Edge Disjoint Paths in Moderately Connected Graphs

Satish Rao*[1] and Shuheng Zhou**[2]

[1] University of California, Berkeley, CA 94720, USA,
satishr@cs.berkeley.edu
[2] Carnegie Mellon University, Pittsburgh, PA 15213, USA,
szhou@ece.cmu.edu

Abstract. We study the Edge Disjoint Paths (EDP) problem in undirected graphs: Given a graph G with n nodes and a set $\mathcal{T}$ of pairs of terminals, connect as many terminal pairs as possible using paths that are mutually edge disjoint. This leads to a variety of classic NP-complete problems, for which approximability is not well understood. We show a polylogarithmic approximation algorithm for the undirected EDP problem in general graphs with a moderate restriction on graph connectivity; we require the global minimum cut of G to be $\Omega(\log^5 n)$. Previously, constant or polylogarithmic approximation algorithms were known for trees with parallel edges, expanders, grids and grid-like graphs, and most recently, even-degree planar graphs. These graphs either have special structure (e.g., they exclude minors) or there are large numbers of short disjoint paths. Our algorithm extends previous techniques in that it applies to graphs with high diameters and asymptotically large minors.

1 Introduction

In this paper, we explore approximation for the edge disjoint paths (EDP) problem: Given a graph with n nodes and a set of terminal pairs, connect as many of the specified pairs as possible using paths that are mutually edge disjoint. EDP has a multitude of applications in areas such as VLSI design, routing and admission control in large-scale, high-speed and optical networks. Moreover, EDP and its variants have also been prominent topics in combinatorics and theoretical computer science for decades. For example, the celebrated theory of graph minors of Robertson and Seymour [29] gives a polynomial time algorithm for routing all the pairs given a constant number of pairs. However, varying the number of terminal pairs leads to a variety of classic NP-complete problems, for which approximability is an interesting problem. In a recent breakthrough [3], Andrews and Zhang showed an $\Omega(\log^{\frac{1}{3}-\epsilon} n)$ lower bound on the hardness of approximation for undirected EDP.

* Supported in part by NSF Award CCF-0515304.
** This material is based on research sponsored in part by the Army Research Office, under agreement number DAAD19–02–1–0389 and NSF grant CNF–0435382. This work was done while the author was visiting UC Berkeley.

In this work, we show a polylogarithmic approximation algorithm for the undirected EDP problem in general graphs with a moderate restriction on graph connectivity; we require that there are $\Omega(\log^5 n)$ edge disjoint paths between every pair of vertices, i.e., the global min cut is of size $\Omega(\log^5 n)$. If this moderately connected case holds, we can route $\Omega(\mathsf{OPT}/\operatorname{polylog} n)$ pairs using disjoint paths with congestion 1, where OPT is the maximum number of pairs that one can route edge disjointly for the given EDP instance. Previously, constant or polylogarithmic approximation algorithms were known for trees with parallel edges, expanders, grids and grid-like graphs, and most recently, even-degree planar graphs [20]. The results rely either on excluding a minor (or other structural properties), or the fact that very short paths exist. Our algorithm extends previous techniques; for example, our graphs can have high diameter and contain very large minors. We are hopeful that this constraint on the global minimum cut can be removed if congestion on each edge is allowed to be $O(\log \log n)$. Formally, we have the following result.

Theorem 1. *There is a* $\operatorname{polylog} n$*-approximation algorithm for the edge disjoint path problem in a general graph* $\mathcal{G}$ *with minimum cut and node degree* $\Omega(\log^5 n)$.

1.1 The Approach

We begin with a fractional relaxation of the problem, where each terminal pair can route a real-valued amount of flow between 0 and 1, and this flow can be split fractionally across a set of distinct paths. This can be expressed as an LP and can be solved efficiently. We denote the value of an optimal fractional LP solution as OPT^*. Our algorithm routes a polylogarithmic fraction of this value using integral edge-disjoint paths.

The algorithm proceeds by decomposing the graph into well-connected subgraphs, based on OPT^*, so that a subset of the terminal pairs, that remain within each subgraph are "well-connected", following a decomposition procedure of Chekuri, Khanna, and Shepherd (CKS05) [11]. Then, for each well connected subgraph G, we construct an expander graph that can be embedded into G using its terminal set. We use a result by Khandekar, Rao and Vazirani in [19], where they show that one can build an expander graph H on a set of nodes V by constructing $O(\log^2 n)$ perfect matchings $M_1, \ldots, M_{O(\log^2 n)}$ between $O(\log^2 n)$ sets of equal partitions of V in an iterative manner.

Our contribution along this line is to route each perfect matching $M_t, \forall t$, on one of the $O(\log^2 n)$ (edge-disjoint) subgraphs of G. The "splitting procedure", motivated by Karger's theorem [18], simply assigns edges of G uniformly at random into $O(\log^2 n)$ subgraphs. Using Karger's arguments, we show that all cuts in each subgraph have approximately the correct size with high probability. Here we crucially use the polylogarithmic lower bound on the min-cut. We then route each matching M_t on a unique split subgraph using a max-flow computation with unit capacities. Thus, we can route all $O(\log^2 n)$ matchings edge disjointly in G and embed an expander graph H integrally with congestion 1 on G.

After we construct such an expander graph H for each G, we route terminal pairs in H greedily via short paths. This is effective since there are plenty of short disjoint paths in an expander graph[7, 21]. Since a node in H maps to a cluster of nodes in G that is connected by a spanning tree, we put a capacity constraint on $V(H)$: we allow only a single path to go through each node. We greedily connect a pair of terminals from G via a path in H while taking both nodes and edges along the chosen path away from H, until no short paths remain between any unrouted terminal pair. For the pairs we indeed route, we know the congestion is 1 in the original graph G, since we use each edge and node in H only once, and edges and nodes of H correspond to disjoint paths of G. We use a lemma in [15] to show that such a greedy method ensures that we route a sufficiently large number of such pairs; We note that this method was proposed but analyzed somewhat differently by Kleinberg and Rubinfeld [21]. Our analysis is more like that of Obata [27], and yields somewhat stronger bounds. Our approximation factor is $O(\log^{10} n)$. (A breakdown of this factor is described in Theorem 4.)

1.2 Related Work

Much of recent work on EDP has focused on understanding the polynomial-time approximability of the problem. Previously, constant or polylogarithmic approximation algorithms were known for trees with parallel edges [15], expanders [21, 26], grids and grid-like graphs [5, 6, 22, 23], and even-degree planar graphs [20]. For general graphs, the best approximation ratio for EDP in directed graphs is $O(\min(n^{2/3}, \sqrt{m}))$ [8, 24, 25, 30, 31], where m denotes number of edges in the input graph. This is matched by the $\Omega(m^{\frac{1}{2}-\epsilon})$-hardness of approximation result by Guruswami et al [17]. For undirected and directed acyclic graphs, the upper bound has been improved to $O(\sqrt{n})$ [13]. For even-degree planar graphs, an $O(\log^2 n)$-approximation [20] is obtained recently.

A variant is the EDP with Congestion (EDPwC) problem, where the goal is to route as many terminals as possible, such that at most ω demands can go through any edge in the graph. For EDPwC on planar graphs, for $\omega = 2$ and 4, $O(\log n)$ [10, 11] and constant [12] approximations have been obtained respectively. For undirected graphs, the hardness results [1] are $\Omega(\log^{1/2-\epsilon} n)$ for EDP and $\Omega(\log^{(1-\epsilon)/(\omega+1)} n)$ for EDPwC.

A closely related problem is the congestion minimization problem: Given a graph and a set of terminal pairs, connect *all* pairs with integral paths while minimizing the maximum number of paths through any edge. Raghavan and Thompson [28] show that by applying a randomized rounding to a linear relaxation of the problem one obtains an $O(\log n/\log\log n)$ approximation for both directed and undirected graphs. For hardness of approximation, Andrews and Zhang [2] show a result of $\Omega((\log\log^{1-\epsilon} m))$ for undirected and an almost-tight result [4] of $\Omega(\log^{1-\epsilon} m)$ for directed graphs, improving that of $\Omega(\log\log m)$ by Chuzhoy and Naor [14]. Finally, the All-or-Nothing Flow (ANF) problem [9, 11] is to choose a subset of terminal pairs such that for each chosen pair, one can fractionally route a unit of flow for all the chosen pairs. The hardness result for ANF and ANF with Congestion is the same as that of EDP and EDPwC [1]. Currently, there exists an $O(\log^2 n)$ [11] approximation for ANF. Indeed, we build on the techniques developed in this approximation algorithm for ANF.

2 Definitions and Preliminaries

We work with graph $G = (V, E)$ with unit-capacity edges, where we allow parallel edges, unless we specify a capacity function for edges explicitly. For a capacitated graph $G = (V, E, c)$, where c is an integer capacity function on edges, one can replace each edge $e \in E$ with $c(e)$ parallel edges. For a cut $(S, \bar{S} = V \setminus S)$ in G, let $\delta_G(S)$, or simply $\delta(S)$ when it is clear, denote the set of edges with exactly one endpoint in S in G. Let $\mathsf{cap}(S, \bar{S}) = |\delta_G(S)|$ denote the total capacity of edges in the cut. The edge expansion of a cut $(S, \bar{S})$, where $|S| \leq |V|/2$, is $\phi(S) = \frac{\mathsf{cap}(S,\bar{S})}{|S|}$. The expansion of a graph G is the minimum expansion over all cuts in G. We call a graph G an expander if its expansion is at least a constant.

An instance of a routing problem consists of a graph $\mathcal{G} = (V, E)$ and a set of terminals pairs $\mathcal{T} = \{(s_1, t_1), (s_2, t_2), \ldots, (s_k, t_k)\}$. Nodes in $\mathcal{T}$ are referred to as terminals. Given an EDP instance $(\mathcal{G}, \mathcal{T})$ with k pairs of terminals, we will use the following LP relaxation as specified in (2.1), to obtain an optimal fractional solution. Let $\mathcal{P}_i, \forall i$, denote the set of paths joining s_i and t_i in $\mathcal{G}$.

$$\max \sum_{i=1}^{k} x_i \ s.t. \tag{2.1}$$

$$x_i - \sum_{p \in \mathcal{P}_i} f(p) \ = \ 0, \forall 1 \leq i \leq k \tag{2.2}$$

$$\sum_{p: e \in p} f(p) \ \leq \ 1, \forall e \in E \tag{2.3}$$

$$x_i, f(p) \ \in \ [0,1], \forall 1 \leq i \leq k, \forall p \tag{2.4}$$

We let $\mathsf{OPT}^*(\mathcal{G}, \mathcal{T})$ be the value of this linear program for the optimal solution $\bar{f}$ of the LP. In the text, where we always refer to a single instance, we primarily use OPT^*. The following definitions come from [11].

Definition 1. (CKS2005 [11]) *Given a non-negative weight function $\boldsymbol{\pi} : X \rightarrow \mathbb{R}^+$ on a set of nodes X in G, X is $\boldsymbol{\pi}$-cut-linked in G if $\forall S$ such that $\boldsymbol{\pi}(S \cap X) = \sum_{x \in S \cap X} \boldsymbol{\pi}(x) \leq \boldsymbol{\pi}(X)/2$, $|\delta(S)| \geq \boldsymbol{\pi}(S \cap X)$; We also refer to (G, X) as a $\boldsymbol{\pi}$-cut-linked instance.*

Definition 2. (CKS2005 [11]) *A set of nodes X is well-linked in G if $\forall S$ such that $|S \cap X| \leq |X|/2$, $|\delta(S)| \geq |S \cap X|$.*

3 Decomposition and an Outline of Routing Procedure

In this section, we first present Theorem 2 regarding a preprocessing phase of our algorithm that decomposes and processes $(\mathcal{G}, \mathcal{T})$ into a collection of cut-linked instances with a min-cut $\Omega(\log^3 n)$ in each subgraph. We then state our main theorem with a breakdown of the polylog n approximation factor. Finally, we give an outline on how we route terminal pairs in each cut-linked instance (G, T); Note that we use G to refer to a subgraph that we obtain through Theorem 2 starting from Section 3.1 till the end of the paper, while $\mathcal{G}$ refers to the original input graph. We first specify the following parameters.

– **Parameters related to original EDP instance** $(\mathcal{G}, \mathcal{T})$
  - $\omega \log^2 n$ is the number of matchings as in Figure 3.1;
  - min-cut $\kappa = \Omega(\log^3 n) = \frac{12(\ln n)(\omega \log^2 n+1)}{\epsilon^2}$, where $\epsilon < 1$;
  - $\beta(\mathcal{G}) = O(\log n)$: the worst-case mincut-maxflow gap on product commodity flow instances on $\mathcal{G}$;
  - $\lambda(n) = 10\beta(\mathcal{G}) \log \mathsf{OPT}^*(\mathcal{G}, \mathcal{T}) = O(\log^2 n)$: as introduced in [11].

Theorem 2. *There is a polynomial time decomposition algorithm, that given an EDP instance $(\mathcal{G}, \mathcal{T})$, where $\mathcal{G}$ has a min-cut of size $\Omega(\kappa \log^2 n)$, and a solution $\bar{f}$ to the fractional EDP problem, with $x_i, \forall i$, being specified as in (2.1), produces a disjoint set of subgraphs and a weight function $\boldsymbol{\pi} : V(\mathcal{G}) \rightarrow \mathbb{R}^+$ on $V(\mathcal{G})$ where*

1. *there are $\alpha_1, \ldots, \alpha_k$ such that $\forall u$ in a subgraph H, $\boldsymbol{\pi}(u) = \sum_{i:s_i=u,t_i \in H} \alpha_i x_i$, (note that this implies $\forall s_i t_i \in \mathcal{T}$, x_i contributes the same amount of weight to $\boldsymbol{\pi}(s_i)$ and $\boldsymbol{\pi}(t_i)$);*
2. *the set of nodes $V(H)$ in each subgraph H is $\boldsymbol{\pi}$-cut-linked in H;*
3. *each subgraph H has min-cut $\kappa = \Omega(\log^3 n)$;*
4. *$\forall u$ in a subgraph H s.t. $\boldsymbol{\pi}(H) \geq \Omega(\log^3 n)$, $\boldsymbol{\pi}(u) \leq \sum_{i:s_i=u,t_i \in H} \frac{x_i}{\beta(\mathcal{G})\lambda(n)}$;*
5. *and $\boldsymbol{\pi}(\mathcal{G}) = \Omega(OPT^*/\beta(\mathcal{G})\lambda(n))$.*

The decomposition essentially says that summing across all subgraphs G, a fair fraction of terminal pairs in $\mathcal{T}$ remain (condition 4, 5); indeed, we lose only a constant fraction of the terminal pairs (by assigning a zero weight to those lost terminals) of $\mathcal{T}$. In addition, each subgraph G is well connected with respect to X, the set of induced terminals of $\mathcal{T}$ in G, in the sense of (G, X) being a $\boldsymbol{\pi}$-cut-linked instance. This decomposition is essentially the same as that of Chekuri, Khanna, and Shepherd [11]. We need to do some additional work to ensure that the min-cut condition (condition 3) holds. We prove a dual (flow-based) version of the result in the full version of the paper.

3.1 Overall Routing Algorithm in Each Decomposed Subgraph G

We assume that we have the $\boldsymbol{\pi}$-cut-linked subgraphs given by Theorem 2. We will treat each subgraph and its induced subproblem (G, T) independently. We use $\boldsymbol{\pi}(G)$ to denote $\boldsymbol{\pi}(V(G))$ in the following sections. Let X be the set of terminals of T that is assigned with a positive weight by function $\boldsymbol{\pi}$ in instance G. We further assume that $\boldsymbol{\pi}(G) = \Omega(\log^7 n)$. If not, we just route an arbitrary pair of terminals in T; otherwise, we use PROCEDURE EMBEDANDROUTE$(G, T, \boldsymbol{\pi})$ in Figure 3.1 to route. We first specify a few more parameters and conditions related to (G, T); We then state Theorem 3, which we prove through the rest of the paper. Combining Theorem 3 and Theorem 2 proves Theorem 4.

– **Parameters and conditions related to an induced subproblem** (G, T)
  - sampling probability $p = 12(\ln n)/\epsilon^2\kappa = 1/(\omega \log^2 n + 1)$

Given graph G with min-cut $\Omega(\log^3 n)$ and a weight function $\boldsymbol{\pi} : V(G) \to \mathbb{R}^+$
$\{G^1, \ldots, G^Z\} = \textsc{Split}(G, Z, \boldsymbol{\pi})$
$\{\mathcal{X}, \mathcal{C}\} = \textsc{Clustering}(G^Z, \boldsymbol{\pi})$, where $\mathcal{X} = \{X_1, \ldots, X_r\}$ and $\mathcal{C} = \{C_1, \ldots, C_r\}$
Given a set of superterminals $\mathcal{X}$ of size r
Let $\mathcal{X}$ map to vertex set $V(H)$ of Expander H
For $t = 1$ to $\omega \log^2 n$
    $(S, \bar{S} = \mathcal{X} \setminus S) = \text{KRV-}\textsc{FindCut}(\mathcal{X}, \{M_k : k < t\})$ s. t. $|S| = |\bar{S}| = r/2$
    Matching $M_t = \textsc{FindMatch}(S, \bar{S}, G^t)$ s.t. M_t is routable in G^t
Combine $M_1, \ldots, M_{\omega \log^2 n}$ to form the edge set F on vertices $V(H)$
$\textsc{ExpanderRoute}(H, T, X)$
End

Fig. 3.1. Procedure $\textsc{EmbedAndRoute}(G, T, \boldsymbol{\pi})$

- number of split subgraphs $Z = 1/p = \omega \log^2 n + 1$
- $W = (\omega \log^2 n + 1)/(1 - \epsilon)$, for some $\epsilon < 1$;
- $r \geq \max\{1, (\boldsymbol{\pi}(G) - (W-1))/(2W-1)\}$, such that $\forall i \in [1, \ldots, r], 2W - 1 \geq \boldsymbol{\pi}(X_i) = \sum_{v \in X_i} \boldsymbol{\pi}(v) \geq W$ and $\boldsymbol{\pi}(\mathcal{X}) \geq \boldsymbol{\pi}(G) - (W-1)$: i.e., at most $W - 1$ unit of weight is not counted in $\mathcal{X}$.

Theorem 3. *Given an induced instance (G, T) with min-cut of G being $\Omega(\log^3 n)$ and a weight function $\boldsymbol{\pi} : V(G) \to \mathbb{R}^+$ such that X is $\boldsymbol{\pi}$-cut-linked in G and $\boldsymbol{\pi}(G) = \Omega(\log^7 n)$,* $\textsc{EmbedAndRoute}$ *routes at least* $\max\{1, \Omega(\boldsymbol{\pi}(G)/\log^7 n)\}$ *pairs of T in G edge disjointly.*

Theorem 4. *Given an EDP instance $(\mathcal{G}, \mathcal{T})$, where $\mathcal{G}$ has a min-cut $\Omega(\lambda(n)\kappa)$, we can route $\Omega(\mathit{OPT}^*(\mathcal{G}, \mathcal{T})/f)$ terminal pairs edge disjointly in $\mathcal{G}$, where the approximation factor f is $O(\lambda(n)\beta(\mathcal{G})W \log^5 n)$.*

4 Obtaining Z Split Subgraphs of G

In this section, we analyze a procedure that splits a graph G, with min-cut $\kappa = \Omega(\log^3 n)$, into Z subgraphs b extending a uniform sampling scheme from Karger [18]. We thus obtain a set of cut-linked instances as in Lemma 1, which immediately follows from Theorem 5.

Procedure Split$(G, Z, \boldsymbol{\pi})$**:** Given a graph $G = (V, E)$ with min-cut $\kappa = \Omega(\log^3 n)$, a weight function $\boldsymbol{\pi} : V(G) \to \mathbb{R}^+$, a set of terminals X in G such that (G, X) is a $\boldsymbol{\pi}$-cut-linked instance, and probability $p = 1/Z$.
Output: A set of randomized split subgraphs $G^1, \ldots, G^Z$ of G.
Each split subgraph $G^j, \forall j = 1, \ldots, Z$ inherits the same set of vertices of G; Edges of G are placed independently and uniformly at random into the Z subgraphs; each $e = (u, v) \in E$ is placed between the same endpoints u, v in the chosen subgraph. We retain the same weight function $\boldsymbol{\pi}$ for all nodes in V in each split subgraph $G^j, \forall j$.

Lemma 1. *With high probability, X is $\frac{(1-\epsilon)\boldsymbol{\pi}}{Z}$-cut-linked in $G^j, \forall j$, for some $\epsilon < 1$.*

Theorem 5 says that all cuts can be preserved in all split graphs $G^1, \ldots, G^Z$ of G we thus obtain. Recall for $S \in V$, $|\delta_G(S)|$ denote the size of $(S, V \setminus S)$ in G. For the same cut $(S, V \setminus S)$, we have $\mathbf{E}[|\delta_{G^j}(S)|] = p\,|\delta_G(S)|$ in $G^j, \forall j$, where p is the probability that an edge $e \in E$ is placed in $G^j, \forall j$.

Theorem 5. *Let $G = (V, E)$ be any graph with unit-weight edges and min cut κ. Let $\epsilon = \sqrt{3(d+2)(\ln n)/p\kappa}$. If $\epsilon \leq 1$, then with probability $1 - O(\log^2 n/n^d)$, every cut $(S, V \setminus S)$ in every subgraph $G^1, G^2, \ldots, G^Z$ of G has value between $(1-\epsilon)$ and $(1+\epsilon)$ times its expected value $p\,|\delta_G(S)|$.*

Proof. We sketch a proof, leaving details in the full paper. We first give a definition by Karger [18], regarding a uniform random sampling scheme on an unweighted graph $G = (V, E)$; Lemma 2 immediately follows from this definition. We then state Karger's theorem regarding preserving all cuts of G in a sampled subgraph, under a certain min-cut condition.

Definition 3. (Karger94 [18]) *A p-skeleton of G is a random subgraph $G(p)$ constructed on the same vertices of G by placing each edge $e \in E$ in $G(p)$ independently with probability p.*

Lemma 2. *Every randomized subgraph $G^j, \forall j$, is a p-skeleton of G.*

Theorem 6. (Karger94 [18]) *Let G be a graph with unit-weight edges and min-cut κ. Let $p = 3(d+2)(\ln n)/\epsilon^2\kappa$. With probability $1 - O(1/n^d)$, every cut in a p-skeleton of G has value between $(1-\epsilon)$ and $(1+\epsilon)$ times its expected value.*

To prove Theorem 6, Karger uses a union bound to show that the sum of probabilities of all *bad* events in a p-skeleton of G is $O(1/n^d)$, where a bad event refers to some cut in a p-skeleton of G diverges from its expected value k by more than ϵk. Given that every random split subgraph $G^j, \forall j$, is a p-skeleton of G by Lemma 2, we apply the essential statement in Karger's proof to all subgraphs G^j with $p = 12(\ln n)/\epsilon^2\kappa$ and $\kappa = 12(\ln n)(\omega \log^2 n + 1)/\epsilon^2$ for a given ϵ. We can then use a union bound to sum up probabilities of bad events across all split subgraphs $G^1, \ldots, G^Z$ of G, which is $O(\log^2 n/n^2)$ for $d = 2$. ■

5 Forming Superterminals that are Well-Linked

The procedure in this section constructs superterminals as follows. It finds connected subgraphs C in G^Z, where $\boldsymbol{\pi}(C) = \Omega(\log^2 n)$, each connecting a subset of terminals. Roughly, the idea is that these clustered terminals are better connected than individual terminals. They are well linked in the sense that any cut that splits off K superterminals as one entity contains at least K edges in $G^j, \forall j$ This allows us to compute congestion-free maximum flows in Section 6.1.

Given split subgraphs $G^1, \ldots, G^Z$ of G, each with the same weight function $\boldsymbol{\pi}$ on its vertex set $V(G^j) = V, \forall j$, that we obtain through PROCEDURE

SPLIT$(G, Z, \boldsymbol{\pi})$, we aim to find a set $\mathcal{X} = \{X_1, \ldots, X_r\}$ of node-disjoint "superterminals", where each superterminal $X_i \in \mathcal{X}$ consists of a subset of terminals in X and each X_i gathers a weight between W and $2W-1$. In addition, we want to find an edge-disjoint set of clusters $\mathcal{C} = \{C_1, \ldots, C_r\}$, where $C_i = (V_i, E_i)$, such that $X_i \subseteq V_i$ and C_i is a connected component, and hence all nodes in X_i are connected through E_i. W.l.o.g., we pick G^Z for forming such clusters $C_i, \forall i$; note that G^Z is a connected graph with a min-cut of $\Omega(\log n), whp$, by Theorem 5.

Procedure Clustering$(G^Z, \boldsymbol{\pi})$**:** Given a split subgraph G^Z and a weight function $\boldsymbol{\pi} : V(G^Z) \to \mathbb{R}^+$ and $\boldsymbol{\pi}(V(G^Z)) = \boldsymbol{\pi}(G) \geq W$.
Output: $\mathcal{X} = \{X_1, \ldots, X_r\}$ and $\mathcal{C} = \{C_1, \ldots, \mathcal{C}_r\}$ as specified in Lemma 3.
We group subsets of vertices of V in an edge-disjoint manner, following a procedure from [9], by choosing an arbitrary rooted spanning tree of G^Z and greedily partitioning the tree into a set $\mathcal{C}$ of edge-disjoint subgraphs of G^Z.

Lemma 3. (CKS2004 [9]) *Let G^Z be a connected graph with a weight function $\boldsymbol{\pi} : V(G^Z) \to [0, W]$ such that $\boldsymbol{\pi}(V(G^Z)) \geq W$. We can find $r \geq \max\{1, (\boldsymbol{\pi}(G) - (W-1))/(2W-1)\}$ edge-disjoint connected subgraphs, $C_1 = (V_1, E_1), \ldots, C_r = (V_r, E_r)$, such that there exist vertex-disjoint subsets $X_1, \ldots, X_r$ and for each i: (a) $X_i \subseteq V_i$ and (b) $2W - 1 \geq \sum_{v \in X_i} \boldsymbol{\pi}(v) \geq W$.*

Result. To get an intuition of the purpose of forming such clusters, consider a cut $(U, V \setminus U)$ in a split subgraph $G^j, \forall j$. Let U be a subset of $V(G)$ such that $\boldsymbol{\pi}(U) = \sum_{x \in U \cap X} \boldsymbol{\pi}(x) \leq \boldsymbol{\pi}(X)/2$. Let K be the number of superterminals that are contained in U. We have the following lemma, which captures the notion of superterminals being "well-linked", with a hint of Definition 2.

Lemma 4. *$\forall$ split subgraphs $G^1, \ldots, G^Z$, where $Z = \omega \log^2 n + 1$, and $\forall U \subset V(G)$ s.t. $\boldsymbol{\pi}(U) \leq \boldsymbol{\pi}(X)/2$, $|\delta_{G^j}(U)| \geq K$, where $K = |\{X_i \in \mathcal{X} : X_i \subseteq U\}|$.*

6 Construct and Embed an Expander H in G

In this section, we use the superterminals from the previous section as nodes in an expander H that we embed in G. The edges of H are defined using a technique in [19] that builds an expander using $O(\log^2 n)$ matchings. We embed this expander in G by routing each matching in one of the split graphs using a maximum flow computation. This allows us to embed H into G with no congestion. The following procedure restates this outline. Theorem 7 is a main technical contribution of this paper.

Procedure EmbedExpander$(G^1, \ldots, G^{\omega \log^2 n}, \mathcal{X})$**:**
Output: An expander $H = (V', F)$ routable in G s.t. $|V'| = r$ and $\forall i \in V'$, $\boldsymbol{\pi}(i) = \boldsymbol{\pi}(X_i)$ and $\boldsymbol{\pi}(H) = \boldsymbol{\pi}(\mathcal{X})$; F consists of $M_1, \ldots, M_{\omega \log^2 n}$.
We use Step (3) to (8) of PROCEDURE EMBEDANDROUTE in Figure 3.1, where we substitute PROCEDURE FINDMATCH with Figure 6.3 while relying on an existing PROCEDURE KRV-FINDCUT [19]. At each round t, we use KRV-FINDCUT

```
Given a set of points V(H) of size k
for t = 1 to ω log^2 n
        (S, S̄ = V(H) \ S) = KRV-FindCut(V(H), {M_k : k < t}) s.t. |S| = |S̄| = k/2
        M_t = FindMatch(S, S̄) s.t. M_t is a matching between S and S̄
Combine M_1, ..., M_{ω log^2 n} to form the edge set F on vertices V(H)
End
```

Fig. 6.2. KRV-Procedure Constructing an α-Expander H.

to generate an equal-sized partition $(S, \mathcal{X} \setminus S = \bar{S})$; we then find a matching M_t between S and $\bar{S}$ by computing a single-commodity max-flow using FindMatch$(S, \bar{S}, G^t)$ in G^t, that we add to F as edges.

Theorem 7. *(a)* EmbedExpander *constructs a $1/4$-expander $H = (V', F)$; (b) in addition, H is embedded into G as follows. Each node i of H corresponds to a superterminal X_i in $\mathcal{X}$ in G such that all superterminals are mutually node disjoint and each superterminal is connected by a spanning tree, T_i, in G. Each edge (i, j) in H corresponds to a path, P_{ij} from a node in X_i to a node in X_j. All paths P_{ij} and trees T_i are mutually edge disjoint in G.*

Proof. The expander property (a) follows from a result of Khandekar, Rao and Vazirani [19]; they show the procedure in Figure 6.2 produces an expander H.

Theorem 8. (KRV2005 [19]) *Given a set of nodes $V(H)$ of size k, $\exists$ a* KRV-FindCut *procedure s.t. given any* FindMatch *procedure, the* KRV-procedure *as in Figure 6.2. produces an α-expander graph H, for $\alpha \geq 1/4$.*

Each edge $e = (i, j)$ in the matching M_t maps to an integral flow path that connects X_i and X_j in G^t; all such flow paths can be simultaneously routed in G^t edge disjointly due to the max-flow computation as we show in Lemma 5. Since each matching M^t is on a unique split subgraph G^t, the entire set of edges in $M_1, \ldots, M_{\omega \log^2 n}$, that comprise the edge set F of H, correspond to edge disjoint paths in $G^1, \ldots, G^{Z-1}$, where $Z = \omega \log^2 n + 1$. Finally, all spanning trees $T_i, \forall i$, are constructed using disjoint set of edges in G^Z as in Lemma 3. ∎

6.1 Finding a Matching through a Max-flow Construction

In this section, we show that given an arbitrary equal partition $(S, \bar{S})$ of the set $\mathcal{X} = \{X_1, \ldots, X_r\}$, that we obtain through Procedure Clustering$(G^Z, \boldsymbol{\pi})$, we can use the following procedure to route a max-flow of size $r/2$, such that the integral flow paths that we obtain through flow decomposition induce a perfect matching between S and $\bar{S}$. Let $S = \{X_{i_1}, \ldots, X_{i_{r/2}}\}$ and $\bar{S} = \{X_{j_1}, \ldots, X_{j_{r/2}}\}$.

Lemma 5. *In each sampled graph G^t,* FindMatch *produces a perfect matching M_t between an equal partition $(S, \bar{S})$ of $\mathcal{X}$ such that for each edge in $e = (i, j) \in M_t$, there is an integral unit-flow path P_{ij} from a terminal in $X_i \in S$ to a terminal in $X_j \in \bar{S}$. All paths $P_{ij}, s.t. (i, j) \in M_t$ are edge disjoint in G^t.*

0. Given an equal partition $(S, \bar{S})$ of $\mathcal{X}$, we form a flow graph G' from G^t
   by adding auxiliary nodes and directed unit-capacity edges:
1. Add a special source and sink nodes s_0 and t_0;
2. Add nodes $s_1, \ldots, s_{r/2}$ and an edge from s_0 to $s_k, \forall k = 1, \ldots, r/2$;
3. Add nodes $t_1, \ldots, t_{r/2}$; from each $t_k, \forall k = 1, \ldots, r/2$, add an edge to t_0
4. From each $s_k, \forall k$, add an edge to each terminal $x \in X_{i_k}$ s.t. $X_{i_k} \in S$
5. To each node t_k, add an edge from each terminal $x \in X_{j_k}$ s.t. $X_{j_k} \in \bar{S}$
6. Route a max-flow from s_0 to t_0
7. Decompose the flow to obtain a matching between S and $\bar{S}$
8. End

Fig. 6.3. Procedure FindMatch$(S, \bar{S}, G^t)$

Lemma 6. *Every $s_0 - t_0$ cut has size at least $r/2$ in the flow graph G'.*

Proof of Lemma 5: By Lemma 6 (proof appears in the full version), and the fact that there $\exists$ a $s_0 - t_0$ cut of size $r/2$, (e.g., $(\{s_0\}, V(G') \setminus \{s_0\})$) we know the $s_0 - t_0$ min-cut is $r/2$. Hence by the max-flow min-cut theorem, we know that there $\exists$ a max-flow of size $r/2$ from s_0 to t_0. We next decompose the max-flow into $r/2$ integer flow paths, which induce a perfect matching M_t between S and $\bar{S}$ as follows. Consider an integral flow path $P_k, \forall k = 1, \ldots, r/2$. Let directed path P_k start with s_0 and go through $s_k, x \in X_{i_k} \in S$ for some x; and let P_k end with $y \in X_{j_{k'}} \in \bar{S}, t_{k'}, t_0$ for some $k' \in [1, \ldots, r/2]$ and some terminal y. No other path in the max-flow can go through the same pair of superterminals X_{i_k}, $X_{j_{k'}}$ due to the capacity constraints on edges (s_0, s_k) and $(t_{k'}, t_0)$. Hence $M_t = \{(i_k, j_{k'}), \forall k \in [1, \ldots, r/2],$ where $k' \in [1, \ldots, r/2]\}$ is a perfect matching between S and $\bar{S}$. ■

7 Routing on an Expander H Node Disjointly

In this section, we show that the following greedy algorithm routes $\Omega(K/\log^5 n)$ pairs of terminals, where $K = |V(H)| = \Omega(\pi(G)/W)$, in H.

Procedure ExpanderRoute(H, T, X)**:** Given an uncapacitated expander H with at least $512 \log^5 n$ nodes, with node degree $\omega \log^2 n$. While there is a pair (s, t) in $T \subseteq \mathcal{T}$ whose path length is less than D in $H = (V, E)$, where $D = a_3 \omega \log^3 n$ and $a_3 = 32$ is a constant; Remove both nodes and edges from H, along a path through which we connect a pair of terminals in T.

Since we take away both nodes and edges as we route a path across the expander H due to the node capacity constraints on $V(H)$, routing the set P of pairs via integral paths on H induces no congestion in G by Theorem 7. We now argue that $|P|$ is large to finish our proof. Let H' be the remaining graph of expander $H = (V, E)$, after we take away nodes and edges along the paths used to route P. Note that all remaining pairs $T' \subseteq T$ in H' must have distance at least D. This is the main condition that allows us to prove the following theorem.

Theorem 9. *The procedure above routes $\Omega(K/\log^5 n)$ pairs, node disjointly, in degree-$(\omega \log^2 n)$ expander $H = (V, E)$ with $K \geq 512 \log^5 n$ nodes.*

Proof Sketch: Let us first state the following lemma regarding a multicut in H' which follows from arguments of Garg, Vazirani and Yannakakis [16].

Lemma 7. *If all remaining terminal pairs in $T' \subseteq T$ have distances at least D in H', then there exists a multicut L in $H' = (V', E')$ of size $|E'| \log n/D$ in H' that separates every source and sink pair $s_i t_i \in T'$.*

Lemma 7 implies that there is a multicut of size at most $K\omega \log^3 n/2D = K/2a_3$ given that $|E'| \leq |E| = K\omega \log^2 n/2$ in the remaining graph H'.

We finish, by noting that condition 1 of Theorem 2 implies that any multicut of the terminals in H' ensures that no piece in H' separated by L contains more than half the weight of all terminals in H. We use this fact to show that the multicut L can be rearranged to find a "weight-balanced" cut in H', which corresponds to a node-balanced cut in H. Any node-balanced cut, however, in H must have at least $\Omega(K)$ edges. Using a proper choice of a_3, we force this balanced cut to contain at most half as many edges in H' as in H. Thus, we show $\Omega(K)$ edges have been removed when routing P. Since routing each such pair removes at most $D\omega \log^2 n(O(\log^5 n)$ edges. We conclude $|P|$ must be $\Omega(K/\log^5 n)$.

In more detail, in H', we alter $\boldsymbol{\pi}$ slightly to generate a new function $\boldsymbol{\pi}'(i), \forall i \in V(H')$, so that only remaining pairs $uv \in T'$ contribute a positive weight to $\boldsymbol{\pi}'(H')$ according to their flow in $\bar{f}$ like that of condition 1 in Theorem 2; hence each connected component in H', separated by multicut L, has a weight of at most $\boldsymbol{\pi}'(H')/2$. We then use L to find a balanced cut $(U', V' \setminus U')$ in H' such that each side has weight at least $\boldsymbol{\pi}'(H')/4$, where $\boldsymbol{\pi}'(H') \geq \boldsymbol{\pi}(G) - (W-1) - 2(2W-1)D\,|P|$. It is straightforward to verify that any partition $(U, V(H) \setminus U)$ in H, such that $U' \subseteq U$ and $(V' \setminus U') \subseteq (V(H) \setminus U)$, is node-balanced in H. ■

References

1. M. Andrews, J. Chuzhoy, S. Khanna, and L. Zhang. Hardness of the undirected edge-disjoint paths problem with congestion. In *Proceedings of the 46th IEEE FOCS*, 2005.
2. M. Andrews and L. Zhang. Hardness of the undirected congestion minimization problem. In *Proceedings of the 37th ACM STOC*, 2005.
3. M. Andrews and L. Zhang. Hardness of the undirected edge-disjoint path problem. In *Proceedings of the 37th ACM STOC*, 2005.
4. M. Andrews and L. Zhang. Logarithmic hardness of the directed congestion minimization problem. In *Proceedings of the 38th ACM STOC*, 2006.
5. Y. Aumann and Y. Rabani. Improved bounds for all-optical routing. In *Proceedings of the 6th ACM-SIAM SODA*, pages 567–576, 1995.
6. B. Awerbuch, R. Gawlick, F. T. Leighton, and Y. Rabani. On-line admission control and circuit routing for high performance computing and communication. In *Proceedings of the 35th IEEE FOCS*, pages 412–423, 1994.
7. A. Broder, A. Frieze, and E. Upfal. Existence and construction of edge-disjoint paths on expander graphs. *SIAM Journal of Computing*, 23:976–989, 1994.

8. C. Chekuri and S. Khanna. Edge disjoint paths revisited. In *Proceedings of the 14th ACM-SIAM SODA*, 2003.
9. C. Chekuri, S. Khanna, and F. B. Shepherd. The all-or-nothing multicommodity flow problem. In *Proceedings of the 36th ACM STOC*, 2004.
10. C. Chekuri, S. Khanna, and F. B. Shepherd. Edge-disjoint paths in planar graphs. In *Proceedings of the 45th IEEE FOCS*, 2004.
11. C. Chekuri, S. Khanna, and F. B. Shepherd. Multicommodity flow, well-linked terminals, and routing problems. In *Proceedings of the 37th ACM STOC*, 2005.
12. C. Chekuri, S. Khanna, and F. B. Shepherd. Edge-disjoint paths in planar graphs with constant congestion. In *Proceedings of the 38th ACM STOC*, 2006.
13. C. Chekuri, S. Khanna, and F. B. Shepherd. An $O(\sqrt{n})$ approximation and integrality gap for disjoint paths and unsplittable flow. *Journal of Theory of Computing*, 2:137–146, 2006.
14. J. Chuzhoy and J. Naor. New hardness results for congestion minimization and machine scheduling. In *Proceedings of the 36th ACM STOC*, pages 28–34, 2004.
15. N. Garg, V. V. Vazirani, and M. Yannakakis. Primal-dual approximation algorithms for integral flow and multicut in trees. In *Proc. of the 20th ICALP*, 1993.
16. N. Garg, V. V. Vazirani, and M. Yannakakis. Approximate max-flow min-(multi)cut theorems and their applications. *SIAM J. of Computing*, 25:235–251, 1996.
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. In *Proceedings of the 31th ACM STOC*, 1999.
18. D. R. Karger. Random sampling in cut, flow, and network design problems. In *Proceedings of the 26th ACM STOC*, 1994.
19. R. Khandekar, S. Rao, and U. Vazirani. Graph partitioning using single commodity flows. In *Proceedings of the 38th ACM STOC*, 2006.
20. J. Kleinberg. An approximation algorithm for the disjoint paths problem in even-degree planar graphs. In *Proceedings of the 46th IEEE FOCS*, 2005.
21. J. Kleinberg and R. Rubinfeld. Short paths in expander graphs. In *Proceedings of the 37th IEEE FOCS*, 1996.
22. J. Kleinberg and E. Tardos. Approximations for the disjoint paths problem in high-diameter planar networks. In *Proceedings of the 27th ACM STOC*, 1995.
23. J. Kleinberg and E. Tardos. Disjoint paths in densely embedded graphs. In *Proceedings of the 36th IEEE FOCS*, pages 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. In *Proceedings of IPCO*, 1998.
26. P. Kolman and C. Scheideler. Simple on-line algorithms for the maximum disjoint paths problem. In *Proceedings of the 13th ACM SPAA*, 2001.
27. K. Obata. Approximate max-integral-flow/min-multicut theorems. In *Proceedings of the 36th ACM STOC*, 2004.
28. P. Raghavan and C. D. Thompson. Randomized roundings: a technique for provably good algorithms and algorithms proofs. *Combinatorica*, 7:365–374, 1987.
29. N. Robertson and P. D. Seymour. An outline of a disjoint paths algorithm. *Paths, Flows and VLSI-design, Algorithms and Combinatorics*, 9:267–292, 1990.
30. A. Srinivasan. Improved approximations for edge-disjoint paths, unsplittable flow, and related routing problems. In *Proceedings of the 38th IEEE FOCS*, 1997.
31. K. Varadarajan and G. Venkataraman. Graph decomposition and a greedy algorithm for edge-disjoint paths. In *Proceedings of the ACM-SIAM SODA*, 2004.