute such a minor in polynomial time.*

We remark that one approach to finding such a large grid is to first deduce that the existence of an α-node-cut-linked X implies that G has treewidth $\Omega(\alpha|X|)$ and to then apply the following result of Robertson, Seymour and Thomas [35]: every planar graph of treewidth at least $6g - 4$ has a grid minor of size g. In [9], such a grid minor is constructed in polytime, and moreover it was shown that if X is edge-cut-linked and cannot route a large fraction to the grid minor, then there is an edge that can be deleted while maintaining that X is edge-cut-linked. This gives an indirect way to find a grid minor that can be reached by a large fraction of X by disjoint paths. Subsequently, Paul Seymour pointed out to the authors that these deletable edge computations should be avoidable by ensuring at the outset that one chooses a grid to which a large fraction of X can be routed. The proof below adapts the scheme in [35] for finding a grid minor. We mention that other adaptions to produce related results were given also in [2, 23].

We start with an easy but useful fact.

PROPOSITION 4.6. *For $\alpha \le 1$, if X is α-node-cut-linked in G, then, for any $Y \subset V(G)$ such that $|Y| < \alpha|X|/4$, $G-Y$ contains a connected component D such that $|D \cap X| > |X|/2$.*

PROOF. Let $S_1, S_2, \dots, S_\ell$ be the node sets of the connected components of $G - Y$ in increasing order according to size. If $\ell = 1$ we are done, so assume that $\ell \ge 2$. Let j be the smallest index such that $|X|/4 \le \sum_{h=1}^{j} |S_h \cap X|$. If $j = \ell$, then S_ℓ yields the desired component. Otherwise either S_j or $\cup_{h=1}^{j} S_h$ is a set Z with $|X|/4 \le |Z \cap X| \le |X|/2$. But then $|N(Z)| \le |Y| < \alpha|X|/4$ contradicting the α-cut-linkedness of X. ■

Proof of Theorem 4.5: We assume that we have some fixed embedding of G on the sphere and that X is α-node-cut-linked in G with $k = |X|$. We also fix some point ι of the sphere. A *G-curve* is a simple curve in the plane, whose image only intersects the embedding of G at nodes. If the image of its "beginning" and "endpoints" are the same, then we call such a *closed* simple curve a *G-contour*, or just a contour. The *length* of a G-curve C, denoted by $len(C)$, is the number of nodes of G whose embedding is contained in the image of the curve. (In the following, we abuse terminology and do not differentiate between nodes, edges, and their images in the embedding.) We call a G-contour *short* if its image does not contain ι and its length is at most $\beta := \lfloor \alpha k/8 \rfloor$. For any short contour C, removing its image from the sphere produces two open regions (disks). We let $ins(C)$ (respectively $ext(C)$) denote the region not containing ι (respectively containing ι). Without loss of generality, there is a G-contour C whose length is 0 and whose interior contains the embedding of G. We seek contours whose interior contains at least $k/2$ nodes of X; call such a contour *fat*. In the following, we make use of the simple observation below.

PROPOSITION 4.7. *If $S \subset V(G)$ contains more than $k/8$ nodes of X and $|N_G(S)| \le \beta$, then S contains at least $k/2$ nodes of X. In particular, any short contour whose interior contains more than $k/8$ nodes of X is fat.*

Indeed, if $k/8 < |S \cap X| \le k/2$, by α-node-cut-linkedness of X, $|N_G(S)| > \alpha k/8 \ge \beta$.

Let C be a short, fat contour. If C has length strictly less than β we may obtain a longer such contour by repeatedly applying one of two operations. The first shifts the curve over an edge. This decreases by 1, the number of edges in the interior. The second nudges the curve into a new node, thus increasing its length by one – see Figure 1. If this operation creates a non-simple closed curve, then we created two more short contours, one of whose interior contains at least $k/4$ nodes of X. Hence by Proposition 4.7 it must be fat. We may then work with this new contour (whose closed interior contains at least 1 less node). If nudging does not create a non-simple closed curve, then the new contour has at most one less node of X in its interior. Hence by Proposition 4.7, the new contour is again fat. Therefore after at most $|E| + |V|$ shifts, we obtain a short, fat contour whose length is exactly β.

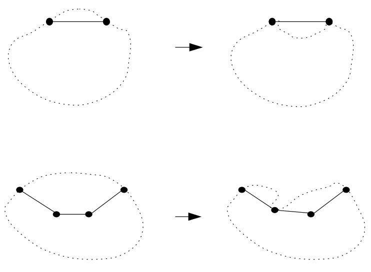

Figure 1: Shifting the curve.

Given a short contour C', we call a simple non-closed curve J a *cut* of C' if its image is contained in the closed interior of C' and whose image only intersects C' at the endpoints of J, and these correspond to distinct nodes a, b of G. Let U_1, U_2 denote the two subcurves of C' joining a, b. We call J a *shortcut* of C' if the length of J is less than the minimum of the lengths of the U_1 and U_2. Note that if J is a shortcut, then each of the two contours, C_1 and C_2, formed respectively from U_1, U_2 and J, has length at most $\beta - 1$. Since $ins(C')$ contained at least $k/2$ nodes of X, the interior of at least one of these two contours, say C_1, contains at least $(k/2 - \beta/2)/2 > k/8$ nodes. Hence by Proposition 4.7, $ins(C_1)$ contains at least $k/2$ terminals. Thus we may work with this new short contour whose closed interior contains at least one less node of G. Note that we can check for the existence of a shortcut by a shortest path computation.

Thus, by the shifting process and the shortcut process, we may find, in polynomial time, a contour C' of length exactly β with no shortcuts, that contains at least $k/2$ terminals in its proper interior. Let S be the set of nodes in the interior of C' and S' be the set of nodes on C'.

Let G' denote the subgraph of G induced by $S \cup S'$. Let $X' = S \cap X$. We now check if there are β node disjoint paths from X' to S' in G'. We check this via Menger's Theorem for node disjoint paths in the graph G^+ obtained from G' by adding two new nodes s, t, where s is adjacent to each node of X' and t to each node of S'. If the maximum number of internally-disjoint $s - t$ paths is strictly less than β, then there is a subset F of nodes such that $|F| < \beta$ and where s, t lie in distinct components of $G^+ - F$. It follows that F separates X' from S' in G'. By Proposition 4.6, some component R of $G - F$ contains more than $k/2$ nodes. Since $|X'| \geq k/2$, R must contain some node of X'. Note that R is also a component of $G' - F$ since otherwise it would contain a node of S', and hence s, t are not disconnected in $G^+ - F$.

It now follows that there is a G'-contour C'' which traverses precisely the nodes $L := N_{G'}(R)$ and whose interior contains R. If we choose F to be minimal, then in fact the contour can be chosen so that its closure contains exactly the embedding of the graph $G'[L \cup R]$. C'' may be chosen to also be a G-contour whose interior is contained in that of C'. Since $L \subseteq F$, $|L| < \beta$, and since $len(C') = \beta$ we thus have a short fat contour C'' for which at least one node on C' lies in $ext(C'')$. We may thus continue the process with this smaller contour, and start shifting and shortcutting once again. Eventually we find a short fat contour C^* such that β nodes of X can route on disjoint paths to the β nodes on C^*.

Starting with C^*, we now follow the same argument as used in [35] for finding a grid minor. In particular, since C^* has no shortcuts, we have that for any two equal-sized disjoint subsets of nodes B_1, B_2 on C', G' contains $|B_1|$ node-disjoint paths each joining a node of B_1 to a node of B_2. We now produce a γ-grid minor for $\gamma = \lfloor \beta/4 \rfloor$ as follows. Order the nodes on C^* as $v_1, v_2, \ldots, v_\beta$ in say clockwise order. Partition these into 4 chunks of size γ: $B_i = \{v_j : i\gamma + 1 \leq j \leq (i+1)\gamma\}$ for $i = 0, 1, 2, 3$. We now find γ paths connecting the nodes of B_1 to the nodes of B_3. By planarity we actually have that node v_i is connected to to node v_{s+i} for each i (where $s = 2\gamma$). Then we find γ paths joining the nodes of B_2 to the nodes of B_4. Clearly the combined collection of paths yields the desired grid minor of size $\lfloor \beta/4 \rfloor \geq \alpha k/32 - 2$. We can arbitrarily choose B_1 to be the interface of the grid minor. We have also seen that X has β node disjoint paths to the boundary S' of C^*, and since $|S'| = \beta$ we have that X has $|B_1|$ node-disjoint paths to B_1 which is the interface of the grid minor. ∎

We now apply this result to obtain the following which then implies Theorem 4.3.

THEOREM 4.8. *Let G be a planar graph and X an α-node-cut-linked set in G. Given a matching M on X, $\Omega(\alpha^2|M|)$ pairs in M can be routed via paths such that at most 4 of these paths intersect any node in G.*

PROOF. From Theorem 4.5, there is a grid minor H in G of size $h = \Omega(\alpha|X|)$ and subset $X' \subset X$ such that $|X'| = h$ and X' can be routed via node-disjoint paths to the interface I of H. Let X'' denote those terminals whose paths terminate at the "odd" nodes of the interface; so $|X''| = (|X'| - 1)/2$. Let Y denote the mates of X'' under M. Note that by cut-linkedness of X, there exists at least $\alpha|Y|$ node-disjoint paths from Y to X''. These paths can be further extended to reach the interface I using the paths from X'' to I and giving congestion at most 2 on each node. Thus by Menger's Theorem, there exist $\alpha|Y|/2$ node-disjoint paths from Y to I. Moreover, we can assume that the endpoints of these paths are the "even" nodes of I, by possibly adding an extra edge at the end of a path. Let Y' denote the endpoints of these paths, and let Z be their mates in X''. We have thus routed each node of $Y' \cup Z$ to distinct nodes of I with congestion at most 2. Let M' be the matching induced on $Y' \cup Z$ by M, note that $|M'| = |Y'|$. We may match up the pairs under M' via paths in the grid, with congestion at most 2. Thus we have routed a total of $|Y'| = \Omega(\alpha^2|M|)$ pairs with congestion at most 4. ∎

5. CONCLUDING REMARKS

We believe that ideas similar to those in our earlier work on edge problems [8, 9] can be used to improve the congestion bounds presented here for node routing problems. In particular, we should be able to obtain congestion 1 for all-or-nothing node flow and a congestion of 2 for node disjoint paths in planar graphs. Combined with the recent work of [12], our results on planar graphs should extend to graphs that exclude a fixed minor. We leave the details of these improvements to the full version of the paper.

Acknowledgments: We thank Jon Kleinberg and Paul Seymour for comments on [9] which helped in the proof of Theorem 4.5.

6. REFERENCES

[1] A. Agarwal, M. Charikar, K. Makarychev, and Y. Makarychev. $O(\sqrt{\log n})$ approximation algorithms for Min UnCut, Min 2CNF Deletion, and directed cut problems. *Proc. of STOC*, 2005.

[2] N. Alon, P. Seymour and R. Thomas. Planar Separators. *SIAM J. Discrete Math.* 7(2):217–241, 1994.

[3] E. Amir, R. Krauthgamer and S. Rao. Constant factor approximation of vertex-cuts in planar graphs. *Proc. of STOC*, 2003.

[4] M. Andrews and L. Zhang. Hardness of the Undirected Edge Disjoint Path Problem. *Proc. of STOC*, 2005.

[5] S. Arora, S. Rao and U. Vazirani. Expander Flows, Geometric Embeddings, and Graph Partitionings. *Proc. of STOC*, 2004.

[6] Y. Aumann and Y. Rabani. An $O(\log k)$ approximate min-cut max-flow theorem and approximation algorithm. *SIAM Journal on Computing*, 27(1):291–301, February 1998.

[7] C. Chekuri and S. Khanna. Edge Disjoint Paths Revisited. *Proc. of SODA*, 2003.

[8] C. Chekuri, S. Khanna, and F. B. Shepherd. The All-or-Nothing Multicommodity Flow Problem. *Proc. of STOC*, June 2004.

[9] C. Chekuri, S. Khanna, and F. B. Shepherd. Edge Disjoint Paths in Planar Graphs. *Proc. of FOCS*, 2004.

[10] C. Chekuri, M. Mydlarz, and F. B. Shepherd. Multicommodity Demand Flow in a Tree and Packing Integer Programs. Submitted. Preliminary version in *Proc. of ICALP*, 2003.

[11] J. Chuzhoy and S. Khanna. Improved Hardness Results and Integrality Gaps for Edge Disjoint Paths. Manuscript, 2005.

[12] E. Demaine and M. Hajiaghayi. Graphs Excluding a Fixed Minor have Grids as Large as Treewidth, with Combinatorial and Algorithmic Applications through Bidimensionality. *Proc. of SODA*, 2005.

[13] S. Even, A. Itai and A. Shamir. On the complexity of timetable and multicommodity flow problems. *SIAM J. on Computing*, Vol 5, 691-703, 1976.

[14] U. Feige, M. Hajiaghayi, and J. Lee. Improved approximation algorithms for minimum-weight vertex separators. *Proc. of STOC*, 2005.

[15] A. Frank. Packing paths, cuts, and circuits - a survey. In B. Korte, L. Lovász, H. J. Prömel, and A. Schrijver, eds., *Paths, Flows and VLSI-Layout*, 49–100. Springer Verlag, Berlin, 1990.

[16] N. Garg, V. Vazirani, M. Yannakakis. Primal-Dual Approximation Algorithms for Integral Flow and Multicut in Trees. *Algorithmica*, 18(1):3-20, 1997.

[17] V. Guruswami, S. Khanna, R. Rajaraman, F. B. Shepherd, and M. Yannakakis. Near-Optimal Hardness Results and Approximation Algorithms for Edge-Disjoint Paths and Related Problems. *JCSS*, 67:473–496, 2003.

[18] M. Hajiaghayi, R. Kleinberg, T. Leighton, and H. Räcke. Oblivious routing in node-capacitated and directed graphs. *Proc. of SODA*, 2005.

[19] C. Harrelson, K. Hildrum, and S. Rao. A polynomial-time tree decomposition to minimize congestion. *Proc. of SPAA*, 2003.

[20] R. M. Karp. Reducibility among combinatorial problems. *Complexity of Computer Computations*, R. E. Miller, J. W. Thatcher, Eds., New York: Plenum Press, 1972, 85–103.

[21] P. Klein, S. Plotkin and S. Rao. Planar graphs, multicommodity flow, and network decomposition. *Proc. of STOC*, 1993.

[22] J. M. Kleinberg and É. Tardos. Approximations for the disjoint paths problem in high-diameter planar networks. *JCSS*, 57:61–73, 1998. Preliminary version in the *Proc. of STOC*, 1995.

[23] J. M. Kleinberg and É. Tardos. Disjoint Paths in Densely Embedded Graphs. *Proc. of FOCS*, pp. 52–61, 1995.

[24] J. M. Kleinberg. Approximation algorithms for disjoint paths problems. PhD thesis, MIT, Cambridge, MA, 1996.

[25] S. G. Kolliopoulos and C. Stein. Approximating Disjoint-Path Problems Using Greedy Algorithms and Packing Integer Programs. *Math. Programming*, series A, (99):63–87, 2004.

[26] P. Kolman and C. Scheideler. Improved bounds for the unsplittable flow problem. In *Proc. of SODA*, 2002.

[27] M. R. Kramer and J. L. van Leeuwen. The complexity of wire-routing and finding minimum area layouts for arbitrary VLSI circuits. *VLSI-Theory* (F. P. Preparata ed.) [Advances in Computing Research, Volume 2], JAI Press, Greenwich, Connecticut, 1984, 129–146.

[28] T. Leighton and S. Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *JACM*, 46(6):787–832, 1999.

[29] N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245, 1995.

[30] J. F. Lynch. The equivalence of theorem proving and the interconnection problem. *ACM SIGDA Newsletter*, 5:3, 1975.

[31] J. K. Park and C. Phillips. Finding minimum-quotient cuts in planar graphs. *Proc. of STOC*, 1993.

[32] H. Räcke. Minimizing congestion in general networks. *Proc. of FOCS*, 2002.

[33] P. Raghavan and C. D. Thompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7:365–374, 1987.

[34] S. Rao Finding near optimal separators in planar graphs. *Proc. of FOCS*, 1987.

[35] N. Robertson, P. D. Seymour and R. Thomas. Quickly Excluding a Planar Graph. *Journal of Combinatorial Theory (B)*, 62: 323-348 (1994).

[36] A. Schrijver. Combinatorial Optimization: Polyhedra and Efficiency. Springer-Verlag, Berlin Hiedelberg, 2003.

[37] A. Srinivasan. Improved approximations for edge-disjoint paths, unsplittable flow, and related routing problems. *Proc. of the FOCS*, pp. 416–425, 1997.