Poly-logarithmic Approximation for Maximum Node Disjoint Paths with Constant Congestion*

Chandra Chekuri Alina Ene

Abstract

We consider the Maximum Node Disjoint Paths (MNDP) problem in undirected graphs. The input consists of an undirected graph $G = (V, E)$ and a collection $\{(s_1, t_1), \ldots, (s_k, t_k)\}$ of k source-sink pairs. The goal is to select a maximum cardinality subset of pairs that can be routed/connected via node-disjoint paths. A relaxed version of MNDP allows up to c paths to use a node, where c is the congestion parameter. We give a polynomial time algorithm that routes $\Omega(\mathrm{OPT}/\mathrm{poly}\log k)$ pairs with $O(1)$ congestion, where OPT is the value of an optimum fractional solution to a natural multicommodity flow relaxation. Our result builds on the recent breakthrough of Chuzhoy [17] who gave the first poly-logarithmic approximation with constant congestion for the Maximum Edge Disjoint Paths (MEDP) problem.

1 Introduction

In this paper, we consider the Maximum Node Disjoint Paths (MNDP) problem in undirected graphs. An instance of MNDP consists of an undirected graph $G = (V, E)$ and a collection $\mathcal{M} = \{(s_1, t_1), \ldots, (s_k, t_k)\} \subseteq V \times V$ of k source-sink pairs. The goal is to *route* a maximum cardinality subset of the source-sink pairs via *node-disjoint* paths. Formally, the goal is to select a maximum cardinality subset $\mathcal{M}' \subseteq \mathcal{M}$ and a collection of node-disjoint[1] paths $\mathcal{P}$ such that, for each pair $(s_i, t_i) \in \mathcal{M}'$, there is a path in $\mathcal{P}$ with endpoints s_i and t_i. MNDP is an optimization version of the classical decision problem NDP in which the goal is to decide whether all the pairs in $\mathcal{M}$ can be routed in G via node disjoint paths. NDP and MNDP are related to, and generalize, the corresponding problems EDP and MEDP where the paths for the pairs are required to be *edge-disjoint*. These disjoint paths problems have been well-studied because of their connections to algorithms, combinatorial optimization, and graph theory. In addition, these problems and their variants can be used to model a variety of routing problems in networks, and have several applications in practice.

Karp showed that NDP is **NP**-complete when k is part of the input (in his original list of **NP**-complete problems). EDP is also **NP**-complete [21]. Over the years it has been shown that very restricted instances of disjoint paths problems are **NP**-complete, see [42] for a survey. In contrast, when k is a fixed constant, Robertson and Seymour [49], building on several tools from their seminal graph minor project, gave a polynomial-time algorithm for NDP. In this paper we are concerned with the case when k is part of the input and consider the approximability of MNDP. MNDP is easily seen to be **NP**-hard via a reduction from NDP. There is also hardness coming from the problem of choosing which pairs to connect. This can be seen from the fact that MEDP is **NP**-hard (and in fact **APX**-hard to approximate) in capacitated trees [26]; routing is trivial in trees since there is a unique path connecting any given pair, nevertheless, the subset selection is hard.

*Dept. of Computer Science, University of Illinois, Urbana, IL 61801, USA. {chekuri,ene1}@illinois.edu. Partially supported by NSF grant CCF-1016684.

[1] It is important that the paths do not share endpoints as well as internal nodes. This is in contrast to some settings in which one may want the paths to be disjoint only on the internal nodes.

Approximation algorithms for MEDP and MNDP have been studied extensively with MEDP receiving the most attention. The starting point for many approximation algorithms for disjoint paths problems is a natural multicommodity flow based linear programming relaxation (see Section 2 for the formal description); random-walk based algorithms for routing on expanders are an exception. However, a simple example shows that the integrality gap of this flow relaxation is $\Omega(\sqrt{n})$ (here n is the number of nodes in G) for both MEDP and MNDP even on planar graph instances [26]. This arises due to a crossing obstruction in the plane that allows only one pair to be routed while the fractional solution can route $1/2$ unit of flow for $\Omega(\sqrt{n})$ pairs. This example naturally led to the question of whether the integrality gap of the flow relaxation becomes substantially smaller if one removes the topological obstruction by allowing up to two paths to use an edge (or assuming that edges/nodes have capacity 2). More generally, what is the trade-off between the congestion c (the number of paths that are allowed to use an edge/node) and the integrality gap of the flow relaxation? Raghavan and Thompson [45] showed via randomized rounding that a constant factor approximation is achievable if $c = \Omega(\log n/\log\log n)$ (even in directed graphs). It is only in the last few years that substantial and exciting progress was achieved for $c = o(\log n/\log\log n)$. In a recent breakthrough, Chuzhoy [17] obtained a poly-logarithmic approximation for MEDP with congestion 14; Chuzhoy and Li [19] improved the congestion to 2. Our main result is a poly-logarithmic approximation for MNDP with constant congestion and is encapsulated in the following theorem.

Theorem 1.1. *There is a randomized polynomial time algorithm that, given an instance of* MNDP *in an undirected graph G with n nodes and k pairs, routes $\Omega(\mathrm{OPT}/\mathrm{poly}\log k)$ pairs with $O(1)$ congestion, where* OPT *is the value of an optimum solution to the multicommodity flow relaxation.*

The congestion we can guarantee is 51. We believe that it can be further reduced, however, we have not attempted to optimize it in the interests of keeping the algorithmic details and proofs simple and clear. An $O(\sqrt{n})$-approximation is known for MNDP [39]; with congestion c the best previous approximation in general graphs is $O(n^{1/c})$ which follows from randomized rounding. Moreover, via known hardness results for MEDP [3], for any constant c there is no $O(1)$-approximation for MNDP with congestion c unless $NP \subseteq ZPTIME(n^{\mathrm{poly}\log n})$. We discuss a specific motivation for our study of MNDP.

MEDP, MNDP and connections to treewidth and graph theory: Chuzhoy's work on MEDP introduces beautiful new ideas while utilizing several concepts and tools from previous work [44, 9, 33, 46, 2, 52]. In particular, the work of [9] on well-linked decompositions for routing problems reduced the poly-logarithmic approximability of MEDP and MNDP with constant congestion to the following graph-theoretic question. *If a graph G has a well-linked set[2] of size k, does it have a constant congestion crossbar of size $\Omega(k/\mathrm{poly}\log k)$?* A crossbar is a switch-like routing structure that can route any permutation at its interface — see [9] for the precise definitions. Chuzhoy's work answered this question affirmatively by embedding, with constant congestion, an expander of size $\Omega(k/\mathrm{poly}\log k)$ into any graph G that has a well-linked set of size k; the edges of the expander are embedded as paths in G that cause constant edge congestion. Our result shows that the graph-theoretic question has an affirmative answer for node-disjoint routing as well. In the node-capacitated case, a graph G has a well-linked set of size k iff the treewidth of G is $\Omega(k)$. Thus, our result implies that if G has treewidth k then one can embed an expander of size $\Omega(k/\mathrm{poly}\log k)$ such that the edges of G can be mapped to paths that cause constant node congestion. This has the following interesting connection to a central result in the graph minor project of Robertson and Seymour.

Theorem 1.2 (Robertson and Seymour [48])**.** *Let $r, h > 0$ be integers and let G be a graph of treewidth greater than $r^{4h^2(r+2)}$. Then G contains either the $r \times r$ grid or the complete graph K_h as a minor.*

The quantitative bounds have been subsequently improved by Robertson, Seymour and Thomas.

[2] A set X is well-linked in G iff, for any two equal-sized subsets Y and Z of X, there is a collection of disjoint Y-Z paths in G.

Theorem 1.3 ([47]). *Let $r, h > 0$ be integers and let G be a graph of treewidth at least 20^{5gh^3}. Then G contains either the $r \times r$ grid or the complete graph K_h as a minor.*

We note that a clique minor K_h is a crossbar of size h; an $h \times h$ grid minor is also a crossbar of size h but it requires congestion 2. Thus, Theorem 1.3 says that if a graph G has treewidth at least k then it is guaranteed to have a congestion 2 crossbar of size $\Omega(\log^{1/4} k)$. Our result shows that, by allowing constant congestion, one can obtain a crossbar of size $\Omega(k/\text{poly}\log k)$, which is a significant improvement. As previously mentioned, Chuzhoy and Li [19] obtained a congestion 2 poly-logarithmic approximation for MEDP, a remarkably tight result! It is conceivable that one can extend their result (although we suspect it will be technically quite challenging) to obtain a similar result for MNDP. These results and some of the techniques developed along the way may have other applications in (algorithmic) graph theory.

Our algorithm for MNDP follows Chuzhoy's high-level framework for MEDP. Node problems in routing and network design have similarities to their corresponding edge problems, but often exhibit non-trivial differences in the technical details or approximability. The algorithm in [17] uses several tools; some of them are straightforward to generalize to the node setting and some are not. In the following subsection, we give a high-level overview of Chuzhoy's algorithm for MEDP and our adaptation of it to MNDP, while indicating which parts generalize easily and which require new technical ideas. This also serves as a roadmap for the reader who may not be familiar with [17]. This description will gloss over several details; Section 3 gives a formal description of the algorithm.

1.1 High-level Overview of Algorithm and Technical Contribution

Let $(G, \mathcal{M})$ be an instance of the MNDP problem. We may assume without loss of generality that the nodes participating in the pairs of $\mathcal{M}$ are distinct and they have degree one in G. Let $\mathcal{T}$ denote the set of all nodes participating in the pairs of $\mathcal{M}$; we refer to the nodes in $\mathcal{T}$ as *terminals*.

Reduction to well-linked instances: The first (and a key) step is to reduce a general instance of the problem to one in which the terminals are *well-linked*. This is done via the well-linked decomposition framework of [9] which applies to edge as well as node problems. An important technical ingredient is a grouping technique that, given a set X of nodes that is approximately well-linked, it identifies a subset X' of X that is well-linked. This boosting technique is simple in the edge-capacitated setting [7] but is more involved in the node-capacitated setting [9, 10].

Embedding an expander in a well-linked instance: Suppose $\mathcal{T}$ is well-linked. The second ingredient is to show that an expander of sufficiently large size can be embedded into G with constant node congestion. Once this is done, a large number of pairs can be routed via the expander (there are technical details on how to embed an expander that can be reached by $\mathcal{T}$). To embed the expander, Chuzhoy [17] builds on a crucial idea from Rao and Zhou's [46] work on approximating MEDP when G has a large (poly-logarithmic) global minimum-cut value. They embedded an expander as follows. Khandekar, Rao, and Vazirani [33] describe an algorithm that, given a graph G with a well-linked set X of size k, embeds an expander of size k in G but with congestion $O(\log^2 k)$. This may not seem so useful but the important fact about the KRV algorithm is that the expander is embedded in $O(\log^2 k)$ rounds where in each round the well-linkedness of X is used to find a collection of $|X|/2$ disjoint paths. Rao and Zhou used the large minimum cut assumption to split G into $\Omega(\log^2 k)$ *edge-disjoint* subgraphs of G via Karger's sampling scheme [30], and simulated each round of the KRV algorithm in a separate subgraph. Subsequently, Andrews [2], using some ideas from [46] and properties of Raecke's hierarchical decomposition [44], obtained a poly-logarithmic approximation with $O(\text{poly}(\log\log n))$ congestion; Andrews' result was the first approximation algorithm that achieves a sub-polynomial approximation factor using $o(\log n)$ congestion. We note that both Rao-Zhou's sampling approach and Andrews' approach based on

the Raecke tree decomposition do not admit an easy extension to node-disjoint routing.

Chuzhoy's key high-level contribution is based on a new approach to simulating the KRV algorithm and it consists of two ingredients.

Finding good subsets: Chuzhoy shows that if G has a well-linked set X of size k, for any integer parameter h, one can find h *node-disjoint* subsets $S_1, S_2, \ldots, S_h$ such that each S_i has a boundary B_i and a subset $D_i \subseteq B_i$ such that $|D_i| = \Omega(k/(\text{poly}(h, \log k))$ and D_i is approximately well-linked in $G[S_i]$; Chuzhoy refers to such a set as a good subset. In other words, G can be split into h disjoint subgraphs each of which has a well-linked set of size $\Omega(k/\text{poly}(h, \log k))$. The idea is to simulate each iteration of the KRV algorithm inside a separate subgraph $G[S_i]$ by choosing $h = \Omega(\log^2 k)$. The algorithm of Chuzhoy for finding such a partitioning relies on edge-well-linked sets and their properties, and it is quite technical and non-trivial. We believe that the algorithm can be generalized to apply directly to node-well-linked sets, however, we do not have such an algorithm yet. Instead, we apply a simple idea to use the algorithm of Chuzhoy in a black box fashion. Using a preprocessing step, we can assume that the graph G has maximum degree $O(\log n)$. An edge-well-linked set is an approximately node-well-linked where the approximation depends linearly on the degree. Since good subsets are already weakly well-linked (the well-linkedness parameters are later boosted using the grouping technique that we mentioned earlier), this loss does not matter too much and we can absorb it into the approximation ratio rather than in the congestion which we cannot afford to do.

Connecting good subsets via disjoint trees: The second ingredient in Chuzhoy's approach for embedding an expander is the following. The good subsets allow each iteration of KRV to be simulated inside a separate part of G so that the edges used in each iteration are disjoint. However, to embed an expander, one needs to have for each node v of the expander a representative v_i in the boundary of $G[S_i]$; further, all the representatives of v have to simulate a single node. For this purpose Chuzhoy ensures that the boundaries of the good sets are in fact connected to the terminals $\mathcal{T}$ and hence are well-linked themselves. Moreover, she uses this property and several technical tools based on connectivity, to find $k' = \Omega(k/\text{poly}\log k)$ trees $T_1, T_2, \ldots, T_{k'}$ such that (i) the trees are nearly edge-disjoint, in the sense that no edge of G is in more than a constant number of trees, (ii) each tree T_j has a leaf in the boundary of each good subset S_i, and (iii) the leaves of the different trees are disjoint. Thus, each T_j simulates a node v of the expander and the leaves of T_j are the representatives of v in each good subset.

Our main technical contribution is in implementing the preceding step for MNDP; we need to find trees that are nearly node-disjoint. The algorithm of Chuzhoy for finding the disjoint trees is involved; she uses the splitting-off operation that preserves edge-connectivity [41, 23, 29] to create an auxiliary graph and then shows the existence of a bounded degree spanning tree T in the auxiliary graph via a result of Singh and Lau [52]. This tree T is then used as a "template" to generate the required nearly edge-disjoint trees. To obtain nearly node-disjoint trees, we instead rely on an element-connectivity reduction step [28, 16, 14]; however, this reduction step is not as clean as the edge-connectivity step and does not eliminate "Steiner" nodes that can have high degree. Nevertheless, we are able to apply the rough high-level idea of finding a bounded-degree spanning tree but we use a different (a weaker but somewhat more intuitive) argument that is based on the notion of toughness of a graph [54, 25]. We then do a postprocessing step to reduce the high-degree Steiner nodes and make them to essentially behave as edges. We refer the reader to Section 3 and Section 5 for more details.

Other Related Work: We refer the reader to [50, 42] for tractable cases of EDP. We mostly restrict our attention to undirected graphs. The literature on approximation algorithms for disjoint paths problems has focused primarily on MEDP. The best known approximation for MEDP is $O(\sqrt{n})$ [11], and there is a matching approximation for MNDP [39]; here n is the number of nodes in the input graph. Various special classes of graphs have been studied; constant factor and poly-logarithmic factor approximations for MEDP are known for trees [26, 15], graphs with bounded treewidth [13], grids and grid-like graphs [34, 37, 38], Eulerian planar graphs [35, 31], graphs with good expansion [5, 24, 36, 40], and graphs with large global minimum cut [46]. MEDP and MNDP with congestion have also been well-studied, especially given the integrality gap of the

flow relaxation; it is known that randomized rounding techniques give an $O(d^{1/c})$ approximation, where d is the maximum flow path length in the fractional solution and c is the congestion parameter [53, 39, 4, 6]; this holds even for directed graphs and it leads to an $O(n^{1/c})$ approximation. Improved bounds are obtained by taking advantage of fractional solutions with short paths, for example, in expanders. The well-linked decomposition ideas in [7, 8, 9] led to an $O(\log k)$-approximation for MEDP and MNDP in planar graphs with congestion 2; the approximation for MEDP was subsequently improved to an $O(1)$-approximation in [12, 51]. Finally, [32] obtained an $O(n^{3/7}\mathrm{poly}\log n)$-approximation ratio for MEDP with congestion 2 prior to the result of [19] that obtained a poly-logarithmic approximation.

In terms of hardness of approximation, despite the polynomial-factor upper bounds, the first non-trivial lower bounds were established fairly recently. It is known that MEDP (and hence also MNDP) does not admit an $O(\log^{1/2-\varepsilon} n)$-approximation unless $NP \subseteq ZPTIME(n^{\mathrm{poly}\log n})$ [3]; under the same hardness assumption there is no $O\left((\log n)^{\frac{1-\varepsilon}{c+1}}\right)$-approximation with congestion c [3]. As we mentioned, the latter result rules out for MEDP and MNDP a constant factor approximation for any constant congestion. MEDP is significantly harder in directed graphs; there is no $n^{1/2-\varepsilon}$-approximation unless $P = NP$ [27], and with congestion c there is no $n^{\Omega(1/c)}$-approximation unless $NP \subseteq ZPTIME(n^{\mathrm{poly}\log n})$ [18]; these hardness bounds match the approximation ratios guaranteed by randomized rounding.

Organization: We build on several tools from previous work; Section 2 discusses the relevant definitions and theorems. Section 3 describes our algorithm and its proof assuming two key technical theorems on embedding an expander in a graph with a well-linked set. These theorems are proved in Section 4 and Section 5, respectively.

2 Preliminaries and Setup

In the following, we work with an instance $(G, \mathcal{M})$ of the MNDP problem, where $G = (V, E)$ is an undirected graph and $\mathcal{M} = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ is a collection k node pairs. We let $\mathcal{T}$ denote the set of all nodes participating in the pairs of $\mathcal{M}$. We refer to the nodes in $\mathcal{T}$ as *terminals*. Each terminal has degree one in G and $\mathcal{M}$ is a perfect matching on $\mathcal{T}$.

For a set S of nodes, we let $\mathrm{out}_G(S)$ denote the set of all edges $e \in E(G)$ such that e has exactly one endpoint in S. Let $\mathrm{bd}_G(S)$ denote the set of all nodes $v \in S$ such that v is incident to some edge of $\mathrm{out}_G(S)$; we refer to $\mathrm{bd}_G(S)$ as the *inner boundary* of S. Let $N_G(S)$ be the set of all nodes $v \notin S$ such that v is incident to some edge of $\mathrm{out}_G(S)$; we refer to $N_G(S)$ as the *outer boundary* of S. We may omit the subscript if the graph G is clear from the context.

Given two disjoint subsets $A, B \subset V$ in a graph G such that $|A| \leq |B|$ we say that $\mathcal{P}$ is a collection of A-B paths if the following properties hold: (i) the endpoints of the paths in $\mathcal{P}$ are in $A \cup B$ and (ii) each node of A is the endpoint of exactly one path and each node of B is the endpoint of at most one path.

LP relaxation: We consider a standard multicommodity flow relaxation for the MNDP problem. Let $\mathcal{P}_i$ denote the collection of all paths that connect s_i to t_i in G, and let $\mathcal{P} = \cup_i \mathcal{P}_i$; from our assumption on $\mathcal{M}$ forming a perfect matching on $\mathcal{T}$, $\mathcal{P}_i$ and $\mathcal{P}_j$ are disjoint for $i \neq j$. For each path $p \in \mathcal{P}$, the relaxation has a variable $f(p)$ which represents the amount of flow that is sent on p. We let x_i denote the total flow routed for the pair (s_i, t_i). We use f to denote the vector that has an entry for each path $p \in \mathcal{P}$ that is equal to $f(p)$, and we let $|f| = \sum_i x_i$.

(**NDP-LP**)

$$\begin{aligned}
\max \quad & \sum_{i=1}^{k} x_i & \\
\text{s.t.} \quad & \sum_{p \in \mathcal{P}_i} f(p) = x_i & 1 \leq i \leq k \\
& \sum_{p:\, v \in p} f(p) \leq 1 & v \in V(G) \\
& 0 \leq x_i \leq 1 & 1 \leq i \leq k \\
& f(p) \geq 0 & p \in \mathcal{P}
\end{aligned}$$

The above path-based relaxation has an exponential (in n) number of variables but a polynomial number of non-trivial constraints. It can be solved in polynomial time since the separation oracle for the dual is a shortest path problem. Alternatively, there is an equivalent LP formulation that is polynomial-sized.

Degree reduction: For technical reasons we need to work with a graph that has low degree. We accomplish this as follows. Using a standard randomized rounding argument, we can convert a feasible solution to the flow relaxation to another feasible solution such that the flow on each path is either zero or $\Omega(1/\log n)$, while losing only a constant factor in the value of the flow.

Lemma 2.1 (Lemma 1.1 in [9])**.** *Let f be a feasible solution to **NDP-LP**. Then there is a solution f' such that $|f'| = \Omega(|f|)$ and $f'(p) = 0$ or $f'(p) = \Omega(1/\log n)$ for all $p \in \mathcal{P}$.*

Let G' be the subgraph of G induced by the support of the modified flow f'; that is, G' consists of all edges of G used in some path p with $f'(p) > 0$. Since each node capacity is 1, the maximum number of edges incident to any node is $O(\log n)$. We will henceforth assume that the graph we are working with has degree at most $c \log n$ for some fixed constant c.

Remark 2.2. *We do not believe that this degree reduction step is crucial but need it since we use a result from [17] in a black box fashion. The degree can be upper bounded by $O(\log^2 k)$ at a loss of a $\operatorname{poly}\log k$ factor in the approximation ratio using techniques from [9, 46, 33]. Using this bound instead of $O(\log n)$ the approximation ratio can be shown to be poly$\log k$ without a dependence on n.*

Sparse node separators: We need an approximation algorithm for finding a sparse node separator. This will be used (in a black box fashion) to obtain a well-linked decomposition. With this goal in mind, let $\text{cap} : V \to \mathbb{R}_+$ be a node-capacity function and let $\pi : V \to \mathbb{R}_+$ be a weight function. A node separator is a set $S \subset V$ that partitions $V - S$ into A and B such that there are no edges between A and B. The sparsity[3] of S is defined to be $\frac{\text{cap}(S)}{\pi(A \cup S)\pi(B \cup S)}$. An $O(\sqrt{\log k})$-approximation for sparse node separators is given in [22] where k is the support of π (the number of nodes with non-zero π value); see also [1].

Well-linked sets: As we discussed already, well-linked sets and the reduction to well-linked instances of disjoint paths is an important ingredient. Let $X \subseteq V$ be a set of nodes and $\pi : X \to [0, 1]$ be a weight function on X. We are primarily concerned here with node-well-linked sets but we start with the easier to define notion of edge-well-linkedness. We say that X is π-edge-well-linked in G if $|\delta_G(S)| \geq \pi(X \cap S)$ for all $S \subset V$ such that $\pi(X \cap S) \leq \pi(X \cap (V \setminus S))$. If $\pi(u) = \alpha$ for all $u \in X$ we say that X is α-edge-well-linked, and in particular X is edge-well-linked if $\alpha = 1$. Using Menger's theorem, it is straightforward to show that

[3] One has to be a somewhat careful in defining sparsity of node separators. This is in contrast to the definitions for sparsity of edge separators. We refer the reader [22]; we follow their definition.

if X is edge-well-linked then for any two disjoint subsets Y, Z of X such that $|Y| = |Z| = k$ there are k edge-disjoint Y-Z paths in G. In the context of node-well-linked sets, a separator-based definition was given in [9]; here we give a slightly refined and precise definition that is helpful. We say that X is node-well-linked if for any two disjoint subsets Y, Z of X such that $|Y| = |Z| = k$ there are k node-disjoint Y-Z paths in G. This is the definition that is standard in the literature on treewidth. We would like to extend it to weight functions $\pi : X \to [0, 1]$. Assume that each node in X has degree 1 in G and no two nodes $u, v \in X$ share a neighbor in G. In this case we say that X is π-node-well-linked if $|N_G(S)| \geq \pi(X \cap S)$ for all sets S such that $\pi(X \cap S) \leq \pi(X \cap (V \setminus S))$. In general, we say that X is π-node-well-linked in G if X' is π'-node-well-linked in G', where G', X', and π' are defined as follows. For each node $u \in X$, we attach a new leaf node u' to u; we let G' denote the resulting graph and we let X' denote the set of all new leaf nodes. For each node $u' \in X'$, we set $\pi'(u') = \pi(u)$. As in the edge case, we say that X is α-node-well-linked if $\pi(u) = \alpha$ for all $u \in X$. The following lemma follows easily from Menger's theorem.

Lemma 2.3. *Let G be a graph and let X be a set that is α-node-well-linked in G, where $\alpha \in (0, 1]$. For any two subsets Y and Z of X such that $|Y| = |Z|$, there is a collection of Y-Z paths $\mathcal{P}$ such that each node of G appears in at most $\lceil 1/\alpha \rceil$ paths of $\mathcal{P}$.*

Well-linked decomposition: The following theorem allows us to reduce a general instance of MNDP to one in which the terminals are (approximately) node-well-linked.

Theorem 2.4 ([9])**.** *Let* OPT *be the value of a solution to **NDP-LP** for a given instance $(G, \mathcal{M})$ of* MNDP *in a graph G. Let $\beta(G) \geq 1$ be an upper bound on the approximation ratio of a polynomial time algorithm for the sparsest node separator problem in G. Then there is a polynomial time algorithm that partitions G into node-disjoint induced subgraphs $G_1, G_2, \ldots, G_\ell$ and it assigns a weight function $\pi_i : V(G_i) \to \mathbb{R}_+$ to each graph G_i such that the function π_i has the following properties. Let $\mathcal{M}_i$ be the set of all pairs of $\mathcal{M}$ that are contained in G_i, and let $\mathcal{T}_i$ be the set of terminals of $\mathcal{M}_i$.*

(1) $\pi_i(u) = \pi_i(v)$ for $uv \in \mathcal{M}_i$.

(2) $\mathcal{T}_i$ is π_i-node-well-linked in G_i.

(3) $\sum_{i=1}^{\ell} \pi_i(\mathcal{T}_i) = \Omega(\mathrm{OPT}/(\beta(G) \log \mathrm{OPT})) = \Omega(\mathrm{OPT}/\log^{1.5} k)$.

Grouping technique: We also need a technique to boost well-linkedness in the following sense: Given a set X that is α-well-linked, find a subset X' that is well-linked. This is relatively easy for edge-well-linkedness [7] but is more involved in the node case. The following theorem strengthens weaker versions that were initially given in [9].

Theorem 2.5 ([10])**.** *Let B be a π-node-well-linked set in G and let M be a perfect matching on B such that $\pi(u) = \pi(v)$ for all $uv \in M$. Then there is a matching $M' \subseteq M$ with endpoints $B' \subseteq B$ such that B' is $1/4$-node-well-linked in G and $|M'| = 2|B'| = \Omega(\pi(B))$. Moreover, we can find B' and M' in polynomial time.*

Combining Theorem 2.4 and Theorem 2.5, we can reduce an arbitrary instance of MNDP to one in which the terminals are $1/4$-node-well-linked at the loss of a $O(\log^{1.5} k)$-factor in the approximation ratio. Here we use the $O(\sqrt{\log k})$-approximation for sparse node separators from [22].

Routing in edge-expanders: A graph $G = (V, E)$ is an α-edge-expander iff $\min_{S \subseteq V:\, |S| \leq |V|/2} \frac{|\delta(S)|}{|S|} \geq \alpha$. We make use of the following theorem on routing along node-disjoint paths in edge-expanders that have a bound on the degree (and hence are also node-expanders with a weaker parameter that depends on the degree).

Theorem 2.6 ([46]). *Let $G = (V, E)$ be a d-regular α-edge-expander on n nodes. Suppose that n is even and the nodes of G are partitioned into $n/2$ disjoint demand pairs $\mathcal{M}$. There is a polynomial time algorithm that routes a subset $\mathcal{M}' \subseteq \mathcal{M}$ of size $\Omega(\alpha n/(d^2 \log n))$ on node-disjoint paths of G.*

The cut-matching game: Following [46, 17], we use the cut-matching game of Khandekar, Rao, and Vazirani [33] in order to embed an expander into G. In the cut-matching game, there is a set V of nodes, where $|V|$ is even, and two players, the cut player and the matching player. The goal of the cut player is to construct an edge-expander in as few iterations as possible, whereas the goal of the matching player is to prevent the construction of the edge-expander for as long as possible. The two players start with a graph $\mathcal{X}$ with node set V and an empty edge set. The game then proceeds in iterations, each of which adds a set of edges to $\mathcal{X}$. In iteration j, the cut player chooses a partition (Y_j, Z_j) of V such that $|Y_j| = |Z_j|$ and the matching player chooses a perfect matching M_j that matches the nodes of Y_j to the nodes of Z_j. The edges of M_j are then added to $\mathcal{X}$. Khandekar, Rao, and Vazirani [33] showed that there is a strategy for the cut player that guarantees that after $O(\log^2 |V|)$ iterations the graph $\mathcal{X}$ is a $1/2$-edge-expander. Orecchia et al. [43] strengthened this result by showing that after $O(\log^2 |V|)$ iterations the graph $\mathcal{X}$ is a $\Omega(\log |V|)$-edge-expander.

Theorem 2.7 ([43]). *There is a probabilistic algorithm for the cut player such that, no matter how the matching player plays, after $\gamma_{\mathrm{CMG}}(|V|) = O(\log^2 |V|)$ iterations, the graph $\mathcal{X}$ is an $\Omega(\log |V|)$-edge-expander with constant probability.*

We use $\gamma_{\mathrm{CMG}}(n)$ to denote the number of iterations of the cut-matching game required in the proof of the preceding theorem for $|V| = n$. Note that the resulting expander is regular with degree equal to $\gamma_{\mathrm{CMG}}(n)$.

Element connectivity and a reduction lemma: Let $G = (V, E)$ be an undirected graph and let V be partitioned into black nodes B and white nodes W. The element connectivity of two black nodes u and v, denoted by $\kappa'_G(u, v)$, is the maximum number of paths in G from u to v that are disjoint in edges and white nodes (the edges and white nodes are the elements). By Menger's theorem, the element connectivity of u and v is equal to the minimum number of elements whose removal disconnects u and v. It is convenient to assume that the black nodes form an independent set by subdividing any edge connecting two black nodes and placing a new white node. In this case $\kappa'_G(u, v)$ for $u, v \in B$ is equal to the maximum number of u-v paths that are disjoint in white nodes. Hind and Oellermann [28] described a graph reduction step that preserves the global element connectivity of the black nodes. Chekuri and Korula [14] generalized this result in order to preserve the pairwise element-connectivity of the black nodes.

Lemma 2.8 ([14]). *Let G be an undirected graph and B be a set of black nodes. Let pq be any edge where $p, q \in V(G) - B$ and let $G_1 = G - pq$ and $G_2 = G/pq$, where $G - pq$ is the graph obtained from G by removing the edge pq and G/pq is the graph obtained from G by contracting the edge pq. Then one of the following holds: (i) for all $u, v \in B$, $\kappa'_{G_1}(u, v) = \kappa'_G(u, v)$, or (ii) for all $u, v \in B$, $\kappa'_{G_2}(u, v) = \kappa'_G(u, v)$.*

The preceding lemma can be used to transform the original graph G into a new graph G' in which the element connectivity of the black nodes is preserved and the white nodes form an independent set; hence G' is bipartite if the black nodes also form an independent set. A white node v in G' corresponds to a connected subgraph of G consisting of only white nodes; this is the subgraph that was contracted to form v.

3 Expander Embedding and Routing Algorithm

In this section we describe the details of the routing algorithm; the main ingredient is the expander embedding algorithm that relies on the framework and approach of [17]. We state and use two theorems that are proved in subsequent sections. The algorithm can be described as follows.

(1) Solve the flow relaxation **NDP-LP** to obtain a fractional solution (x, f). Use degree reduction (Lemma 2.1), well-linked decomposition (Theorem 2.4), and grouping (Theorem 2.5) to reduce the original instance to a collection of separate instances such that the graph in each instance has maximum degree $\Delta = O(\log n)$ and the terminals are $1/4$-node-well-linked.

(2) Let $(G, \mathcal{M})$ have k pairs such that the terminals are $1/4$-node-well-linked and $\Delta(G) = O(\log n)$. Embed, with $O(1)$ congestion, an expander in G of size $k' = \Omega(k/\text{poly}(\Delta, \log k))$ and degree $\text{poly}\log(k)$.

(3) Use the expander to route $\Omega(k'/\text{poly}\log k)$ pairs from $\mathcal{M}$ with $O(1)$ congestion.

The first step incurs an $O(\log^{1.5} k)$-factor loss in the approximation. The heart of the matter is the second step. Following the outline in Subsection 1.1, it consists of two steps: (i) finding a node-disjoint collection of "good" clusters to simulate the KRV cut-matching game, and (ii) finding representatives in each cluster and connecting them via (nearly-disjoint) trees to simulate each node of the expander. We start with the clustering step and we prove Theorem 1.1 at the end of the section.

3.1 Family of good clusters

We extend the definition of a good set from [17] to the node-capacitated setting as follows; we refer to such sets as clusters.

Definition 3.1. *A subset $S \subseteq V(G) - \mathcal{T}$ of nodes is a* (h, α)-good-cluster *iff there is a subset $B \subseteq \text{bd}_G(S)$ of nodes with the following properties:*

- $|B| \geq h$.
- *B is α-node-well-linked in $G[S]$.*
- *There is a collection of B-$\mathcal{T}$ paths $\mathcal{P}$ in G that are node-disjoint.*

The subset $B \subseteq \text{bd}_G(S)$ of boundary nodes is part of the definition of a good cluster. In the following, when we say that we are given a good cluster S, we mean that we are given the set S and the subset $B \subseteq \text{bd}_G(S)$. The parameters in the definition give flexibility in finding good clusters; the grouping technique allows us to boost the well-linkedness later.

Theorem 3.2 below shows that one can find a family of node-disjoint good clusters in graph G (with appropriate parameters) if G has a well-linked set. We prove it in Section 4 via a corresponding theorem in [17] for the edge-capacitated case.

Theorem 3.2. *Let G be a graph that contains a set $\mathcal{T}$ of nodes with the following properties: $\mathcal{T}$ is $1/4$-node-well-linked in G, $|\mathcal{T}| = 2k$, and each node in $\mathcal{T}$ has degree one in G. Let Δ be the maximum degree in G. There is an efficient randomized algorithm that with high probability constructs a family $\mathcal{F} = \{S_1, \ldots, S_\gamma\}$ of $\gamma = \gamma_{\text{CMG}}(k) = \Theta(\log^2 k)$ node-disjoint sets such that each S_i is a $(k^*, 1/4)$ good cluster, where $k^* = \Omega\left(\frac{k}{\Delta^3 \log^{14} k \log\log k}\right)$. Moreover, the algorithm also outputs for each S_j a set $B_j \subset \text{bd}_G(S_j)$ such that (i) $|B_j| = k^*$, (ii) B_j is $1/4$-well-linked in $G[S_j]$, and (iii) there is a collection of node-disjoint $B_j - \mathcal{T}$ paths in G.*

Remark 3.3. *There is a technical requirement in the preceding theorem that Δ is sufficiently small compared to k. A poly-logarithmic approximation is easy if the condition is not satisfied. This is explained in Section 4.*

3.2 Connecting the good sets and expander embedding

We now describe the algorithm that uses the good clustering from the previous subsection to embed an expander $\mathcal{X}$ in G. We work with two parameters, k^* and k', where k^* is the parameter guaranteed by Theorem 3.2; it is helpful to simply think of k^* as $k/\mathrm{poly}\log(k+n)$. We set $k' = k^*/(6\gamma^4)$; we round k' down to the nearest even integer. The embedding follows the approach from [17, 46].

The expander $\mathcal{X} = (V(\mathcal{X}), E(\mathcal{X}))$ has k' nodes $v_1, v_2, \ldots v_{k'}$. The embedding maps each node $v_i \in V(\mathcal{X})$ to a connected subgraph C_i of G; the k' connected subgraphs $C_1, \ldots, C_{k'}$ are nearly disjoint in that each node of G appears in only a constant number of them. The embedding of each edge $v_i v_{i'} \in E(\mathcal{X})$ is a path of G connecting C_i to $C_{i'}$; the paths corresponding to the edges of $\mathcal{X}$ also have constant node congestion in G. We also need the expander to be reachable from the terminals. For this purpose an additional property that is guaranteed is that each C_i contains a unique terminal; by relabeling terminals C_i contains $t_i \in \mathcal{T}$. We can thus identify v_i with the terminal t_i and interpret the expander as being embedded on a subset of the terminals.

Recall that the plan for embedding the expander is to simulate iteration j of the KRV cut-matching game [33] in cluster S_j (thus the number of clusters is $\gamma = \gamma_{\mathrm{CMG}}(k)$) using the well-linked set $B_j \subset \mathrm{bd}_G(S_j)$. Each node v_i of $\mathcal{X}$ has a representative $b_{i,j} \in B_j$ for each j such that $1 \leq j \leq \gamma$, and the subgraph C_i connects the nodes $b_{i,1}, \ldots, b_{i,\gamma}$ and t_i. In fact, the algorithm first finds the nearly-disjoint connected subgraphs $C_1, \ldots, C_{k'}$ such that each C_i has a representative in each B_j; the well-linkedness of B_j implies that the identity of the representative for C_i in B_j is not important. The theorem, whose proof is given in Section 5, formally states the properties of the polynomial time algorithm that finds the desired sets $C_1, \ldots, C_{k'}$.

Theorem 3.4. *Let $\mathcal{F} = \{S_1, S_2, \ldots, S_\gamma\}$ be the good clusters guaranteed by Theorem 3.2. There is a polynomial time algorithm that finds a subset $\mathcal{T}' \subset \mathcal{T}$ of k' terminals, connected subgraphs $C_1, \ldots, C_{k'}$, and a collection of node sets $D_1, \ldots, D_{k'}$ with the following properties.*

- *Each node of G belongs to at most 43 of the subgraphs $C_1, \ldots, C_{k'}$.*
- *For each i such that $1 \leq i \leq k'$, $D_i \subset C_i$ and $D_i = \{b_{i,1}, b_{i,2}, \ldots, b_{i,\gamma}\}$, where $b_{i,j} \in B_j$ for all j such that $1 \leq j \leq \gamma$.*
- *The sets $D_1, \ldots, D_{k'}$ are mutually disjoint.*
- *We can label the terminals in $\mathcal{T}'$ as $t_1, \ldots, t_{k'}$ such that $t_i \in V(C_i)$ for each i such that $1 \leq i \leq k'$.*

Theorem 3.4 gives us an embedding of the nodes of the edge-expander in which each node v_i is embedded into G using a connected subgraph containing the terminal t_i. We use the cut-matching game of Theorem 2.7 to define the edges of $\mathcal{X}$ and an embedding of these edges into G.

Recall that we have γ good clusters $S_1, S_2, \ldots, S_\gamma$, where $\gamma = \gamma_{\mathrm{CMG}}(k) \geq \gamma_{\mathrm{CMG}}(k')$. We use the cut-matching game as follows. The cut player will follow the strategy guaranteed by Theorem 2.7. In each iteration j of the cut-matching game, the algorithm implements the matching-player as follows. The matching player receives a partition of $V(\mathcal{X})$ into two sets Y_j and Z_j of equal size and needs to find a perfect matching between them. Let $Y'_j = \{b_{i,j} : v_i \in Y\} \subset D_j$ be the representatives in D_j of the expander nodes Y_j, and similarly let $Z'_j = \{b_{i,j} : v_i \in Z\} \subset D_j$ be the representatives in D_j of the expander nodes Z_j. From Theorem 3.4, the sets Y'_j and Z'_j are disjoint and D_j is $1/4$-node-well-linked in $G[S_j]$. Hence there is a collection $\mathcal{P}$ of Y'_j-Z'_j paths in $G[S_j]$ with node congestion 4 (moreover, given Y'_j, Z'_j such a collection of paths can be found in polynomial time via a maximum-flow algorithm). This collection of paths induces a perfect matching M'_j between Y'_j and Z'_j and hence also a perfect matching M_j between Y_j and Z_j; each edge $u'v' \in M'_j$ corresponds to an edge uv in M_j where u' is the representative of u in D_j and v' is the representative of v in D_j. The matching player outputs M_j in iteration j. We associate each edge $uv \in M_j$ with the path between u' and v' in $G[S_j]$.

It follows from Theorem 2.7 that, after γ iterations, the graph $\mathcal{X}$ is an edge-expander with constant probability. Since the sets $S_1, \ldots, S_\gamma$ are node disjoint, the collection of paths in G corresponding to all the edges added to $\mathcal{X}$ in the cut-matching game has node congestion 4. Based on the preceding argument we obtain the following theorem on embedding $\mathcal{X}$.

Theorem 3.5. *There exists a set $\mathcal{T}' \subseteq \mathcal{T}$ of k' terminals and a graph $\mathcal{X}$ with node set $\mathcal{T}'$ with the following properties.*

- *The graph $\mathcal{X}$ is a γ-regular $\Omega(\log k')$-edge-expander.*
- *For each $t_i \in \mathcal{T}'$ there is a connected subgraph C_i of G such that C_i contains t_i and the node congestion of the subgraphs $\{C_i \mid 1 \leq i \leq k'\}$ is at most 43.*
- *For each edge $e = v_i v_{i'} \in E(\mathcal{X})$, there is a path q_e in G connecting C_i to $C_{i'}$ such that the node congestion of the paths $\{q_e \mid e \in E(\mathcal{X})\}$ is at most 4.*

Moreover, there is a randomized polynomial time algorithm that constructs a set $\mathcal{T}'$ and a graph $\mathcal{X}$ with these properties with constant probability.

3.3 Routing using the embedded expander

Let $\mathcal{T}'$ and $\mathcal{X}$ be the set of terminals and the edge-expander guaranteed by Theorem 3.5. We can use the edge-expander $\mathcal{X}$ to route a large subset of the pairs of $\mathcal{M}$.

Theorem 3.6. *Let $\mathcal{M}_0 \subseteq \mathcal{M}$ be any subset of $k'/2$ pairs. There is a randomized polynomial time algorithm that, with high probability, routes in G a subset of $\Omega(|\mathcal{M}_0|/\gamma^2)$ of the pairs in $\mathcal{M}_0$ with node congestion at most 51.*

Proof: Let $\mathcal{T}_0$ denote the set of all terminals participating in the pairs in $\mathcal{M}_0$. Note that $|\mathcal{T}_0| = 2|\mathcal{M}_0| = k'$. Since the set $\mathcal{T}$ of terminals is $1/4$-node-well-linked in G, it follows from Lemma 2.3, there is a collection $\mathcal{P}$ of $\mathcal{T}_0 - \mathcal{T}'$ paths with congestion 4. Using the paths in $\mathcal{P}$ we can translate the matching $\mathcal{M}_0$ to a matching $\mathcal{M}'$ on $\mathcal{T}'$ as follows: for any pair $uv \in \mathcal{M}_0$, we add the pair $u'v'$ to $\mathcal{M}'$, where u' and v' are the nodes of $\mathcal{T}'$ that are the endpoints of the paths of $\mathcal{P}$ that start at u and v, respectively.

Since $\mathcal{X}$ is a γ-regular $\Omega(\log k')$-edge-expander, we can route a subset a subset $\mathcal{M}'' \subseteq \mathcal{M}'$ of $\Omega(|\mathcal{M}_0|/\gamma^2)$ demand pairs on node-disjoint paths of $\mathcal{X}$ (see Theorem 2.6). Let $\mathcal{P}$ be the collection of these paths. We map these paths to a collection of paths $\mathcal{P}'$ in G resulting in a routing of $\mathcal{M}''$ in G. Let p be a t_i-$t_{i'}$ path in $\mathcal{X}$ for a pair $(t_i, t_{i'}) \in \mathcal{M}''$. Let $e = t_h t_\ell$ be an edge on p. From Theorem 3.5 there is a path $q(e)$ in G connecting C_h and C_ℓ; since C_h and C_ℓ are connected subgraphs of G containing t_h and t_ℓ respectively, there is a t_h-t_ℓ path $q'(e)$ in G whose nodes are contained in $C_h \cup C_\ell \cup q(e)$. We map the t_i-$t_{i'}$ path p in $\mathcal{X}$ to a t_i-$t_{i'}$ walk p' in G obtained by replacing each edge $e \in p$ by the path $q'(e)$; this walk contains a simple t_i-$t_{i'}$ path in G. This procedure, done separately for each path $p \in \mathcal{P}$, gives the desired path collection $\mathcal{P}'$ for routing the pairs $\mathcal{M}''$ in G. It is easy to see that $\mathcal{P}'$ can be generated efficiently given the embedding from Theorem 3.5. We now consider the node congestion in G caused by the path collection $\mathcal{P}'$. Since the paths in $\mathcal{P}$ are node disjoint in $\mathcal{X}$ (and hence also trivially edge disjoint) the node congestion of $\mathcal{P}'$ is upper bounded by the sum of the node congestion of the collection $\{C_i \mid 1 \leq i \leq k'\}$ and $\{q(e) \mid e \in E(\mathcal{X})\}$, which, by Theorem 3.5, is at most $43 + 4 = 47$.

The paths in $\mathcal{P}(\mathcal{T}_0, \mathcal{T}')$ concatenated with the paths corresponding to the routing of $\mathcal{M}''$ in G gives a routing with congestion $47 + 4 = 51$ of a subset $\mathcal{M}_1 \subseteq \mathcal{M}_0$ of size $\Omega(|\mathcal{M}_0|/\gamma^2)$. Theorem 3.5 guarantees that $\mathcal{X}$ has the desired expansion properties with constant probability. One can independently repeat the expander embedding algorithm and the subsequent routing via the expander, to obtain a high probability bound on the success of routing a poly-logarithmic fraction of the pairs. □

We can now put the ingredients together to prove the main result of the paper.

Proof of Theorem 1.1: Let $(G, \mathcal{M})$ be an instance of MNDP on a graph with n nodes and k pairs. We follow the outline of the algorithm stated at the beginning of this section. The algorithm solves the **NDP-LP** relaxation to obtain an optimal fractional solution (x, f). Let OPT be the value of this solution. First, we assume that the number of pairs and OPT are at least $\log^c n$ for a sufficiently large constant c, since otherwise we can obtain a poly-logarithmic approximation by routing an arbitrarily chosen pair from $\mathcal{M}$ (if no pair can be connected in G then $\mathrm{OPT} = 0$ and the problem is trivial). We then use the fractional solution and apply the degree reduction and well-linked decomposition to reduce the given instance to a collection of separate instances on subgraphs $G_1, \ldots, G_\ell$ of G such that the resulting instances are $1/4$-node-well-linked for the terminals and the graph in each instance has degree $O(\log n)$; an α-approximation for these restricted instances implies an $O(\alpha \log^{1.5} k)$-approximation for the original instance (from Theorem 2.4 and Theorem 2.5) where the approximation is with respect to the fractional solution value OPT.

Given a well-linked instance with k terminals and a graph with n nodes and maximum degree $O(\log n)$, Theorems 3.2, 3.4, and 3.5 together give an efficient algorithm that embeds an expander $\mathcal{X}$ of size k' in G with constant congestion where $k' = \Omega(k/\mathrm{poly}\log(k+n))$; the expansion of $\mathcal{X}$ is $\Omega(\log k')$. Theorem 3.6 shows that the expander can be used to route $\Omega(k'/\log^4 k)$ pairs from the given instance with congestion 51. Thus, the algorithm routes $\Omega(k/\mathrm{poly}\log(k+n))$ pairs in G with congestion 51. The dependence on n in the approximation ratio is due to the fact that the maximum degree is $\Delta = O(\log n)$. We can ensure that $\Delta = O(\log^2 k)$ with additional work (see Remark 2.2). This leads to an efficient algorithm that routes $\Omega(\mathrm{OPT}/\mathrm{poly}\log k)$ pairs. □

4 Proof of Theorem 3.2 on good clustering

Chuzhoy [17] gave a clustering algorithm for the edge-capacitated case. We believe there should be a "natural" extension of it to the node-capacitated case; however, the proof in [17] is rather technical and non-trivial. Here, we use her result in a black-box fashion and take advantage of the fact that edge-well-linkedness and node-well-linkedness can be related if we have an upper bound on the degree; we lose factors that are polynomial in the degree in this translation; since the degree bound we have is $O(\log n)$, it affects the final approximation ratio by only a poly-logarithmic factor.

Chuzhoy [17] uses the following definition of well-linkedness, which we call *edge-well-linkedness*. For a set S of nodes, we let $\mathrm{out}_G(S)$ denote the set of all edges of G with an endpoint in S and the other endpoint outside of S.

Definition 4.1 ([17]). *Let G be a graph and let S be a set of nodes. Let $F \subseteq \mathrm{out}_G(S)$ be a set of edges. We say that S is α-edge-well-linked for F iff, for any partition (X, Y) such that $X \cup Y = S$, we have*

$$|E_G(X,Y)| \geq \alpha \cdot \min\{|\mathrm{out}_G(X) \cap F|, |\mathrm{out}_G(Y) \cap F|\}$$

where $E_G(X,Y)$ is the set of all edges of G with one endpoint in X and the other in Y.

We observe that Chuzhoy's definition is equivalent to the following. Subdivide each edge $e \in F$ using a node v_e; let X be the set of these new nodes. The set S is α-edge-well-linked for F iff X is α-edge-well-linked (according to the definition given in Section 2) in the graph $G[S \cup X]$.

One of the main technical ingredients of the algorithm of [17] is a clustering procedure that selects a family of $\gamma = \gamma_{\mathrm{CMG}} = \Theta(\log^2 k)$ disjoint subsets of nodes called *good subsets*. Following [17], we use the parameters $k_1 = \Omega\left(\frac{k}{\log^6 k \log\log k}\right)$ and $\alpha_{\mathrm{WL}}(k) = \Omega\left(\frac{1}{\log^{3.5} k}\right)$.

Definition 4.2 (Definition 4 in [17]). *A subset $S \subseteq V(G) - \mathcal{T}$ of nodes is a* good subset *iff there is a subset $\Gamma \subseteq \mathrm{out}_G(S)$ of edges with the following properties:*

- $|\Gamma| = k_1$.
- *S is $\alpha_{\mathrm{WL}}(k)$-edge-well-linked for Γ.*
- *There is a flow F in graph G, where every edge $e \in \Gamma$ sends one flow unit to a distinct terminal $t_e \in \mathcal{T}$ (so for $e \neq e'$, $t_e \neq t_{e'}$), and the congestion caused by F is at most $O(\log^{4.5} k)$.*

A family $\mathcal{F} = \{S_1, \ldots, S_\gamma\}$ of $\gamma = \gamma_{\mathrm{CMG}}(k) = \Theta(\log^2 k)$ subsets of nodes is good *iff each subset S_j is a good subset of nodes of G, and $S_1, \ldots, S_\gamma$ are pairwise disjoint.*

Theorem 4.3 (Corollary 2 in [17])**.** *Let G be a graph that contains a set $\mathcal{T}$ such that $|\mathcal{T}| = 2k$ and $\mathcal{T}$ is $1/4$-edge-well-linked in G. If the maximum degree Δ of G is at most k_1, there is a polynomial time randomized algorithm that computes a good family of subsets in G with high probability.*

Remark 4.4. *In [17] the statement of Theorem 4.3 assumes that $\mathcal{T}$ is flow-well-linked. However, the proof only uses cut-well-linkedness. Further, this distinction is not crucial since flow and cut well-linkedness are approximately the same (within a logarithmic factor). We refer the reader to [9] for a definition of flow and cut well-linkedness and the approximate equivalence between the two notions. Additionally, $\mathcal{T}$ is assumed to be $1/2$-edge-well-linked. Replacing the $1/2$ by $1/4$ only weakens the parameters for the good sets by a constant factor. Alternatively, we can make a copy of each edge of the graph in order to boost the well-linkedness of $\mathcal{T}$ from $1/4$ to $1/2$ by losing a constant factor in the congestion.*

In our setting we have $\Delta = O(\log n)$. We will assume that $\Delta \leq k_1$, since otherwise $k = O(\mathrm{poly} \log n)$ and it is trivial to get a k approximation for MNDP by simply routing one pair.

The following propositions relate the notions of edge and node well-linkedness, and they are straightforward to verify.

Proposition 4.5. *Let G be a graph with maximum degree Δ. Let S be a set of nodes and let $F \subseteq \mathrm{out}_G(S)$ be a set of edges. Let B be the set of all endpoints of edges in F that are in S. If S is α-edge-well-linked for F then B is $\Omega(\alpha/\Delta)$-node-well-linked in $G[S]$, where $G[S]$ is the subgraph of G induced by S.*

Proposition 4.6. *If X is α-node-well-linked in G then X is α-edge-well-linked in G.*

We can use Theorem 4.3 to complete the proof of Theorem 3.2 as follows.

Proof of Theorem 3.2: Since $\mathcal{T}$ is $1/4$-node-well-linked in G, it follows from Proposition 4.6 that $\mathcal{T}$ is $1/4$-edge-well-linked in G. Therefore G and $\mathcal{T}$ satisfy the conditions of Theorem 4.3. Let $\mathcal{F} = \{S_1, \ldots, S_\gamma\}$ be the good family guaranteed by Theorem 4.3. These will be our good clusters, however, the parameters will be weaker and we also need to identify a set $B_j \subset \mathrm{bd}_G(S_j)$ for each S_j.

For each set $S_j \in \mathcal{F}$, we have a subset $\Gamma_j \subseteq \mathrm{out}_G(S_j)$ of edges. For each index j, let B_j be the set of all endpoints of the edges in Γ_j that are in S_j. We select a subset $B''_j \subseteq B_j$ such that S_j and B''_j form a $(k^*, 1/4)$-good-cluster as follows.

Consider an index j. We start by selecting a subset $B'_j \subseteq B_j$ such that there is a collection of B'_j-$\mathcal{T}$ paths in G that are node-disjoint. Recall that there is a flow F in G where each edge $e \in \Gamma_j$ sends one flow unit to a distinct terminal in $\mathcal{T}$ and the edge congestion caused by F is at most $O(\log^{4.5} k)$. We reinterpret the flow F as originating at the nodes in B_j and ending in $\mathcal{T}$. Note that, since the maximum degree in G is Δ, the node congestion caused by F is at most $O(\Delta \log^{4.5} k)$. Additionally, every node in B_j sends at least one unit of flow and at most Δ units of flow. Thus, if we scale down the flow F by $O(\Delta^2 \log^{4.5} k)$, we get a flow F' from B_j to $\mathcal{T}$ that respects node capacities (1 on every node) of the graph including the endpoints B_j and $\mathcal{T}$. We can use the flow F' to select the subset B'_j as follows. We add a source node s and an edge from s to each node in B_j. We add a sink t and an edge from each node in $\mathcal{T}$ to t. We assign a capacity of one to each node. Note that

the flow F' gives us a feasible s-t flow of value $\Omega(k_1/(\Delta^2 \log^{4.5} k))$. Since the node capacities are integral, there is an integral s-t flow of value $\Omega(k_1/(\Delta^2 \log^{4.5} k))$. We take a path decomposition of such an integral flow in order to get a collection of s-t paths that are internally node-disjoint; by removing the endpoints s, t from these paths, we get a collection of node-disjoint paths in G connecting a subset $B'_j \subseteq B_j$ to $\mathcal{T}$, where $|B'_j| = \Omega(k_1/(\Delta^2 \log^{4.5} k))$.

Since S_j is $\alpha_{\mathrm{WL}}(k)$-edge-well-linked for Γ_j, it follows from Proposition 4.5 that B_j is $\Omega(\alpha_{\mathrm{WL}}(k)/\Delta)$-node-well-linked in $G[S_j]$. We apply Theorem 2.5 to B'_j in order to get a subset $B''_j \subseteq B'_j$ such that B''_j is $1/4$-node-well-linked in $G[S_j]$ and $|B''_j| = \Omega(|B'_j|\alpha_{\mathrm{WL}}(k)/\Delta) = \Omega(k_1\alpha_{\mathrm{WL}}(k)/(\Delta^3 \log^{4.5} k))$.

Therefore the set S_j together with the boundary set $B''_j \subseteq \mathrm{bd}_G(S_j)$ gives us a $(k^*, 1/4)$-good-cluster, where $k^* = \Omega(k_1\alpha_{\mathrm{WL}}(k)/(\Delta^3 \log^{4.5} k))$. □

5 Proof of Theorem 3.4 on connecting good clusters

We recall the properties of good clusters guaranteed by Theorem 3.2. There are γ good clusters $S_1, \ldots, S_\gamma$; each S_j has a set $B_j \subset \mathrm{bd}_G(S_j)$ such that B_j is $1/4$-node-well-linked in $G[S_j]$ and there is a collection of $B_j - \mathcal{T}$ node disjoint paths in G. Recall that $|B_j| \geq k^*$ for $1 \leq j \leq \gamma$. In this section we assume that $|B_j| = k^*$, which we can ensure by selecting arbitrarily a subset of B_j of cardinality k^*; this will be important later.

We prove Theorem 3.4 in this section; it guarantees $k' = k^*/(64\gamma^2)$ connected subgraphs $C_1, \ldots, C_{k'}$ in G. Each C_i has a representative $b_{i,j}$ in B_j for each j. At a high level we find these subgraphs via the same approach as that in [17]; Chuzhoy uses edge-connectivity based techniques to find many trees, each of which has a boundary node from each good set. We use element-connectivity techniques to find many trees, each of which has a boundary node from each good cluster. Moreover, each node of G is in at most a constant number of these trees. Once we have the trees, we connect a subset of the terminals to the trees to get the desired connected subgraphs.

To construct the trees, we first create a new graph G' from G as follows. We add γ new (super-)nodes $s_1, s_2, \ldots, s_\gamma$; for $1 \leq j \leq \gamma$, s_j is connected to each node in B_j. We think of the nodes of G' as being partitioned into black and white nodes; the black nodes are the new super-nodes and the white nodes are the nodes of G. We note that the degree of each black node is exactly equal to k^*. We refer the reader to element-connectivity definitions from Section 2.

Proposition 5.1. *For any two black nodes s_i and s_j, we have $\kappa'_{G'}(s_i, s_j) \geq \lceil k^*/6 \rceil$; that is, s_i and s_j are $\lceil k^*/6 \rceil$ element-connected in G'.*

Proof: Consider B_i and B_j; we show a collection of B_i-B_j paths $\mathcal{P}$ in G such that any node in G is in at most 6 paths in $\mathcal{P}$. This implies that there are $\lceil k^*/6 \rceil$ paths from s_i to s_j in G' that are disjoint in the white nodes, which proves the proposition. Recall that in G there is a collection $\mathcal{P}_1$ of node-disjoint B_i-$\mathcal{T}$ paths and similarly there is a collection $\mathcal{P}_2$ of node disjoint B_j-$\mathcal{T}$ paths. Let $Y \subset \mathcal{T}$ be the endpoints of the paths in $\mathcal{P}_1$ and $Z \subset \mathcal{T}$ be the endpoints of the paths in $\mathcal{P}_2$. Note that $|Y| = |Z| = k^*$. Since $\mathcal{T}$ is $1/4$-node-well-linked in G, it follows from Lemma 2.3 that there is a collection of Y-Z paths $\mathcal{P}_3$ with node congestion 4. We obtain a collection of paths $\mathcal{P}$ by concatenating the paths in $\mathcal{P}_1$, $\mathcal{P}_3$, and $\mathcal{P}_2$ in the natural way. That is, if p is a u-t path in $\mathcal{P}_1$ from a node $u \in B_i$ to a terminal $t \in Y$, q is a t-t' path in $\mathcal{P}_3$ from t to $t' \in Z$, and p' is a t'-v path in $\mathcal{P}_2$ from t' to $v \in B_j$ then we obtain a u-v path in $\mathcal{P}$ from the union of the paths p, q, p'. The node congestion of $\mathcal{P}$ is at most 6 since $\mathcal{P}_1$ and $\mathcal{P}_2$ have congestion 1, and $\mathcal{P}_3$ has congestion at most 4. □

We now apply the element-connectivity reduction step from Lemma 2.8 to G'. This results in a bipartite graph G'' in which the element-connectivity between each pair of black nodes is at least $\lceil k^*/6 \rceil$. Moreover, each white node v in G'' is obtained by contracting a connected subgraph of G. We now create an auxiliary graph H as follows. The node set of H is $V(H) = \{s_1, s_2, \ldots, s_\gamma\}$. There is an edge $s_i s_j$ in H iff there are at least

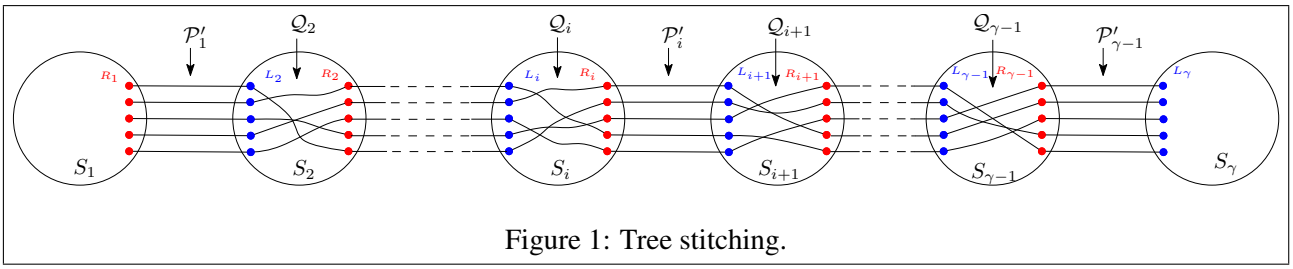

Figure 1: Tree stitching.

$k^*/(6\gamma^3)$ white nodes v in G'' such that v is adjacent to both s_i and s_j. Theorem 5.2 captures the important facts about the auxiliary graph, in particular the existence of a constant degree spanning tree that will be used as a "template" to find many trees.

Theorem 5.2. *There is a spanning tree T^* in H with the following properties:*

- *The maximum degree of T^* is at most 10.*
- *For each edge $e = s_i s_j$ of T^*, there is a collection $\mathcal{P}_e$ of at least $k^*/(6\gamma^4)$ paths of G' such that, for each path $p \in \mathcal{P}_e$, the endpoints of p are s_i and s_j, and the internal nodes of p are white.*
- *The paths in $\mathcal{P} = \cup_{e \in E(T^*)} \mathcal{P}_e$ are disjoint in white nodes.*

Moreover, we can find the tree T^* and the collections of paths $\{\mathcal{P}_e \mid e \in E(T^*)\}$ in polynomial time.

Remark 5.3. *Chuzhoy [17] uses the edge-connectivity preserving splitting-off operation [41, 23, 29] to remove all white nodes to directly obtain an auxiliary graph on the super-nodes and then uses a theorem of Singh and Lau [52] to find a spanning tree of degree at most 3 in the auxiliary graph (via a feasible solution to an LP relaxation). The presence of white nodes in the graph G' which have different degrees does not allow us to use a similar argument. Hence, we rely on a different argument based on the notion of toughness; this gives a bound of 10 on the degree of the spanning tree (which affects the final congestion bound) and we also lose additional poly-logarithmic factors in the approximation.*

We give the proof of Theorem 5.2 in Subsection 5.1. Using Theorem 5.2, we can complete the proof of Theorem 3.4 as follows.

Proof of Theorem 3.4: Let T^* be the tree guaranteed by Theorem 5.2. We first consider the special case in which T^* is a path; as we will see shortly, we can extend the argument for this special case to the general case by making a copy of each edge of T^* and considering an Eulerian walk of the resulting graph.

Suppose that T^* is a path. We think of the nodes and edges of T^* as being ordered from left to right. By relabeling the nodes, we may assume that the nodes of T^* are $s_1, s_2, \ldots, s_\gamma$ from left to right. Let $\mathcal{P}_j$ be the collection of $k' = k^*/(6\gamma^4)$ s_j-s_{j+1} paths guaranteed by Theorem 5.2. By removing the endpoints of the paths in $\mathcal{P}_j$, we get a collection $\mathcal{P}'_j$ of R_j-L_{j+1} paths in G where $R_j \subseteq B_j$ and $L_{j+1} \subseteq B_{j+1}$. (Recall that $B_j \subseteq \mathrm{bd}_G(S_j)$ is the set of boundary nodes of the good cluster S_j and s_j is connected only to B_j in G'.) Note that the paths in $\mathcal{P}' = \uplus_j \mathcal{P}'_j$ are node disjoint. From this it follows that the node sets $R_1, L_2, R_2, L_3, R_3, \ldots, L_{\gamma-1}, R_{\gamma-1}, L_\gamma$ are disjoint and all have the same cardinality k'. Since $|L_j| = |R_j| = k'$ and B_j is $1/4$-node-well-linked in $G[S_j]$, there is a collection $\mathcal{Q}_j$ of L_j-R_j paths that are contained in $G[S_j]$ and they have node congestion at most 4. These paths can be concatenated together to generate k' walks in G (see Figure 1). Once we have the walks, we attach a subset of the terminals in order to get the connected subgraphs $C_1, \ldots, C_{k'}$.

In the following, we give a more formal overview of the stitching described in Figure 1. The path collection $\mathcal{P}'_j$ defines a perfect matching M_j between R_j and L_{j+1}, and the path collection $\mathcal{Q}_j$ defines a perfect matching M'_j between L_j and R_j. Consider the layered graph with node set $R_1 \uplus \left(\uplus_{j=2}^{\gamma-1} L_j \uplus R_j\right) \uplus L_\gamma$ and

edge set $M_1 \cup \left(\cup_{j=2}^{\gamma-1} M'_j \cup M_j\right)$. Each layer has the same cardinality k' and the edge set consists of perfect matchings between adjacent layers. It is easy to see that the edge set can be decomposed into k' paths where each path consists of a sequence of nodes, one from each layer, starting with a node in the first layer R_1 and ending at the last layer L_γ; these paths also partition the nodes. Let $p_1, \ldots, p_{k'}$ be these paths. Let $p_i = v_{i,1}, u_{i,2}, v_{i,2}, \ldots, u_{i,\gamma-1}, v_{i,\gamma-1}, u_{i,\gamma}$ where $u_{i,j}$ is the node from L_j on p_i and $v_{i,j}$ is the node from R_j on p_i. For each i we obtain a connected subgraph (in fact a walk) C_i in G by replacing the edges of p_i with corresponding paths from G: an edge $v_{i,j}u_{i,j+1}$ in p_i corresponds to a unique path in $\mathcal{P}'_j$ and an edge $u_{i,j}v_{i,j}$ corresponds to a unique path in $\mathcal{Q}'_j$. Recall that the paths in $\mathcal{P}'$ are node-disjoint in G and the paths in $\mathcal{Q}'_j$ are contained in $G[S_j]$ and have node congestion 4. It therefore follows that no node of G is in more than 5 of the subgraphs $C_1, \ldots, C_{k'}$. We also need to choose $D_i \subset C_i$ such that $|D_i \cap B_j| = 1$ for each $1 \leq j \leq \gamma$ and such that $D_1, \ldots, D_{k'}$ are disjoint; we let $D_i = \{v_{i,j} | 1 \leq j < \gamma\} \cup \{u_{i,\gamma}\}$; in other words we set $b_{i,j} = v_{i,j}$ for $1 \leq j < \gamma$ and $b_{i,\gamma} = u_{i,\gamma}$. It is easy to verify that $D_1, \ldots, D_{k'}$ satisfy the desired properties.

Finally, we need to ensure that each C_i contains a distinct terminal. Let $B'_1 = \{b_{i,1} \mid 1 \leq i \leq k'\} \subseteq B_1$. Since S_1 is a good cluster, there is a collection of node-disjoint paths in G connecting B'_1 to a subset $\mathcal{T}' \subseteq \mathcal{T}$; for each i, we add to C_i the path connecting $b_{i,1}$ to $\mathcal{T}$. This final step increases the congestion by 1 so we have that no node is in more than 6 of the subgraphs $C_1, C_2, \ldots, C_{k'}$.

Now we extend the argument to the case in which T^* is not a path. We make a copy of each edge of T^* and consider an Eulerian walk of the resulting Eulerian graph. We view this walk as a path (after removing the final edge in the walk) in which each edge of T^* appears at most twice and each node s_i of T^* appears at most 10 times (the degree of s_i to be precise). We apply the previous argument to this path. For this purpose each edge and node of T^* that occurs more than once is assumed to be distinct; the path collection associated with each edge and node of T^* in the previous argument will use separate copies of the nodes of G. We will subsequently analyze the congestion caused by these copies. The path argument gives k' connected components $C_1, \ldots, C_{k'}$ and k' sets $D_1, \ldots, D_{k'}$. We claim that the connected components have node congestion at most $1 \cdot 2 + 4 \cdot 10 + 1 = 43$. Since each edge of T^* appears twice in the Eulerian walk, the stitching described in Figure 1 uses the paths of $\{\mathcal{P}'_e \mid e \in E(T^*)\}$ at most twice; thus we have a congestion of at most $1 \cdot 2$ from these paths. Since each node s_i appears at most 10 times in the Eulerian walk, the stitching uses the subgraph $G[S_i]$ at most 10 times. Each of those uses requires a collection of paths Q'_i between two disjoint sets $L_i, R_i \subset B_i$; since B_i is $1/4$-node-well-linked in $G[S_i]$ such a path collection with node congestion 4 can be found. Hence the overall congestion of all these paths in $G[S_i]$ is $4 \cdot 10$; since the good clusters $S_1, S_2, \ldots, S_\gamma$ are disjoint, the total congestion of the union of these paths is also upper bounded by $4 \cdot 10$. Finally, we use a collection of node-disjoint paths to connect a subset of B_1 to a subset of $\mathcal{T}$. □

Remark 5.4. *In the proof of Theorem 3.4, we used an Eulerian walk of T^* in order to make the argument more transparent. In order to get an Eulerian walk, we duplicated the edges of T^*. Instead, we can root T^* at an arbitrary leaf and stitch the paths in a bottom-up fashion in order to get the desired connected components; this is the scheme used in [17] and requires a more involved description and proof. The bottom-up argument avoids duplicating the edges of T^* and therefore it improves the congestion of the resulting connected subgraphs by* 1.

5.1 Proof of Theorem 5.2

First, we show that H has a spanning tree T^* that has constant degree. Chvátal [20] introduced the notion of graph toughness. Win [54] showed existence of a low-degree spanning tree in a graph where the degree bound was related to its toughness.

The toughness of a graph G, denoted by $\tau(G)$, is defined as follows. Consider a subset $S \subset V(G)$ of the nodes of G. Let $c(S)$ denote the number of connected components of the graph $G - S$ obtained from G by removing the nodes in S. The *toughness* of G is the ratio $\tau(G) = \min_{S \subset V} |S|/c(S)$, where the minimum is

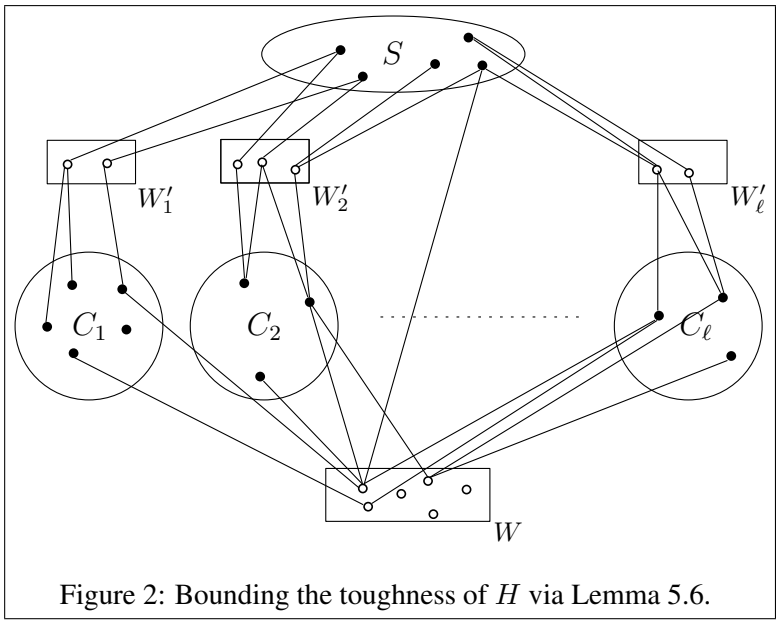

Figure 2: Bounding the toughness of H via Lemma 5.6.

taken over all sets $S \subset V$ such that $c(S) > 1$. We use the following result of Fürer and Raghavachari [25] that is slightly stronger than the result of Win [54].

Theorem 5.5 ([25]). *Let G be a graph and let $\tau(G)$ be its toughness. Let Δ^* be the smallest number such that G has a spanning tree with maximum degree Δ^*. Then $\Delta^* - 3 < 1/\tau(G) \leq \Delta^*$.*

Fürer and Raghavachari also described a polynomial time algorithm that constructs a spanning tree of degree at most $\Delta^* + 1$, where Δ^* is the optimal degree. Therefore, in order to prove that we can find in polynomial time a spanning tree of H with degree at most 10, it suffices to show that $1/\tau(H)$ is at most 7.

Lemma 5.6. *We have $1/\tau(H) \leq 6\gamma/(\gamma - 1) \leq 7$.*

Proof: It follows from the definition of $\tau(H)$ that we need to verify that $c(S) \leq (6\gamma/(\gamma-1))|S|$ for each set $S \subset V(H)$ such that $c(S) > 1$. Consider such a set S and let $C_1, C_2, \ldots, C_\ell$ denote the connected components of $H - S$ (see Figure 2).

Let M be the set of all pairs $s_i s_j$ such that s_i and s_j are in different components of $H - S$. Trivially, $|M| \leq \gamma^2$, since there are γ black nodes. Let W be the set of all white nodes v in G'' such that v is adjacent to s_i and s_j for some pair $s_i s_j$ in M.

We claim that $|W| \leq k^*/(6\gamma)$. If not, there is some pair $s_i s_j \in M$ such that there are more than $(1/\gamma^2) \cdot k^*/(6\gamma) = k^*/(6\gamma^3)$ white nodes in G'' that are adjacent to both s_i and s_j. But then $s_i s_j$ is an edge in H, which contradicts the assumption that s_i and s_j are in different connected components of $H - S$.

Now we claim that, for each connected component C_i of $H - S$, there is a set W_i' of white nodes in G'' with the following properties: (1) $|W_i'| \geq (1 - 1/\gamma)k^*/6$, (2) $N_{G''}(W_i') \subseteq C_i \cup S$, and (3) each node in W_i' is adjacent in G'' to S. Since the black nodes are $\lceil k^*/6 \rceil$ element-connected in G'', there is a set W_i of white nodes such that $|W_i| \geq \lceil k^*/6 \rceil$ and each node in W_i is adjacent in G'' to C_i and $V(H) - C_i$. Let W_i' be the subset of W_i consisting of all nodes that are only adjacent in G'' to $C_i \cup S$. By the claim in the preceding paragraph, there are at most $k^*/(6\gamma)$ nodes in $W_i - W_i'$ and therefore $|W_i'| \geq (1 - 1/\gamma)k^*/6$.

Let $W' = W_1' \cup \ldots \cup W_\ell'$. Note that, if $i \neq j$, W_i' and W_j' are disjoint and therefore $|W'| \geq (1-1/\gamma)k^*\ell/6$. Since each node in W' contributes at least one edge to the total degree in G'' of the nodes in S, it follows that $\sum_{u \in S} d_{G''}(u) \geq |W'|$. Since each black node has degree k^* in G'', we have $|W'| \leq k^*|S|$. Therefore

$c(S) = \ell \leq 6|S|/(1-1/\gamma) \leq 7|S|$. (Recall that $\gamma = \gamma_{\mathrm{CMG}} = \Omega(\log^2 k)$; we can ensure that $6/(1-1/\gamma) \leq 7$ by assuming that k is a sufficiently large constant.) □

It follows from Theorem 5.5 and Lemma 5.6 that H has a spanning tree with maximum degree $\Delta^* < 3 + (1/\tau(H)) \leq 10$; since Δ^* is an integer, we have $\Delta^* \leq 9$. As shown by Fürer and Raghavachari [25], we can find in polynomial time a spanning tree T^* with maximum degree at most $\Delta^* + 1 \leq 10$. This proves the first part of Theorem 5.2.

For each edge $e = s_i s_j$ of T^*, we construct a collection $\mathcal{P}_e$ of paths as follows. Since T^* is a subgraph of H, for each edge $s_i s_j$ of T^*, the graph G'' has a set W'_e of at least $k^*/(6\gamma^3)$ white nodes that are adjacent in G'' to both s_i and s_j. The sets $\{W'_e \mid e \in E(T^*)\}$ may not be disjoint, but we can select a subset $W''_e \subseteq W'_e$ for each edge e such that $|W''_e| \geq k^*/(6\gamma^4)$ and the sets $\{W''_e \mid e \in E(T^*)\}$ are disjoint. We construct the sets $\{W''_e \mid e \in E(T^*)\}$ greedily as follows. We order the edges of T^* arbitrarily. We consider the edges in this order. Let e be the current edge and suppose that, for each edge e' that comes before e in the order, we have already selected a set $W''_{e'} \subseteq W'_{e'}$ of size $k^*/(6\gamma^4)$. Since W'_e has at least $k^*/(6\gamma^3)$ nodes and there are less than $\gamma - 1$ edges that appear before e, W'_e contains a subset W''_e of $k^*/(6\gamma^4)$ nodes such that $W''_e \cap W''_{e'} = \emptyset$ for each edge e' that appears before e in the ordering.

Recall that the white nodes of G'' resulted from the contraction of disjoint subgraphs of G' that are connected and they consist of only white nodes of G'. Since $v \in W''_e$ is adjacent to s_i and s_j — where s_i and s_j are the endpoints of e — we can find a path p_v in G' from s_i to s_j through the subgraph corresponding to v. Thus we obtain $|W''_e|$ paths in G' from s_i to s_j. This is the desired collection $\mathcal{P}_e$ for edge $e = s_i s_j$. The sets $\{W''_e \mid e \in E(T^*)\}$ are mutually disjoint by construction, and hence the paths in $\{\mathcal{P}_e \mid e \in E(T^*)\}$ have the desired properties.

This completes the proof of Theorem 5.2. □

Acknowledgments: We thank Julia Chuzhoy for several discussions and clarifications on her work [17, 19], and for sharing a draft of [19]. Her algorithm in [17] is the inspiration for this paper. CC thanks Sanjeev Khanna and Bruce Shepherd for many past and ongoing discussions on disjoint paths and related problems that influence this work.

References

[1] A. Agarwal, M. Charikar, K. Makarychev, and Y. Makarychev. $O(\sqrt{\log n})$ approximation algorithms for min UnCut, min 2CNF deletion, and directed cut problems. In *Proc. of ACM STOC*, pages 573–581, 2005.

[2] M. Andrews. Approximation algorithms for the edge-disjoint paths problem via Raecke decompositions. In *Proc. of IEEE FOCS*, pages 277–286, 2010.

[3] M. Andrews, J. Chuzhoy, V. Guruswami, S. Khanna, K. Talwar, and L. Zhang. Inapproximability of edge-disjoint paths and low congestion routing on undirected graphs. *Combinatorica*, 30(5):485–520, 2010.

[4] Y. Azar and O. Regev. Combinatorial algorithms for the unsplittable flow problem. *Algorithmica*, 44(1):49–66, 2006.

[5] A. Z. Broder, A. M. Frieze, and E. Upfal. Existence and construction of edge-disjoint paths on expander graphs. *SIAM Journal on Computing*, 23(5):976–989, 1994.

[6] A. Chakrabarti, C. Chekuri, A. Gupta, and A. Kumar. Approximation algorithms for the unsplittable flow problem. *Algorithmica*, 47(1):53–78, 2007.

[7] C. Chekuri, S. Khanna, and F.B. Shepherd. The all-or-nothing multicommodity flow problem. In *Proc. of ACM STOC*, pages 156–165, 2004.

[8] C. Chekuri, S. Khanna, and F.B. Shepherd. Edge-disjoint paths in planar graphs. In *Proc. of IEEE FOCS*, pages 71–80, 2004.

[9] C. Chekuri, S. Khanna, and F.B. Shepherd. Multicommodity flow, well-linked terminals, and routing problems. In *Proc. of ACM-STOC*, pages 183–192, 2005.

[10] C. Chekuri, S. Khanna, and F.B. Shepherd. Well-linked terminals for node-capacitated routing problems. Manuscript, 2005.

[11] C. Chekuri, S. Khanna, and F.B. Shepherd. An $O(\sqrt{n})$ approximation and integrality gap for disjoint paths and unsplittable flow. *Theory of Computing*, 2(7):137–146, 2006.

[12] C. Chekuri, S. Khanna, and F.B. Shepherd. Edge-disjoint paths in planar graphs with constant congestion. *SIAM Journal on Computing*, 39:281–301, 2009.

[13] C. Chekuri, S. Khanna, and F.B. Shepherd. A note on multiflows and treewidth. *Algorithmica*, 54(3):400–412, 2009.

[14] C. Chekuri and N. Korula. A graph reduction step preserving element-connectivity and applications. *Proc. of ICALP*, pages 254–265, 2009.

[15] C. Chekuri, M. Mydlarz, and F.B. Shepherd. Multicommodity demand flow in a tree and packing integer programs. *ACM Transactions on Algorithms*, 3(3):27, 2007.

[16] J. Cheriyan and M.R. Salavatipour. Packing element-disjoint steiner trees. *ACM Transactions on Algorithms*, 3(4):47, 2007.

[17] J. Chuzhoy. Routing in undirected graphs with constant congestion. *Arxiv preprint ArXiv:1107.2554*, 2011. Extended abstract in STOC 2012.

[18] J. Chuzhoy, V. Guruswami, S. Khanna, and K. Talwar. Hardness of routing with congestion in directed graphs. In *Proc. of ACM STOC*, pages 165–178, 2007.

[19] J. Chuzhoy and S. Li. A polylogarithimic approximation algorithm for edge-disjoint paths with congestion 2. In *Proc. of IEEE FOCS*, 2012.

[20] V. Chvátal. Tough graphs and hamiltonian circuits. *Discrete Mathematics*, 5(3):215–228, 1973.

[21] S. Even, A. Itai, and A. Shamir. On the complexity of timetable and multicommodity flow problems. *SIAM Journal on Computing*, 5(4):691–703, 1976.

[22] U. Feige, M.T. Hajiaghayi, and J.R. Lee. Improved approximation algorithms for minimum weight vertex separators. *SIAM Journal on Computing*, 38:629–657, 2008.

[23] A. Frank. On connectivity properties of Eulerian digraphs. *Annals of Discrete Mathematics*, 41:179–194, 1989.

[24] A.M. Frieze. Edge-disjoint paths in expander graphs. *SIAM Journal on Computing*, 30(6):1790–1801, 2001.

[25] M. Fürer and B. Raghavachari. Approximating the minimum-degree Steiner tree to within one of optimal. *Journal of Algorithms*, 17(3):409–423, 1994.

[26] N. Garg, V.V. Vazirani, and M. Yannakakis. Primal-dual approximation algorithms for integral flow and multicut in trees. *Algorithmica*, 18(1):3–20, 1997.

[27] V. Guruswami, S. Khanna, R. Rajaraman, B. Shepherd, and M. Yannakakis. Near-optimal hardness results and approximation algorithms for edge-disjoint paths and related problems. *Journal of Computer and System Sciences*, 67(3):473–496, 2003.

[28] H.R. Hind and O. Oellermann. Menger-type results for three or more vertices. *Congressus Numerantium*, pages 179–204, 1996.

[29] B. Jackson. Some remarks on arc-connectivity, vertex splitting, and orientation in graphs and digraphs. *Journal of Graph Theory*, 12(3):429–436, 1988.

[30] D. R. Karger. Random sampling in cut, flow, and network design problems. *Mathematics of Operations Research*, 24(2):383–413, 1999.

[31] K. Kawarabayashi and Y. Kobayashi. The edge disjoint paths problem in Eulerian graphs and 4-edge-connected graphs. In *Proc. of ACM-SIAM SODA*, pages 345–353, 2010.

[32] K. Kawarabayashi and Y. Kobayashi. Breaking $O(n^{1/2})$-approximation algorithms for the edge-disjoint paths problem with congestion two. In *Proc. of ACM STOC*, pages 81–88, 2011.

[33] R. Khandekar, S. Rao, and U. Vazirani. Graph partitioning using single commodity flows. *Journal of the ACM*, 56(4):19, 2009.

[34] J. Kleinberg. Approximation algorithms for disjoint paths problems. Ph.D. thesis, MIT, 1996.

[35] J. Kleinberg. An approximation algorithm for the disjoint paths problem in even-degree planar graphs. In *Proc. of IEEE FOCS*, pages 627–636, 2005.

[36] J. Kleinberg and R. Rubinfeld. Short paths in expander graphs. In *Proc. of IEEE FOCS*, pages 86–95, 1996.

[37] J. Kleinberg and E. Tardos. Disjoint paths in densely embedded graphs. In *Proc. of IEEE FOCS*, pages 52–61, 1995.

[38] J. Kleinberg and E. Tardos. Approximations for the disjoint paths problem in high-diameter planar networks. *Journal of Computers and System Sciences*, 57(1):61–73, 1998.

[39] S.G. Kolliopoulos and C. Stein. Approximating disjoint-path problems using packing integer programs. *Mathematical Programming*, 99(1):63–87, 2004.

[40] P. Kolman and C. Scheideler. Improved bounds for the unsplittable flow problem. *Journal of Algorithms*, 61(1):20–44, 2006.

[41] W. Mader. A reduction method for edge-connectivity in graphs. *Annals of Discrete Mathematics*, 3:145–164, 1978.

[42] G. Naves and A. Sebo. Multiflow feasibility: An annotated tableau. *Research Trends in Combinatorial Optimization: Bonn 2008*, page 261, 2008.

[43] L. Orecchia, L.J. Schulman, U.V. Vazirani, and N.K. Vishnoi. On partitioning graphs via single commodity flows. In *Proc. of ACM STOC*, pages 461–470, 2008.

[44] H. Racke. Minimizing congestion in general networks. In *Proc. of IEEE FOCS*, pages 43–52, 2002.

[45] P. Raghavan and C.D. Tompson. Randomized rounding: a technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, 1987.

[46] S. Rao and S. Zhou. Edge disjoint paths in moderately connected graphs. *SIAM Journal on Computing*, 39(5):1856–1887, 2010.

[47] N. Robertson, P. Seymour, and R. Thomas. Quickly excluding a planar graph. *Journal of Combinatorial Theory, Series B*, 62(2):323–348, 1994.

[48] N. Robertson and P.D. Seymour. Graph minors V. Excluding a planar graph. *Journal of Combinatorial Theory, Series B*, 41(1):92–114, 1986.

[49] N. Robertson and P.D. Seymour. Graph minors XIII. The disjoint paths problem. *Journal of Combinatorial Theory, Series B*, 63(1):65–110, 1995.

[50] A. Schrijver. *Combinatorial optimization*. Springer, 2003.

[51] L. Seguin-Charbonneau and F.B. Shepherd. Maximum edge-disjoint paths in planar graphs with congestion 2. In *Proc. of IEEE FOCS*, pages 200–209, 2011.

[52] M. Singh and L.C. Lau. Approximating minimum bounded degree spanning trees to within one of optimal. In *Proc. of ACM STOC*, pages 661–670, 2007.

[53] A. Srinivasan. Improved approximations for edge-disjoint paths, unsplittable flow, and related routing problems. In *Proc. of IEEE FOCS*, pages 416–425, 1997.

[54] S. Win. On a connection between the existence of k-trees and the toughness of a graph. *Graphs and Combinatorics*, 5(1):201–205, 1989.