Improved Approximation for Node-Disjoint Paths in Grids with Sources on the Boundary

Julia Chuzhoy[1]
Toyota Technological Institute at Chicago, 6045 S. Kenwood Ave., Chicago, Illinois 60637, USA
cjulia@ttic.edu

David H. K. Kim[2]
Computer Science Department, University of Chicago, 1100 East 58th Street, Chicago, Illinois 60637, USA
hongk@cs.uchicago.edu

Rachit Nimavat[3]
Toyota Technological Institute at Chicago, 6045 S. Kenwood Ave., Chicago, Illinois 60637, USA
nimavat@ttic.edu

Abstract

We study the classical Node-Disjoint Paths (NDP) problem: given an undirected n-vertex graph G, together with a set $\{(s_1,t_1),\ldots,(s_k,t_k)\}$ of pairs of its vertices, called source-destination, or demand pairs, find a maximum-cardinality set $\mathcal{P}$ of mutually node-disjoint paths that connect the demand pairs. The best current approximation for the problem is achieved by a simple greedy $O(\sqrt{n})$-approximation algorithm. Until recently, the best negative result was an $\Omega(\log^{1/2-\epsilon} n)$-hardness of approximation, for any fixed ϵ, under standard complexity assumptions.

A special case of the problem, where the underlying graph is a grid, has been studied extensively. The best current approximation algorithm for this special case achieves an $\tilde{O}(n^{1/4})$-approximation factor. On the negative side, a recent result by the authors shows that NDP is hard to approximate to within factor $2^{\Omega(\sqrt{\log n})}$, even if the underlying graph is a subgraph of a grid, and all source vertices lie on the grid boundary. In a very recent follow-up work, the authors further show that NDP in grid graphs is hard to approximate to within factor $\Omega(2^{\log^{1-\epsilon} n})$ for any constant ϵ under standard complexity assumptions, and to within factor $n^{\Omega(1/(\log\log n)^2)}$ under randomized ETH.

In this paper we study the NDP problem in grid graphs, where all source vertices $\{s_1,\ldots,s_k\}$ appear on the grid boundary. Our main result is an efficient randomized $2^{O(\sqrt{\log n}\cdot\log\log n)}$-approximation algorithm for this problem. Our result in a sense complements the $2^{\Omega(\sqrt{\log n})}$-hardness of approximation for sub-graphs of grids with sources lying on the grid boundary, and should be contrasted with the above-mentioned almost polynomial hardness of approximation of NDP in grid graphs (where the sources and the destinations may lie anywhere in the grid).

Much of the work on approximation algorithms for NDP relies on the multicommodity flow relaxation of the problem, which is known to have an $\Omega(\sqrt{n})$ integrality gap, even in grid graphs, with all source and destination vertices lying on the grid boundary. Our work departs from this paradigm, and uses a (completely different) linear program only to select the pairs to be routed, while the routing itself is computed by other methods.

2012 ACM Subject Classification Theory of computation → Routing and network design problems

Keywords and phrases Node-disjoint paths, approximation algorithms, routing and layout

[1] Supported in part by NSF grants CCF-1318242 and CCF-1616584.
[2] Supported in part by NSF grants CCF-1318242 and CCF-1616584.
[3] Supported in part by NSF grant CCF-1318242.

45th International Colloquium on Automata, Languages, and Programming (ICALP 2018).
Editors: Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella;
Article No. 38; pp. 38:1–38:14

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Digital Object Identifier 10.4230/LIPIcs.ICALP.2018.38

Related Version A full version of the paper is available at `https://arxiv.org/abs/1805.09956`.

1 Introduction

We study the classical Node-Disjoint Paths (NDP) problem, where the input consists of an undirected n-vertex graph G and a collection $\mathcal{M} = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ of pairs of its vertices, called *source-destination* or *demand* pairs. We say that a path P *routes* a demand pair (s_i, t_i) iff the endpoints of P are s_i and t_i. The goal is to compute a maximum-cardinality set $\mathcal{P}$ of node-disjoint paths, where each path $P \in \mathcal{P}$ routes a distinct demand pair in $\mathcal{M}$. We denote by NDP-Planar the special case of the problem when the underlying graph G is planar, and by NDP-Grid the special case where G is a square grid[4]. We refer to the vertices in set $S = \{s_1, \ldots, s_k\}$ as *source vertices*; to the vertices in set $T = \{t_1, \ldots, t_k\}$ as *destination vertices*, and to the vertices in set $S \cup T$ as *terminals*.

NDP is a fundamental graph routing problem that has been studied extensively in both graph theory and theoretical computer science communities. Robertson and Seymour [31, 33] explored the problem in their Graph Minor series, providing an efficient algorithm for NDP when the number k of the demand pairs is bounded by a constant. But when k is a part of input, the problem becomes NP-hard [20, 18], even in planar graphs [27], and even in grid graphs [26]. The best current approximation factor of $O(\sqrt{n})$ for NDP is achieved by a simple greedy algorithm [25]. Until recently, this was also the best approximation algorithm for NDP-Planar and NDP-Grid. A natural way to design approximation algorithms for NDP is via the multicommodity flow relaxation: instead of connecting each routed demand pair with a path, send maximum possible amount of (possibly fractional) flow between them. The optimal solution to this relaxation can be computed via a standard linear program. The $O(\sqrt{n})$-approximation algorithm of [25] can be cast as an LP-rounding algorithm of this relaxation. Unfortunately, it is well-known that the integrality gap of this relaxation is $\Omega(\sqrt{n})$, even when the underlying graph is a grid, with all terminals lying on its boundary. In a recent work, Chuzhoy and Kim [12] designed an $\tilde{O}(n^{1/4})$-approximation for NDP-Grid, thus bypassing this integrality gap barrier. Their main observation is that, if all terminals lie close to the grid boundary (say within distance $O(n^{1/4})$), then a simple dynamic programming-based algorithm yields an $O(n^{1/4})$-approximation. On the other hand, if, for every demand pair, either the source or the destination lies at a distance at least $\Omega(n^{1/4})$ from the grid boundary, then the integrality gap of the multicommodity flow relaxation improves, and one can obtain an $\tilde{O}(n^{1/4})$-approximation via LP-rounding. A natural question is whether the integrality gap improves even further, if all terminals lie further away from the grid boundary. Unfortunately, the authors show in [12] that the integrality gap remains at least $\Omega(n^{1/8})$, even if all terminals lie within distance $\Omega(\sqrt{n})$ from the grid boundary. The $\tilde{O}(n^{1/4})$-approximation algorithm for NDP-Grid was later extended and generalized to an $\tilde{O}(n^{9/19})$-approximation algorithm for NDP-Planar [13].

On the negative side, until recently, only an $\Omega(\log^{1/2-\epsilon} n)$-hardness of approximation was known for the general version of NDP, for any constant ϵ, unless NP $\subseteq$ ZPTIME$(n^{\text{poly}\log n})$ [4, 3], and only APX-hardness was known for NDP-Planar and NDP-Grid [12]. In a recent

[4] We use the standard convention of denoting $n = |V(G)|$, and so the grid has dimensions $(\sqrt{n} \times \sqrt{n})$; we assume that $\sqrt{n}$ is an integer.

work [15], the authors have shown that NDP is hard to approximate to within a $2^{\Omega(\sqrt{\log n})}$ factor unless $\mathsf{NP} \subseteq \mathsf{DTIME}(n^{O(\log n)})$, even if the underlying graph is a planar graph with maximum vertex degree at most 3, and all source vertices lie on the boundary of a single face. The result holds even when the input graph G is a vertex-induced subgraph of a grid, with all sources lying on the grid boundary. In a very recent work [14], the authors show that NDP-Grid is $2^{\Omega(\log^{1-\epsilon} n)}$-hard to approximate for any constant ϵ assuming $\mathsf{NP} \not\subseteq \mathsf{BPTIME}(n^{\mathrm{poly}\log n})$, and moreover, assuming randomized ETH, the hardness of approximation factor becomes $n^{\Omega(1/(\log\log n)^2)}$. We note that the instances constructed in these latter hardness proofs require all terminals to lie far from the grid boundary.

In this paper we explore NDP-Grid. This important special case of NDP was initially motivated by applications in VLSI design, and has received a lot of attention since the 1960's. We focus on a restricted version of NDP-Grid, that we call Restricted NDP-Grid: here, in addition to the graph G being a square grid, we also require that all source vertices $\{s_1, \ldots, s_k\}$ lie on the grid boundary. We do not make any assumptions about the locations of the destination vertices, that may appear anywhere in the grid. The best current approximation algorithm for Restricted NDP-Grid is the same as that for the general NDP-Grid, and achieves a $\tilde{O}(n^{1/4})$-approximation [12]. Our main result is summarized in the following theorem.

▶ **Theorem 1.** *There is an efficient randomized $2^{O(\sqrt{\log n}\cdot\log\log n)}$-approximation algorithm for* Restricted NDP-Grid.

This result in a sense complements the $2^{\Omega(\sqrt{\log n})}$-hardness of approximation of NDP on sub-graphs of grids with all sources lying on the grid boundary of [15][5], and should be contrasted with the recent almost polynomial hardness of approximation of [14] for NDP-Grid mentioned above. Our algorithm departs from previous work on NDP in that it does not use the multicommodity flow relaxation. Instead, we define sufficient conditions that allow us to route a subset $\mathcal{M}'$ of demand pairs via disjoint paths, and show that there exists a subset of demand pairs satisfying these conditions, whose cardinality is at least $\mathsf{OPT}/2^{O(\sqrt{\log n}\cdot\log\log n)}$, where OPT is the value of the optimal solution. It is then enough to compute a maximum-cardinality subset of the demand pairs satisfying these conditions. We write an LP-relaxation for this problem and design a $2^{O(\sqrt{\log n}\cdot\log\log n)}$-approximation LP-rounding algorithm for it. We emphasize that the linear program is only used to select the demand pairs to be routed, and not to compute the routing itself.

We then generalize this result to instances where the source vertices lie within a prescribed distance from the grid boundary.

▶ **Theorem 2.** *For every integer $\delta \geq 1$, there is an efficient randomized $\left(\delta \cdot 2^{O(\sqrt{\log n}\cdot\log\log n)}\right)$-approximation algorithm for the special case of* NDP-Grid *where all source vertices lie within distance at most δ from the grid boundary.*

We note that for instances of NDP-Grid where both the sources and the destinations are within distance at most δ from the grid boundary, it is easy to obtain an efficient $O(\delta)$-approximation algorithm (see, e.g. [12]).

A problem closely related to NDP is the Edge-Disjoint Paths (EDP) problem. It is defined similarly, except that now the paths chosen to route the demand pairs may share vertices,

[5] Note that the two results are not strictly complementary: our algorithm only applies to grid graphs, while the hardness result is only valid for sub-graphs of grids.

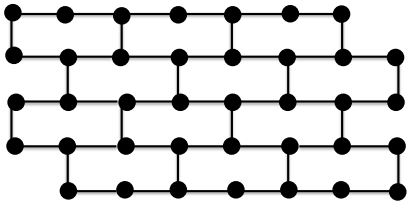


Figure 1 A wall graph.

and are only required to be edge-disjoint. The approximability status of EDP is very similar to that of NDP: there is an $O(\sqrt{n})$-approximation algorithm [10], and an $\Omega(\log^{1/2-\epsilon} n)$-hardness of approximation for any constant ϵ, unless NP $\subseteq$ ZPTIME$(n^{\text{poly}\log n})$ [4, 3]. As in the NDP problem, we can use the standard multicommodity flow LP-relaxation of the problem, in order to obtain the $O(\sqrt{n})$-approximation algorithm, and the integrality gap of the LP-relaxation is $\Omega(\sqrt{n})$ even in planar graphs. Recently, Fleszar et al. [19] designed an $O(\sqrt{r}\cdot\log(kr))$-approximation algorithm for EDP, where r is the feedback vertex set number of the input graph $G=(V,E)$ — the smallest number of vertices that need to be deleted from G in order to turn it into a forest.

Several special cases of EDP have better approximation algorithms: an $O(\log^2 n)$-approximation is known for even-degree planar graphs [9, 8, 22], and an $O(\log n)$-approximation is known for nearly-Eulerian uniformly high-diameter planar graphs, and nearly-Eulerian densely embedded graphs, including grid graphs [5, 24, 23]. Furthermore, an $O(\log n)$-approximation algorithm is known for EDP on 4-edge-connected planar, and Eulerian planar graphs [21]. It appears that the restriction of the graph G to be Eulerian, or near-Eulerian, makes the EDP problem on planar graphs significantly simpler, and in particular improves the integrality gap of the standard multicommodity flow LP-relaxation.

The analogue of the grid graph for the EDP problem is the wall graph (see Figure 1): the integrality gap of the multicommodity flow relaxation for EDP on wall graphs is $\Omega(\sqrt{n})$. The $\tilde{O}(n^{1/4})$-approximation algorithm of [12] for NDP-Grid extends to EDP on wall graphs, and the $2^{\Omega(\sqrt{\log n})}$-hardness of approximation of [15] for NDP-Planar also extends to EDP on sub-graphs of walls, with all sources lying on the top boundary of the wall. The recent hardness result of [14] for NDP-Grid also extends to an $2^{\Omega(\log^{1-\epsilon} n)}$-hardness of EDP on wall graphs, assuming NP $\not\subseteq$ BPTIME$(n^{\text{poly}\log n})$, and to $n^{\Omega(1/(\log\log n)^2)}$-hardness assuming randomized ETH. We extend our results to EDP and NDP on wall graphs:

▶ **Theorem 3.** *There is an efficient randomized $2^{O(\sqrt{\log n}\cdot\log\log n)}$-approximation algorithm for* EDP *and for* NDP *on wall graphs, when all source vertices lie on the wall boundary.*

Other related work

Cutler and Shiloach [17] studied an even more restricted version of NDP-Grid, where all source vertices lie on the top row R^* of the grid, and all destination vertices lie on a single row R' of the grid, far enough from its top and bottom boundaries. They considered three different settings of this special case. In the packed-packed setting, all sources appear consecutively on R^*, and all destinations appear consecutively on R' (but both sets may appear in an arbitrary order). They show a necessary and a sufficient condition for all demand pairs to be routable via node-disjoint paths in this setting. The second setting is the packed-spaced setting. Here, the sources again appear consecutively on R^*, but all destinations are at a distance at least

d from each other. For this setting, the authors show that if $d \geq k$, then all demand pairs can be routed. We note that [12] extended their algorithm to a more general setting, where the destination vertices may appear anywhere in the grid, as long as the distance between any pair of the destination vertices, and any destination vertex and the boundary of the grid, is at least $\Omega(k)$. Robertson and Seymour [32] provided sufficient conditions for the existence of node-disjoint routing of a given set of demand pairs in the more general setting of graphs drawn on surfaces, and they designed an algorithm whose running time is $\text{poly}(n) \cdot f(k)$ for finding the routing, where $f(k)$ is at least exponential in k. Their result implies the existence of the routing in grids, when the destination vertices are sufficiently far from each other and from the grid boundaries, but it does not provide an efficient algorithm to compute such a routing. The third setting studied by Cutler and Shiloach is the spaced-spaced setting, where the distances between every pair of source vertices, and every pair of destination vertices are at least d. The authors note that they could not come up with a better algorithm for this setting, than the one provided for the packed-spaced case. Aggarwal, Kleinberg, and Williamson [1] considered a special case of NDP-Grid, where the set of the demand pairs is a permutation: that is, every vertex of the grid participates in exactly one demand pair. They show that $\Omega(\sqrt{n}/\log n)$ demand pairs are routable in this case via node-disjoint paths. They further show that if all terminals are at a distance at least d from each other, then at least $\Omega(\sqrt{nd}/\log n)$ pairs are routable.

A variation of the NPD and EDP problems, where small congestion is allowed, has been a subject of extensive study, starting with the classical paper of Raghavan and Thompson [29] that introduced the randomized rounding technique. We say that a set $\mathcal{P}$ of paths causes congestion c, if at most c paths share the same vertex or the same edge, for the NDP and the EDP settings respectively. A recent line of work [9, 28, 2, 30, 11, 16, 7, 6] has lead to an $O(\text{poly}\log k)$-approximation for both NDP and EDP problems with congestion 2. For planar graphs, a constant-factor approximation with congestion 2 is known [34].

Organization

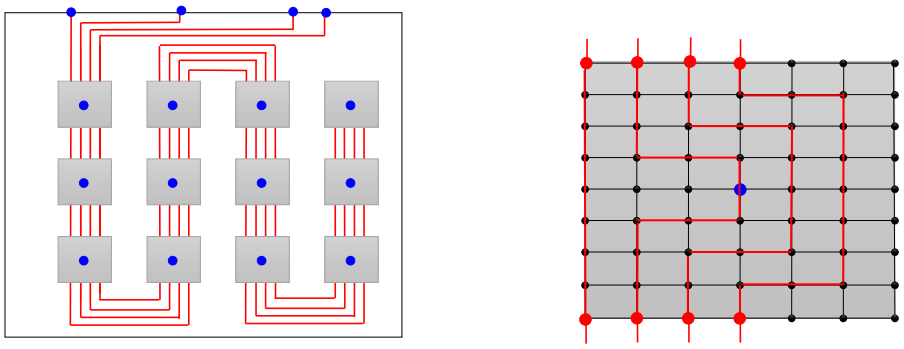
The majority of this extended abstract is dedicated to a detailed but informal overview of the proofs of Theorem 1 and Theorem 2. The formal proofs, as well as the extension to EDP and NDP on wall graphs, are deferred to the full version of the paper.

2 High-Level Overview of the Algorithm

The goal of this section is to provide an informal high-level overview of the main result of the paper – the proof of Theorem 1. With this goal in mind, the values of various parameters are given imprecisely in this section, in a way that best conveys the intuition. The full version of the paper contains a formal description of the algorithm and the precise settings of all parameters.

We first consider an even more restricted special case of NDP-Grid, where all source vertices appear on the top boundary of the grid, and all destination vertices appear far enough from the grid boundary, and design an efficient randomized $2^{O(\sqrt{\log n \cdot \log\log n})}$-approximation algorithm $\mathcal{A}$ for this problem. We later show how to reduce Restricted NDP-Grid to this special case of the problem; we focus on the description of the algorithm $\mathcal{A}$ for now.

We assume that our input graph G is the $(\ell \times \ell)$-grid, and we denote by $n = \ell^2$ the number of its vertices. We further assume that the set of the demand pairs is $\mathcal{M} = \{(s_1, t_1), \ldots, (s_k, t_k)\}$, with the vertices in set $S = \{s_1, \ldots, s_k\}$ called source vertices; the vertices in set $T = \{t_1, \ldots, t_k\}$ called destination vertices; and the vertices in $S \cup T$ called



(a) Global routing. In this figure, the sub-grids B_i are aligned vertically and horizontally. A similar (but somewhat more complicated) routing can be performed even if they are not aligned. For convenience we did not include all source vertices and all paths.

(b) Local routing inside B_i

Figure 2 Schematic view of routing of spaced-out instances.

terminals. Let OPT denote the value of the optimal solution to the NDP instance $(G, \mathcal{M})$. We assume that the vertices of S lie on the top boundary of the grid, that we denote by R^*, and the vertices of T lie sufficiently far from the grid boundary – say, at a distance at least OPT from it. For a subset $\mathcal{M}' \subseteq \mathcal{M}$ of the demand pairs, we denote by $S(\mathcal{M}')$ and $T(\mathcal{M}')$ the sets of the source and the destination vertices of the demand pairs in $\mathcal{M}'$, respectively. As our starting point, we consider a simple observation of Chuzhoy and Kim [12], that generalizes the results of Cutler and Shiloach [17]. Suppose we are given an instance of NDP-Grid with k demand pairs, where the sources lie on the top boundary of the grid, and the destination vertices may appear anywhere in the grid, but the distance between every pair of the destination vertices, and every destination vertex and the boundary of the grid, is at least $(8k + 8)$ – we call such instances *spaced-out instances*. In this case, all demand pairs in $\mathcal{M}$ can be efficiently routed via node-disjoint paths, as follows. Consider, for every destination vertex $t_i \in T$, a square sub-grid B_i of G, of size $(2k \times 2k)$, such that t_i lies roughly at the center of B_i. We construct a set $\mathcal{P}$ of k node-disjoint paths, that originate at the vertices of S, and traverse the sub-grids B_i one-by-one in a snake-like fashion (see a schematic view on Figure 2a). We call this part of the routing *global routing*. The *local routing* needs to specify how the paths in $\mathcal{P}$ traverse each box B_i. This is done in a straightforward manner, while ensuring that the unique path originating at vertex s_i visits the vertex t_i (see Figure 2b). By suitably truncating the final set $\mathcal{P}$ of paths, we obtain a routing of all demand pairs in $\mathcal{M}$ via node-disjoint paths.

Unfortunately, in our input instance $(G, \mathcal{M})$, the destination vertices may not be located sufficiently far from each other. We can try to select a large subset $\mathcal{M}' \subseteq \mathcal{M}$ of the demand pairs, so that every pair of destination vertices in $T(\mathcal{M}')$ appear at a distance at least $\Omega(|\mathcal{M}'|)$ from each other; but in some cases the largest such set $\mathcal{M}'$ may only contain $O(\mathsf{OPT}/\sqrt{k})$ demand pairs (for example, suppose all destination vertices lie consecutively on a single row of the grid). One of our main ideas is to generalize this simple algorithm to a number of recursive levels.

For simplicity, let us first describe the algorithm with just two recursive levels. Suppose we partition the top row of the grid into z disjoint intervals, $I_1, \ldots, I_z$. Let $\mathcal{M}' \subseteq \mathcal{M}$ be a set of demand pairs that we would like to route. Denote $|\mathcal{M}'| = k'$, and assume that we are given a collection $\mathcal{Q}$ of square sub-grids of G, of size $(4k' \times 4k')$ each (that we call *squares*),

such that every pair $Q, Q' \in \mathcal{Q}$ of distinct squares is at a distance at least $4k'$ from each other. Assume further that each such sub-grid $Q \in \mathcal{Q}$ is assigned a color $\chi(Q) \in \{c_1, \ldots, c_z\}$, such that, if Q is assigned the color c_j, then all demand pairs $(s,t) \in \mathcal{M}'$ whose destination t lies in Q have their source $s \in I_j$ (so intuitively, each color c_j represents an interval I_j). Let $\mathcal{M}'_j \subseteq \mathcal{M}'$ be the set of all demand pairs $(s,t) \in \mathcal{M}'$ with $s \in I_j$. We would like to ensure that $|\mathcal{M}'_j|$ is roughly k'/z, and that all destination vertices of $T(\mathcal{M}'_j)$ are at a distance at least $|\mathcal{M}'_j|$ from each other. We claim that if we could find the collection $\{I_1, \ldots, I_z\}$ of the intervals of the first row, a collection $\mathcal{Q}$ of sub-grids of G, a coloring $\chi : \mathcal{Q} \to \{c_1, \ldots, c_z\}$, and a subset $\mathcal{M}' \subseteq \mathcal{M}$ of the demand pairs with these properties, then we would be able to route all demand pairs in $\mathcal{M}'$.

In order to do so, for each square $Q \in \mathcal{Q}$, we construct an augmented square Q^+, by adding a margin of k' rows and columns around Q. Our goal is to construct a collection $\mathcal{P}$ of node-disjoint paths routing the demand pairs in $\mathcal{M}'$. We start by constructing a global routing, where all paths in $\mathcal{P}$ originate from the vertices of $S(\mathcal{M}')$ and then visit the squares in $\{Q^+ \mid Q \in \mathcal{Q}\}$ in a snake-like fashion, just like we did for the spaced-out instances described above (see Figure 2a). Consider now some square $Q \in \mathcal{Q}$ and the corresponding augmented square Q^+. Assume that $\chi(Q) = c_j$, and let $\mathcal{P}_j \subseteq \mathcal{P}$ be the set of paths originating at the source vertices that lie in I_j. While traversing the square Q^+, we ensure that only the paths in $\mathcal{P}_j$ enter the square Q; the remaining paths use the margins on the left and on the right of Q in order to traverse Q^+. This can be done because the sources of the paths in $\mathcal{P}_j$ appear consecutively on R^*, relatively to the sources of all paths in $\mathcal{P}$. In order to complete the local routing inside the square Q, observe that the destination vertices appear far enough from each other, and so we can employ the simple algorithm for spaced-out instances inside Q.

In order to optimize the approximation factor that we achieve, we extend this approach to $\rho = O(\sqrt{\log n})$ recursive levels. Let $\eta = 2^{\lceil \sqrt{\log n} \rceil}$. We define auxiliary parameters $d_1 > d_2 > \cdots > d_\rho > d_{\rho+1}$. Roughly speaking, we can think of $d_{\rho+1}$ as being a constant (say 16), of d_1 as being comparable to OPT, and for all $1 \leq h \leq \rho$, $d_{h+1} = d_h/\eta$. The setup for the algorithm consists of three ingredients: (i) a hierarchical decomposition $\tilde{\mathcal{H}}$ of the grid into square sub-grids (that we refer to as squares); (ii) a hierarchical partition $\mathfrak{I}$ of the first row R^* of the grid into intervals; and (iii) a hierarchical coloring f of the squares in $\tilde{\mathcal{H}}$ with colors that correspond to the intervals of $\mathfrak{I}$, together with a selection of a subset $\mathcal{M}' \subseteq \mathcal{M}$ of the demand pairs to route. We define sufficient conditions on the hierarchical system $\tilde{\mathcal{H}}$ of squares, the hierarchical partition $\mathfrak{I}$ of R^* into intervals, the coloring f and the subset $\mathcal{M}'$ of the demand pairs, under which a routing of all pairs in $\mathcal{M}'$ exists and can be found efficiently. For a fixed hierarchical system $\tilde{\mathcal{H}}$ of squares, a triple $(\mathfrak{I}, f, \mathcal{M}')$ satisfying these conditions is called a *good ensemble*. We show that a good ensemble with a large enough set $\mathcal{M}'$ of demand pairs exists, and then design an approximation algorithm for computing a good ensemble maximizing $|\mathcal{M}'|$. We now describe each of these ingredients in turn.

2.1 A Hierarchical System of Squares.

A hierarchical system $\tilde{\mathcal{H}}$ of squares consists of a sequence $\mathcal{Q}_1, \mathcal{Q}_2, \ldots, \mathcal{Q}_\rho$ of sets of sub-grids of G. For each $1 \leq h \leq \rho$, $\mathcal{Q}_h$ is a collection of disjoint sub-grids of G (that we refer to as *level-h squares*); every such square $Q \in \mathcal{Q}_h$ has size $(d_h \times d_h)$, and every pair of distinct squares $Q, Q' \in \mathcal{Q}_h$ are within distance at least d_h from each other (see Figure 3). We require that for each $1 < h \leq \rho$, for every square $Q \in \mathcal{Q}_h$, there is a unique square $Q' \in \mathcal{Q}_{h-1}$ (called the *parent-square* of Q) that contains Q. We say that a demand pair (s,t) *belongs* to the hierarchical system $\tilde{\mathcal{H}} = (\mathcal{Q}_1, \mathcal{Q}_2, \ldots, \mathcal{Q}_\rho)$ of squares iff $t \in \bigcup_{Q \in \mathcal{Q}_\rho} Q$. We show a

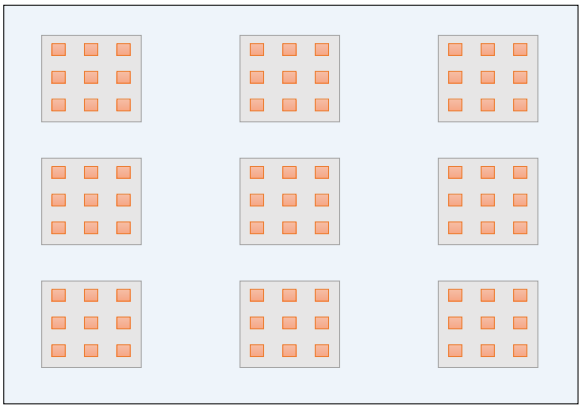


Figure 3 A schematic view of a hierarchical system of squares with 2 levels.

simple efficient algorithm to construct $2^{O(\sqrt{\log n})}$ such hierarchical systems of squares, so that every demand pair belongs to at least one of them. Each such system $\tilde{\mathcal{H}}$ of squares induces an instance of NDP — the instance is defined over the same graph G, and the set $\tilde{\mathcal{M}} \subseteq \mathcal{M}$ of demand pairs that belong to the system $\tilde{\mathcal{H}}$. It is then enough to obtain a factor $2^{O(\sqrt{\log n} \cdot \log\log n)}$-approximation algorithm for each resulting instance $(G, \tilde{\mathcal{M}})$ separately. From now on we fix one such hierarchical system $\tilde{\mathcal{H}} = (\mathcal{Q}_1, \mathcal{Q}_2, \ldots, \mathcal{Q}_\rho)$ of squares, together with the set $\tilde{\mathcal{M}} \subseteq \mathcal{M}$ of demand pairs, containing all pairs (s,t) that belong to $\tilde{\mathcal{H}}$, and focus on designing a $2^{O(\sqrt{\log n} \cdot \log\log n)}$-approximation algorithm for instance $(G, \tilde{\mathcal{M}})$.

2.2 A Hierarchical Partition of the Top Grid Boundary

Recall that R^* denotes the first row of the grid. A hierarchical partition $\mathfrak{I}$ of R^* is a sequence $\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_\rho$ of sets of sub-paths of R^*, such that for each $1 \leq h \leq \rho$, the paths in $\mathcal{I}_h$ (that we refer to as *level-h intervals*) partition the vertices of R^*. We also require that for all $1 < h \leq \rho$, every level-h interval $I \in \mathcal{I}_h$ is contained in a unique level-$(h-1)$ interval $I' \in \mathcal{I}_{h-1}$, that we refer to as the *parent-interval* of I. For every level $1 \leq h \leq \rho$, we define a collection χ_h of colors, containing one color $c_h(I)$ for each level-h interval $I \in \mathcal{I}_h$. If $I' \in \mathcal{I}_h$ is a parent-interval of $I \in \mathcal{I}_{h+1}$, then we say that color $c_h(I')$ is a *parent-color* of $c_{h+1}(I)$.

2.3 Coloring the Squares and Selecting Demand Pairs to Route

The third ingredient of our algorithm is an assignment f of colors to the squares, and a selection of a subset of the demand pairs to be routed. For every level $1 \leq h \leq \rho$, for every level-h square $Q \in \mathcal{Q}_h$, we would like to assign a single level-h color $c_h(I) \in \chi_h$ to Q, denoting $f(Q) = c_h(I)$. Intuitively, if color $c_h(I)$ is assigned to Q, then the only demand pairs (s,t) with $t \in Q$ that we may route are those whose source vertex s lies on the level-h interval I. We require that the coloring is consistent across levels: that is, for all $1 < h \leq \rho$, if a level-h square is assigned a level-h color c_h, and its parent-square is assigned a level-$(h-1)$ color c_{h-1}, then c_{h-1} must be a parent-color of c_h. We call such a coloring f a *valid coloring* of $\tilde{\mathcal{H}}$ with respect to $\mathfrak{I}$.

Finally, we would like to select a subset $\mathcal{M}' \subseteq \tilde{\mathcal{M}}$ of the demand pairs to route. Consider some demand pair (s,t) and some level $1 \leq h \leq \rho$. Let I_h be the level-h interval to which s belongs. Then we say that s has the level-h color $c_h(I_h)$. Therefore, for each level $1 \leq h \leq \rho$,

vertex s is assigned the unique level-h color $c_h(I_h)$, and for $1 \leq h < \rho$, $c_h(I_h)$ is the parent-color of $c_{h+1}(I_{h+1})$. Let $Q_\rho \in \mathcal{Q}_\rho$ be the level-ρ square to which t belongs. We may only add (s,t) to $\mathcal{M}'$ if the level-ρ color of Q_ρ is $c_\rho(I_\rho)$ (that is, it is the same as the level-ρ color of s). Notice that in particular, this means that for every level $1 \leq h \leq \rho$, if Q_h is the level-h square containing t, and it is assigned the color $c_h(I_h)$, then s is assigned the same level-h color, and so $s \in I_h$. Finally, we require that for all $1 \leq h \leq \rho$, for every level-h color c_h, the total number of all demand pairs $(s,t) \in \mathcal{M}'$, such that the level-h color of s is c_h, is no more than $d_{h+1}/16$ (if $h = \rho$, then the number is no more than 1). If $\mathcal{M}'$ has all these properties, then we say that it *respects the coloring* f. We say that $(\mathfrak{I}, f, \mathcal{M}')$ is a *good ensemble* iff $\mathfrak{I}$ is a hierarchical partition of R^* into intervals; f is a valid coloring of the squares in $\tilde{\mathcal{H}}$ with respect to $\mathfrak{I}$; and $\mathcal{M}' \subseteq \tilde{\mathcal{M}}$ is a subset of the demand pairs that respects the coloring f. The *size* of the ensemble is $|\mathcal{M}'|$.

2.4 The Routing

We show that, if we are given a good ensemble $(\mathfrak{I}, f, \mathcal{M}')$, then we can route all demand pairs in $\mathcal{M}'$. The routing itself follows the high-level idea outlined above. We gradually construct a collection $\mathcal{P}$ of node-disjoint paths routing the demand pairs in $\mathcal{M}'$. At the highest level, all these paths depart from their sources and then visit the level-1 squares one-by-one, in a snake-like fashion, as in Figure 2a. Consider now some level-1 square Q, and assume that its level-1 color is $c_1(I)$, where $I \in \mathcal{I}_1$ is some level-1 interval of R^*. Then only the paths $P \in \mathcal{P}$ that originate at the vertices of I will enter the square Q; the remaining paths will exploit the spacing between the level-1 squares in order to bypass it; the spacing between the level-1 squares is sufficient to allow this. Once we have defined this global routing, we need to specify how the routing is carried out inside each square. We employ the same procedure recursively. Consider some level-1 square Q, and let $\mathcal{P}' \subseteq \mathcal{P}$ be the set of all paths that visit Q. Assume further that the level-1 color of Q is $c_1(I)$. Since we are only allowed to have at most $d_2/16$ demand pairs in $\mathcal{M}'$ whose level-1 color is $c_1(I)$, $|\mathcal{P}'| \leq d_2/16$. Let $\mathcal{Q}' \subseteq \mathcal{Q}_2$ be the set of all level-2 squares contained in Q. The paths in $\mathcal{P}'$ will visit the squares of $\mathcal{Q}'$ one-by-one in a snake-like fashion (but this part of the routing is performed inside Q). As before, for every level-2 square $Q' \subseteq Q$, if the level-2 color of Q' is $c_2(I')$, then only those paths of $\mathcal{P}'$ that originate at the vertices of I' will enter Q'; the remaining paths will use the spacing between the level-2 squares to bypass Q'. Since $|\mathcal{P}'| \leq d_2/16$, and all level-2 squares are at distance at least d_2 from each other, there is a sufficient spacing to allow this routing. We continue this process recursively, until, at the last level of the recursion, we route at most one path per color, to its destination vertex.

In order to complete the proof of the theorem, we need to show that there exists a good ensemble $(\mathfrak{I}, f, \mathcal{M}')$ of size $|\mathcal{M}'| \geq |\mathsf{OPT}|/2^{O(\sqrt{\log n \cdot \log\log n})}$, and that we can find such an ensemble efficiently.

2.5 The Existence of the Ensemble

The key notion that we use in order to show that a large good ensemble $(\mathfrak{I}, f, \mathcal{M}')$ exists is that of a *shadow property*. Suppose Q is some $(d \times d)$ sub-grid of G, and let $\hat{\mathcal{M}} \subseteq \mathcal{M}$ be some subset of the demand pairs. Among all demand pairs $(s,t) \in \hat{\mathcal{M}}$ with $t \in Q$, let (s_1, t_1) be the one with s_1 appearing earliest on the first row R^* of G, and let (s_2, t_2) be the one with s_2 appearing latest on R^*. The *shadow of* Q *with respect to* $\hat{\mathcal{M}}$ is the sub-path of R^* between s_1 and s_2. Let $N_{\hat{\mathcal{M}}}(Q)$ be the number of all demand pairs $(s,t) \in \hat{\mathcal{M}}$ with s lying in the shadow of Q (that is, s lies between s_1 and s_2 on R^*). We say that $\hat{\mathcal{M}}$ has the

shadow property with respect to Q iff $N_{\hat{\mathcal{M}}}(Q) \leq d$. We say that $\hat{\mathcal{M}}$ has the *shadow property with respect to the hierarchical system* $\tilde{\mathcal{H}} = (\mathcal{Q}_1, \ldots, \mathcal{Q}_\rho)$ *of squares*, iff $\hat{\mathcal{M}}$ has the shadow property with respect to every square in $\bigcup_{h=1}^{\rho} \mathcal{Q}_h$. Let $\mathcal{P}^*$ be the optimal solution to the instance $(G, \tilde{\mathcal{M}})$ of NDP, where $\tilde{\mathcal{M}}$ only includes the demand pairs that belong to $\tilde{\mathcal{H}}$. Let $\mathcal{M}^* \subseteq \tilde{\mathcal{M}}$ be the set of the demand pairs routed by $\mathcal{P}^*$. For every demand pair $(s,t) \in \mathcal{M}^*$, let $P(s,t) \in \mathcal{P}^*$ be the path routing this demand pair. Intuitively, it feels like $\mathcal{M}^*$ should have the shadow property. Indeed, let $Q \in \bigcup_{h=1}^{\rho} \mathcal{Q}_h$ be some square of size $(d_h \times d_h)$, and let $(s_1,t_1), (s_2,t_2) \in \mathcal{M}^*$ be defined for Q as before, so that the shadow of Q with respect to $\mathcal{M}^*$ is the sub-path of R^* between s_1 and s_2. Let P be any path of length at most $2d_h$ connecting t_1 to t_2 in Q, and let γ be the closed curve consisting of the union of $P(s_1,t_1)$, P, $P(s_2,t_2)$, and the shadow of Q. Consider the disc D whose boundary is γ. The intuition is that, if $(s,t) \in \mathcal{M}^*$ is a demand pair whose source lies in the shadow of Q, and destination lies outside of D, then $P(s,t)$ must cross the path P, as it needs to escape the disc D. Since path P is relatively short, only a small number of such demand pairs may exist. The main difficulty with this argument is that we may have a large number of demand pairs (s,t), whose source lies in the shadow of Q, and the destination lies in the disc D. Intuitively, this can only happen if $P(s_1,t_1)$ and $P(s_2,t_2)$ "capture" a large area of the grid. We show that, in a sense, this cannot happen too often, and that there is a subset $\mathcal{M}^{**} \subseteq \mathcal{M}^*$ of at least $|\mathcal{M}^*|/2^{O(\sqrt{\log n} \cdot \log\log n)}$ demand pairs, such that $\mathcal{M}^{**}$ has the shadow property with respect to $\tilde{\mathcal{H}}$.

Finally, we show that there exists a good ensemble $(\mathfrak{I}, f, \mathcal{M}')$ with $|\mathcal{M}'| \geq |\mathcal{M}^{**}|/2^{O(\sqrt{\log n} \cdot \log\log n)}$. We construct the ensemble over the course of ρ iterations, starting with $\mathcal{M}' = \mathcal{M}^{**}$. In the hth iteration we construct the set $\mathcal{I}_h$ of the level-h intervals of R^*, assign level-h colors to all level-h squares of $\tilde{\mathcal{H}}$, and discard some demand pairs from $\mathcal{M}'$. Recall that $\eta = 2^{\lceil \sqrt{\log n} \rceil}$. In the first iteration, we let $\mathcal{I}_1$ be a partition of the row R^* into intervals, each of which contains roughly $\frac{d_1}{16\eta} = \frac{d_2}{16} \leq \frac{|\mathcal{M}^*|}{\eta}$ vertices of $S(\mathcal{M}')$. Assume that these intervals are $I_1, \ldots, I_r$, and that they appear in this left-to-right order on R^*. We call all intervals I_j where j is odd *interesting intervals*, and the remaining intervals I_j *uninteresting intervals*. We discard from $\mathcal{M}'$ all demand pairs (s,t), where s lies on an uninteresting interval. Consider now some level-1 square Q, and let $\mathcal{M}(Q) \subseteq \mathcal{M}'$ be the set of all demand pairs whose destination lies in Q. Since the original set $\mathcal{M}^{**}$ of demand pairs had the shadow property with respect to Q, it is easy to verify that all source vertices of the demand pairs in $\mathcal{M}(Q)$ must belong to a single interesting interval of $\mathcal{I}_1$. Let I be that interval. Then we color the square Q with the level-1 color $c_1(I)$ corresponding to the interval I. This completes the first iteration. Notice that for each level-1 color $c_1(I)$, at most $d_2/16$ demand pairs $(s,t) \in \mathcal{M}'$ have $s \in I$. In the following iteration, we similarly partition every interesting level-1 interval into level-2 intervals that contain roughly $d_3/16 \leq |\mathcal{M}^*|/\eta^2$ source vertices of $\mathcal{M}'$ each, and then define a coloring of all level-2 squares similarly, while suitably updating the set $\mathcal{M}'$ of the demand pairs. We continue this process for ρ iterations, eventually obtaining a good ensemble $(\mathfrak{I}, f, \mathcal{M}')$. Since we only discard a constant fraction of the demand pairs of $\mathcal{M}'$ in every iteration, at the end, $|\mathcal{M}'| \geq |\mathcal{M}^{**}|/2^{\rho} = |\mathcal{M}^{**}|/2^{O(\sqrt{\log n})} \geq |\mathcal{M}^*|/2^{O(\sqrt{\log n} \cdot \log\log n)}$.

2.6 Finding the Good Ensemble

In our final step, our goal is to find a good ensemble $(\mathfrak{I}, f, \mathcal{M}')$ maximizing $|\mathcal{M}'|$. We show an efficient randomized $2^{O(\sqrt{\log n} \cdot \log\log n)}$-approximation algorithm for this problem. First, we show that, at the cost of losing a small factor in the approximation ratio, we can restrict our attention to a small collection $\mathfrak{I}_1, \mathfrak{I}_2, \ldots, \mathfrak{I}_z$ of hierarchical partitions of R^* into intervals,

and that it is enough to obtain a $2^{O(\sqrt{\log n}\cdot\log\log n)}$-approximate solution for the problem of finding the largest ensemble $(\mathfrak{I}_j, f, \mathcal{M}')$ for each such partition $\mathfrak{I}_j$ separately.

We then fix one such hierarchical partition $\mathfrak{I}_j$, and design an LP-relaxation for the problem of computing a coloring f of $\tilde{\mathcal{H}}$ and a collection $\mathcal{M}'$ of demand pairs, such that $(\mathfrak{I}_j, f, \mathcal{M}')$ is a good ensemble, while maximizing $|\mathcal{M}'|$. Finally, we design an efficient randomized LP-rounding $2^{O(\sqrt{\log n}\cdot\log\log n)}$-approximation algorithm for the problem.

2.7 Completing the Proof of Theorem 1

So far we have assumed that all source vertices lie on the top boundary of the grid, and all destination vertices are at a distance at least $\Omega(\mathsf{OPT})$ from the grid boundary. Let $\mathcal{A}$ be the randomized efficient $2^{O(\sqrt{\log n}\cdot\log\log n)}$-approximation algorithm for this special case. We now extend it to the general Restricted NDP-Grid problem. For every destination vertex t, we identify the closest vertex $\tilde{t}$ that lies on the grid boundary. Using standard grouping techniques, at the cost of losing an additional $O(\log n)$ factor in the approximation ratio, we can assume that all source vertices lie on the top boundary of the grid, all vertices in $\left\{\tilde{t} \mid t \in T(\mathcal{M})\right\}$ lie on a single boundary edge of the grid (assume for simplicity that it is the bottom boundary), and that there is some integer d, such that for every destination vertex $t \in T(\mathcal{M})$, $d \leq d(t,\tilde{t}) < 2d$. We show that we can define a collection $\mathcal{Z} = \{Z_1, \ldots, Z_r\}$ of disjoint square sub-grids of G, and a collection $\mathcal{I} = \{I_1, \ldots, I_r\}$ of disjoint sub-intervals of R^*, such that the bottom boundary of each sub-grid Z_i is contained in the bottom boundary of G, the top boundary of Z_i is within distance at least OPT from R^*, $Z_1, \ldots, Z_r$ appear in this left-to-right order in G, and $I_1, \ldots, I_r$ appear in this left-to-right order on R^*. For each $1 \leq j \leq r$, we let $\mathcal{M}_j$ denote the set of all demand pairs with the sources lying on I_j and the destinations lying in Z_j. For each $1 \leq j \leq r$, we then obtain a new instance $(G, \mathcal{M}_j)$ of the NDP problem. We show that there exist a collection $\mathcal{Z}$ of squares and a collection $\mathcal{I}$ of intervals, such that the value of the optimal solution to each instance $(G, \mathcal{M}_j)$, that we denote by OPT_j, is at most d, while $\sum_{j=1}^{r} \mathsf{OPT}_j \geq \mathsf{OPT}/2^{O(\sqrt{\log n}\cdot\log\log n)}$. Moreover, it is not hard to show that, if we can compute, for each $1 \leq j \leq r$, a routing of some subset $\mathcal{M}'_j \subseteq \mathcal{M}_j$ of demand pairs in G, then we can also route all demand pairs in $\bigcup_{j=1}^{r} \mathcal{M}'_j$ simultaneously in G.

There are two problems with this approach. First, we do not know the set $\mathcal{Z}$ of sub-grids of G and the set $\mathcal{I}$ of intervals of R^*. Second, it is not clear how to solve each resulting problem $(G, \mathcal{M}_j)$. To address the latter problem, we define a simple mapping of all source vertices in $S(\mathcal{M}_j)$ to the top boundary of grid Z_j, obtaining an instance of Restricted NDP-Grid, where all source vertices lie on the top boundary of the grid Z_j, and all destination vertices lie at a distance at least $\mathsf{OPT}_j \leq d$ from its boundary. We can then use algorithm $\mathcal{A}$ in order to solve this problem efficiently. It is easy to see that, if we can route some subset $\mathcal{M}'_j$ of the demand pairs via node-disjoint paths in Z_j, then we can extend this routing to the corresponding set of original demand pairs, whose sources lie on R^*.

Finally, we employ dynamic programming in order to find the set $\mathcal{Z}$ of sub-grids of G and the set $\mathcal{I}$ of intervals of I. For each such potential sub-grid Z and interval I, we use algorithm $\mathcal{A}$ in order to find a routing of a large set of demand pairs of the corresponding instance defined inside Z, and then exploit the resulting solution values for each such pair (I, Z) in a simple dynamic program, that allows us to compute the set $\mathcal{Z}$ of sub-grids of G, the set $\mathcal{I}$ of intervals of I, and the final routing.

3 Approximation Algorithm for the Special Case with Sources Close to the Grid Boundary

In this section we provide a sketch of the proof of Theorem 2. We assume that we are given an instance $(G, \mathcal{M})$ of NDP-Grid and an integer $\delta > 0$, such that every source vertex is at a distance at most δ from the grid boundary. Our goal is to design an efficient randomized factor-$(\delta \cdot 2^{O(\sqrt{\log n} \cdot \log\log n)})$-approximation algorithm for this special case. For every terminal $v \in S(\mathcal{M}) \cup T(\mathcal{M})$, let $\tilde{v}$ be the vertex lying closest to v on the boundary of the grid G. Using standard grouping techniques, at the cost of losing an $O(\log n)$-factor in the approximation ratio, we can assume that there is some integer d, such that for all $t \in T(\mathcal{M})$, $d \leq d(t, \tilde{t}) < 2d$.

Assume first that $d \leq \delta \cdot 2^{O(\sqrt{\log n} \log\log n)}$. Let $\hat{\mathcal{M}} = \left\{(\tilde{s}, \tilde{t}) \mid (s,t) \in \mathcal{M}\right\}$ be a new set of demand pairs, so that all vertices participating in these demand pairs lie on the boundary of G. We can efficiently find an optimal solution to the NDP problem instance $(G, \hat{\mathcal{M}})$ using standard dynamic programming. We then show that $\mathsf{OPT}(G, \hat{\mathcal{M}}) = \Omega(\mathsf{OPT}(G, \mathcal{M})/(\delta \cdot 2^{O(\sqrt{\log n} \cdot \log\log n)}))$, obtaining an $(\delta \cdot 2^{O(\sqrt{\log n} \cdot \log\log n)})$-approximation algorithm.

From now on we assume that $d > \delta \cdot 2^{\Omega(\sqrt{\log n} \log\log n)}$. Next, we define a new set $\tilde{\mathcal{M}}$ of demand pairs: $\tilde{\mathcal{M}} = \{(\tilde{s}, t) \mid (s,t) \in \mathcal{M}\}$, so all source vertices of the demand pairs in $\tilde{\mathcal{M}}$ lie on the boundary of G, obtaining an instance of Restricted NDP-Grid. Let OPT' be the value of the optimal solution to problem $(G, \tilde{\mathcal{M}})$. We show that $\mathsf{OPT}' \geq \Omega(\mathsf{OPT}(G, \mathcal{M})/\delta)$.

We then focus on instance $(G, \tilde{\mathcal{M}})$ of Restricted NDP-Grid. We say that a path P routing a demand pair $(\tilde{s}, t) \in \tilde{\mathcal{M}}$ is *canonical* iff it contains the original source s. The crux of the proof is to show that we can modify the routing produced by the $2^{O(\sqrt{\log n} \cdot \log\log n)}$-approximation algorithm to instance $(G, \tilde{\mathcal{M}})$, so that in the resulting routing all paths are canonical. In order to do so, we utilize the fact that the destination vertices lie much further from the grid boundaries than the source vertices. This creates sufficient margins around the grid boundaries that allow us to modify the routing to turn it a canonical one.

References

1. Alok Aggarwal, Jon Kleinberg, and David P. Williamson. Node-disjoint paths on the mesh and a new trade-off in VLSI layout. *SIAM J. Comput.*, 29(4):1321–1333, 2000. doi:10.1137/S0097539796312733.
2. Matthew Andrews. Approximation algorithms for the edge-disjoint paths problem via Raecke decompositions. In *Proceedings of IEEE FOCS*, pages 277–286, 2010. doi:10.1109/FOCS.2010.33.
3. Matthew Andrews, Julia Chuzhoy, Venkatesan Guruswami, Sanjeev Khanna, Kunal Talwar, and Lisa Zhang. Inapproximability of edge-disjoint paths and low congestion routing on undirected graphs. *Combinatorica*, 30(5):485–520, 2010. doi:10.1007/s00493-010-2455-9.
4. Matthew Andrews and Lisa Zhang. Logarithmic hardness of the undirected edge-disjoint paths problem. *J. ACM*, 53(5):745–761, sep 2006. doi:10.1145/1183907.1183910.
5. Yonatan Aumann and Yuval Rabani. Improved bounds for all optical routing. In *Proceedings of the sixth annual ACM-SIAM symposium on Discrete algorithms*, SODA '95, pages 567–576, Philadelphia, PA, USA, 1995. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=313651.313820.
6. Chandra Chekuri and Julia Chuzhoy. Half-integral all-or-nothing flow. Unpublished Manuscript.
7. Chandra Chekuri and Alina Ene. Poly-logarithmic approximation for maximum node disjoint paths with constant congestion. In *Proc. of ACM-SIAM SODA*, 2013.

8 Chandra Chekuri, Sanjeev Khanna, and F Bruce Shepherd. Edge-disjoint paths in planar graphs. In *Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on*, pages 71–80. IEEE, 2004.

9 Chandra Chekuri, Sanjeev Khanna, and F. Bruce Shepherd. Multicommodity flow, well-linked terminals, and routing problems. In *Proc. of ACM STOC*, pages 183–192, 2005.

10 Chandra Chekuri, Sanjeev Khanna, and F. Bruce Shepherd. An $O(\sqrt{n})$ approximation and integrality gap for disjoint paths and unsplittable flow. *Theory of Computing*, 2(1):137–146, 2006. doi:10.4086/toc.2006.v002a007.

11 Julia Chuzhoy. Routing in undirected graphs with constant congestion. *SIAM J. Comput.*, 45(4):1490–1532, 2016. doi:10.1137/130910464.

12 Julia Chuzhoy and David H. K. Kim. On approximating node-disjoint paths in grids. In Naveen Garg, Klaus Jansen, Anup Rao, and José D. P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2015, August 24-26, 2015, Princeton, NJ, USA*, volume 40 of *LIPIcs*, pages 187–211. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015. URL: http://www.dagstuhl.de/dagpub/978-3-939897-89-7, doi:10.4230/LIPIcs.APPROX-RANDOM.2015.187.

13 Julia Chuzhoy, David H. K. Kim, and Shi Li. Improved approximation for node-disjoint paths in planar graphs. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2016, pages 556–569, New York, NY, USA, 2016. ACM. doi:10.1145/2897518.2897538.

14 Julia Chuzhoy, David H. K. Kim, and Rachit Nimavat. Almost polynomial hardness of node-disjoint paths in grids. Unpublished Manuscript, 2017.

15 Julia Chuzhoy, David H. K. Kim, and Rachit Nimavat. New hardness results for routing on disjoint paths. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 86–99. ACM, 2017. doi:10.1145/3055399.3055411.

16 Julia Chuzhoy and Shi Li. A polylogarithmic approximation algorithm for edge-disjoint paths with congestion 2. *J. ACM*, 63(5):45:1–45:51, 2016. URL: http://dl.acm.org/citation.cfm?id=2893472, doi:10.1145/2893472.

17 M. Cutler and Y. Shiloach. Permutation layout. *Networks*, 8:253–278, 1978. doi:10.1002/net.3230080308.

18 Shimon Even, Alon Itai, and Adi Shamir. On the complexity of timetable and multicommodity flow problems. *SIAM J. Comput.*, 5(4):691–703, 1976. doi:10.1137/0205048.

19 Krzysztof Fleszar, Matthias Mnich, and Joachim Spoerhase. New Algorithms for Maximum Disjoint Paths Based on Tree-Likeness. In Piotr Sankowski and Christos Zaroliagis, editors, *24th Annual European Symposium on Algorithms (ESA 2016)*, volume 57 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 42:1–42:17, Dagstuhl, Germany, 2016. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.ESA.2016.42.

20 R. Karp. On the complexity of combinatorial problems. *Networks*, 5:45–68, 1975.

21 Ken-Ichi Kawarabayashi and Yusuke Kobayashi. An O(log n)-approximation algorithm for the edge-disjoint paths problem in Eulerian planar graphs. *ACM Trans. Algorithms*, 9(2):16:1–16:13, 2013. doi:10.1145/2438645.2438648.

22 Jon Kleinberg. An approximation algorithm for the disjoint paths problem in even-degree planar graphs. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '05, pages 627–636, Washington, DC, USA, 2005. IEEE Computer Society. doi:10.1109/SFCS.2005.18.

23 Jon M. Kleinberg and Éva Tardos. Disjoint paths in densely embedded graphs. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science*, pages 52–61, 1995.

24 Jon M. Kleinberg and Éva Tardos. Approximations for the disjoint paths problem in high-diameter planar networks. *J. Comput. Syst. Sci.*, 57(1):61–73, 1998. doi:10.1006/jcss.1998.1579.

25 Stavros G. Kolliopoulos and Clifford Stein. Approximating disjoint-path problems using packing integer programs. *Mathematical Programming*, 99:63–87, 2004. doi:10.1007/s10107-002-0370-6.

26 MR Kramer and Jan van Leeuwen. The complexity of wire-routing and finding minimum area layouts for arbitrary vlsi circuits. *Advances in computing research*, 2:129–146, 1984.

27 James F. Lynch. The equivalence of theorem proving and the interconnection problem. *SIGDA Newsl.*, 5(3):31–36, 1975. doi:10.1145/1061425.1061430.

28 Harald Räcke. Minimizing congestion in general networks. In *Proc. of IEEE FOCS*, pages 43–52, 2002.

29 Prabhakar Raghavan and Clark D. Tompson. Randomized rounding: a technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7:365–374, December 1987. doi:10.1007/BF02579324.

30 Satish Rao and Shuheng Zhou. Edge disjoint paths in moderately connected graphs. *SIAM J. Comput.*, 39(5):1856–1887, 2010. doi:10.1137/080715093.

31 N. Robertson and P. D. Seymour. Outline of a disjoint paths algorithm. In *Paths, Flows and VLSI-Layout*. Springer-Verlag, 1990.

32 Neil Robertson and Paul D. Seymour. Graph minors. VII. disjoint paths on a surface. *J. Comb. Theory, Ser. B*, 45(2):212–254, 1988. doi:10.1016/0095-8956(88)90070-6.

33 Neil Robertson and Paul D Seymour. Graph minors. XIII. the disjoint paths problem. *Journal of Combinatorial Theory, Series B*, 63(1):65–110, 1995.

34 Loïc Seguin-Charbonneau and F. Bruce Shepherd. Maximum edge-disjoint paths in planar graphs with congestion 2. In *Proceedings of the 2011 IEEE 52Nd Annual Symposium on Foundations of Computer Science*, FOCS '11, pages 200–209, Washington, DC, USA, 2011. IEEE Computer Society. doi:10.1109/FOCS.2011.30.