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@cs.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 [31] 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 [22]. 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 [21], 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 [19], 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, 23]. 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 [16] 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 [23]. Our analysis is more like that of Obata [29], 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 [16], expanders [23, 28], grids and grid-like graphs [5, 6, 24, 25], and even-degree planar graphs [22]. For general graphs, the best approximation ratio for EDP in directed graphs is $O(\min(n^{2/3}, \sqrt{m}))$ [8, 26, 27, 32, 33], 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 [18]. 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 [22] 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 [30] 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 [15]. 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^*.

Given a non-negative weight function $\boldsymbol{\pi} : X \to \mathbb{R}^+$ on a set of nodes X in G, we use following definitions from [11].

Definition 1. ([11]) *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. ([11]) *A set X is $\boldsymbol{\pi}$-flow-linked in G if there is a feasible multi-commodity flow for the problem with demand $\mathsf{dem}(u, v) = \boldsymbol{\pi}(u)\boldsymbol{\pi}(v)/\boldsymbol{\pi}(X)$ between every unordered pair of terminals $u, v \in X$.*

Remark 1. Note this is a product flow with $\mathsf{dem}(u, v) = w(u)w(v)$, where $w(u) = \boldsymbol{\pi}(u)/\sqrt{\boldsymbol{\pi}(X)}$.

We have the following proposition immediately from the definitions above.

Proposition 1. ([11]) *If a set X is $\boldsymbol{\pi}$-flow-linked in G, then it is $\boldsymbol{\pi}/2$-cut-linked. If X is $\boldsymbol{\pi}$-cut-linked in G, then it is $\boldsymbol{\pi}/\beta(G)$-flow-linked, where $\beta(G)$ is the worst-case mincut-maxflow gap on product multicommodity flow instances on $\mathcal{G}$.*

Definition 3. ([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 $\boldsymbol{\pi} : V(\mathcal{G}) \to \mathbb{R}^+$ 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 $\boldsymbol{\pi}(\mathcal{G}) = \Omega(\mathsf{OPT}^*/\beta(\mathcal{G})\lambda(n))$.*

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 is in Section 8.

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

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

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 $\textsc{EmbedAndRoute}(G, T, \boldsymbol{\pi})$ 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 $p = 12(\ln n)/\epsilon^2\kappa = 1/(\omega \log^2 n + 1)$
  - 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 $\Omega(\mathsf{OPT}^*(\mathcal{G}, \mathcal{T})/f)$ 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 by extending a uniform sampling scheme from [19].

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 $\epsilon < 1$.*

Proof. Since X is $\boldsymbol{\pi}$-cut-linked in G hence $|\delta(S)| \geq \boldsymbol{\pi}(S \cap X)$, $\forall S$ such that $\boldsymbol{\pi}(S \cap X) \leq \boldsymbol{\pi}(X)/2$ in G. Let $\delta_j(S)$ denote the size of cut $(S, V \setminus S)$ in G^j. With probability $1 - O(\log^2 n/n^2)$, we have $|\delta_j(S)| \geq (1-\epsilon)p\,|\delta(S)| \geq (1-\epsilon)p\boldsymbol{\pi}(S \cap X))$, for all S such that $\boldsymbol{\pi}(S \cap X) \leq \boldsymbol{\pi}(X)/2$ and all j as shown in Theorem 5. Hence X is $(1 - \epsilon)\boldsymbol{\pi}/Z$-cut-linked in $G^j, \forall j$.

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 $p\,|\delta_G(S)|$.*

We give an overview of our proof by introducing a definition by [19], 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. Finally, we show the details of our proof to Theorem 5. For the sake of completeness, we also give Karger's proof to Theorem 6.

Definition 4. ([20]) *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.*

Proof. Recall the construction of a random subgraph $G^j, \forall j$, of G: on the same set of vertices as G, each edge $e \in E$ of the original graph G is placed in G^j independently with probability p. Hence, $G^j, \forall j$, is a p-skeleton of G by Definition4.

Theorem 6. ([20]) *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.*

Proof of Theorem 5: Define an indicator variable $X_e^j, \forall j, \forall e \in E$, such that $X_e^j = 1$ when e is placed in G^j, and 0 otherwise; hence X_e^j is a Bernoulli random variable with success probability p, $\forall j, \forall e$. Note that random variables $X_e^j, \forall j = 1, \ldots, 1/p$, are not independent; in fact, $\sum_{j=1}^{1/p} X_e^j = 1$ for all e.

Consider a cut $(S, \bar{S})$ of size c in G. Let $X_1^j, X_2^j, \ldots, X_c^j$ be the indicator variables that signal whether edges $e_1, e_2, \ldots, e_c$ of cut $(S, \bar{S})$ appear in G^j. Define $X_j(S, \bar{S}) = \sum_{y=1}^{c} X_y^j$ as the size of the cut in a random subgraph G^j of G. Given that $X_1^j, X_2^j, \ldots, X_c^j$ are i.i.d. random variables whose common distribution is the Bernoulli distribution with parameter p, we can apply Chernoff bound to obtain the following lemma.

Lemma 3. *Consider a cut $(S, V \setminus S)$ of size c in unweighted graph $G = (V, E)$. Let $X_j(S, \bar{S})$ be the size of the corresponding cut in a randomized split graph G^j. Then $\forall S$, $\forall j$, we have*

$$\mathbf{Pr}[|X_j(S, V \setminus S) - pc| \geq \epsilon pc] \leq 2e^{-\epsilon^2 pc/3}. \tag{4.5}$$

Lemma 4. ([14]) *Let X be a sum of independent Bernoulli random variables with success probability $p_1, \ldots, p_m$ and expected value $\mu = \sum p_i$. Then for $\epsilon \leq 1$*

$$\mathbf{Pr}[|X - \mu| \geq \epsilon\mu] \leq 2e^{-\epsilon^2\mu/3}.$$

Let $r = 2^n - 2$ be the number of cuts in graph G, and hence $G^1, \ldots, G^Z$, and let $c_1, \ldots, c_r$ be the expected values of the r cuts in a p-skeleton listed in nondecreasing order so that $p\kappa = c_1 \leq c_2, \ldots, \leq c_r$. Given a split graph $G^j, \forall j$, let $\mathcal{E}_k^j, \forall j, \forall k$ be the event that the value of a cut $X_j(S, V/S)$ in G^j diverges from its expectation c_k by more than ϵc_k. First we have $\mathbf{Pr}\left[\mathcal{E}_k^j\right] \leq 2e^{-\epsilon^2 c_k/3}$ by Lemma 3. We then apply a union bound to sum up (4.5) for all r cuts in $G^j, \forall j$.

Given that every random split subgraph $G^j, \forall j$, is a p-skeleton of G by Lemma 2, we apply Karger's statement as in the form below, to all subgraphs G^j with the following parameters: $p = 12(\ln n)/\epsilon^2\kappa$ and $\kappa = 12(\ln n)(\omega \log^2 n + 1)/\epsilon^2$ for a given ϵ; following Karger's proof to Theorem 6, we have:

Lemma 5. ([20]) $\forall G^j$, $\sum_{k=1}^{r} \mathbf{Pr}\left[\mathcal{E}_k^j\right] \leq O(1/n^d)$.

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 yields following:

$$\sum_{j=1}^{Z} \sum_{k=1}^{r} \mathbf{Pr}\left[\mathcal{E}_k^j\right] \leq O(\log^2 n/n^d) \tag{4.6}$$

Note that $\mathcal{E}_k^j, \forall j = 1, \ldots, Z$, are not independent, since the indicator random variables that contribute to value of $X_j(S, V/S)$ are not at all independent across all subgraphs. However, we only use a union bound that does not assume anything about dependency among events. ■

Proof of Theorem 6: ([20]) We give a sketch of Karger's proof as shown in [20] here for the sake of completeness. 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. The proof of this claim follows by using two lemmas:

Lemma 6. ([20]) *In an undirected graph, the number of α-minimum cuts is less than $n^{2\alpha}$.*

The "expected value" graph $\bar{G}$ of $G^j, \forall j$, is a weighted graph with all vertices and edges of the original unweighted graph $G = (V, E)$, and with edge weight p assigned to edge $e, \forall e \in E$. Note that the minimum cut of $\bar{G}$ is $p\kappa$, where κ is the minimum cut of G. Lemma 6 applied to $\bar{G}$, the "expected value" graph of a p-skeleton of G, states that the number of cuts within α factor of the minimum $p\kappa$ increases exponentially with α. On the other hand, the Chernoff bound says that one such cut diverges too far from its expected value decreases exponentially with α as shown in (4.5). Combining these two lemmas and balancing the exponential rates proves Theorem 6 using a union bound. ■

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 7.

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 7. ([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 3.

Lemma 8. *$\forall$ 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 $K = |\{X_i \in \mathcal{X} : X_i \subseteq U\}|$.*

Proof. With high probability, X is $\frac{(1-\epsilon)\boldsymbol{\pi}}{(\omega \log^2 n+1)}$-cut-linked in $G^1, \ldots, G^Z$, as shown in Lemma 1. Recall that in our clustering scheme, total weight of all terminals in one cluster is at least $W = \frac{(\omega \log^2 n+1)}{1-\epsilon}$, then $\forall j$,

$$\begin{aligned}
|\delta_{G^j}(U)| &\geq \sum_{x \in U} \frac{(1-\epsilon)\boldsymbol{\pi}(x)}{(\omega \log^2 n + 1)} \\
&\geq \frac{(1-\epsilon)}{(\omega \log^2 n + 1)} \sum_{i: X_i \subseteq U} \sum_{x \in X_i} \boldsymbol{\pi}(x) \\
&\geq \frac{(1-\epsilon)KW}{(\omega \log^2 n + 1)} \\
&\geq K
\end{aligned}$$

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 [21] 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 $M_1, \ldots, M_{\omega \log^2 n}$.

0. Given a set of points $V(H)$ of size k
1. for $t = 1$ to $\omega \log^2 n$
2. $\quad (S, \bar{S} = V(H) \setminus S) =$ KRV-FindCut$(V(H), \{M_k : k < t\})$ s.t. $|S| = \left|\bar{S}\right| = k/2$
3. $\quad M_t =$ FindMatch$(S, \bar{S})$ s.t. M_t is a matching between S and $\bar{S}$
4. Combine $M_1, \ldots, M_{\omega \log^2 n}$ to form the edge set F on vertices $V(H)$
5. End

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

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 [21]. At each round t, we use KRV-FindCut 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 [21]; they show the procedure in Figure 6.2 produces an expander H.

Theorem 8. ([21]) *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 9. 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 7.

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 $\bar{S} = \{X_{j_1}, \ldots, X_{j_{r/2}}\}$.

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 9. *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.*

We first prove the following lemma.

Lemma 10. *Every $s_0 - t_0$ cut has size at least $r/2$ in the flow graph G'.*

Proof. Let $(U, \bar{U})$ be a cut in the flow graph that separates s_0 from t_0; w.l.o.g., let U be subset such that $\boldsymbol{\pi}(U \cap X) \leq \boldsymbol{\pi}(X)/2$, and let $s_0 \in U$ (otherwise, we can just rename all the auxiliary nodes and the two subsets S and $\bar{S}$).

Consider any superterminal $X \in \mathcal{X}$ that we obtained through lemma 7; if X is contained either in U or $\bar{U}$, we call such a superterminal X uncut; otherwise, we say X is cut by $(U, \bar{U})$.

1. Let $K_c^s = \left|\{X \in S : X \cap U, X \cap \bar{U} \neq \emptyset\}\right|$ denote the number of superterminals in S that is cut by $(U, \bar{U})$.
2. Let $\overline{K_{uc}^s} = \left|\{X \in S : X \subseteq \bar{U}\}\right|$ be the number of superterminals in S that is contained in $\bar{U}$;
3. Let $K_{uc}^s = |\{X \in S : X \subseteq U\}|$ denote the number of superterminals in S that is contained in U; hence $K_{uc}^s + \overline{K_{uc}^s} + K_c^s = r/2$, where $r = |\mathcal{X}|$.
4. Let $K_c^t = \left|\{X \in \bar{S} : X \cap U, X \cap \bar{U} \neq \emptyset\}\right|$ denote the number of superterminals in $\bar{S}$ that is cut.
5. Let $K_{uc}^t = \left|\{X \in \bar{S} : X \subseteq \bar{U}\}\right|$ denote the number of superterminals in $\bar{S}$ that is contained in U.

Given that G is $\boldsymbol{\pi}$-cut-linked, we know that the sampled graph G^j is $(1 - \epsilon)\boldsymbol{\pi}/(\omega \log^2 n + 1)$-cut-linked *whp* by Lemma 1. Recall that in our clustering scheme, total weight of all terminals in one superterminal is at least $W = \frac{(\omega \log^2 n + 1)}{1-\epsilon}$. Note that there is at least one directed auxiliary edge crossing the cut for all superterminals except those in S that is contained in U or those in $\bar{S}$ that is contained in $\bar{U}$.

Thus we know

$$\begin{aligned}
|\delta_{G'}(U)| &\geq |\delta_{G^t}(U)| + K^t_{uc} + \overline{K^s_{uc}} + K^s_c + K^t_c \\
&\geq \frac{(1-\epsilon)\sum_{x\in U}\boldsymbol{\pi}(x)}{\omega\log^2 n+1} + K^t_{uc} + \overline{K^s_{uc}} + K^s_c + K^t_c \\
&\geq \frac{(1-\epsilon)(K^s_{uc}+K^t_{uc})W}{\omega\log^2 n+1} + K^t_{uc} + \overline{K^s_{uc}} + K^s_c + K^t_c \\
&\geq K^s_{uc} + \overline{K^s_{uc}} + K^s_c \\
&\geq r/2.
\end{aligned}$$

Hence we have shown that the size of every cut $(U, \bar{U})$ in the flow graph G' has size at least $r/2$.

Proof of Lemma 9: By Lemma 10, 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 $K = |V(H)| = \Omega(\boldsymbol{\pi}(G)/W)$, 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.*

We first prove the following lemma regarding a multicut L in H'.

Lemma 11. *$\exists$ a multicut of size at most $K/2a_3$ in the remaining graph H' of H.*

Proof. Let us first state the following lemma which follows from arguments of [17].

Lemma 12. *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 $s_i t_i \in T'$.*

Applying Lemma 12 to H', we have that there exists 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 prove Theorem 9, 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 $\Omega(K/\log^5 n)$.

Proof of Theorem 9: Recall that initially $\boldsymbol{\pi}(H) = \boldsymbol{\pi}(\mathcal{X}) \geq \boldsymbol{\pi}(G) - (W-1)$, since at most $W-1$ of $\boldsymbol{\pi}(G)$ is not assigned to any node in H, and each node in H has weight between W and $2W-1$ as in shown in proof of Lemma 7. Hence the total weight taken away from routing P terminal pairs of distance at most D is at most $DP(2W-1)$.

To facilitate our analysis, we first alter $\boldsymbol{\pi}$ slightly to generate a new function $\boldsymbol{\pi}'(i), \forall i \in V(H')$,

Procedure Alter($\boldsymbol{\pi}$, $\boldsymbol{\pi}'$): For a pair of terminals $uv \in T$ such that u takes away a certain amount of weight, we remove the same amount (as specified in condition 1 of Theorem 2) from $\boldsymbol{\pi}(v)$ if v remains in H' and define this updated weight of H' as $\boldsymbol{\pi}'(H')$. Thus we have $\boldsymbol{\pi}'(H') \geq \boldsymbol{\pi}(G) - (W-1) - 2DP(2W-1)$.

It is easy to see 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 $\boldsymbol{\pi}'(H')/2$.

Let L be the multicut that separates all remaining terminals pairs $T' \in T$ in H'. L cuts the graph H' and hence group nodes in $V(H')$ into clusters, such that weight of each cluster according to $\boldsymbol{\pi}'$ is less than half of the total remaining weight $\boldsymbol{\pi}'(H')$ of H', since each pair of terminals that contribute the same amount of weight to $\boldsymbol{\pi}'(H')$ must belong to different multicut clusters.

We then use L to find a weight-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 as shown in Lemma 13 and Lemma 14.

We build a $(1/4, 3/4)$-weight-balanced partition of H' in the following way: start two empty sides A and B, and start adding the connected components (after removing the multicut L) of H' to the smaller side repeatedly. Each component contains at most $\boldsymbol{\pi}'(H')/2$ due to condition 1 of Theorem 2 and Procedure Alter; in the end neither side can contain more than $3\boldsymbol{\pi}'(H')/4$ of weight; indeed, consider the step where, w.l.o.g, side A were put over $3/4$ of $\boldsymbol{\pi}'(H')$ by adding a component d: in that step, d could not have been added to A, since $\boldsymbol{\pi}'(A) \geq \boldsymbol{\pi}'(H')/4 \geq \boldsymbol{\pi}'(B)$ before d were added, given that $d \leq \boldsymbol{\pi}'(H')/2$.

Lemma 13. *Let $(U', V(H') \setminus U')$ be a $(1/4, 3/4)$-weight-balanced cut in H'. Consider any cut $(U, V(H) \setminus U)$ in H, such that $U' \subseteq U$ and $(V(H') \setminus U') \subseteq (V(H) \setminus U)$ before we route any of the P paths:*

$$\min\left(|U|, |V \setminus U|\right) \geq \frac{\boldsymbol{\pi}(G) - (W-1)}{4(2W-1)} - DP/2$$

Proof. Indeed, if U is the smaller side, $|U| \geq |U'|$; otherwise, we have $|V(H) \setminus U| \geq |V(H') \setminus U'|$. For both U' and $V(H') \setminus U'$, we have

$$\begin{aligned} |U'| &\geq \boldsymbol{\pi}'(H')/4(2W-1) \\ |V(H') \setminus U'| &\geq \boldsymbol{\pi}'(H')/4(2W-1) \\ &\geq \frac{(\boldsymbol{\pi}(G) - (W-1) - 2DP(2W-1))}{4(2W-1)} \end{aligned}$$

since each node in $V(H')$ has weight at most $2W-1$ despite alterations on terminal weights and $\boldsymbol{\pi}'(H') \geq \boldsymbol{\pi}(G) - (W-1) - 2DP(2W-1)$. Therefore

$$\min\left(|U|, |V \setminus U|\right) \geq \frac{\boldsymbol{\pi}(G) - (W-1)}{4(2W-1)} - DP/2$$

Lemma 14. *$|\delta_H(U)| \geq \phi(H)\left(\frac{K}{8} - DP/2\right)$, for any $(U, V(H) \setminus U)$ as defined in Lemma 13.*

Proof. By the edge expansion property of expander H, we get the lower bound on the size of the cut $(U, V \setminus U)$:

$$\begin{aligned} |\delta_H(U)| &\geq \phi(H) \min\left(|U|, |V \setminus U|\right) \\ &\geq \phi(H)\left(\frac{\boldsymbol{\pi}(G) - (W-1)}{4(2W-1)} - DP/2\right) \\ &\geq \phi(H)\left(\boldsymbol{\pi}(G)/8W + \boldsymbol{\pi}(G)/16W^2 - \frac{1}{8} - DP/2\right) \\ &\geq \phi(H)\left(\frac{K}{8} - DP/2\right) \end{aligned}$$

since $K \leq \boldsymbol{\pi}(G)/W$, given that every cluster must have weight at least W in H and $\boldsymbol{\pi}(G)/16W^2 = \Omega(\log^3 n) \geq 1/8$.

On the other hand, by Lemma 12, we know that the current size of the balanced cut in H' is at most the size of the multicut L given the construction of $(U', V(H') \setminus U')$:

$$|\delta_{H'}(U')| \leq |E| \log n/D = \frac{K\omega \log^2 n \log n}{2D} = \frac{K\omega}{2a_3} \tag{7.7}$$

The edge loss from the balanced cut $\mathsf{cap}(U, V(H) \setminus U)$ in H is caused by routing the P paths, which can take away at most $DP\omega \log^2 n$ number of edges. Thus we have:

$$\begin{aligned}
|\delta_{H'}(U')| + DP\omega \log^2 n &\geq |\delta_H(U)| \\
\frac{\omega K}{2a_3} + DP\omega \log^2 n &\geq \phi(H)\left(\frac{K}{8} - DP/2\right) \\
DP(\omega \log^2 n + \phi(H)/2) &\geq \phi(H)\frac{K}{8} - \frac{K\omega}{2a_3} \\
P &\geq \frac{\left(\phi(H)\frac{K}{8} - \frac{K\omega}{2a_3}\right)}{a_3 \log^3 n(\omega \log^2 n + \phi(H)/2)}
\end{aligned}$$

By taking $\phi(H) = 1/4$, $a_3 = 32\omega$, we have $D = 32\omega \log^3 n$ and $P \geq K/2048\omega^2 \log^5 n$, for a constant ω. ■

8 An Outline of the Decomposition Procedure

In this section, we first sketch a proof to Theorem 11, which states a more refined and stronger version of Theorem 2. Actual proof of Theorem 11 is shown in Section 10.

8.1 The CKS Flow-Linked Decomposition Theorem

We first transform $(\mathcal{G}, \mathcal{T})$ to a set of flow-linked instances by following a decomposition procedure in [11], the outcome of which is summarized in the following theorem.

Theorem 10. (CKS2005 [11]) *Let $\mathsf{OPT}^*(\mathcal{G}, \mathcal{T})$ be a solution to the LP for a given instance $(\mathcal{G}, \mathcal{T})$ of EDP in an input graph $\mathcal{G}$. One can efficiently compute a partition of $\mathcal{G}$ into node-disjoint induced subgraphs $G_1, G_2, \ldots, G_\ell$, and weight functions $\boldsymbol{\pi} : V(G_i) \to \mathbb{R}^+$ 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. *$\boldsymbol{\pi}_i(u) = \boldsymbol{\pi}_i(v)$ for $uv \in \mathcal{T}_i$.*
2. *X_i is $\boldsymbol{\pi}_i$-flow-linked in G_i.*

3. $\sum_{i=1}^{\ell} \boldsymbol{\pi}_i(X_i) = \Omega(\mathsf{OPT}^*(\mathcal{G}, \mathcal{T})/\lambda(n))$, *where* $\lambda(n) = 10\beta(\mathcal{G}) \log \mathsf{OPT}^*(\mathcal{G}, \mathcal{T})$.

Remark 2. Although the statement of condition 1 in the CKS decomposition theorem assumes that each node u belongs to only a single terminal pair in $\mathcal{T}$, their actual proof does not depend on such an assumption.

The proof of the theorem appears in [11]. They use this procedure as the first step in a two-step transformation from the optimal multicommodity flow solutions $\bar{f}$ to obtain sets of well-linked terminal sets, that eventually leads to an $O(\log^2 K)$-approximation for the ANF problem described in Section 1, where $K = |\mathcal{T}|$. We place details regarding this decomposition in the Section 10. From now on, we refer to both $(G_i, \mathcal{T}_i)$ and (G_i, X_i) as a $\boldsymbol{\pi}_i$-flow-linked instance without differentiation.

8.2 Processing Subgraphs to Maintain Mincut Condition

We treat the induced subproblems $(G_i, \mathcal{T}_i), \forall i$ independently. Given (G_i, X_i) such that X_i is $\boldsymbol{\pi}_i$-flow-linked in G_i, there are two post-processing stages.

1. **Min-cut processing stage.** Formally, let $V(G_i)$ be the current set of vertices of G_i. We keep cutting off the smaller side S of a minimum cut, in terms of weight $\boldsymbol{\pi}_i$, from G_i when $\mathsf{cap}(S, V(G_i) \setminus S)$ is less than $\hat{c}$, until every cut in G_i is at least $\hat{c}$, where we set $\hat{c} = \Omega(\log^3 n)$.
   By cutting off, we remove both nodes in S and edges that are adjacent to S in current G_i; this includes the cases when we get rid of any single node whose degree fall below $\hat{c}$ from its original degree of $\Omega(\log^5 n)$. We call such a stage a min-cut processing stage.
2. **Sparsest-cut processing stage.** In order to guarantee that we have an instance X_i' that is $\boldsymbol{\pi}_i'$-flow-linked in G_i for a new weight function $\boldsymbol{\pi}_i'$, we need to further "mute" some terminals with a positive weight under $\boldsymbol{\pi}$ by setting their weight to zero under $\boldsymbol{\pi}_i'$. This way, we can guarantee that every cut in G_i is good with respect to a product multicommodity flow demand that is defined based on the new weight function $\boldsymbol{\pi}_i'$. We emphasize that we do not remove any nodes or edges in this stage; hence the min-cuts are guaranteed to be $\Omega(\log^3 n)$.

8.3 A Modified Flow-Linked Decomposition Theorem

Therefore, we have the following theorem about the instances that we have by the end of this post-processing stage. The proof of this theorem is in Section 10.

Theorem 11. *Given a graph $\mathcal{G}$ with min-cut value $C^0 \geq (4a_0\lambda(n)+a_0+2)\hat{c}$, for some $a_0 \geq 2$. By the end of the sparsest-cut processing, we obtain a set of node-disjoint induced subgraphs $\hat{G}_1, \ldots, \hat{G}_\ell$, all with min-cut $\hat{c}$, and the corresponding disjoint subsets $\mathcal{T}_1', \ldots, \mathcal{T}_\ell'$ of $\mathcal{T}$, such that terminals pairs in $\mathcal{T}_i'$ belong to $\hat{G}_i$ and there exist a set of weight functions $\boldsymbol{\pi}_i' : V(\hat{G}_i) \to \mathbb{R}^+$ with the following properties. Let X_i' be the set of terminals of T_i'.*

1. *there are* $\alpha_1, \ldots, \alpha_k$ *such that* $\forall u$ *in a subgraph* $\hat{G}_i$, $\boldsymbol{\pi}(u) = \sum_{i:s_i=u, t_i \in \hat{G}_i} \alpha_i x_i$, *(note that this implies* $\forall s_i t_i \in \mathcal{T}'_i$, x_i *contributes the same amount of weight to* $\boldsymbol{\pi}'_i(s_i)$ *and* $\boldsymbol{\pi}'_i(t_i)$*);*
2. X'_i *is* $\boldsymbol{\pi}'_i$*-flow-linked in* G'_i*;*
3. $\forall u$ *in a subgraph* $\hat{G}_i$ *s.t.* $\boldsymbol{\pi}'_i(X'_i) \geq \Omega(\log^3 n)$, $\boldsymbol{\pi}'_i(u) \leq \sum_{i:s_i=u, t_i \in \hat{G}_i} \frac{x_i}{\beta(\mathcal{G})\lambda(n)}$*;*
4. $\sum_{i=1}^{\ell} \boldsymbol{\pi}'_i(X'_i) = \Omega\left(\frac{\mathsf{OPT}^*(\mathcal{G},\mathcal{T})}{\lambda(n)\beta(\mathcal{G})}\right)$, *where* $\lambda(n) = \beta(\mathcal{G}) \log \mathsf{OPT}^*(\mathcal{G}, T)$ *and* $\beta(G)$ *is the worst-case mincut-maxflow gap on product multicommodity flow instances on* $\mathcal{G}$*.*

Finally, we define a weight function on $V(\mathcal{G})$ as follows: (a) $\forall i, \forall u \in \hat{G}_i$, where $\hat{G}_i$ is a subgraph of $\mathcal{G}$, we assign $\boldsymbol{\pi}(u) = \boldsymbol{\pi}'_i(u)/2$; and (b) assign $\boldsymbol{\pi}(u) = 0$, for nodes of $V(G)$ not in any $\hat{G}_i$. We thus have defined the weight function $\boldsymbol{\pi} : V(G) \to \mathbb{R}^+$ on the entire set of nodes of $\mathcal{G}$ as required by Theorem 2 with the same decomposition as we obtain for Theorem 11.

9 Details Regarding CKS Flow-linked Decompositions

The following notation appears in proof of Theorem 10 as in [11]. We will inherit these in our proofs in Section 10. Let $H = (V(H), E(H))$ be a node induced subgraph of $G = (V, E)$.

- $\gamma(\mathcal{G}) = \mathsf{OPT}^*(\mathcal{G}, \mathcal{T})$.
- $\gamma(H) = \sum_{P \in \mathcal{P}: P \in H} \bar{f}(P)$: the total flow induced in H by the original flow $\bar{f}$; it counts flow only on flows paths $\bar{f}(P)$ from the the original flow path decomposition that are completely contained in H. $\mathcal{P}$ refers to the entire set of paths from the original flow decomposition.
- $\gamma(u, H)$: the flow in H for u, hence $\gamma(H) = 1/2 \sum_{u \in V(H)} \gamma(u, H)$.

Recall the following results from their decomposition procedure. Let $G_1, G_2, \ldots, G_\ell$ be the subgraphs produced by the decomposition.

1. If $\gamma(G_i) \leq \lambda(n)/10$, assign $\boldsymbol{\pi}_i(u) = \boldsymbol{\pi}_i(v) = 1$ for some pair $uv \in \mathcal{T}_i$ with positive flow in G_i; and $\boldsymbol{\pi}_i(y) = 0$ for $y \neq u, v$. Hence one can just route a unit flow between the chosen pair $uv \in \mathcal{T}_i$ along an integral path; such a path exists since G_i is a connected component.
2. Else, for $\gamma(G_i) > \lambda(n)/10$, X_i is $\boldsymbol{\pi}_i$-flow-linked in G_i, where $\boldsymbol{\pi}_i$ is defined as follows for G_i; Recall $\lambda(n) = 10\beta(\mathcal{G}) \log \mathsf{OPT}^*(\mathcal{G}, \mathcal{T})$.
   (a) $\boldsymbol{\pi}_i(u) = \frac{\gamma(u, G_i)}{\lambda(n)}, \forall u \in X_i$
   (b) $\boldsymbol{\pi}_i(u) = \gamma(u, G_i) = 0$ for $u \notin X_i$

Remark 3. For both cases, the CKS weight function on $V(G_i)$ satisfy $\boldsymbol{\pi}_i(G_i) = \Omega\left(\frac{\gamma(G_i)}{\lambda(n)}\right)$, given that $\boldsymbol{\pi}_i(G_i) = \sum_{x \in X_i} \boldsymbol{\pi}_i(x) = \sum_{x \in V(G_i)} \frac{\gamma(x, G_i)}{\lambda(n)} = \frac{2\gamma(G_i)}{\lambda(n)}$; And the flow that one route in G_i satisfies the following two equivalent conditions.

1. Define $\forall uv \in V(G_i)$,

$$\mathsf{dem}^{\omega}(u,v) = \frac{\gamma(u,G_i)\gamma(v,G_i)}{\gamma(G_i)}, \tag{9.8}$$

as demands for the multicommodity product flow problem based on original induced flow values at each node $u \in V(G_i)$ of $\bar{f}$ in G_i; in $G_i, \forall i$, the concurrent max-flow value f for product flow $\mathsf{dem}^{\omega}(u,v)$, satisfy

$$f \geq f_0 = \frac{1}{2\lambda(n)}. \tag{9.9}$$

Thus $f_0\mathsf{dem}^{\omega}(u,v)$ units of demands can be simultaneously routed $\forall uv$ in G_i with congestion 1.

2. For a scaled-down product flow problem $\mathsf{dem}^{\boldsymbol{\pi}_i}(u,v)$, such that each demand is f_0 of the original, $\forall uv \in V(G_i)$,

$$\mathsf{dem}^{\boldsymbol{\pi}_i}(u,v) = \frac{\boldsymbol{\pi}_i(u)\boldsymbol{\pi}_i(v)}{\boldsymbol{\pi}_i(X_i)} = \frac{\gamma(u,G_i)\gamma(v,G_i)}{2\lambda(n)\gamma(G_i)} = \frac{\mathsf{dem}^{\omega}(u,v)}{2\lambda(n)} = f_0\mathsf{dem}^{\omega}(u,v),$$

there is is a feasible flow in G_i since the concurrent max-flow value is at least 1.

Depending on the context, we may prefer to use the original product flow $\mathsf{dem}^{\omega}(u,v)$ than the feasible product flow $\mathsf{dem}^{\boldsymbol{\pi}_i}(u,v)$, or the other way around.

10 An Analysis on Postprocessing to Maintain Cut Conditions

The analysis of this section will lead to the proof of Theorem 11 eventually. Throughout this section, we keep reducing the set of terminals pairs of $\mathcal{T}_i$ that are relevant, in the sense that these pairs will remain to be candidate pairs that we eventually route edge disjointly in $\mathcal{G}$. Therefore, we keep track of the following set of parameters in each subgraph G_i that we obtain through flow decomposition:

- $\mathcal{T}_i$: the induced pairs of $\mathcal{T}$ in G_i that we still consider to route edge disjointly.
- A weight function $\boldsymbol{\pi}_i$ defined on the $V(G_i)$, with positive values only on terminals X_i of $\mathcal{T}_i$.

Finally, we use $\mathsf{remaining\text{-}flow}$ to keep track of the total remaining flows of $\bar{f}$ between terminal pairs in $\mathcal{T}_i$, across all i; note that $\mathsf{remaining\text{-}flow}$ is the lower bound on $\sum_i |\mathcal{T}_i|$.

By the end of the CKS flow decomposition, $\mathcal{T}_i$ is the induced pairs of $\mathcal{T}$ in G_i. There exists at least one flow path between a pair of terminals $uv \in T_i$, with a positive amount of flow, from original flow path decomposition of $\bar{f}$ that is entirely contained in G_i. We lose at most half of $\bar{f}$, where $|\bar{f}| = \mathsf{OPT}^*(\mathcal{G},\mathcal{T})$,

because the number of edges that were cut during flow decomposition is at most $\mathsf{OPT}^*/2 = \gamma(G)/2$; hence

$$\textsf{remaining-flow} \geq \sum_{i=1}^{\ell} \gamma(G_i) \geq \mathsf{OPT}^*(\mathcal{G}, \mathcal{T})/2 = \gamma(G)/2, \tag{10.10}$$

and the total amount of the weights across all clusters is at least:

$$\sum_{i=1}^{\ell} \boldsymbol{\pi}_i(X_i) = \sum_{i=1}^{\ell} 2\gamma(G_i)/\lambda(n) = \Omega(\mathsf{OPT}^*(\mathcal{G}, \mathcal{T})/\lambda(n)). \tag{10.11}$$

We are going to keep computing the original flows of $\bar{f}$ that we lose during the post-processing stages.

We specify the following parameters that are related to minimum cuts:

1. $\hat{c}$: the smallest minimum cut value that we allow in $G_i, \forall i$, which is $\theta(\log^3 n)$.
2. $\mathcal{C}^0$: the minimum cut value in original graph $\mathcal{G}$, which is $\Omega(\log^5 n)$.
3. $\ell(S) = \mathsf{cap}(S, V \setminus S)$: size of a cut $(S, V \setminus S)$ in original graph $G = (V, E)$.
4. $\mathsf{LOSS} \leq \mathsf{OPT}^*(\mathcal{G}, \mathcal{T})/2$: number of edges that are cut during the CKS flow-decomposition process.

We analyze the minimum cut processing stage in the next two sections. Formally, let $V(G_i)$ be the current set of vertices of G_i. We keep cutting off the smaller side S of a minimum cut, in terms of weight $\boldsymbol{\pi}_i$, from G_i when $\mathsf{cap}(S, V(G_i) \setminus S)$ is less than $\hat{c}$, until every cut in G_i is at least $\hat{c}$. By cutting off, we remove both nodes in S and edges that are adjacent to S in current G_i.

Let $S_i^1, S_i^2, \ldots, S_i^{x_i}$ be the sets of vertices that we take away from G_i and in that order. We define the following notation to track this process of updating G_i.

- $G_i^0 = (V_i^0, E_i^0)$: the subgraph G_i before any of $S_i^t, t = 1, \ldots, x_i$ have been take out.
- X_i^0: the set of terminals of G_i^0 right after flow decomposition, such that X_i^0 is $\boldsymbol{\pi}_i$-flow-linked in G_i^0 as guaranteed by CKS decomposition.
- $G_i^t = (V_i^t, E_i^t), \forall t = 1, \ldots, x_i$: the remaining subgraph of G_i^0 after removing $S_i^1, \ldots, S_i^t$ and their adjacent edges; hence $V_i^t = V_i^0 \setminus \cup_{j=1,\ldots,t} S_i^j$.
- $\hat{G}_i = (\hat{V}_i, \hat{E}_i) = G_i^{x_i} = (V_i^{x_i}, E_i^{x_i})$ be the remaining subgraph of G_i^0 by the end of the min-cut processing stage.

10.1 Bound Edges Lost Due to Min-Cut Processing

Denote the number of edges that we take away from G_i^0 due to the min-cut processing by $\textsf{edge-loss}_i, \forall i$.

Definition 5. *$\textsf{edge-loss}_i$ is the sum of capacities of the minimum cuts that have caused $S_i^1, \ldots, S_i^{x_i}$ to be cut off from $G_i, \forall i$. Denote the sum of $\textsf{edge-loss}_i$ across all i with $\textsf{edge-loss}$,*

$$\textit{edge-loss} = \sum_{i=1,2,\ldots} \textit{edge-loss}_i = \sum_{i=1,2,\ldots} \sum_{t=1,\ldots,x_i} \textit{cap}(S_i^t, V_i^t).$$

Remark 4. Note that the number of edges that we take away from the final set of nodes $V(G_i) = V_i^{x_i} = V_i^0 \setminus \cup_{j=1,\ldots,x_i} S_i^j$ during the min-cut processing stage is upper bounded, and in fact may be smaller than $\mathsf{edge\text{-}loss}_i, \forall i$.

We prove the following lemma in this section.

Lemma 15. *The total number of edges that we take away from decomposed subgraphs $G_i^0, G_i^1, \ldots$ is at most*

$$\mathit{edge\text{-}loss} = \sum_{i=1,2,\ldots} \mathit{edge\text{-}loss}_i \leq \frac{2\mathit{LOSS} \cdot \hat{c}}{\mathcal{C}^0 - 2\hat{c}}. \tag{10.12}$$

Proof. We use a potential function $\psi(G_i)$ to count the number of edges we lose from nodes currently in G_i, as compared to the original graph $G = (V, E)$, while G_i keeps shrinking due to its min-cut processing. The counting process is as follows. We start with a component G_i such that $\psi_i^0 = \mathsf{LOSS}_i$ denotes the number of edges that we initially lose from nodes in G_i^0 right after the CKS flow decomposition procedure. Hence

$$\psi_i^0 = \psi(G_i^0) = \mathsf{LOSS}_i \geq 0, \tag{10.13}$$

and

$$\sum_{i=1,2,\ldots} \mathsf{LOSS}_i = 2\mathsf{LOSS}. \tag{10.14}$$

When a subset S is cut off, it claims away some credit from the current $\psi(G_i)$, since S is cut off because $\mathsf{cap}(S, V \setminus S)$ has decreased from above $\mathcal{C}^0$ to its current size in G_i, $\mathsf{cap}(S, V(G_i) \setminus S) \leq \hat{c}$ due to edges lost from nodes in S during CKS flow decomposition. That is, the amount of edge loss from nodes in S has contributed to the current value of $\psi(G_i)$.

Let ψ_i^t be value of $\psi(G_i)$ after taking t sets of vertices $S_i^1, \ldots, S_i^t$ and their adjacent edges away from G_i. Let $(S_i^{t+1}, V_i^t \setminus S_i^{t+1})$ be the minimum cut in G_i^t, and hence S_i^{t+1} be the $(t+1)^{st}$ set of vertices that we cut off from G_i because $\mathsf{cap}(S_i^{t+1}, V_i^t \setminus S_i^{t+1})$ is less than $\hat{c}$. The amount of credit S_i^{t+1} takes away from $\psi(G_i)$ is $(\mathsf{cap}(S_i^{t+1}, V \setminus S_i^{t+1}) - \mathsf{cap}(S_i^{t+1}, V_i^t \setminus S_i^{t+1}))$ and the credit it puts back is $\mathsf{cap}(S_i^{t+1}, V_i^t \setminus S_i^{t+1})$, since we remove edges in $(S_i^{t+1}, V_i^t \setminus S_i^{t+1})$ from G_i^t, in addition to the subgraph induced by S_i^{t+1} in G_i^t.

Let us denote the size of the original cut $(S_i^{t+1}, V \setminus S_i^{t+1})$ in $\mathcal{G}$ with

$$\ell_i^{t+1} = \ell(S_i^{t+1}) = \mathsf{cap}(S_i^{t+1}, V \setminus S_i^{t+1}) \geq \mathcal{C}^0. \tag{10.15}$$

Hence, we update $\psi(G_i)$ as follows,

$$\begin{aligned}\psi_i^{t+1} &= \psi_i^t - (\mathsf{cap}(S_i^{t+1}, V \setminus S_i^{t+1}) - \mathsf{cap}(S_i^{t+1}, V_i^t \setminus S_i^{t+1})) + \mathsf{cap}(S_i^{t+1}, V_i^t \setminus S_i^{t+1}) \\ &= \psi_i^t - (\ell_i^{t+1} - \mathsf{cap}(S_i^{t+1}, V_i^{t+1})) + \mathsf{cap}(S_i^{t+1}, V_i^{t+1}).\end{aligned}$$

Since $\mathsf{cap}(S_i^{t+1}, V_i^{t+1}) \leq \hat{c}$, we have $\psi_i^{t+1} \leq \psi_i^t - (\ell_i^{t+1} - \hat{c}) + \hat{c}$.

Since the credit that a cut puts back is much less than the credit that it spent, there is only finite number x_i of such small cuts in G_i, $\forall i$. By the end of x_i rounds, there must be a non-negative credit in $\psi(G_i)$, since nodes in current G_i can never gain any edges. Hence

$$0 \le \psi(G_i) = \psi_i^x \le \mathsf{LOSS}_i - (\ell_i^1 - \hat{c}) + \hat{c} - (\ell_i^2 - \hat{c}) + \hat{c} - \ldots - (\ell_i^{x_i} - \hat{c}) + \hat{c}.$$

Summing the above inequalities over all i,

$$\sum_{i=1,2,\ldots} x_i \cdot C^0 \le \sum_{i=1,2,\ldots} \sum_{j=1,2,\ldots,x_i} \ell_i^j \le 2 \cdot \mathsf{LOSS} + 2 \sum_{i=1,2,\ldots} x_i \cdot \hat{c}.$$

Hence the total number of minimum cuts across all G_i that we process is

$$\sum_{i=1,2,\ldots} x_i \le \frac{2\mathsf{LOSS}}{\mathcal{C}^0 - 2\hat{c}}. \tag{10.16}$$

Denote the sum of $\mathsf{edge\text{-}loss}_i$ across all i with $\mathsf{edge\text{-}loss}$ and thus

$$\mathsf{edge\text{-}loss} = \sum_{i=1,2,\ldots} \mathsf{edge\text{-}loss}_i = \sum_{i=1,2,\ldots} \sum_{t=1,\ldots,x_i} \mathsf{cap}(S_i^t, V_i^t) \tag{10.17}$$

$$\le \sum_{i=1,2,\ldots} x_i \cdot \hat{c} \le \frac{2\mathsf{LOSS} \cdot \hat{c}}{\mathcal{C}^0 - 2\hat{c}}. \tag{10.18}$$

10.2 Bound the Flow Lost Due to Min-Cut Processing

Lemma 16. *The total flow of $\bar{f}$ that we lose from min-cut processing is*

$$\mathit{flow\text{-}loss}_1 \le \frac{2\mathit{LOSS} \cdot \hat{c}}{\mathcal{C}^0 - 2\hat{c}}(2\lambda(n) + 1/2). \tag{10.19}$$

Proof. For a set of nodes $S_i^t \in V_i^0, \forall t = 1, \ldots, x_i$, in $G_i^0 = (V_i^0, E_i^0)$, we denote the size of cut $(S_i^t, V_i^0 \setminus S_i^t)$ with $\mathcal{B}_i^t = \mathsf{cap}(S_i^t, V_i^0 \setminus S_i^t)$. $\mathcal{B}_i^t$ determines the amount of flow of $\bar{f}$ that we take away from $\gamma(G_i)$ as we remove S_i^t from G_i as the smaller side of a min-cut (S_i^t, V_i^t) in G^{t-1}.

A closer examination of the above cutting process shows that

$$\sum_{t=1,2,\ldots,x_i} \mathcal{B}_i^t \le 2\mathsf{edge\text{-}loss}_i, \tag{10.20}$$

and

$$\sum_{i=1,2,\ldots} \sum_{t=1,2,\ldots,x_i} \mathcal{B}_i^t \le 2\mathsf{edge\text{-}loss}, \tag{10.21}$$

since the edges in $\mathcal{B}_i^t$ come either from previous min-cuts: $\{(S_i^j, V_i^j), \forall j < t\}$, or from new edges that contribute to $\{(S_i^t, V_i^t)\}$; in addition, each edge e counted in $\mathsf{edge\text{-}loss}_i$ can be used at most twice toward $\sum_{t=1}^{x_i} \mathcal{B}_i^t$, once for each of the two neighboring sets in $\{S_i^t, t = 1, \ldots, x_i\}$ that share $e \in G_i^0$.

Hence fix $\mathcal{B}_i^t$ for some t. We now calculate the amount of flows of $\bar{f}$ that we lose by cutting off S_i^t. The flow that we lose falls into one of the four types:

1. flow whose paths are entirely contained in the subgraph of G_i induced by S_i^t;
2. flow that has to go through edges that are counted in $\mathcal{B}_i^t$, but not counted in (S_i^t, V_i^t);
3. flow that has to cross (S_i^t, V_i^t) with at least one endpoint in S_i^t;
4. flow with both endpoints $u'v' \in V_i^t$ such that the flow path intersects the min-cut (S_i^t, V_i^t) at least twice.

Flow of type 1 is counted in $\sum_{u \in S_i^t} \gamma(u, G_i)$ twice. Flow of type 2 has been counted before when S_i^j were cut off for some $j < t$. Flow of type 3 contributes its flow amount once to $\sum_{u \in S_i^t} \gamma(u, G_i)$ and once to the usage of $\mathsf{cap}(S_i^t, V_i^t)$. Flow of type 4 are counted twice in the usage of $\mathsf{cap}(S_i^t, V_i^t)$.

Note that flow that crosses cut (S_i^t, V_i^t) either has been counted in $\sum_{u \in S_i^t} \gamma(u, G_i)$ at least once or it crosses (S_i^t, V_i^t) at least twice. Hence $1/2(\sum_{u \in S_i^t} \gamma(u, G_i) + \mathsf{cap}(S_i^t, V_i^t))$ upper bounds the amount of flow that we lose from $\bar{f}$, that has not been counted earlier, due to cutting off induced subgraph of S_i^t from G_i^{t-1}:

$$\begin{aligned}\frac{1}{2}\sum_{u \in S_i^t} \gamma(u, G_i) + \frac{1}{2}\mathsf{cap}(S_i^t, V_i^t) &\leq \frac{1}{2}\boldsymbol{\pi}_i(S_i^t \cap X_i^0)\lambda(n) + \frac{1}{2}\mathsf{cap}(S_i^t, V_i^t) \\ &\leq \mathsf{cap}(S_i^t, V_i^0 \setminus S_i^t)\lambda(n) + \frac{1}{2}\mathsf{cap}(S_i^t, V_i^t) \\ &\leq \mathcal{B}_i^t\lambda(n) + \frac{1}{2}\mathsf{cap}(S_i^t, V_i^t),\end{aligned}$$

where second inequality is due to the fact that X_i^0 is $\boldsymbol{\pi}_i$-flow-linked and Proposition 1, which implies that X_i^0 is $\boldsymbol{\pi}_i/2$-cut-linked in G_i^0.

Sum over all $S_i^t, \forall t$, we obtain the total flow lost:

$$\begin{aligned}\mathsf{flow\text{-}loss}_1 &= \sum_{i=1,2,\ldots} \sum_{t=1,\ldots,x_i} (\mathcal{B}_i^t\lambda(n) + \frac{1}{2}\mathsf{cap}(S_i^t, V_i^t)) && (10.22)\\ &\leq 2\mathsf{edge\text{-}loss} \cdot \lambda(n) + \frac{1}{2}\mathsf{edge\text{-}loss} && (10.23)\\ &\leq \frac{2\mathsf{LOSS} \cdot \hat{c}}{\mathcal{C}^0 - 2\hat{c}}(2\lambda(n) + 1/2). && (10.24)\end{aligned}$$

Let $1/a_0$ denote the ratio of amount of flow of $\bar{f}$ that we lose during min-cut processing with respect to LOSS in CKS flow decomposition:

$$\frac{\mathsf{flow\text{-}loss}_1}{\mathsf{LOSS}} \leq \frac{1}{a_0}. \tag{10.25}$$

Thus we require

$$\frac{\mathsf{flow\text{-}loss}_1}{\mathsf{LOSS}} \leq \frac{(2\lambda(n) + 1/2) \cdot 2\hat{c}}{\mathcal{C}^0 - 2\hat{c}} = \frac{(4\lambda(n) + 1) \cdot \hat{c}}{\mathcal{C}^0 - 2\hat{c}} \leq \frac{1}{a_0}. \tag{10.26}$$

Given an a_0, in order to satisfy (10.26), we require

$$\mathcal{C}^0 \geq (4a_0\lambda(n) + a_0 + 2) \cdot \hat{c}. \tag{10.27}$$

Plugging (10.27) in (10.17), we obtain the following bound on edge loss due to post-processing of G_i:

$$\mathsf{edge\text{-}loss} \leq \frac{2\mathsf{LOSS} \cdot \hat{c}}{\mathcal{C}^0 - 2\hat{c}} \leq \frac{2\mathsf{LOSS} \cdot \hat{c}}{a_0(4\lambda(n)+1) \cdot \hat{c}} = \frac{\mathsf{LOSS}}{a_0(2\lambda(n) + \frac{1}{2})}. \tag{10.28}$$

10.3 Obtain the Final Set of Terminals

Recall that $G_i^0 = (V_i^0, E_i^0)$ denote the subgraph G_i we obtain through CKS flow decomposition before any subset of nodes have been removed; $\hat{G}_i = (\hat{V}_i, \hat{E}_i), \forall i$ are the remaining subgraphs of $G_i, \forall i$ at the end of the min-cut processing stage. By (10.26), the total flow of $\bar{f}$ that remains is the sum of flow of $\bar{f}$ induced in $\hat{G}_i$, across all i,

$$\mathsf{remaining\text{-}flow} = \sum_{i=1,2,\ldots} \gamma(\hat{G}_i) \geq \frac{\mathsf{OPT}^*(\mathcal{G}, \mathcal{T})}{2} - \mathsf{flow\text{-}loss}_1 \tag{10.29}$$

$$\geq \frac{1}{2}\mathsf{OPT}^*(\mathcal{G}, \mathcal{T})(1 - \frac{1}{a_0}), \tag{10.30}$$

where $\mathsf{flow\text{-}loss}_1 = \mathsf{LOSS}/a_0$ and $\mathsf{LOSS} \leq \mathsf{OPT}^*(\mathcal{G}, \mathcal{T})/2$.

In the sparsest-cut processing, we remove $P_i^1, P_i^2, \ldots, P_i^{y_i}$ from the graph $\hat{G}_i$ that do not meet a certain sparsest cut condition. In the end, we have a subgraph G_i' that does meet the sparsest cut condition on the demands in the remaining subgraph. Now we assign a zero weight to all vertices in the removed regions to zero out demands on these regions and put $P_i^1, P_i^2, \ldots, P_i^{y_i}$ all back in. This graph $\hat{G}_i$ is only more connected with regard to the non removed demand induced by $\bar{f}$ inside $G_i', \forall i$. Hence we emphasize that $\hat{G}_i, \forall i = 1, \ldots, \ell$ are the set of subgraphs that we pass on to the next stage. We give an algorithm for computing the final disjoint subsets $\mathcal{T}_1', \ldots, \mathcal{T}_\ell'$ of $\mathcal{T}$ such that terminal pairs in $\mathcal{T}_i'$ belong to G_i', and hence $\hat{G}_i, \forall i$, and assigning a positive weight to the set of terminals in $\mathcal{T}_i', \forall i$.

In the rest of this section, we prove Theorem 11.

Proof of Theorem 11: Given a subgraph $\hat{G}_i = (\hat{V}_i, \hat{E}_i)$, we use the procedure as in Figure 10.3 to update $\hat{G}_i$ recursively by muting regions that do not satisfy the sparsest cut condition; by "muting" a region P, we treat nodes in P and their adjacent edges as if they were removed from $\hat{G}_i$ during the sparsest-cut processing stage, although in the end, we retain these regions entirely in $\hat{G}_i$. We define the following parameters given a remaining subgraph $\hat{G}_i^t$ of G^i after muting some regions, $P_i^1, \ldots, P_i^{t-1}$.

1. $\hat{G}_i^t = (\hat{V}_i^t, \hat{E}_i^t)$: the remaining subgraph of $\hat{G}_i$ after muting nodes in $P_i^1, \ldots, P_i^t$ and their adjacent edges. $\hat{V}_i^t = \hat{V}_i \backslash \cup_{j=1,\ldots,t} P_i^j$ is the remaining set of vertices in $\hat{G}_i$ at stage t.

0. Given a subgraph $\hat{G}_i$.
1. If $\gamma(\hat{G}_i) \le (a_1/4)\beta\lambda(n)$, $\boldsymbol{\pi}'_i(u) = \boldsymbol{\pi}'_i(v) = 1$ for some pair $uv \in \mathcal{T}'_i$ with positive flow in $\hat{G}_i$; and $\boldsymbol{\pi}'_i(y) = 0$ for $y \neq u, v$.
   Hence we can just route a unit flow between the chosen pair $uv \in \mathcal{T}'_i$ along an integral path; such a path exists since $\hat{G}_i$ is a connected component.
2. Suppose that $\gamma(\hat{G}_i) > (a_1/4)\beta\lambda(n)$. For $\mathsf{dem}(u,v) = \gamma(u,\hat{G}_i)\gamma(v,\hat{G}_i)/\gamma(\hat{G}_i)$, let f' be the maximum concurrent flow for this instance.
   (a) if $f' \ge f_1$, set $\boldsymbol{\pi}'_i(u) = \frac{\gamma(u,\hat{G}_i)}{(a_1/2)\beta\lambda(n)}$ $\forall u \in \hat{V}_i$ and stop.
   (b) else $f' < f_1$, find an approximate sparsest cut such that $\frac{\mathsf{cap}(S,\hat{V}_i \setminus S)}{\mathsf{dem}(S,\hat{V}_i \setminus S)} \le \beta f'$.
   set $\boldsymbol{\pi}'_i(u) = 0$, $\forall u \in S$, and
   shut off edges in $\delta^0(S) = (S, \hat{V}_i \setminus S)$
   so that we recurse on $\hat{G}_i[V(\hat{G}_i) \setminus S]$.
3. End

Fig. 10.4. Algorithm FINDING SPARSEST CUTS

2. $\delta^t(S) = \mathsf{cap}(S, \hat{V}_i^t \setminus S)$ denotes the size of cut $(S, \hat{V}_i^t \setminus S)$ in subgraph $\hat{G}_i^t$.
3. $\Delta(S) = \mathsf{cap}(S, V_i^0 \setminus S)$ denotes the size of cut $(S, V_i^0 \setminus S)$ in subgraph G_i^0.

Given $\hat{G}_i^t$, we try to route the following multicommodity product flow between any unordered pair of vertices u, v:

$$\mathsf{dem}^t(u,v) = \frac{\gamma(u,\hat{G}_i^t)\gamma(v,\hat{G}_i^t)}{\gamma(\hat{G}_i^t)}, \tag{10.31}$$

where $\gamma(u,\hat{G}_i^t)$ is the flow of $\bar{f}$ at node $u \in \hat{V}_i^t$ that is induced in $\hat{G}_i^t$.

We define $f_1 = \frac{1}{a_1\beta(G)\lambda(n)}$, where $a_1 > 8$, as the minimum concurrent flow value that one needs to obtain for $\mathsf{dem}^t(u,v)$ in order for subgraph $\hat{G}_i^t$ to satisfy the flow-linked property. When the actual flow value $f' < f_1$, we can find a set P_i^{t+1} such that $\delta^t(P_i^{t+1}) \le \mathsf{dem}^t(P_i^{t+1}, \hat{V}_i^t \setminus P_i^{t+1})\beta f'$. We say P_i^{t+1} does not meet the sparsest cut condition for the demands $\mathsf{dem}^t(u,v)$ in subgraph $\hat{G}_i^t$, and we mute P_i^{t+1} in $\hat{G}_i^t$ and recurse on $\hat{G}_i[\hat{V}_i^t \setminus P_i^{t+1}] = \hat{G}_i^{t+1}$. When the flow value $f' \ge f_1$, we stop the recursion, and assign $\boldsymbol{\pi}'_i(u) = \frac{\gamma(u,\hat{G}_i^t)}{(a_1/2)\beta\lambda(n)}$ for all $u \in \hat{V}_i^t$.

Let $G'_i = \hat{G}_i^{y_i} = (\hat{V}_i^{y_i}, \hat{E}_i^{y_i})$ be the remaining subgraph of $\hat{G}_i$ by end of sparsest-cut processing after muting nodes in $P_i^1, \ldots, P_i^{y_i}$ and their adjacent edges. Let the set of terminal pairs $\mathcal{T}'_i$ be the subset of $\mathcal{T}_i$ that are contained in subgraph G'_i and let X'_i be the set of terminals of $\mathcal{T}'_i$.

If $\gamma(G'_i) \le (a_1/4)\beta\lambda(n)$ when the algorithm terminates, we have obtained a terminal pair $\mathcal{T}'_i$ to route in G'_i and a weight assignment that satisfy all three conditions in the theorem. And we are done with this subgraph.

When $\gamma(G'_i) > (a_1/4)\beta\lambda(n)$, a product flow based on the flow of $\bar{f}$ induced in G'_i is routable with throughput at least $f_1 = \frac{1}{a_1\beta(G)\lambda(n)}$ in G'_i, where $a_1 > 8$.

Hence by assigning a new weight

$$\boldsymbol{\pi}'_i(u) = \frac{\gamma(u, G'_i)}{(a_1/2)\beta\lambda(n)}, \tag{10.32}$$

for all $u \in V(G'_i)$, and $\boldsymbol{\pi}'_i(u) = 0$ for all other nodes $u \in \hat{V}_i$ in $\hat{G}_i$, we can define a multicommodity flow problem, where for any unordered pair of vertices $u, v \in V(G'_i)$, $\mathsf{dem}^{\boldsymbol{\pi}'_i}(u, v) = \boldsymbol{\pi}'_i(u)\boldsymbol{\pi}'_i(v)/\boldsymbol{\pi}'_i(X'_i)$, that is feasible in both G' and $\hat{G}_i$. Hence X'_i is $\boldsymbol{\pi}'_i$-flow-linked in $\hat{G}_i, \forall i$. Finally, we put $P_i^t, \forall t$ back in $\hat{G}_i$ with zero node weight, while retaining the same weight assignment for nodes in G'_i. Hence the sum of the total weight is:

$$\boldsymbol{\pi}'_i(\hat{G}_i) = \boldsymbol{\pi}'_i(G'_i) = \sum_{u \in G'_i} \frac{\gamma(u, G'_i)}{(a_1/2)\beta\lambda(n)} = \frac{\gamma(G'_i)}{(a_1/4)\beta\lambda(n)}. \tag{10.33}$$

Hence for both terminating conditions of the algorithm, we have $\boldsymbol{\pi}'_i(G'_i) \geq \frac{\gamma(G'_i)}{(a_1/4)\beta\lambda(n)}$, and thus

$$\sum_{i=1}^{\ell} \boldsymbol{\pi}'_i(X'_i) \geq \sum_{i=1}^{\ell} \frac{\gamma(G'_i)}{(a_1/4)\beta\lambda(n)} \geq \frac{\mathsf{OPT}^*(\mathcal{G}, \mathcal{T})}{16\beta\lambda(n)},$$

where $\sum_{i=1}^{\ell} \gamma(G'_i) \geq \mathsf{OPT}^*(\mathcal{G}, \mathcal{T})/4$ by taking $a_0 = 4$ and $a_1 = 16$ in Lemma 17 and requiring that $\mathcal{C}^0 > 16\lambda(n)\hat{c}$. ■ Hence by the end of sparsest-cut processing, we get a new instance X'_i on $\hat{G}_i = (\hat{V}_i, \hat{E}_i)$ with min-cut at least $\hat{c} = \Omega(\log^3 n)$, such that X'_i is $\boldsymbol{\pi}'_i$-flow-linked in $\hat{G}_i$, which can be only more connected than G'_i. We tune two parameters: a_0 and a_1, to balance the the initial node degree requirement and the amount of flow of $\bar{f}$ that we retain by the end of min-cut and sparsest-cut processing.

Lemma 17. *Given a graph $\mathcal{G}$ with min-cut value $C^0 \geq (4a_0\lambda(n) + a_0 + 2)\hat{c}$, where $a_0 \geq 2$. By the end of sparsest-cut processing, the total amount of flow of $\bar{f}$ that we will pass on to next stage of the algorithm for finding EDP in $\mathcal{G}$ is the sum of flow of $\bar{f}$ induced in G'_i, across all i,*

$$\sum_{i=1,2,\ldots} \gamma(G'_i) \geq \frac{1}{2}\mathit{OPT}^*(\mathcal{G}, \mathcal{T}) \left(1 - \frac{1}{a_0} - \frac{1}{2a_0(1 - 8/a_1)}\right), \tag{10.34}$$

where $a_1 > 8$.

Proof. In the beginning of the sparsest-cut processing stage, we have

$$\mathsf{remaining\text{-}flow} = \sum_{i=1,2,\ldots} \gamma(\hat{G}_i) \geq \frac{1}{2}\mathsf{OPT}^*(\mathcal{G}, \mathcal{T})(1 - \frac{1}{a_0}). \tag{10.35}$$

Combine this initial condition with Lemma 18, we have

$$\mathsf{remaining\text{-}flow} = \sum_{i=1,2,\ldots} \gamma(G'_i) \geq \frac{1}{2}\mathsf{OPT}^*(\mathcal{G}, \mathcal{T}) \left(1 - \frac{1}{a_0} - \frac{1}{2a_0(1 - 8/a_1)}\right), \tag{10.36}$$

where $\mathsf{LOSS} \leq \mathsf{OPT}^*(\mathcal{G}, \mathcal{T})/2$.

Lemma 18. *The amount of flow that we lose from* $\sum_{i=1,2,\ldots} \gamma(\hat{G}_i)$ *due to sparsest-cut processing is* $\mathsf{flow\text{-}loss}_2 \leq \frac{\mathsf{LOSS}}{2a_0(1-8/a_1)}$, *where* $a_1 > 8$.

Proof. To analyze the amount of flow that we lose from sparsest-cut processing, we use a potential function $\varphi(\hat{G}_i)$ to keep track of the edges of $G_i^0 = (V_i^0, E_i^0)$ that we take away from nodes currently in $\hat{G}_i$, after min-cut and during sparsest-cut processing. Note that those lost edges connect to other nodes in V_i^0 from nodes internal to $\hat{G}_i^t$ at stage t. The counting process is the following. We start with a component G_i such that some nodes in $\hat{G}_i$ have lost some of their edges right after min-cut processing and

$$\varphi_i^0 = \mathsf{edge\text{-}loss}_i \geq 0.$$

Let φ_i^t be value of $\varphi(\hat{G}_i)$ after removing t sets of vertices $P_i^1, \ldots, P_i^t$ and their adjacent edges from $\hat{G}_i$. Let P_i^{t+1} be the $(t+1)^{st}$ set of vertices that we shut off from $\hat{G}_i$ because internal boundary capacity of P_i^{t+1} has decreased from $\Delta(P_i^{t+1})$ to $\delta^t(P_i^{t+1}) \leq \mathsf{dem}^t(P_i^{t+1}, \hat{V}_i^t \setminus P_i^{t+1})\beta f'$.

We update $\varphi(G_i)$ as the following,

$$\varphi_i^{t+1} = \varphi_i^t - (\Delta(P_i^{t+1}) - \delta^t(P_i^{t+1})) + \delta^t(P_i^{t+1}).$$

Since the credit that a cut puts back is less than the credit that it spent, there is a only finite number y_i of such small cuts. By the end of y_i rounds, there must be non-negative credit in $\varphi(\hat{G}_i)$, since nodes in current $\hat{G}_i$ can never gain any internal edges:

$$\begin{aligned}
\varphi(\hat{G}_i) &= \varphi_i^{y_i} \\
&= \mathsf{edge\text{-}loss}_i - (\Delta(P_i^1) - \delta^0(P_i^1)) + \delta^0(P_i^1) - (\Delta(P_i^2) - \delta^1(P_i^2)) + \delta^1(P_i^2) \\
&\quad - \ldots - (\Delta(P_i^{y_i}) - \delta^{y_i-1}(P_i^{y_i})) + \delta^{y_i-1}(P_i^{y_i}) \\
&\geq 0.
\end{aligned}$$

Hence by summing above inequalities over all i,

$$\sum_{i=1,2,\ldots} \sum_{j=1,2,\ldots,y_i} (\Delta(P_i^j) - 2\delta^{j-1}(P_i^j)) \leq \sum_{i=1,2,\ldots} \mathsf{edge\text{-}loss}_i \tag{10.37}$$

$$= \mathsf{edge\text{-}loss} \leq \frac{\mathsf{LOSS}}{a_0(2\lambda(n) + \frac{1}{2})} \tag{10.38}$$

Fix P_i^{t+1} for some $i, t \in [0, \ldots, y_i - 1]$, we have the following two lemmas on $\Delta(P_i^{t+1})$ and $\delta^($$P_i^{t+1})$.

Lemma 19. *For all* i *and all* $t \in [0, \ldots, y_i - 1]$,

$$\Delta(P_i^{t+1}) = \mathit{cap}(P_i^{t+1}, V_i^0 \setminus P_i^{t+1}) \geq \sum_{u \in P_i^{t+1}} \gamma(u, \hat{G}_i^{t+1})/2\lambda(n).$$

Lemma 20. *For all i and all $t \in [0, \ldots, y_i - 1]$,*

$$\delta^t(P_i^{t+1}) \leq \frac{4}{a_1}\Delta(P_i^{t+1}). \tag{10.39}$$

Plugging (10.39) in (10.37), we get

$$\begin{aligned}\sum_{i=1,2,\ldots}\sum_{t=1,2,\ldots,y_i}\Delta(P_i^j)(1-\frac{8}{a_1}) &\leq \sum_{i=1,2,\ldots}\sum_{t=1,2,\ldots,y_i}\Delta(P_i^j) - 2\delta^{t-1}(P_i^t))\\ &\leq \frac{\mathsf{LOSS}}{a_0(2\lambda(n)+\frac{1}{2})}.\end{aligned}$$

Hence

$$\sum_{i=1,2,\ldots}\sum_{t=1,2,\ldots,y_i}\Delta(P_i^j) = \frac{\mathsf{LOSS}}{a_0(2\lambda(n)+\frac{1}{2})(1-8/a_1)}. \tag{10.40}$$

Fix $\hat{G}_i^t = (\hat{V}_i^t, \hat{E}_i^t)$ for some $t \in [1, \ldots, y_i]$. We now calculate the amount of flows of $\bar{f}$ that we lose from $\sum_{i=1,2,\ldots}\gamma(\hat{G}_i)$ by shutting off P_i^{t+1} in $\hat{G}_i$. The flow that we lose falls into one of the four types:

1. its path are entirely contained in the subgraph of $\hat{G}_i$ induced by nodes in P_i^{t+1};
2. its path contains edges counted in $\Delta(P_i^{t+1})$ but not those in $\delta^t(P_i^{t+1})$;
3. its path contains edges counted in $\delta^t(P_i^{t+1})$, but with at least one endpoint in P_i^{t+1};
4. those flow with both endpoints $u'v' \in \hat{V}_i^{t+1}$, such that its path intersects edges counted in $\delta^t(P_i^{t+1})$ for at least twice.

Flow of type 1 contributes to the sum $\sum_{u\in P_i^{t+1}}\gamma(u, \hat{G}_i^t)$ twice. Flow of type 3 contribute its flow value to $\sum_{u\in P_i^{t+1}}\gamma(u, \hat{G}_i^t)$ once and to the usage of $\delta^t(P_i^{t+1}) = \mathsf{cap}(P_i^{t+1}, \hat{V}_i^{t+1})$ at least once. Flow of type 4 contribute its flow amount at least twice to the usage of $\mathsf{cap}(P_i^{t+1}, \hat{V}_i^{t+1})$. Flow of type 2 has been counted before when P_i^j were mute for some $j \leq t$ from $\hat{G}_i$. Note that those flow that crosses (P_i^{t+1}, V_i^{t+1}) either has been counted in $\sum_{u\in P_i^{t+1}}\gamma(u, G_i)$ at least once or it goes through the cut (P_i^{t+1}, V_i^{t+1}) in $\hat{G}_i^t$ at least twice.

Hence the total amount of flow of $\bar{f}$ that we lose from $\gamma(\hat{G}_i)$, that has not been counted in earlier stages than t, by muting the induced subgraph of P_i^{t+1} and its adjacent edges in $\hat{G}_i^t$:

$$\begin{aligned}\frac{1}{2}(\sum_{u\in P_i^{t+1}}\gamma(u, G_i) + \mathsf{cap}(P_i^{t+1}, V_i^{t+1})) &= \frac{1}{2}\sum_{u\in P_i^{t+1}}\gamma(u, G_i) + \frac{1}{2}\delta^t(P_i^{t+1})\\ &\leq \Delta(P_i^{t+1})\lambda(n) + \frac{1}{2}\delta^t(P_i^{t+1})\\ &\leq \Delta(P_i^{t+1})(\lambda(n) + 2/a_1),\end{aligned}$$

where the last two inequalities are due to Lemma 19 and 20.

Summing over all $P_i^t, \forall t, \forall i$, given that $a_1 \geq 8$, the total flow lost in sparsest-cut processing stage is

$$\begin{aligned}
\mathsf{flow\text{-}loss}_2 &= \sum_{i=1,2,\ldots} \sum_{t=1,\ldots,y_i} \Delta(P_i^t)\lambda(n) + \frac{1}{2}\delta^{t-1}(P_i^t) \\
&\leq \sum_{i=1,2,\ldots} \sum_{t=1,\ldots,y_i} \Delta(P_i^t)(\lambda(n) + 2/a_1) \\
&\leq \frac{(\lambda(n) + 2/a_1)\mathsf{LOSS}}{2a_0(\lambda(n) + 1/4)(1 - 8/a_1)} \\
&\leq \frac{\mathsf{LOSS}}{2a_0(1 - 8/a_1)}.
\end{aligned}$$

Proof of Lemma 19: Given that

$$\sum_{u \in P_i^{t+1}} \gamma(u, \hat{G}_i^t) \leq \sum_{v \in \hat{V}_i^{t+1}} \gamma(v, \hat{G}_i^t)$$

and $\sum_{u \in P_i^{t+1}} \gamma(u, \hat{G}_i^t) + \sum_{v \in \hat{V}_i^{t+1}} \gamma(v, \hat{G}_i^t) = \sum_{u \in \hat{V}_i^t} \gamma(u, \hat{G}_i^t)$, we have

$$\sum_{u \in P_i^{t+1}} \gamma(u, \hat{G}_i^t) \leq \frac{1}{2} \sum_{u \in \hat{V}_i^t} \gamma(u, \hat{G}_i^t). \tag{10.41}$$

Next let us define a_2 as the additional flow of $\bar{f}$ for node u in G_i^0 as compared to that in subgraph $\hat{G}_i^t$,

$$\sum_{u \in P_i^{t+1}} \gamma(u, G_i^0) = \sum_{u \in P_i^{t+1}} \gamma(u, \hat{G}_i^t) + a_2. \tag{10.42}$$

Since each unit of flow of a_2 uses at least one unit capacity from edges that connect two set of vertices P_i^{t+1} and $V_i^0 \setminus \hat{V}_i^t$ in G_i^0, we have

$$a_2 \leq \Delta(P_i^{t+1}) - \delta^t(P_i^{t+1}). \tag{10.43}$$

In addition, we know that

$$\sum_{u \in X_i^0} \gamma(u, G_i^0) \geq \sum_{u \in \hat{V}_i^t} \gamma(u, G_i^0) \geq \sum_{u \in \hat{V}_i^t} \gamma(u, \hat{G}_i^t) + a_2. \tag{10.44}$$

Thus we have

$$\sum_{u \in P_i^{t+1}} \gamma(u, G_i^0) = \sum_{u \in P_i^{t+1}} \gamma(u, \hat{G}_i^t) + a_2 \leq \frac{\sum_{u \in \hat{V}_i^t} \gamma(u, \hat{G}_i^t) + a_2}{2} + \frac{a_2}{2} \tag{10.45}$$

$$\leq \sum_{u \in \hat{V}_i^t} \frac{1}{2}\gamma(u, G_i^0)/2 + \frac{a_2}{2} \leq \sum_{u \in X_i^0} \frac{1}{2}\gamma(u, G_i^0)/2 + \frac{a_2}{2} \tag{10.46}$$

Now if

$$\sum_{u\in P_i^{t+1}} \gamma(u,G_i^0) \le \frac{1}{2}\sum_{u\in V_i^0} \gamma(u,G_i^0) \;\text{ i.e.}\boldsymbol{\pi}_i(P_i^{t+1}\cap X_i^0) \le \frac{1}{2}\boldsymbol{\pi}_i(X_i^0),$$

we have

$$\begin{aligned}\Delta(P_i^{t+1}) &\ge \mathsf{cap}(P_i^{t+1}, V_i^0\setminus P_i^{t+1}) \ge \frac{1}{2}\boldsymbol{\pi}_i(P_i^{t+1}\cap X_i^0)\\ &\ge \frac{\sum_{u\in P_i^{t+1}}\gamma(u,G_i^0)}{2\lambda(n)} \ge \frac{\sum_{u\in P_i^{t+1}}\gamma(u,\hat{G}_i^t)}{2\lambda(n)}.\end{aligned}$$

Otherwise, $\sum_{u\in P_i^{t+1}}\gamma(u,G_i^0) \ge \frac{1}{2}\sum_{u\in V_i^0}\gamma(u,G_i^0)$. First we have

$$\sum_{u\in V_i^0\setminus P_i^{t+1}} \gamma(u,G_i^0) \ge \sum_{u\in V_i^0}\gamma(u,G_i^0)/2 - a_2/2 \tag{10.47}$$

due to (10.46) and $\sum_{u\in P_i^{t+1}}\gamma(u,G_i^0)+\sum_{u\in V_i^0\setminus P_i^{t+1}}\gamma(u,G_i^0)=\sum_{u\in V_i^0}\gamma(u,G_i^0)$.

Therefore,

$$\begin{aligned}\Delta(P_i^{t+1}) = \mathsf{cap}(P_i^{t+1}, V_i^0\setminus P_i^{t+1}) &\ge \frac{1}{2}\boldsymbol{\pi}_i((V_i^0\setminus P_i^{t+1})\cap X_i^0) = \frac{\sum_{u\in V_i^0\setminus P_i^{t+1}}\gamma(u,G_i^0)}{2\lambda(n)}\\ \ge \frac{(\sum_{u\in V_i^0}\gamma(u,G_i^0)/2 - a_2/2}{2\lambda(n)} &\ge \frac{\sum_{u\in P_i^{t+1}}\gamma(u,G_i^0)-a_2}{2\lambda(n)} = \frac{\sum_{u\in P_i^{t+1}}\gamma(u,\hat{G}_i^t)}{2\lambda(n)},\end{aligned}$$

where the last three (in)equalities are due to (10.47), (10.46), and (10.42), and in this order. ∎

Next, let us bound the size of $\delta^t(P_i^{t+1})$.

Proof of Lemma 20: Given that

$$\sum_{u\in P_i^{t+1}}\gamma(u,\hat{G}_i^t) \le \sum_{v\in\hat{V}_i^{t+1}}\gamma(v,\hat{G}_i^t)$$

and $2\gamma(\hat{G}_i^t)=\sum_{u\in P_i^{t+1}}\gamma(u,\hat{G}_i^t)+\sum_{v\in\hat{V}_i^{t+1}}\gamma(v,\hat{G}_i^t)$, we have

$$\sum_{v\in\hat{V}_i^{t+1}}\gamma(v,\hat{G}_i^t) \le 2\gamma(\hat{G}_i^t).$$

By terminating condition 2(b) in Figure 10.3, we have

$$\begin{aligned}\delta^t(P_i^{t+1}) &\le \mathsf{dem}^t(P_i^{t+1},\hat{V}_i^t\setminus P_i^{t+1})\beta f_1 \le \mathsf{dem}^t(P_i^{t+1},\hat{V}_i^t\setminus P_i^{t+1})\frac{1}{a_1\lambda(n)}\\ &= \frac{\sum_{u\in P_i^{t+1}}\gamma(u,\hat{G}_i^t)\cdot\sum_{v\in\hat{V}_i^{t+1}}\gamma(v,\hat{G}_i^t)}{a_1\lambda(n)\cdot\gamma(\hat{G}_i^t)} \le \frac{2\sum_{u\in P_i^{t+1}}\gamma(u,\hat{G}_i^t)}{a_1\lambda(n)}\\ &\le \frac{4}{a_1}\left(\frac{\sum_{u\in P_i^{t+1}}\gamma(u,\hat{G}_i^t)}{2\lambda(n)}\right) \le \frac{4}{a_1}\Delta(P_i^{t+1}),\end{aligned}$$

where $\mathsf{dem}^t(P_i^{t+1},\hat{V}_i^t\setminus P_i^{t+1})$ is defined in (10.31). ∎

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. H. Chernoff. A measure of asymptotic efficiency of tests of a hypothesis based on the sum of observations. *Annals of Mathematical Statistics*, 23:493–507, 1952.
15. 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.
16. 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.
17. 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.
18. 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.
19. D. R. Karger. Random sampling in cut, flow, and network design problems. In *Proceedings of the 26th ACM STOC*, 1994.
20. D. R. Karger. Random sampling in cut, flow, and network design problems. *Mathematics of Operations Research*, 24(2):383–413, 1999.
21. R. Khandekar, S. Rao, and U. Vazirani. Graph partitioning using single commodity flows. In *Proceedings of the 38th ACM STOC*, 2006.

22. J. Kleinberg. An approximation algorithm for the disjoint paths problem in even-degree planar graphs. In *Proceedings of the 46th IEEE FOCS*, 2005.
23. J. Kleinberg and R. Rubinfeld. Short paths in expander graphs. In *Proceedings of the 37th IEEE FOCS*, 1996.
24. J. Kleinberg and E. Tardos. Approximations for the disjoint paths problem in high-diameter planar networks. In *Proceedings of the 27th ACM STOC*, 1995.
25. J. Kleinberg and E. Tardos. Disjoint paths in densely embedded graphs. In *Proceedings of the 36th IEEE FOCS*, pages 52–61, 1995.
26. J. M. Kleinberg. *Approximation Algorithms for Disjoint Paths Problems*. PhD thesis, MIT, Cambridge, MA, 1996.
27. S. G. Kolliopoulos and C. Stein. Approximating disjoint-path problems using greedy algorithms and packing integer programs. In *Proceedings of IPCO*, 1998.
28. P. Kolman and C. Scheideler. Simple on-line algorithms for the maximum disjoint paths problem. In *Proceedings of the 13th ACM SPAA*, 2001.
29. K. Obata. Approximate max-integral-flow/min-multicut theorems. In *Proceedings of the 36th ACM STOC*, 2004.
30. P. Raghavan and C. D. Thompson. Randomized roundings: a technique for provably good algorithms and algorithms proofs. *Combinatorica*, 7:365–374, 1987.
31. 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.
32. A. Srinivasan. Improved approximations for edge-disjoint paths, unsplittable flow, and related routing problems. In *Proceedings of the 38th IEEE FOCS*, 1997.
33. K. Varadarajan and G. Venkataraman. Graph decomposition and a greedy algorithm for edge-disjoint paths. In *Proceedings of the ACM-SIAM SODA*, 2004.