New Hardness Results for Routing on Disjoint Paths

Extended Abstract[*]

Julia Chuzhoy[†]
Toyota Technological Institute at Chicago
6045 S. Kenwood Ave
Chicago, IL 60637, U.S.A
cjulia@ttic.edu

David H. K. Kim[‡]
Computer Science Department
University of Chicago
1100 E. 58th Street
Chicago, IL 60637, U.S.A
hongk@cs.uchicago.edu

Rachit Nimavat[§]
Toyota Technological Institute at Chicago
6045 S. Kenwood Ave
Chicago, IL 60637, U.S.A
nimavat@ttic.edu

ABSTRACT

In the classical Node-Disjoint Paths (NDP) problem, 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. The goal is to route the largest possible number of the demand pairs via node-disjoint paths. The best current approximation for the problem is achieved by a simple greedy algorithm, whose approximation factor is $O(\sqrt{n})$, while the best current negative result is an $\Omega(\log^{1/2-\delta} n)$-hardness of approximation for any constant δ, under standard complexity assumptions. Even seemingly simple special cases of the problem are still poorly understood: when the input graph is a grid, the best current algorithm achieves an $\tilde{O}(n^{1/4})$-approximation, and when it is a general planar graph, the best current approximation ratio of an efficient algorithm is $\tilde{O}(n^{9/19})$. The best currently known lower bound on the approximability of both these versions of the problem is APX-hardness.

In this paper we prove that NDP is $2^{\Omega(\sqrt{\log n})}$-hard to approximate, unless all problems in NP have algorithms with running time $n^{O(\log n)}$. Our result holds even when the underlying graph is a planar graph with maximum vertex degree 3, and all source vertices lie on the boundary of a single face (but the destination vertices may lie anywhere in the graph). We extend this result to the closely related Edge-Disjoint Paths problem, showing the same hardness of approximation ratio even for sub-cubic planar graphs with all sources lying on the boundary of a single face.

CCS CONCEPTS

•**Theory of computation → Problems, reductions and completeness; Routing and network design problems;** *Network flows;*

KEYWORDS

Approximation algorithms, Hardness of approximation, Routing problems, Node-Disjoint Paths, Edge-Disjoint Paths

ACM Reference format:

Julia Chuzhoy, David H. K. Kim, and Rachit Nimavat. 2017. New Hardness Results for Routing on Disjoint Paths. In *Proceedings of 49th Annual ACM SIGACT Symposium on the Theory of Computing, Montreal, Canada, June 2017 (STOC'17),* 14 pages.

DOI: 10.1145/3055399.3055411

1 INTRODUCTION

The main focus of this paper is the Node-Disjoint Paths (NDP) problem: given an undirected n-vertex graph G, together with a collection $\mathcal{M} = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ of pairs of its vertices, called *source-destination*, or *demand* pairs, route the largest possible number of the demand pairs via node-disjoint paths. In other words, a solution to the problem is a collection $\mathcal{P}$ of node-disjoint paths, with each path connecting a distinct source-destination pair, and the goal is to maximize $|\mathcal{P}|$. The vertices participating in the demand pairs of $\mathcal{M}$ are called *terminals*. NDP is a classical routing problem, that has been extensively studied in both Graph Theory and Theoretical Computer Science communities. One of the key results in Robertson and Seymour's Graph Minors series is an efficient algorithm for the special case of the problem, where the number k of the demand pairs is bounded by a constant [28, 29]; the running time of their algorithm is $f(k) \cdot \text{poly}(n)$ for some large function f. However, when k is a part of input, the problem becomes NP-hard [16, 17], even on planar graphs [24], and even on grid graphs [23]. The following simple greedy algorithm provides an $O(\sqrt{n})$-approximation for NDP [22]: Start with $\mathcal{P} = \emptyset$. While G contains a path connecting any demand pair, select the shortest

[*] A full version of this paper is available at `https://arxiv.org/abs/1611.05429`
[†] Supported in part by NSF grants CCF-1318242 and CCF-1616584.
[‡] Supported in part by NSF grant CCF-1318242.
[§] Supported in part by NSF grant CCF-1318242.

STOC'17, Montreal, Canada
© 2017 ACM. 978-1-4503-4528-6/17/06…$15.00
DOI: 10.1145/3055399.3055411

such path P, add it to $\mathcal{P}$, and delete all vertices of P from G. Surprisingly, despite the extensive amount of work on the problem and its variations, this elementary algorithm remains the best currently known approximation algorithm for the problem, and until recently, this was true even for the special cases where G is a planar graph, or a grid graph. The latter two special cases have slightly better algorithms now: a recent result of Chuzhoy and Kim [13] gives a $\tilde{O}(n^{1/4})$-approximation for NDP on grid graphs, and Chuzhoy, Kim and Li [14] provide a $\tilde{O}(n^{9/19})$-approximation algorithm for the problem on planar graphs. The best current negative result shows that NDP has no $O(\log^{1/2-\delta} n)$-approximation algorithms for any constant δ, unless $\mathsf{NP} \subseteq \mathsf{ZPTIME}(n^{\operatorname{poly}\log n})$ [2, 3]. For the special case of grids and planar graphs only APX-hardness is currently known on the negative side [13].

The main result of this paper is that NDP is $2^{\Omega(\sqrt{\log n})}$-hard to approximate unless $\mathsf{NP} \subseteq \mathsf{DTIME}(n^{O(\log n)})$, even if the underlying graph is a planar graph with maximum vertex degree 3, and all source vertices $\{s_1, \ldots, s_k\}$ lie on the boundary of a single face. We note that NDP can be solved efficiently on graphs whose maximum vertex degree is 2 (see the full version of the paper for details).

A problem closely related to NDP is Edge-Disjoint Paths (EDP). The input to this problem is the same as to NDP, and the goal is again to route the largest possible number of the demand pairs. However, the routing paths are now allowed to share vertices, as long as they remain disjoint in their edges. The two problems are closely related: it is easy to see that EDP is a special case of NDP, by using the line graph of the EDP instance to obtain an equivalent NDP instance[1]. This relationship is not known for planar graphs, as the line graph of a planar graph is not necessarily planar. The current approximability status of EDP is similar to that of NDP: the best current approximation algorithm achieves an $O(\sqrt{n})$-approximation factor [11], and the best current negative result is an $\Omega(\log^{1/2-\delta} n)$-hardness of approximation for any constant δ, unless $\mathsf{NP} \subseteq \mathsf{ZPTIME}(n^{\operatorname{poly}\log n})$ [2, 3]. An analogue of the special case of NDP on grid graphs for the EDP problem is when the input graph is a wall, and the work of [13] gives an $\tilde{O}(n^{1/4})$-approximation algorithm for EDP on wall graphs. For planar graphs, no better than $O(\sqrt{n})$-approximation is currently known for EDP, and it is not clear whether the algorithm of [14] can be adapted to this setting. Our hardness result extends to EDP on planar sub-cubic graphs, where all source vertices lie on the boundary of a single face.

Interestingly, better algorithms are known for several special cases of EDP on planar graphs. Kleinberg [19], building on the work of Chekuri, Khanna and Shepherd [9, 10], has shown an $O(\log^2 n)$-approximation algorithm for even-degree planar graphs. Aumann and Rabani [6] showed an $O(\log^2 n)$-approximation algorithm for EDP on grids, and Kleinberg and Tardos [20, 21] showed $O(\log n)$-approximation algorithms for broader classes of nearly-Eulerian uniformly high-diameter planar graphs, and nearly-Eulerian densely embedded graphs. Recently, Kawarabayashi and Kobayashi [18] gave an $O(\log n)$-approximation algorithm for EDP on 4-edge-connected planar graphs and on Eulerian planar graphs. It seems that the restriction of the graph G to be planar Eulerian, or nearly-Eulerian, makes the EDP problem significantly more tractable. In contrast, the graphs we construct in our hardness of approximation proof are sub-cubic, and far from being Eulerian.

A variation of the NDP and EDP problems, where small congestion is allowed, has been a subject of extensive study. In the NDP with congestion (NDPwC) problem, the input is the same as in the NDP problem, and we are additionally given an integer $c \geq 1$. The goal is to route as many of the demand pairs as possible with congestion at most c: that is, every vertex may participate in at most c paths in the solution. EDP with Congestion (EDPwC) is defined similarly, except that now the congestion bound is imposed on the graph edges and not vertices. The classical randomized rounding technique of Raghavan and Thompson [26] gives a constant-factor approximation for both problems, if the congestion c is allowed to be as high as $\Theta(\log n / \log\log n)$. A long line of work [1, 7, 8, 10, 12, 15, 25, 27] has lead to an $O(\operatorname{poly}\log k)$-approximation for both NDPwC and EDPwC problems, with congestion bound $c = 2$. For planar graphs, a constant-factor approximation with congestion 2 is known for EDP [30]. Our new hardness results demonstrate that there is a dramatic difference in the approximability of routing problems with congestion 1 and 2.

Our Results and Techniques. Our main result is the proof of the following two theorems.

THEOREM 1.1. *There is a constant c, such that no efficient algorithm achieves a factor* $2^{c\sqrt{\log n}}$*-approximation for* NDP*, unless* $\mathsf{NP} \subseteq \mathsf{DTIME}(n^{O(\log n)})$*. This result holds even for planar graphs with maximum vertex degree* 3*, where all source vertices lie on the boundary of a single face.*

THEOREM 1.2. *There is a constant c, such that no efficient algorithm achieves a factor* $2^{c\sqrt{\log n}}$*-approximation for* EDP*, unless* $\mathsf{NP} \subseteq \mathsf{DTIME}(n^{O(\log n)})$*. This result holds even for planar graphs with maximum vertex degree* 3*, where all source vertices lie on the boundary of a single face.*

We now provide an informal high-level overview of the proof of Theorem 1.1. It is somewhat easier to describe the proof of the theorem for the case where the maximum vertex degree is allowed to be 4 instead of 3. This proof can then be easily modified to ensure that the maximum vertex degree in the instances we obtain does not exceed 3, and also extended to the EDP problem. We perform a reduction from the 3SAT(5) problem. In this problem, we are given a SAT formula φ defined over a set of n Boolean variables. The formula consists of m clauses, each of which is an OR of three literals, where every literal is either a variable or its negation. Every variable of φ participates in exactly 5 distinct clauses, and the literals of every clause correspond to three distinct variables. We say that φ is a YES-INSTANCE, if there is an assignment to its variables that satisfies all clauses, and we say that it is a NO-INSTANCE, if no assignment satisfies more than a $(1-\epsilon)$-fraction of the clauses, for some fixed constant $0 < \epsilon < \frac{1}{2}$. The famous PCP theorem [4, 5] shows that, unless P = NP, no efficient algorithm can distinguish between the YES- and the NO-INSTANCES of 3SAT(5).

[1] This transformation may inflate the number of vertices, and so approximation factors depending on n may not be preserved.

We perform $\Theta(\log n)$ iterations, where in iteration i we construct what we call a *level-i* instance of NDP. We use two parameters, N_i and N'_i, and ensure that, if the reduction is performed from a formula φ which is a YES-INSTANCE, then there is a solution to the level-i instance of NDP that routes N_i demand pairs, while if φ is a NO-INSTANCE, then no solution routes more than N'_i demand pairs. We let $g_i = N_i/N'_i$ be the gap achieved by the level-i instance. Our construction ensures that the gap grows by a small constant factor in every iteration, so $g_i = 2^{\Theta(i)}$, while the instance size grows by roughly factor-$\Theta(n \cdot g_{i-1})$ in iteration i. Therefore, after $\Theta(\log n)$ iterations, the gap becomes $2^{\Omega(\log n)}$, while the instance size becomes $n' = 2^{O(\log^2 n)}$, giving us the desired $2^{\Omega(\sqrt{\log n'})}$-hardness of approximation, unless $\mathsf{NP} \subseteq \mathsf{DTIME}(n^{O(\log n)})$.

In all our instances of NDP, the underlying graph is a subgraph of a grid, with all sources lying on the top boundary of the grid; all vertices participating in the demand pairs are distinct. In the first iteration, a level-1 instance is constructed by a simple reduction from 3SAT(5), achieving a small constant gap g_1. Intuitively, once we construct a level-i instance, in order to construct a level-$(i+1)$ instance, we replace every demand pair from a level-1 instance with a collection of level-i instances. In order to be able to do so, we need the instances to be "flexible", so that, for example, we have some freedom in choosing the locations of the source and the destination vertices of a given level-i instance in the grid.

We achieve this flexibility by defining, for each level i, a family of level-i instances. The graph associated with a level-i instance $\mathcal{I}$ is a subgraph of a large enough grid G_i. The construction of the instance consists of two parts. First, we construct a path $Z(\mathcal{I})$, and place all source vertices on this path. Second, we construct a vertex-induced subgraph $B(\mathcal{I})$ of a relatively small grid G'_i, and we call $B(\mathcal{I})$ a *box*. We place all the destination vertices inside the box $B(\mathcal{I})$. Graphs $Z(\mathcal{I})$ and G'_i are completely disjoint from the grid G_i and from each other. In order to construct a specific level-i instance, we select a placement of the path $Z(\mathcal{I})$ on the first row of the grid G_i, and a placement of the box $B(\mathcal{I})$ in G_i, far enough from its boundaries (see Figure 1). We also choose a sub-grid G''_i of G_i, of the same dimensions as G'_i, and map the vertices of G'_i to the vertices of G''_i in a natural way. Since $B(\mathcal{I}) \subseteq G'_i$, this also defines a mapping of the vertices of $B(\mathcal{I})$ to the vertices of G_i. Once these placements are selected, the mapping of the vertices of $Z(\mathcal{I})$ to the vertices of P determines the identities of the source vertices, and the mapping of the vertices of $B(\mathcal{I})$ to the vertices of G''_i determines the identities of the destination vertices. We delete from G_i all vertices to which the vertices of $G'_i \setminus B(\mathcal{I})$ are mapped. In other words, all vertices that were removed from G'_i to construct $B(\mathcal{I})$, are also removed from G''_i, and hence from G_i. In order to reduce the maximum vertex degree to 3, we can use wall graphs instead of grid graphs and employ a similar proof. Alternatively, a simple modification of the final instance we obtain can directly reduce its maximum vertex degree to 3.

The most natural intuitive way to think about our construction is the one described above. An equivalent, and somewhat easier way to define our construction is slightly different: we let a level-0 instance be an instance consisting of a single demand pair (s,t), with s lying on the first row of the grid and t lying far from the grid boundary. We then show, for each $i > 0$, a procedure that constructs a level-i instance by combining a number of level-$(i-1)$ instances. The latter definition is somewhat more convenient, because it saves us the need to provide a separate correctness proof for level-1 instances, which is essentially identical to the proof for higher-level instances. However, we still feel that defining level-1 instances explicitly is useful for the sake of intuition. Therefore, we start with preliminaries in Section 2 and describe our construction of level-1 instances in Section 3, together with an intuition for constructing higher-level instances. In Section 4, we define our final construction in two steps: by first defining level-0 instances, and then showing how to construct level-i instances from level-$(i-1)$ instances. The resulting level-1 instances will be similar to those defined in Section 3. We provide a sketch of the correctness proof in Sections 4–6; a complete proof can be found in the full version of the paper. This gives a proof of Theorem 1.1 for the case where the maximum vertex degree is allowed to be 4. Section 7 extends our result to EDP in planar graphs, and shows how to reduce the degree of the hard NDP instances to 3, completing the proofs of Theorems 1.1 and 1.2.

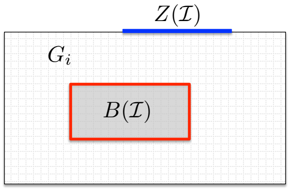

Figure 1: A schematic view of a level-i instance $\mathcal{I}$.

2 PRELIMINARIES

For a pair $\ell, h > 0$ of integers, we let $G^{\ell,h}$ denote a grid of length ℓ and height h. The set of vertices of grid $G^{\ell,h}$ is $V(G^{\ell,h}) = \{v(i,j) \mid 1 \le i \le h, 1 \le j \le \ell\}$. The set of edges of the grid $G^{\ell,h}$ is the union of two subsets: the set of horizontal edges $E^H = \left\{(v_{i,j}, v_{i,j+1}) \mid 1 \le i \le h, 1 \le j < \ell\right\}$ and the set of vertical edges $E^V = \left\{(v_{i,j}, v_{i+1,j}) \mid 1 \le i < h, 1 \le j \le \ell\right\}$. The subgraph of $G^{\ell,h}$ induced by the edges of E^H consists of h paths, that we call the *rows* of the grid; for $1 \le i \le h$, the ith row R_i is the row containing the vertex $v(i,1)$. Similarly, the subgraph induced by the edges of E^V consists of ℓ paths that we call the *columns* of the grid, and for $1 \le j \le \ell$, the jth column W_j is the column containing $v(1,j)$. We think of the rows as ordered from top to bottom and the columns as ordered from left to right. Row $R_{\lceil h/2 \rceil}$ is called the *middle row* of the grid $G^{\ell,h}$. Given two vertices $u = v(i,j), u' = v(i',j')$ of the grid, the distance between them is $d(u,u') = |i - i'| + |j - j'|$.

Given a set $\mathcal{R}$ of consecutive rows of a grid $G = G^{\ell,h}$ and a set $\mathcal{W}$ of consecutive columns of G, we let $\Upsilon(\mathcal{R},\mathcal{W})$ be the subgraph of G induced by the set $\left\{v(j,j') \mid R_j \in \mathcal{R}, W_{j'} \in \mathcal{W}\right\}$ of vertices. We say that $\Upsilon = \Upsilon(\mathcal{R},\mathcal{W})$ is the *sub-grid of G spanned by the set $\mathcal{R}$ of rows and the set $\mathcal{W}$ of columns.*

Assume now that we are given a grid G, a sequence $\mathcal{S} = (G_1, \ldots, G_r)$ of disjoint sub-grids of G, and an integer N. We say that the grids of $\mathcal{S}$ are *aligned and N-separated* iff the middle row of each grid G_i is a sub-path of the middle row of G; the grids in $\{G_1, \ldots, G_r\}$ appear in this left-to-right order inside G; every pair of consecutive grids G_i is separated by at least N columns of G; and every grid in $\mathcal{S}$ is separated by at least N columns from the right and the left boundaries of G.

Throughout our construction, we use the notion of a box. A box B of length ℓ and height h is a vertex-induced subgraph of $G^{\ell,h}$. We denote $U(B) = V(G^{\ell,h}) \setminus V(B)$, and we sometimes think of $U(B)$ as the "set of vertices deleted from B". We say that B is a *cut-out box* iff $U(B)$ contains all vertices lying on the left, right, and bottom boundaries of $G^{\ell,h}$; note that $U(B)$ may contain additional vertices of $G^{\ell,h}$. The vertices of the top boundary of $G^{\ell,h}$ that belong to $V(B)$ are called *the opening of* B. We sometimes say that the vertices of B that belong to row $R_{\lceil h/2 \rceil}$ of $G^{\ell,h}$ lie on the middle row of B.

Given any set $\mathcal{M}$ of demand pairs, we let $S(\mathcal{M})$ denote the set of all source vertices participating in $\mathcal{M}$ and $T(\mathcal{M})$ the set of all destination vertices. Given a path P, the length of the path is the number of vertices on it.

As already described in the introduction, for every level $0 \leq i \leq \Theta(\log n)$, we construct a level-$i$ instance $\mathcal{I}$. In fact, it is a family of instances, but it is more convenient to think of it as one instance with different instantiations. A definition of a level-i instance $\mathcal{I}$ consists of the following ingredients: (i) integral parameters L_i, L'_i and an even integer H_i; (ii) a path $Z(\mathcal{I})$ of length L_i; (iii) a grid G'_i of length L'_i and height H_i, together with a cut-out box $B(\mathcal{I}) \subseteq G'_i$; and (iv) a set $\mathcal{M}$ of demand pairs, together with a mapping of the vertices of $S(\mathcal{M})$ to distinct vertices of $Z(\mathcal{I})$ and a mapping of the vertices of $T(\mathcal{M})$ to distinct vertices on the middle row of $B(\mathcal{I})$.

In order to instantiate a level-i instance $\mathcal{I}$, we select a grid G_i of length at least $2L_i + 2L'_i + 4H_i$ and height at least $3H_i$, a sub-path P of the first row of G_i of length L_i, and a sub-grid G''_i of G_i of height H_i and length L'_i, so that the distance from the vertices of G''_i to the vertices lying on the boundary of G_i is at least H_i. We map every vertex of $Z(\mathcal{I})$ to the corresponding vertex of P in a natural way, and this determines the identities of the source vertices in the instance we construct. We also map every vertex of G'_i to the corresponding vertex of G''_i, and this determines the identities of the destination vertices. Finally, for every vertex $u \in U(B(\mathcal{I}))$, we delete the vertex of G''_i to which u is mapped from G_i. This defines an instance of NDP on a subgraph of G_i, where all the sources lie on the top boundary of G_i and all source and destination vertices are distinct. We note that box $B(\mathcal{I})$ may be constructed recursively, by placing several boxes $B(\mathcal{I}')$ representing lower-level instances $\mathcal{I}'$ inside it. The mapping of the vertices of $B(\mathcal{I}')$ to the vertices of $B(\mathcal{I})$, the placement of the destination vertices, and the removal of the vertices of $B(\mathcal{I})$ corresponding to the grid vertices removed from $B(\mathcal{I}')$ is done similarly.

Assume now that we are given an instantiation of a level-i instance $\mathcal{I}$ and a set $\mathcal{P}$ of node-disjoint paths routing a subset $\mathcal{M}'$ of the demand pairs in that instance. Assume for convenience that $\mathcal{M}' = \{(s_1, t_1), \ldots, (s_r, t_r)\}$, that the vertices $s_1, \ldots, s_r$ appear in this left-to-right order on $Z(\mathcal{I})$, and that $\mathcal{P} = \{P_1, \ldots, P_r\}$, where path P_j connects s_j to t_j. Let A be the set of all vertices of the top row of the grid G''_i that were not deleted (that is, these are the vertices lying on the opening of $B(\mathcal{I})$). We say that the set $\mathcal{P}$ of paths *respects* the box $B(\mathcal{I})$ iff for all $1 \leq j \leq r$, $P_j \cap A$ is a single vertex, that we denote by u_j, and u_j is the jth vertex of A from the left. Intuitively, the paths in $\mathcal{P}$ connect the sources to a set of consecutive vertices on the opening of $B(\mathcal{I})$ in a straightforward manner, and the actual routing occurs inside the box $B(\mathcal{I})$.

We perform a reduction from the 3SAT(5) problem. In this problem, we are given a SAT formula φ on a set $\{x_1, \ldots, x_n\}$ of n Boolean variables and a set $C = \{C_1, \ldots, C_m\}$ of $m = 5n/3$ clauses. Each clause contains 3 literals, each of which is either a variable or its negation. The literals of each clause correspond to 3 distinct variables, and each variable participates in exactly 5 clauses. We denote the literals of the clause C_q by ℓ_{q_1}, ℓ_{q_2} and ℓ_{q_3}. A clause is satisfied by an assignment to the variables iff at least one of its literals evaluates to True. We say that φ is a Yes-Instance if there is an assignment to its variables satisfying all its clauses. We say that it is a No-Instance with respect to some parameter ϵ, if no assignment satisfies more than $(1-\epsilon)m$ clauses. The following well-known theorem follows from the PCP theorem [4, 5].

Theorem 2.1. *There is a constant* $0 < \epsilon < \frac{1}{2}$, *such that no efficient algorithm can distinguish between the* Yes- *and the* No-Instances *(defined with respect to* ϵ*) of the 3SAT(5) problem, unless* P = NP.

Given an input formula φ, we will construct an instance $(G, \mathcal{M})$ of the NDP problem with $|V(G)| = n' = n^{O(\log n)}$, that has the following properties: if φ is a Yes-Instance, then there is a solution to the NDP instance routing N demand pairs, for some parameter N; if φ is a No-Instance, then at most N/g demand pairs can be routed, where $g = 2^{\Omega(\log n)} = 2^{\Omega(\sqrt{\log n'})}$. This will prove that no efficient algorithm can achieve a better than factor $2^{O(\sqrt{\log n})}$-approximation for NDP, unless NP $\subseteq$ DTIME$(n^{O(\log n)})$. The instance we construct is a subgraph of a grid with all source vertices lying on its top boundary, so the hardness result holds for planar graphs with maximum vertex degree 4, with all sources lying on the boundary of a single face. In Section 7, we modify this instance to reduce its maximum vertex degree to 3.

3 THE LEVEL-1 INSTANCE

In this section we define our level-1 instance $\mathcal{I}$ and provide intuition for generalizing it to higher-level instances. Since Section 4 contains all formal definitions, including those for the level-1 instance, the description here is somewhat informal.

We assume that we are given a 3SAT(5) formula φ defined over a set $\{x_1, \ldots, x_n\}$ of variables and a set $C = \{C_1, \ldots, C_m\}$ of clauses, so $m = 5n/3$. For every variable x_j of φ, we will define a set $\mathcal{M}(x_j)$ of demand pairs representing x_j, and similarly, for every clause $C_q \in C$ we will define a set $\mathcal{M}(C_q)$ of demand pairs representing it. We call the demand pairs in set $\mathcal{M}^V = \bigcup_{j=1}^{n} \mathcal{M}(x_j)$ *variable-pairs* and the demand pairs in set $\mathcal{M}^C = \bigcup_{C_q \in C} \mathcal{M}(C_q)$ *clause-pairs.*

Let $h = 1000/\epsilon$ and $\delta = 8\epsilon^2/10^{12}$, where ϵ is the parameter from Theorem 2.1. We set $N_1 = (200h/3 + 1)n$ and $N'_1 = (1 - \delta)N_1$. Our construction ensures that, if the input formula φ is a Yes-Instance, then for every instantiation of $\mathcal{I}$, there is a set $\mathcal{P}$ of node-disjoint paths that respects the box $B(\mathcal{I})$ and routes N_1 demand pairs. On the other hand, if φ is a No-Instance, then no solution can route more than N'_1 demand pairs in any instantiation of $\mathcal{I}$. This gives a gap of $1/(1 - \delta)$ between the Yes- and No-Instance solution costs. In the following levels we gradually amplify this gap.

We set $L_1 = (80h + 2)n$, $L'_1 = 20N_1^3$ and $H_1 = 20N_1$. In order to construct a level-1 instance $\mathcal{I}$, we start with a path $Z(\mathcal{I})$ of length L_1 and a grid G'_1 of length L'_1 and height H_1. We delete all vertices lying on the bottom, left and right boundaries of G'_1 to obtain the initial cut-out box $B(\mathcal{I})$; we will later delete some additional vertices from $B(\mathcal{I})$.

We define two sub-grids of G'_1: grid B^V, that will contain all vertices of $T(\mathcal{M}^V)$ (the destination vertices of the demand pairs in $\mathcal{M}^V$), and grid B^C, that will contain all vertices of $T(\mathcal{M}^C)$. Both grids have sufficiently large length and height: length $9N_1^3$ and height $16N_1$ for each grid are sufficient. We place both grids inside G'_1, so that the middle row of each grid is contained in the middle row of G'_1, there is a horizontal spacing of at least $2N_1$ between the two grids, and both grids are disjoint from the left and the right boundaries of G'_1. It is easy to see that at least $2N_1$ rows of G'_1 lie above and below both grids (see Figure 2).

Next, we define smaller sub-grids of the grids B^V and B^C. For every variable x_j of φ, we select a sub-grid $B(x_j)$ of B^V of height $H^V = 4N_1 + 3$ and length $L^V = 4N_1 + (70h + 2)$. This is done so that the sequence $B(x_1), \ldots, B(x_n)$ of grids is aligned and $(2N_1)$-separated in B^V (see Figure 2). Recall that this means that the middle row of each grid coincides with the middle row of B^V, and the horizontal distance between every pair of these grids, and between each grid and the right and the left boundaries of B^V is at least $2N_1$. It is easy to verify that grid B^V is large enough to allow this. For each variable x_j of φ, the vertices of $T(\mathcal{M}(x_j))$ will lie in $B(x_j)$. Note that there are at least $2N_1$ rows of B^V above and below each box $B(x_1), \ldots, B(x_n)$.

Similarly, for every clause $C_q \in \mathcal{C}$, we define a sub-grid $B(C_q)$ of B^C of length $L^C = 3h$ and height $H^C = 3$. We select the sub-grids $B(C_1), \ldots, B(C_q)$ of B^C so that they are aligned and $(4N_1)$-separated. As before, box B^C is sufficiently large to allow this, and there are at least $2N_1$ rows of B^C both above and below each such grid $B(C_q)$ (see Figure 2).

Recall that the length of the path $Z(\mathcal{I})$ is $L_1 = (80h + 2)n$. We partition $Z(\mathcal{I})$ into n disjoint sub-paths $I(x_1), I(x_2), \ldots, I(x_n)$ of length $80h+2$ each, that we refer to as intervals. For each $1 \le j \le n$, vertices of $S(\mathcal{M}(x_j))$ will lie on $I(x_j)$. Additionally, for every clause C_q in which variable x_j participates, path $I(x_j)$ will contain some vertices of $S(\mathcal{M}(C_q))$. The remainder of the construction consists of two parts — variable gadgets and clause gadgets, that we define next, starting with the variable gadgets.

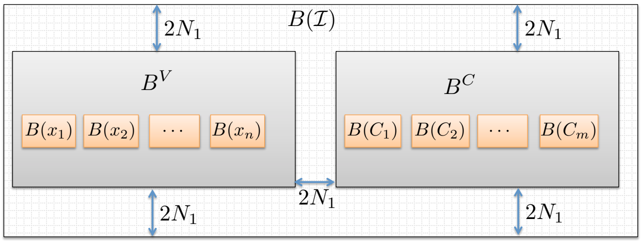

Figure 2: High-level view of the level-1 construction.

Variable gadgets. Consider some variable x of the formula φ and its corresponding interval $I(x)$ of $Z(\mathcal{I})$. We partition $I(x)$ as follows. Let $I^T(x), I^F(x) \subseteq I(x)$ denote the sub-intervals of $I(x)$ containing the first and the last $(10h + 1)$ consecutive vertices of $I(x)$, respectively, and let $I^X(x)$ be the interval containing the remaining $60h$ vertices. Consider the box $B(x)$, and recall that it has length $4N_1 + 70h + 2$ and height $4N_1 + 3$. Let $B'(x) \subseteq B(x)$ be a sub-grid of $B(x)$ of length $70h + 2$ and height 3, so that there are exactly $2N_1$ rows of $B(x)$ above and below $B'(x)$, and $2N_1$ columns of $B(x)$ to the left and to the right of $B'(x)$. Notice that the middle row of $B'(x)$, that we denote by $R'(x)$, is aligned with the middle row of $B(x)$ and hence of $B(\mathcal{I})$. We delete all vertices of $B'(x)$ that lie on its bottom row, and we will place all destination vertices of the demand pairs in $\mathcal{M}(x)$ on $R'(x)$. In order to do so, we further partition $R'(x)$ into three intervals: interval $\hat{I}^T(x)$ contains the first $5h + 1$ vertices of $R'(x)$; interval $\hat{I}^F(x)$ contains the following $5h + 1$ vertices of $R'(x)$; and interval $\hat{I}^X(x)$ contains the remaining $60h$ vertices of $R'(x)$ (see Figure 3). The set $\mathcal{M}(x)$ of demand pairs consists of three subsets:

(Extra Pairs). Let $\mathcal{M}^X(x) = \left\{(s_y^X(x), t_y^X(x)) \mid 1 \le y \le 60h\right\}$ be a set of $60h$ demand pairs that we call the Extra pairs for x. The vertices $s_1^X(x), \ldots, s_{60h}^X(x)$ appear on $I^X(x)$ in this order, and the vertices $t_1^X(x), \ldots, t_{60h}^X(x)$ appear on $\hat{I}^X(x)$ in this order.

(True Pairs). We denote the vertices appearing on $I^T(x)$ by $a_1^T, b_1^T, a_2^T, b_2^T, \ldots, b_{5h}^T, a_{5h+1}^T$ in this left-to-right order. Let $\mathcal{M}^T(x) = \left\{(s_y^T(x), t_y^T(x)) \mid 1 \le y \le 5h + 1\right\}$ be a set of $(5h + 1)$ demand pairs that we call the True demand pairs for x. For each $1 \le y \le 5h + 1$, we identify $s_y^T(x)$ with the vertex a_y^T of $I^T(x)$, and we let $t_y^T(x)$ be the yth vertex on $\hat{I}^T(x)$.

(False Pairs). Similarly, we denote the vertices appearing on $I^F(x)$ by $a_1^F, b_1^F, a_2^F, b_2^F, \ldots, b_{5h}^F, a_{5h+1}^F$ in this left-to-right order. Let $\mathcal{M}^F(x) = \left\{(s_y^F(x), t_y^F(x)) \mid 1 \le y \le 5h + 1\right\}$ be a set of $(5h + 1)$ demand pairs that we call the False demand pairs for x. For each $1 \le y \le 5h + 1$, we identify $s_y^F(x)$ with the vertex a_y^F, and we let $t_y^F(x)$ be the yth vertex on $\hat{I}^F(x)$.

We let $\mathcal{M}(x) = \mathcal{M}^X(x) \cup \mathcal{M}^T(x) \cup \mathcal{M}^F(x)$ be the set of demand pairs representing x.

Consider the set $\mathcal{C}(x) \subseteq \mathcal{C}$ of clauses in which variable x appears without negation. Assume w.l.o.g. that $\mathcal{C}(x) = \{C_1, \ldots, C_r\}$, where $r \le 5$. For each $1 \le r' \le r$, we will create h demand

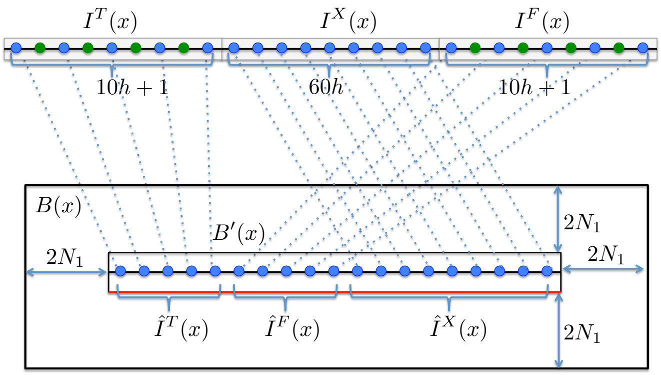

Figure 3: A variable gadget.

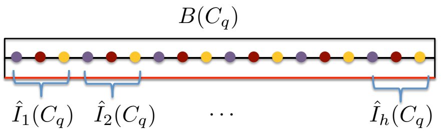

Figure 4: A clause gadget. Vertices of different colors correspond to different literals.

pairs $\left\{(s_j(C_{r'},x), t_j(C_{r'},x)) \mid 1 \leq j \leq h\right\}$, representing the literal x of $C_{r'}$. Consider the interval $I^F(x)$. We will use its vertices b^F_j as the sources of these demand pairs, where, intuitively, sources corresponding to the same clause-literal pair appear consecutively. Formally, for each $1 \leq r' \leq r$ and $1 \leq j \leq h$, we identify the vertex $s_j(C_{r'},x)$ with the vertex $b^F_{(r'-1)h+j}$ of $I^F(x)$. Intuitively, if x is assigned the value FALSE, then we will route all demand pairs in $\mathcal{M}^F(x)$. The paths routing these pairs will "block" the vertices b^F_j, thus preventing us from routing demand pairs that represent clause-literal pairs $(C_{r'},x)$. We treat the subset $C'(x) \subseteq C$ of clauses containing the literal $\neg x$ similarly, except that we identify their source vertices with the vertices of $\left\{b^T_1,\ldots,b^T_{5h}\right\}$ of $I^T(x)$.

Clause Gadgets. Consider some clause $C_q = (\ell_{q_1} \vee \ell_{q_2} \vee \ell_{q_3})$. For each one of the three literals $\ell \in \left\{\ell_{q_1},\ell_{q_2},\ell_{q_3}\right\}$ of C_q, let $\mathcal{M}(C_q,\ell) = \left\{(s_j(C_q,\ell), t_j(C_q,\ell)) \mid 1 \leq j \leq h\right\}$ be a set of h demand pairs representing the literal ℓ for clause C_q. Recall that $B(C_q)$ is a grid of height 3 and length $3h$. We delete all vertices that appear on the bottom row of this grid, and we let $R(C_q)$ be the middle row of $B(C_q)$. Partition $R(C_q)$ into h intervals $\hat{I}_1(C_q),\ldots,\hat{I}_h(C_q)$, each containing three consecutive vertices (see Figure 4). Fix some $1 \leq j \leq h$. We identify the three vertices of $\hat{I}_j(C_q)$ with the destination vertices $t_j(C_q,\ell_{q_1})$, $t_j(C_q,\ell_{q_2})$, and $t_j(C_q,\ell_{q_3})$ in this order. For each $1 \leq z \leq 3$, the corresponding source vertex $s_j(C_q,\ell_{q_z})$ has already been defined as part of the definition of the variable gadget corresponding to the literal ℓ_{q_z}. Let $\mathcal{M}(C_q) = \bigcup_{z=1}^{3} \mathcal{M}(C_q,\ell_{q_z})$ be the set of all demand pairs representing C_q, so $|\mathcal{M}(C_q)| = 3h$. Let $\mathcal{M}^C = \bigcup_{q=1}^{m} \mathcal{M}(C_q)$ be the set of all clause-pairs, and let $\mathcal{M}^V = \bigcup_{j=1}^{n} \mathcal{M}(x_j)$ be the set of all variable-pairs. Our final set of demand pairs is $\mathcal{M} = \mathcal{M}^V \cup \mathcal{M}^C$. This concludes the definition of the level-1 instance. We now proceed to analyze it.

Yes-Instance Analysis. Assume that φ is a YES-INSTANCE. We show that for every instantiation of the level-1 instance $\mathcal{I}$, there is a set $\mathcal{P}$ of node-disjoint paths routing $N_1 = (200h/3+1)n$ demand pairs, that respect the box $B(\mathcal{I})$. Assume that we are given some instantiation of $\mathcal{I}$. We first select the set $\hat{\mathcal{M}} \subseteq \mathcal{M}$ of demand pairs to route. Fix some assignment $\mathcal{A}$ to the variables of φ that satisfies all the clauses. For every variable x, if $\mathcal{A}$ assigns the value TRUE to x, then we let $\hat{\mathcal{M}}(x) = \mathcal{M}^T(x) \cup \mathcal{M}^X(x)$, and otherwise, we let $\hat{\mathcal{M}}(x) = \mathcal{M}^F(x) \cup \mathcal{M}^X(x)$. Notice that in either case, $|\hat{\mathcal{M}}(x)| = 65h+1$. For each clause $C_q = (\ell_{q_1} \vee \ell_{q_2} \vee \ell_{q_3}) \in C$, let ℓ_{q_z} be a literal which evaluates to TRUE by $\mathcal{A}$ (if there are several such literals, select one of them arbitrarily). We let $\hat{\mathcal{M}}(C_q) = \mathcal{M}(C_q,\ell_{q_z})$. Let $\hat{\mathcal{M}}^V = \bigcup_{j=1}^{n} \hat{\mathcal{M}}(x_j)$ and $\hat{\mathcal{M}}^C = \bigcup_{q=1}^{m} \hat{\mathcal{M}}(C_q)$. We then set $\hat{\mathcal{M}} = \hat{\mathcal{M}}^V \cup \hat{\mathcal{M}}^C$, so $|\hat{\mathcal{M}}| = (65h+1)n + mh = (200h/3+1)n = N_1$, as $m = 5n/3$. In the full version of the paper, we show that all demand pairs in $\hat{\mathcal{M}}$ can be routed by a set $\mathcal{P}$ of paths that respect the box $B(\mathcal{I})$. Due to lack of space, the proof is omitted here.

No-Instance Analysis. Assume now that φ is a NO-INSTANCE, and that we are given some instantiation of the level-1 instance $\mathcal{I}$ and a set $\tilde{\mathcal{P}}$ of node-disjoint paths routing some subset $\tilde{\mathcal{M}} \subseteq \mathcal{M}$ of demand pairs. Our goal is to prove that $|\tilde{\mathcal{M}}| \leq N'_1 = (1-\delta)N_1 = (1-\delta)(200h/3+1)n$. Assume for contradiction that $|\tilde{\mathcal{M}}| > N'_1$. In order to analyze the NO-INSTANCE, it is convenient to view the construction slightly differently. Let C' be the set of clauses obtained by adding, for each clause $C_q \in C$, h copies $C^1_q,\ldots,C^h_q$ of C_q to C'. We will refer to the clauses in C as the *original clauses*, and the clauses in C' as the *new clauses*. Notice that $|C'| = mh$, and it is easy to verify that no assignment to the variables of φ can satisfy more than $(1-\epsilon)hm$ clauses of C'. We will reach a contradiction by defining an assignment to the variables of φ that satisfies more than $(1-\epsilon)hm$ clauses of C'. For each new clause $C^j_q \in C'$, we let $\mathcal{M}(C^j_q) \subseteq \mathcal{M}$ be the set of all demand pair whose destinations lie on interval $\hat{I}_j(C_q)$; we view these demand pairs as representing the clause C^j_q.

For every variable x of φ, let $\tilde{\mathcal{M}}(x) = \tilde{\mathcal{M}} \cap \mathcal{M}(x)$. Let $\tilde{\mathcal{M}}^T(x), \tilde{\mathcal{M}}^F(x)$ and $\tilde{\mathcal{M}}^X(x)$ denote $\mathcal{M}^T(x) \cap \tilde{\mathcal{M}}$, $\mathcal{M}^F(x) \cap \tilde{\mathcal{M}}$ and $\mathcal{M}^X(x) \cap \tilde{\mathcal{M}}$, respectively. For every new clause $C^j_q \in C'$, let $\tilde{\mathcal{M}}(C^j_q) = \tilde{\mathcal{M}} \cap \mathcal{M}(C^j_q)$. We also denote by $\tilde{\mathcal{M}}^V = \bigcup_{j=1}^{n} \tilde{\mathcal{M}}(x_j)$ and by $\tilde{\mathcal{M}}^C = \bigcup_{C^j_q \in C'} \tilde{\mathcal{M}}(C^j_q)$, the sets of the variable-pairs and the clause-pairs, respectively, that are routed by $\tilde{\mathcal{P}}$. We use the following claim, whose proof is omitted due to lack of space.

CLAIM 3.1. *For each variable x of the instance φ, at least one of the sets $\tilde{\mathcal{M}}^T(x), \tilde{\mathcal{M}}^F(x), \tilde{\mathcal{M}}^X(x)$ is empty.*

Notice that from the above claim, $|\tilde{\mathcal{M}}^V| \leq (65h+1)n$. Consider now some variable x. Assume first that $\tilde{\mathcal{M}}^F(x) = \emptyset$. We then assign x the value TRUE. We say that an index $1 \leq j \leq 5h+1$ is *bad* for

x, iff the pair $(s^T_j(x), t^T_j(x)) \notin \tilde{\mathcal{M}}(x)$. Otherwise, if $\tilde{\mathcal{M}}^F(x) \neq \emptyset$, then we assign x the value False. In this case, we say that an index $1 \le j \le 5h+1$ is bad for x, iff $(s^F_j(x), t^F_j(x)) \notin \tilde{\mathcal{M}}(x)$.

Consider some new clause C^j_q. We say that it is an *interesting* clause if $|\tilde{\mathcal{M}}(C^j_q)| \ge 1$ (in other words, at least one pair in the set $\left\{(s_j(C_q, \ell_{q_z}), t_j(C_q, \ell_{q_z})) \mid 1 \le z \le 3\right\}$ is in $\tilde{\mathcal{M}}$), and we say that it is uninteresting otherwise. We say that clause C^j_q is *troublesome* iff $|\tilde{\mathcal{M}}(C^j_q)| > 1$. The proof of the following simple observation is omitted here due to lack of space.

Observation 3.2. *For each clause C_q, at most three of its copies are troublesome.*

We conclude that $|\tilde{\mathcal{M}}^C| \le m(h+6) = 5n(h+6)/3$. Let $C'_1 \subseteq C'$ be the set of all interesting new clauses. A simple accounting shows that, if $|\tilde{\mathcal{M}}| \ge (1-\delta)(200h/3+1)n$, then $|C'_1| \ge (1-\epsilon/10)hm$ must hold. Notice that for each new clause $C^j_q \in C'_1$, at least one demand pair from the set $\left\{(s_j(C_q, \ell_{q_z}), t_j(C_q, \ell_{q_z})) \mid 1 \le z \le 3\right\}$ is in $\tilde{\mathcal{M}}$. We select any literal $\ell \in \left\{\ell_{q_1}, \ell_{q_2}, \ell_{q_3}\right\}$ such that $(s_j(C_q, \ell), t_j(C_q, \ell)) \in \tilde{\mathcal{M}}$, and we say that clause C^j_q *chooses* the literal ℓ. Let x be the variable corresponding to the literal ℓ. We say that C^j_q is a *cheating* clause iff the assignment that we chose for x is not consistent with the literal ℓ: that is, if $\ell = x$, then $\mathcal{A}(x) =$ False, and if $\ell = \neg x$, then $\mathcal{A}(x) =$ True. Notice that, if C^j_q is an interesting and a non-cheating clause, then the current assignment satisfies C^j_q. Therefore, in order to compete the analysis, it is enough to prove the following claim.

Claim 3.3. *The number of cheating clauses in C'_1 is bounded by $\epsilon mh/2$.*

Due to lack of space, we only provide a proof sketch here. Let $C^j_q \in C'$ be a cheating clause, and suppose it has chosen the literal ℓ, whose corresponding variable is x. We say that C^j_q is a *bad* cheating clause, iff at least one of the indices $j, j+1$ is a bad index for variable x (recall that j is a bad index for x if $\mathcal{A}(x) =$ True and $(s^T_j(x), t^T_j(x)) \notin \tilde{\mathcal{M}}$, or $\mathcal{A}(x) =$ False and $(s^F_j(x), t^F_j(x)) \notin \tilde{\mathcal{M}}$). Otherwise, we say that C^j_q is a good cheating clause. A simple accounting shows that the number of pairs (x, j), where j is a bad index for x is bounded by $\epsilon mh/16$. Each such pair (x, j) may contribute to at most two bad cheating clauses, and so there are at most $\epsilon mh/8$ bad cheating clauses.

Our final step is to show that the number of good cheating clauses is bounded by $\epsilon mh/4$. We show that for each original clause C_q, at most 3 copies of C_q are good cheating clauses. It then follows that the total number of good cheating clauses is at most $3m < \epsilon mh/4$, since $h = 1000/\epsilon$. Consider some original clause C_q. It is enough to show that for each literal ℓ of C_q, the number of copies of C_q that choose ℓ and are good cheating clauses is at most 1. Assume for contradiction that there are two such copies C^j_q and $C^{j'}_q$. Assume w.l.o.g. that the variable x that corresponds to ℓ is assigned the value True, so $\ell = \neg x$. Then vertex $s_j(C_q, \ell)$ lies on interval $I^T(x)$, between $s^T_j(x)$ and $s^T_{j+1}(x)$, while vertex $s_{j'}(C_q, \ell)$ lies on interval $I^T(x)$, between $s^T_{j'}(x)$ and $s^T_{j'+1}(x)$. Assume w.l.o.g. that $j < j'$. Consider the plane with only the top boundary of the grid G_1, the bottom boundary of the box $B'(x)$, and the images of the paths of $\tilde{\mathcal{P}}$ routing the pairs $(s^T_j(x), t^T_j(x)), (s^T_{j+1}(x), t^T_{j+1}(x))$, and $(s^T_{j'+1}(x), t^T_{j'+1}(x))$ present. Let f, f' be the two faces of this drawing that differ from the outer face, such that f has $s_j(C_q, \ell)$ on its boundary and f' has $s_{j'}(C_q, \ell)$ on its boundary. Then $f \neq f'$, and the bottom boundary of $B(C_q)$ must belong to a single face of the resulting drawing. Assume w.l.o.g. that this face is $f^* \neq f$. Then $t_j(C_q, \ell)$ lies on f^*, and so it is impossible that a path of $\tilde{\mathcal{P}}$ connects $s_j(C_q, \ell)$ to $t_j(C_q, \ell)$. We conclude that the current assignment satisfies at least $(1-\epsilon/10)hm - \epsilon hm/2 > (1-\epsilon)hm$ clauses of C', a contradiction.

Generalization to Higher Levels and the Hardness Gap. Assume now that we are given a construction of a level-i instance, and we would like to construct a level-$(i+1)$ instance. Intuitively, we would like to start with the level-1 instance described above, and then replace each source-destination pair (s, t) with a distinct copy of a level-i instance $\mathcal{I}'$. So we would replace the vertex s with the path $Z(\mathcal{I}')$, and the vertex t with the cut-out box $B(\mathcal{I}')$. We say that a level-i instance $\mathcal{I}'$ is routed by a solution $\mathcal{P}$ to this resulting instance, iff $\mathcal{P}$ routes a significant number of the demand pairs in $\mathcal{I}'$. The idea is that, due to the level-1 instance analysis, the number of such level-i instances that can be routed in the Yes- and the No-Instance cases differ by a constant factor, while within each such instance we already have some gap g_i between the Yes- and the No-Instance solutions, and so the gap grows by a constant factor in every level. Unfortunately this idea does not quite work. If we consider, for example, a level-1 instance $\mathcal{I}'$, then its destination vertices appear quite far – at distance $\Theta(N_1)$ – from the bottom boundary of the box $B(\mathcal{I}')$. In general, in a level-i instance, this distance needs to be roughly $\Theta(N_i)$, to allow the routing in the Yes-Instance case (recall that N_i is the number of the demand pairs that can be routed in the Yes-Instance case). Therefore, if we replace each level-1 demand pair by a level-i instance, some of the paths in the new level-$(i+1)$ instance may "cheat" by passing through the boxes $B(\mathcal{I}')$ of level-i instances $\mathcal{I}'$, and exploiting the spacing between the destination vertices and the bottom boundary of each such box. For example, it is now possible that in a variable gadget, we will be able to route many demand pairs from each set $\mathcal{M}^X(x)$, $\mathcal{M}^T(x)$ and $\mathcal{M}^F(x)$ simultaneously. A simple way to get around this problem is to create more level-i instances, namely: we replace each source-destination pair from a level-1 instance by a collection of c_{i+1} level-i instances. The idea is that, if the number of the demand pairs we try to route in many such c_{i+1}-tuples of level-i instances is large enough, then on average only a small fraction of the routing paths may cheat by exploiting the spacing between the destination vertices and their corresponding box boundaries, and this will not affect the overall accounting by too much. If the formula φ is a No-Instance, then we will only attempt to route N'_i demand pairs from each level-i instance, and therefore we need to ensure that $c_{i+1}N'_i \gg N_i$ in order for the gap to grow in the current level. In other words, the number of copies of the level-i instances that we use in the level-$(i+1)$ instance construction should be proportional to the gap between the Yes- and the No-Instance cost at level i (times n). A simple calculation shows that, if we follow this approach, we will obtain a gap of $2^{\Omega(i)}$ in level-i instances, with construction size roughly $n^{\Theta(i)} \cdot 2^{\Theta(i^2)}$. Therefore, after $i^* = \Theta(\log n)$ iterations, we obtain a gap of $2^{\Omega(\sqrt{\log n'})}$, where n' is the size of the level-i^* instance. This rapid growth in the instance size is the main obstacle to obtaining a stronger hardness of approximation factor using this approach.

4 THE FULL CONSTRUCTION

In this section we provide a full description of our construction. The resulting graphs will have maximum vertex degree 4. We show in Section 7 how to modify the resulting instances in order to obtain the proof of Theorem 1.1 for sub-cubic graphs. We start with setting the parameters.

Parameters. The two main parameters that we use are $h = 1000/\epsilon$ and $\delta = 8\epsilon^2/10^{12}$, where ϵ is the constant from Theorem 2.1. We define the remaining parameters in terms of these two parameters.

For every level $i \geq 0$ of our construction, we use two parameters, N_i and N'_i. We will ensure that for every instantiation of the level-i instance $\mathcal{I}$, if the initial 3SAT(5) formula φ is a Yes-Instance, then there is a solution to $\mathcal{I}$ routing N_i demand pairs, that respects the box $B(\mathcal{I})$, and if φ is a No-Instance, then no solution to $\mathcal{I}$ can route more than N'_i demand pairs. We define the parameters N_i, N'_i inductively, starting with $N_0 = N'_0 = 1$. Assume now that for some $i \geq 0$, we are given the values of N_i and N'_i. Let $g_i = N_i/N'_i$ be the gap between the Yes- and the No-Instance solution values at level i, and let $c_{i+1} = 10^8 h^2 g_i = O(g_i)$. Parameter c_{i+1} will be used in our construction of the level-$(i+1)$ instance. We then set $N_{i+1} = nc_{i+1}(200h/3+1)N_i$ and $N'_{i+1} = (1-\delta)nc_{i+1}(200h/3+1)N'_i$. It is immediate to verify that $g_{i+1} = \frac{g_i}{1-\delta}$, and so for all $i \geq 0$, $g_i = \left(\frac{1}{1-\delta}\right)^i$, and $N_i = O(n \cdot g_{i-1}) \cdot N_{i-1} = (\rho n)^i \cdot 2^{O(i^2)}$, for some absolute constant ρ. We set the parameters L_i, L'_i and H_i below, but we will ensure that each of these parameters is bounded by $20N_i^3$. Our construction has $i^* = \log n$ levels, giving us a gap of $2^{\Omega(\log n)}$ between the Yes- and the No-Instance solution values. For our final level-i^* instance, we can choose the grid G_{i^*} to be of size $(Q \times Q)$, where $Q = 2L_{i^*} + 2L'_{i^*} + 4H_{i^*} = O(N_{i^*}^3)$, and so the instance size is bounded by $n' = O(N_{i^*}^6) = n^{O(\log n)} \cdot 2^{O(\log^2 n)} = 2^{O(\log^2 n)}$. Overall, we obtain a factor $2^{\Omega(\sqrt{\log n'})}$-hardness of approximation, unless all problems in NP have deterministic algorithms running in time $n^{O(\log n)}$.

For $i \geq 0$, we set the parameter $H_i = 20N_i$. For all $i \geq 0$, we set $L'_i = 20N_i^3$. Parameter L_i is defined as follows: $L_0 = 1$, and for $i > 0$, $L_i = (80h+2)c_i L_{i-1} n \leq (80h+2)c_i \cdot 20N_{i-1}^3 n \leq 20N_i^3$.

Level-0 Instance. A level-0 instance $\mathcal{I}$ consists of a single demand pair (s, t). In order to be consistent with our definitions, we let $Z(\mathcal{I})$ be a path of length $L_0 = 1$, with s mapped to the unique vertex of $Z(\mathcal{I})$. Recall that $N_0 = N'_0 = 1$, $H_0 = 20N_0 = 20$, and $L'_0 = 20N_0^3 = 20$. Let G'_0 be a grid of length $L'_0 = 20$ and height $H_0 = 20$. We obtain the box $B(\mathcal{I})$ from G'_0 by deleting all vertices lying on the left, bottom, and right boundaries of G'_0. Let R' be the middle row of G'_0. We map t to any vertex of $B(\mathcal{I})$ that belongs to R'. It is immediate to verify that for every instantiation of this level-0 instance, there is a solution that routes one demand pair and respects $B(\mathcal{I})$, regardless of whether we are in the Yes or the No-Instance.

From now on we focus on constructing instances of levels $i > 0$. As already mentioned, the construction we obtain for level $i = 1$ is similar to that described in Section 3.

4.1 Level-$(i+1)$ Construction

A level-$(i+1)$ instance is obtained by combining a number of level-i instances. We start with a quick overview of the level-i construction.

Level-i Construction Overview. Recall that a definition of a level-i instance $\mathcal{I}$ consists of a path $Z(\mathcal{I})$ of length L_i, a grid G'_i of height H_i and length L'_i, together with a cut-out box $B(\mathcal{I}) \subseteq G'_i$, and a collection $\mathcal{M}$ of demand pairs, such that all vertices of $S(\mathcal{M})$ are mapped to vertices of $Z(\mathcal{I})$, while all vertices of $T(\mathcal{M})$ are mapped to distinct vertices of $B(\mathcal{I}) \cap R'$, where R' is the middle row of G'_i. Path $Z(\mathcal{I})$ will eventually become a sub-path of the first row of the larger grid G_i, and box $B(\mathcal{I})$ will be placed inside G_i, within distance at least H_i from its boundaries.

For every destination vertex $t \in T(\mathcal{M})$, we draw a straight line Q_t from t to the bottom of $B(\mathcal{I})$. This line contains at most $H_i/2$ vertices of the graph. We will use these lines in the analysis of the No-Instance case of the level-$(i+1)$ construction.

It will sometimes be useful to place several level-i instances side-by-side. For an integer $c > 0$, a *c-wide level-i instance $\mathcal{I}$* is constructed as follows. Intuitively, we construct c disjoint level-i instances $\mathcal{I}_1, \ldots, \mathcal{I}_c$, placing their intervals $Z(\mathcal{I}_j)$ side-by-side on $Z(\mathcal{I})$ and placing their boxes $B(\mathcal{I}_j)$ side-by-side inside $B(\mathcal{I})$. Formally, for each $1 \leq j \leq c$, let $\mathcal{M}_j$ be the set of the demand pairs of the level-i instance $\mathcal{I}_j$, and let G^j be the corresponding grid G'_i for that instance. The set of the demand pairs of instance $\mathcal{I}$ is $\mathcal{M} = \bigcup_{j=1}^{c} \mathcal{M}_j$. We let $Z(\mathcal{I})$ be a path of length $c \cdot L_i$, partitioned into c equal-length intervals $A_1, \ldots, A_c$. We let G' be a grid of length cL'_i and height H_i, that we partition into c sub-grids of length L'_i and height H_i each. For $1 \leq j \leq c$, we map the vertices of $Z(\mathcal{I}_j)$ to the vertices of A_j in a natural way. This defines the mapping of the vertices of $S(\mathcal{M})$ to the vertices of $Z(\mathcal{I})$. For each $1 \leq j \leq c$, we map the vertices of G^j to the jth sub-grid of G', and delete from G' all vertices to which the vertices of $G^j \setminus B(\mathcal{I}_j)$ are mapped. The resulting subgraph of G' becomes the box $B(\mathcal{I})$, and the above mapping defines the mapping of the destination vertices in $T(\mathcal{M})$ to the vertices of $B(\mathcal{I})$. Note that if R' denotes the middle row of G', then all vertices of $T(\mathcal{M})$ lie on R'. In order to instantiate this instance, we need to select a grid G of length at least $c(2L_i + 2L'_i + 4H_i)$ and height at least $3cH_i$, a sub-path P of the first row of G of length cL_i, to which $Z(\mathcal{I})$ will be mapped, and a sub-grid G'' of G of the same dimensions as G', to which the vertices of G' will be mapped. The vertices of G'' must be at a distance at least cH_i from the boundaries of G. Clearly, for any instantiation of this instance, in the YES-INSTANCE case, there is a solution $\mathcal{P}$ routing cN_i demand pairs, such that, if we denote, for each $1 \le j \le c$, by $\mathcal{P}_j \subseteq \mathcal{P}$ the set of paths routing demand pairs in $\mathcal{M}_j$, then $|\mathcal{P}_j| = N_i$ and $\mathcal{P}_j$ respects the box $B(\mathcal{I}_j)$. On the other hand, in the NO-INSTANCE case, no solution to $\mathcal{I}$ can route more than cN'_i demand pairs.

We now assume that we are given a construction of a level-i instance, for $i \ge 0$, and describe a construction of a level-$(i+1)$ instance $\mathcal{I}$. For convenience, we denote c_{i+1} by c. We use parameters $L_{i+1}, H_{i+1}, L'_{i+1}$ described above, so $H_{i+1} = 20N_{i+1}, L'_{i+1} = 20N^3_{i+1}$, and $L_{i+1} = (80h+2)cL_in$.

In order to construct the box $B(\mathcal{I})$, we start with a grid G'_{i+1} of length L'_{i+1} and height H_{i+1}. We define two sub-grids of $B(\mathcal{I})$, of length $9N^3_{i+1}$ and height $16N_{i+1}$ each: grid B^V that will contain all destination vertices of the demand pairs representing the variables of the formula φ, and grid B^C that will contain all destination vertices of the demand pairs representing the clauses of the formula φ. In order to construct both grids, let $\mathcal{R}$ be the set of all rows of G'_{i+1}, excluding the top $2N_{i+1}$ and the bottom $2N_{i+1}$ rows, so that $|\mathcal{R}| = H_{i+1} - 4N_{i+1} = 16N_{i+1}$. Let $\mathcal{W}$ be a consecutive set of $9N^3_{i+1}$ columns of G'_{i+1}, starting from the second column, and let $\mathcal{W}'$ be a consecutive set of $9N^3_{i+1}$ columns of G'_{i+1}, terminating at the penultimate column. We then let B^V be the sub-grid of G'_{i+1} spanned by the rows in $\mathcal{R}$ and the columns in $\mathcal{W}$, and we let B^C be the sub-grid of G'_{i+1} spanned by the rows in $\mathcal{R}$ and the columns in $\mathcal{W}'$ (see Figure 5). Notice that at least $2N_{i+1}$ columns of G'_{i+1} separate the two grids. We delete the bottom, left, and right boundaries of G'_{i+1} to turn it into a cut-out box, that we refer to as $B(\mathcal{I})$ from now on. We will later delete some additional vertices from $B(\mathcal{I})$.

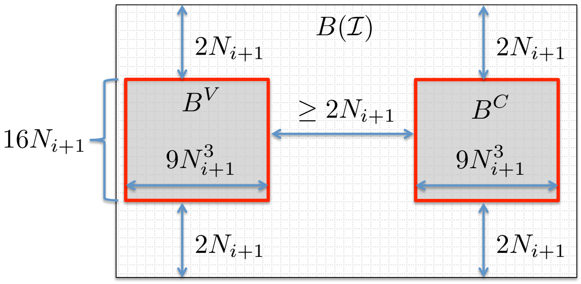

Figure 5: High-level view of box $B(\mathcal{I})$

Next, we define smaller sub-grids of the two grids B^V and B^C. For every variable x_j of φ, we define a sub-grid $B(x_j)$ of B^V, of length $L^V = 4N_{i+1} + (70h+2)cL'_i$ and height $H^V = H_i + 2N_{i+1}$. This box will contain all destination vertices of the demand pairs that represent the variable x_j. We place the boxes $B(x_1), \ldots, B(x_n)$ inside grid B^V, so that they are aligned and $2N_{i+1}$-separated. In other words, the middle row of each box is contained in the middle row of B^V, and the horizontal distance between every pair of these boxes, and between each box and the left and right boundaries of B^V is at least $2N_{i+1}$. Since $n \cdot L^V + (n+1) \cdot 2N_{i+1} \le 7nN_{i+1} + (70h+2)cL'_in \le 7nN_{i+1} + 1500hcN^3_in < 9N^3_{i+1}$, we can find such grids $B(x_1), \ldots, B(x_n)$. Since $H^V = H_i + 2N_{i+1} = 20N_i + 2N_{i+1} < 3N_{i+1}$, there are at least $2N_{i+1}$ rows of B^V above and below these new grids (see Figure 6).

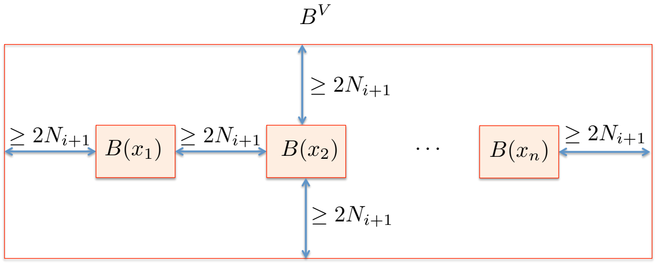

Figure 6: Schematic view of box B^V.

We similarly define sub-grids $B(C_1), \ldots, B(C_m)$ of B^C. Each such sub-grid has height $H^C = H_i$ and width $L^C = 3chL'_i$. For each clause $C_q \in C$, box $B(C_q)$ will contain all destination vertices of the demand pairs that represent this clause. We let $B(C_1), \ldots, B(C_m)$ be sub-grids of H^C that are aligned and $4N_{i+1}$-separated. Since $m \cdot L^C + (m+1) \cdot 4N_{i+1} \le 20nN_{i+1} + 15hcL'_in \le 20nN_{i+1} + 300hcN^3_in < 9N^3_{i+1}$, we can find such grids (see Figure 7). Since $H^C = H_i = 20N_i < N_{i+1}$, there are at least $2N_{i+1}$ rows of B^C above and below these new grids.

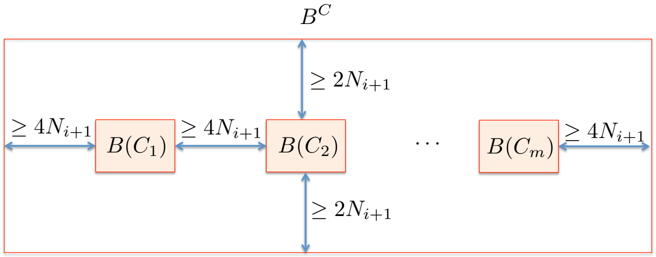

Figure 7: Schematic view of box B^C.

Our construction consists of two parts, called variable gadgets and clause gadgets. For each variable x_j, we construct a number of level-i instances $\mathcal{I}'$, whose corresponding boxes $B(\mathcal{I}')$ are placed inside $B(x_j)$. Whenever we do so, we delete the corresponding vertices of $B(x_j)$ as described in the preliminaries. We also construct clause gadgets similarly.

Variable Gadgets. Let $Z(\mathcal{I})$ be a path of length L_{i+1}, and let Π be a partition of $Z(\mathcal{I})$ into disjoint contiguous sub-paths (that we sometimes refer to as *intervals*) of length cL_i each. For each $1 \le j \le n$, we let $I(x_j)$ be a sub-path of $Z(\mathcal{I})$, containing exactly $80h+2$ consecutive intervals of Π, so that $I(x_1), I(x_2), \ldots, I(x_n)$ appear on $Z(\mathcal{I})$ in this left-to-right order.

Consider some variable x of the 3SAT(5) formula φ and the corresponding interval $I(x)$ of $Z(\mathcal{I})$, containing $80h+2$ consecutive intervals of Π. We further partition $I(x)$ as follows. Let $I^T(x), I^F(x) \subseteq$

$I(x)$ denote the subpaths of $I(x)$ containing the first $(10h+1)$ and the last $(10h+1)$ consecutive intervals of Π, respectively. Let $I^X(x)$ denote the union of the remaining $60h$ consecutive intervals of Π (see Figure 8).

(Extra Pairs). We use $60h$ copies of c-wide level-i instances, that we denote by $\mathcal{I}_j^X(x)$, for $1 \le j \le 60h$. For each $1 \le j \le 60h$, we let $Z(\mathcal{I}_j^X(x))$ be the jth interval of $I^X(x)$. We place the corresponding boxes $B(\mathcal{I}_1^X(x)), \dots, B(\mathcal{I}_{60h}^X(x))$ side-by-side, obtaining one box $B_X(x)$ of width $60hcL'_i$ and height H_i. We define the placement of this box inside $B(x)$ later. We denote by $\mathcal{M}^X(x)$ the resulting set of demand pairs, and we refer to them as *the* EXTRA *demand pairs of* x.

(TRUE Pairs). We denote the intervals of Π appearing on $I^T(x)$ by: $A_1^T, Y_1^T, A_2^T, Y_2^T, \dots, Y_{5h}^T, A_{5h+1}^T$, and we assume that they appear on $I^T(x)$ in this left-to-right order. We use $(5h+1)$ copies of the c-wide level-i instance, that we denote by $\mathcal{I}_j^T(x)$, for $1 \le j \le 5h+1$. For each $1 \le j \le 5h+1$, we let $Z(\mathcal{I}_j^T)$ be the interval A_j^T. We place the corresponding boxes $B(\mathcal{I}_1^T(x)), \dots, B(\mathcal{I}_{5h+1}^T(x))$ side-by-side, obtaining one box $B_T(x)$ of width $(5h+1)cL'_i$ and height H_i. We denote by $\mathcal{M}^T(x)$ the resulting set of demand pairs, and we refer to them as *the* TRUE *demand pairs of* x.

(FALSE Pairs). Similarly, we denote the intervals of Π appearing on $I^F(x)$ by: $A_1^F, Y_1^F, A_2^F, Y_2^F, \dots, Y_{5h}^F, A_{5h+1}^F$, and we assume that they appear on $I^F(x)$ in this left-to-right order. We use $(5h+1)$ copies of the c-wide level-i instance, that we denote by $\mathcal{I}_j^F(x)$, for $1 \le j \le 5h+1$. For each $1 \le j \le 5h+1$, we let $Z(\mathcal{I}_j^F)$ be the interval A_j^F. We place the corresponding boxes $B(\mathcal{I}_1^F(x)), \dots, B(\mathcal{I}_{5h+1}^F(x))$ side-by-side, obtaining one box $B_F(x)$ of width $(5h+1)cL'_i$ and height H_i. We denote by $\mathcal{M}^F(x)$ the resulting set of demand pairs, and we refer to them as *the* FALSE *demand pairs of* x.

We let $\mathcal{M}(x) = \mathcal{M}^X(x) \cup \mathcal{M}^T(x) \cup \mathcal{M}^F(x)$. We call the demand pairs in $\mathcal{M}(x)$ *variable-pairs representing* x.

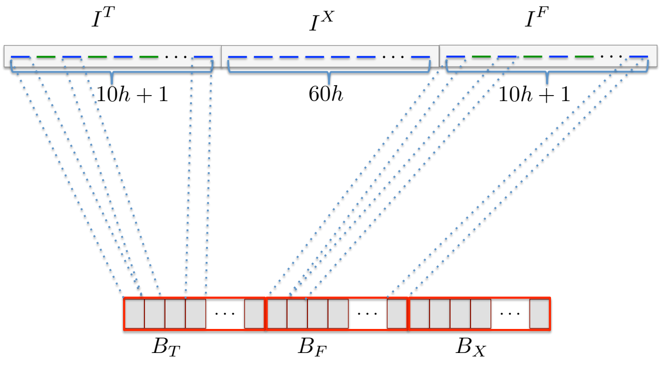

Figure 8: Variable gadget for level-$(i+1)$ instance.

Recall that the length of box $B_X(x)$ is $60hcL'_i$, while boxes $B_T(x), B_F(x)$ have length $(5h+1)cL'_i$ each. The height of each box is H_i. Recall also that box $B(x)$ has length $L^V = 4N_{i+1} + (70h+2)cL'_i$ and height $H^V = H_i + 2N_i$.

We place the boxes $B_T(x), B_F(x)$ and $B_X(x)$ side-by-side inside $B(x)$ in this order, so that the middle row of each box is contained in the middle row of $B(x)$, and there is a horizontal spacing of $2N_{i+1}$ between the left boundaries of $B_T(x)$ and $B(x)$, and between the right boundaries of $B_X(x)$ and $B(x)$ (see Figure 9). Notice that there is no horizontal spacing between $B_T(x), B_F(x)$ and $B_X(x)$, and all destination vertices lying in $B(x)$ belong to the middle row of $B(x)$, and hence to the middle row of $B(\mathcal{I})$. Recall that all vertices lying on the bottom, left and right boundaries of boxes $B_T(x), B_F(x)$ and $B_X(x)$ are deleted from $B(x)$.

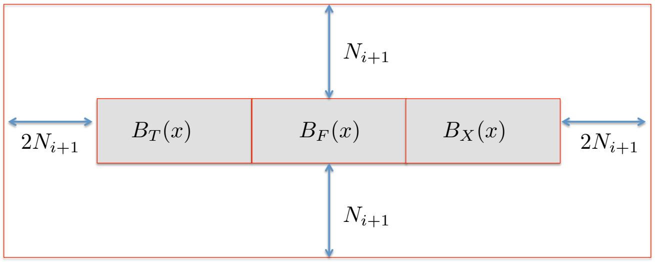

Figure 9: Box $B(x)$. Height: $H^V = H_i + 2N_{i+1}$, length: $L^V = 4N_{i+1} + (70h+2)cL'_i$.

Consider the set $C(x) \subseteq C$ of clauses in which variable x appears without negation. Assume without loss of generality that $C(x) = \{C_1, \dots, C_r\}$, where $r \le 5$. For each $1 \le r' \le r$, we will create h level-i instances of width c, that represent the variable x of $C_{r'}$. We denote these instances by $\mathcal{I}_j(C_{r'}, x)$, for $1 \le j \le h$. Consider the interval $I^F(x)$. We will use the sub-intervals Y_j^F of $I^F(x)$ as intervals $Z(\mathcal{I}_j(C_{r'}, x))$, where, intuitively, intervals corresponding to the same clause-variable pair appear consecutively. Formally, for each $1 \le r' \le r$, for each $1 \le j \le h$, we use the interval $Y^F_{(r'-1)h+j}$ of $I^F(x)$ as $Z(\mathcal{I}_j(C_{r'}, x))$, and we say that it is the sub-interval of $I^F(x)$ that belongs to instance $\mathcal{I}_j(C_{r'}, x)$. Intuitively, if x is assigned the value FALSE, then we will route a large number of demand pairs in $\mathcal{M}^F(x)$. The paths routing these pairs will "block" the intervals Y_j^F of $I^F(x)$, thus preventing us from routing demand pairs that belong to instances $\mathcal{I}_j(C_{r'}, x)$, for $1 \le j \le h$ and $C_{r'} \in C(x)$.

We treat the subset $C'(x) \subseteq C$ of clauses containing the negation of x similarly, except that we assign to each resulting instance an interval Y_j^T of $I^T(x)$.

Clause Gadgets. Consider some clause $C_q = (\ell_{q_1} \vee \ell_{q_2} \vee \ell_{q_3})$. For each one of the three literals of C_q, we construct h level-i width-c instances, with instances $\left\{\mathcal{I}_j(C_q, \ell_{q_z}) \mid 1 \le j \le h\right\}$ representing the literal ℓ_{q_z}, for $1 \le z \le 3$. Recall that $B(C_q)$ is a grid of height $H^C = H_i$ and length $L^C = 3chL'_i$. We partition $B(C_q)$ into h sub-grids $B^1(C_q), \dots, B^h(C_q)$, each of which has height H_i and length $3cL'_i$. For each $1 \le j \le h$, we place the boxes $B(\mathcal{I}_j(C_q, \ell_{q_1}))$, $B(\mathcal{I}_j(C_q, \ell_{q_2}))$ and $B(\mathcal{I}_j(C_q, \ell_{q_3}))$ inside $B^j(C_q)$ side-by-side in this order (see Figure 10). As before, all vertices lying on the bottom, left and right boundaries of boxes $B^j(C_q)$, for all $1 \le j \le h$, are deleted from $B(C_q)$.

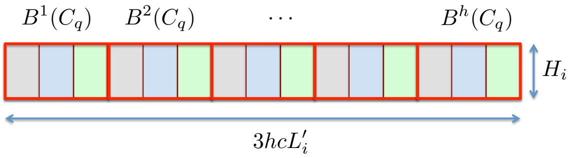

Figure 10: Box $B(C_q)$. Different colors show boxes that represent the three different literals of C_q.

The intervals $Z(\mathcal{I}_j(C_q, \ell_{q_z}))$ are the same as the ones defined in the constructions of the variable gadgets. We denote by $\mathcal{M}(C_q)$ the set of all demand pairs whose destinations lie in $B(C_q)$, and we call them *clause-pairs representing* C_q. For each $1 \le z \le 3$, we denote by $\mathcal{M}(C_q, \ell_{q_z})$ the set of all demand pairs that belong to instances $\mathcal{I}_j(C_q, \ell_{q_z})$, for $1 \le j \le h$, and we sometimes say that they represent literal ℓ_{q_z} of clause C_q. We then let $\mathcal{M}^C = \bigcup_{q=1}^m \mathcal{M}(C_q)$ be the set of all clause-pairs, and $\mathcal{M}^V = \bigcup_{j=1}^n \mathcal{M}(x_j)$ the set of all variable-pairs. Our final set of demand pairs is $\mathcal{M} = \mathcal{M}^V \cup \mathcal{M}^C$. This completes the definition of the level-$(i+1)$ instance.

5 YES-INSTANCE ANALYSIS

The goal of this section is to prove the following theorem.

Theorem 5.1. *Assume that the input 3SAT(5) formula φ is a* Yes-Instance. *Then for all $i \ge 0$, for every instantiation of the level-i instance $\mathcal{I}$, there is a solution routing N_i demand pairs, that respects the box $B(\mathcal{I})$.*

The remainder of this section is devoted to proving this theorem. The proof proceeds by induction. For $i = 0$, $N_0 = 1$, and it is easy to see that for any instantiation of the level-0 instance $\mathcal{I}$, there is a solution that routes the unique demand pair of this instance and respects the box $B(\mathcal{I})$. We now assume that the theorem holds for some $i \ge 0$, and prove it for a level-$(i+1)$ instance, that we denote by $\mathcal{I}$. We assume that we are given an instantiation of instance $\mathcal{I}$, that consists of a grid G_{i+1} of length at least $2L_{i+1} + 2L'_{i+1} + 4H_{i+1}$ and height at least $3H_{i+1}$, the placement of the path $Z(\mathcal{I})$ on the top boundary of G_{i+1}, and the placement of the box $B(\mathcal{I})$ inside G_{i+1}, at distance at least H_{i+1} from its boundaries. We denote the resulting graph by G, and the resulting set of demand pairs by $\mathcal{M}$. Recall that our construction combines a number of level-i instances. From the induction hypothesis, for each such instance $\mathcal{I}'$, for every instantiation of instance $\mathcal{I}'$, there is a set $\mathcal{P}(\mathcal{I}')$ of disjoint paths, routing a set of N_i demand pairs of $\mathcal{I}'$, such that the paths in $\mathcal{P}(\mathcal{I}')$ respect the box $B(\mathcal{I}')$. It is easy to verify that, if a set $\mathcal{M}^*(\mathcal{I}')$ of demand pairs of $\mathcal{I}'$ has a routing that respects $B(\mathcal{I}')$ in one instantiation of $\mathcal{I}'$, then it has such a routing in every instantiation of $B(\mathcal{I}')$. Therefore, for every level-i instance $\mathcal{I}'$, we can fix one such set $\mathcal{M}^*(\mathcal{I}')$ of demand pairs with $|\mathcal{M}^*(\mathcal{I}')| = N_i$, and assume that $\mathcal{M}^*(\mathcal{I}')$ has a routing that respects $B(\mathcal{I}')$ in every instantiation of $\mathcal{I}'$.

Recall that our construction combines level-i instances into c_{i+1}-wide level-i instances. Let $\mathcal{I}''$ be any such c_{i+1}-wide level-i instance, and assume that it consists of level-i instances $\mathcal{I}_1, \ldots, \mathcal{I}_{c_{i+1}}$. We set $\mathcal{M}^*(\mathcal{I}'') = \bigcup_{j=1}^{c_{i+1}} \mathcal{M}^*(\mathcal{I}_j)$, so $|\mathcal{M}^*(\mathcal{I}'')| = c_{i+1}N_i$. It is easy to see that for any instantiation of $\mathcal{I}''$, there is a routing of all demand pairs in $\mathcal{M}^*(\mathcal{I}'')$, such that for each $1 \le j \le c_{i+1}$, the demand pairs in $\mathcal{M}^*(\mathcal{I}_j)$ are routed via paths that respect the box $B(\mathcal{I}_j)$.

Consider now the given instantiation $(G, \mathcal{M})$ of the level-$(i+1)$ instance $\mathcal{I}$. We first select the set $\hat{\mathcal{M}} \subseteq \mathcal{M}$ of demand pairs that we route, and then compute their routing. We fix some assignment $\mathcal{A}$ to the variables $\{x_1, \ldots, x_n\}$ of φ, that satisfies all clauses.

Variable Pairs. Let x be a variable. We let $\hat{\mathcal{M}}^X(x) = \bigcup_{j=1}^{60h} \mathcal{M}^*(\mathcal{I}_j^X(x))$ — the set of all demand pairs that are routed by the solutions to the c_{i+1}-wide level-i instances $\mathcal{I}_1^X(x), \ldots, \mathcal{I}_{60h}^X(x)$. Notice that $|\hat{\mathcal{M}}^X(x)| = 60hc_{i+1}N_i$. If x is assigned the value True, then we let $\hat{\mathcal{M}}^T(x) = \bigcup_{j=1}^{5h+1} \mathcal{M}^*(\mathcal{I}_j^T(x))$, and we set $\hat{\mathcal{M}}^F(x) = \emptyset$. Notice that $|\hat{\mathcal{M}}^T(x)| = (5h+1)c_{i+1}N_i$ in this case. Otherwise, we let $\hat{\mathcal{M}}^F(x) = \bigcup_{j=1}^{5h+1} \mathcal{M}^*(\mathcal{I}_j^F(x))$, so $|\hat{\mathcal{M}}^F(x)| = (5h+1)c_{i+1}N_i$, and we set $\hat{\mathcal{M}}^T(x) = \emptyset$. We denote $\hat{\mathcal{M}}(x) = \hat{\mathcal{M}}^X(x) \cup \hat{\mathcal{M}}^T(x) \cup \hat{\mathcal{M}}^F(x)$, and we let $\hat{\mathcal{M}}^V = \bigcup_x \hat{\mathcal{M}}(x)$, so $|\hat{\mathcal{M}}^V| = nc_{i+1}(65h+1)N_i$.

Clause Pairs. Let $C_q \in \mathcal{C}$ be a clause, and let ℓ_q be a literal of C_q whose value is True (if there are several such literals, we select any one of them arbitrarily). We say that clause C_q *chooses* the literal ℓ_q. For simplicity, we denote $\hat{\mathcal{M}}_j(C_q) = \mathcal{M}^*(\mathcal{I}_j(C_q, \ell_q))$, for all $1 \le j \le h$, and we let $\hat{\mathcal{M}}(C_q) = \bigcup_{j=1}^h \hat{\mathcal{M}}_j(C_q)$. Let $\hat{\mathcal{M}}^C = \bigcup_{q=1}^m \hat{\mathcal{M}}(C_q)$. Clearly, for each $1 \le q \le m$, $|\hat{\mathcal{M}}(C_q)| = hc_{i+1}N_i$, and overall, $|\hat{\mathcal{M}}^C| = mhc_{i+1}N_i = 5nhc_{i+1}N_i/3$.

Finally, we let $\hat{\mathcal{M}} = \hat{\mathcal{M}}^V \cup \hat{\mathcal{M}}^C$, so $|\hat{\mathcal{M}}| = nc_{i+1} \cdot (65h+1)N_i + 5nc_{i+1}hN_i/3 = n \cdot N_ic_{i+1}(200h/3+1) = N_{i+1}$. It is now enough to prove the following lemma. The proof of the lemma is omitted due to lack of space, and can be found in the full version of the paper.

Lemma 5.2. *There is a set $\mathcal{P}$ of node-disjoint paths in graph G, routing all demand pairs in $\hat{\mathcal{M}}$, such that $\mathcal{P}$ respects box $B(\mathcal{I})$.*

6 NO-INSTANCE ANALYSIS

In this section we analyze the No-Instance case, by proving the following theorem.

Theorem 6.1. *Assume that φ is a* No-Instance. *Then for every integer $i \ge 0$, for every instantiation of the level-i instance $\mathcal{I}$, and for every solution $\mathcal{P}$ to this instance, $|\mathcal{P}| \le N'_i$.*

The proof is again by induction. For the base case of $i = 0$, $N'_i = 1$, and the corresponding level-0 instance contains a single demand pair, so the theorem clearly holds. We now assume that the theorem holds for some value $i \ge 0$ and prove it for a level-$(i+1)$ instance $\mathcal{I}$. We assume that we are given some instantiation of $\mathcal{I}$, and from now on our goal is to prove that no solution to this instance of NDP can route more than $N'_{i+1} = (1-\delta)nc_{i+1} \cdot (200h/3+1)N'_i$ demand pairs, where $\delta = 8\epsilon^2/10^{12}$. We assume for contradiction that this is not the case, and we let $\tilde{\mathcal{P}}$ be a collection of more than N'_{i+1} node-disjoint paths, routing a set $\tilde{\mathcal{M}} \subseteq \mathcal{M}$ of demand pairs.

For every demand pair $(s,t) \in \tilde{\mathcal{M}}$, we let $P(s,t) \in \tilde{\mathcal{P}}$ be the path routing this pair in the solution.

Recall that our construction of the level-$(i+1)$ instance $\mathcal{I}$ consists of a number of copies of c_{i+1}-wide level-i instances: For every variable x of φ, we have constructed $(70h+2)$ such instances ($60h$ instances for the extra pairs, and $(5h+1)$ instances each for True and False pairs); for every clause $C \in \mathcal{C}$, we have constructed $3h$ such instances. Therefore, overall we use $(70h+2)n + 3hm = 75nh + 2n$ copies of c_{i+1}-wide level-i instances (we have used the fact that $m = 5n/3$). We assume (by induction) that at most $c_{i+1}N'_i$ pairs from each such instance are in $\tilde{\mathcal{M}}$. We say that a c_{i+1}-wide level-i instance $\mathcal{I}'$ is *interesting* iff at least $25H_i$ demand pairs of $\mathcal{M}(\mathcal{I}')$ belong to $\tilde{\mathcal{M}}$; otherwise we say that it is uninteresting. We let $\hat{\mathcal{M}} \subseteq \tilde{\mathcal{M}}$ be the set of all demand pairs that belong to uninteresting instances, and we call them *excess pairs*. We need the following simple observation, whose proof is omitted due to lack of space.

Observation 6.2. $|\hat{\mathcal{M}}| \le \delta(200h/3+1)nc_{i+1}N'_i$.

It would be convenient for us to assume that no excess pairs exist. In order to do so, we discard all excess pairs from $\tilde{\mathcal{M}}$. From Observation 6.2, $|\tilde{\mathcal{M}}| \ge (1-2\delta)nc_{i+1}(200h/3+1)N'_i$ still holds.

For every variable x, we let $\tilde{\mathcal{M}}(x) = \tilde{\mathcal{M}} \cap \mathcal{M}(x)$, and for every clause C_q, we let $\tilde{\mathcal{M}}(C_q) = \tilde{\mathcal{M}} \cap \mathcal{M}(C_q)$. We also denote by $\tilde{\mathcal{M}}^V = \bigcup_{j=1}^n \tilde{\mathcal{M}}(x_j)$ and by $\tilde{\mathcal{M}}^C = \bigcup_{q=1}^m \tilde{\mathcal{M}}(C_q)$.

For the sake of the No-Instance-analysis, it is convenient to view our construction slightly differently. Let φ be the input 3SAT(5) formula. Recall that $\mathcal{C} = \{C_1,\ldots,C_m\}$ is the set of its clauses. For each clause $C_q \in \mathcal{C}$, we create h new clauses $C_q^1,\ldots,C_q^h$, each of which is a copy of the original clause. We let $\mathcal{C}' = \left\{C_q^j \mid 1 \le q \le m, 1 \le j \le h\right\}$ be the resulting set of clauses, and φ' the corresponding 3SAT formula. In order to avoid confusion, we refer to the clauses in $\mathcal{C}$ as the *original* clauses, to the clauses of $\mathcal{C}'$ as the *new clauses*, and for each $1 \le q \le m$, $1 \le j \le h$, we call C_q^j *the* j*th copy of clause* C_q. Recall that the clause gadget for $C_q \in \mathcal{C}$ contains h boxes $B^1(C_q),\ldots,B^h(C_q)$, where box $B^j(C_q)$ is the union of three boxes: $B(\mathcal{I}_j(C_q,\ell_{q_1}))$, $B(\mathcal{I}_j(C_q,\ell_{q_2}))$ and $B(\mathcal{I}_j(C_q,\ell_{q_3}))$ (see Figure 10). We think of the box $B^j(C_q)$ as representing the new clause $C_q^j \in \mathcal{C}'$. For convenience, we denote by $\tilde{\mathcal{M}}(C_q^j) \subseteq \tilde{\mathcal{M}}(C_q)$ the set of all demand pairs routed by our solution whose destinations lie in $B^j(C_q)$. This set is further partitioned into three subsets, $\tilde{\mathcal{M}}(C_q^j,\ell_{q_1})$, $\tilde{\mathcal{M}}(C_q^j,\ell_{q_2})$, $\tilde{\mathcal{M}}(C_q^j,\ell_{q_3})$, each of which contains demand pairs from the instances $\mathcal{I}_j(C_q,\ell_{q_1})$, $\mathcal{I}_j(C_q,\ell_{q_2})$, and $\mathcal{I}_j(C_q,\ell_{q_3})$ respectively. The following observation is immediate:

Observation 6.3. *If φ is a* No-Instance, *then for any assignment to its variables, at most $(1-\epsilon)mh$ clauses of $\mathcal{C}'$ are satisfied.*

Encircling and its Resolution. Let $(s,t) \in \tilde{\mathcal{M}}$ be any demand pair routed by the solution. Recall that (s,t) belongs to some level-i instance $\mathcal{I}'$, and we have defined a line Q_t containing at most $H_i/2$ vertices of the graph, that connects t to the bottom of the box $B(\mathcal{I}')$ (which is a cut-out box). We say that a demand pair $(s',t') \in \tilde{\mathcal{M}}$ *encircles* pair (s,t) iff path $P(s',t')$ contains a vertex lying on Q_t. Since Q_t contains at most $H_i/2$ vertices, at most $H_i/2$ demand pairs may encircle (s,t). We repeatedly use the following simple lemma, whose proof is omitted due to lack of space.

Lemma 6.4. *Let $S_1,\ldots,S_r$ be a collection of disjoint subsets of $\tilde{\mathcal{M}}$, such that for all $1 \le j \le r$, $|S_j| \ge r^2H_i/2$. Then there is a collection $\mathcal{M}' = \{(s_1,t_1),\ldots,(s_r,t_r)\}$ of demand pairs, such that for all $1 \le j \le r$, $(s_j,t_j) \in S_j$, and for all distinct $(s,t),(s',t') \in \mathcal{M}'$, pair (s',t') does not encircle pair (s,t).*

Variable Gadget Analysis. We fix some variable x and consider its corresponding gadget. We start with the following simple lemma, whose proof relies on the proof of Lemma 6.4, and is omitted due to lack of space.

Lemma 6.5. *Let $\tilde{\mathcal{M}}_X = \tilde{\mathcal{M}} \cap \mathcal{M}^X(x)$, $\tilde{\mathcal{M}}_T = \tilde{\mathcal{M}} \cap \mathcal{M}^T(x)$, and $\tilde{\mathcal{M}}_F = \tilde{\mathcal{M}} \cap \mathcal{M}^F(x)$ be the subsets of $\mathcal{M}^X(x)$, $\mathcal{M}^T(x)$, and $\mathcal{M}^F(x)$, respectively, that are routed by our solution. Then at least one of the sets $\tilde{\mathcal{M}}_X, \tilde{\mathcal{M}}_T, \tilde{\mathcal{M}}_F$ is empty.*

The following corollary is now immediate.

Corollary 6.6. $|\tilde{\mathcal{M}}^V| \le (65h+1)\cdot n \cdot c_{i+1}N'_i$.

Consider some variable x of φ. If $\tilde{\mathcal{M}} \cap \mathcal{M}^T(x) = \emptyset$, then we assign it the value False, and otherwise we assign it the value True.

Fix some variable $x \in X$ and some index $1 \le j \le 5h+1$. We say that index j is bad for variable x if either (i) x is assigned the value True, and instance $\mathcal{I}_j^T(x)$ is uninteresting; or (ii) x is assigned the value False, and instance $\mathcal{I}_j^F(x)$ is uninteresting.

Clause Gadget Analysis. Consider a new clause $C_q^j \in \mathcal{C}'$ and its three literals $\ell_{q_1},\ell_{q_2},\ell_{q_3}$. We say that clause C_q^j is a *troublesome clause*, or a *troublesome copy of* C_q, iff there are at least two values $1 \le z < z' \le 3$, for which instances $\mathcal{I}_j(C_q,\ell_{q_z})$, $\mathcal{I}_j(C_q,\ell_{q_{z'}})$ are both interesting. The proofs of the following lemma and its corollary are omitted due to lack of space.

Lemma 6.7. *For every original clause $C_q \in \mathcal{C}$, at most three of its copies are troublesome.*

Corollary 6.8. *For each original clause $C_q \in \mathcal{C}$, $|\tilde{\mathcal{M}}(C_q)| \le (6+h)N'_ic_{i+1}$, and overall $|\tilde{\mathcal{M}}^C| \le 5n(6+h)N'_ic_{i+1}/3$.*

In the rest of our proof, we will reach a contradiction by proving that the current assignment to the variables of φ satisfies more than $(1-\epsilon)hm$ clauses in $\mathcal{C}'$. In order to do so, we gradually discard clauses from $\mathcal{C}'$, until we obtain a large enough subset of clauses that is guaranteed to be satisfied by the current assignment.

Our first step is to define uninteresting clauses. Recall that for each new clause $C_q^j \in \mathcal{C}'$, there are three corresponding c_{i+1}-wide level-i instances, $\mathcal{I}_j(C_q,\ell_{q_1})$, $\mathcal{I}_j(C_q,\ell_{q_2})$, and $\mathcal{I}_j(C_q,\ell_{q_3})$. We say that clause C_q^j is interesting iff at least one of these three instances is interesting, and we say that it is uninteresting otherwise. Let $\mathcal{C}'_0 \subseteq \mathcal{C}'$ be the set of all uninteresting clauses. The proof of the following claim is omitted due to lack of space.

Claim 6.9. $|C'_0| \le 12n$.

Consider some clause $C^j_q \in C'$ that is interesting. Then there is an index $z \in \{1,2,3\}$, such that instance $\mathcal{I}_j(C_q, \ell_{q_z})$ is interesting. If there are several such indices z (if C^j_q is troublesome), then we choose one of them arbitrarily. We say that clause C^j_q *chooses* the literal ℓ_{q_z}. We say that C^j_q is a *cheating clause* iff the variable x_{q_z} corresponding to literal ℓ_{q_z} is assigned the opposite value: In other words, if $\ell_{q_z} = x_{q_z}$, then x is assigned the value False, and otherwise, $\ell_{q_z} = \neg x_{q_z}$, and x_{q_z} is assigned the value True. We further say that it is a *bad cheating clause* iff at least one of the indices $j, j+1$ is bad for the variable x_{q_z}, and we say that it is a *good cheating clause* otherwise. Let $C'_1 \subseteq C' \setminus C'_0$ be the set of all the cheating clauses. In the following lemma, whose proof is omitted due to lack of space, we bound the number of the cheating clauses.

Lemma 6.10. *There are at most* $24n$ *bad cheating clauses, and at most* $3m$ *good cheating clauses.*

Notice that if clause C^j_q is an interesting non-cheating clause, then the current assignment must satisfy it. From Claim 6.9 and Lemma 6.10, there are at least $hm - 12n - 24n - 3m = hm - 123m/5 = (1 - 123\epsilon/5000)hm > (1-\epsilon)hm$ such clauses, contradicting Observation 6.3.

7 HARDNESS OF ROUTING ON SUB-CUBIC PLANAR GRAPHS

In this section we prove Theorem 1.2, and show that Theorem 1.1 holds for sub-cubic planar graphs. We start with proving Theorem 1.2. Let $G = G^{\ell,h}$ be a grid of length ℓ and height h, where $\ell > 0$ is an even integer, and $h > 0$. For every column W_j of the grid, let $e^j_1, \ldots, e^j_{h-1}$ be the edges of W_j indexed in their top-to-bottom order. Let $E^*(G) \subseteq E(G)$ contain all edges e^j_z, where $z \ne j \mod 2$, and let $\hat{G}$ be the graph obtained from $G \setminus E^*(G)$, by deleting all degree-1 vertices from it. The resulting graph is called a *wall of length* $\ell/2$ *and height* h (see Figure 11). Consider the subgraph of $\hat{G}$ induced by all horizontal edges of the grid G that belong to $\hat{G}$. This graph is a collection of h node-disjoint paths, that we refer to as the *rows* of $\hat{G}$, and denote them by $R_1, \ldots, R_h$ in this top-to-bottom order; notice that R_j is a sub-path of the jth row of G for all j. Graph $\hat{G}$ contains a unique collection $\mathcal{W}$ of $\ell/2$ node-disjoint paths that connect vertices of R_1 to vertices of R_h and are internally disjoint from R_1 and R_h. We refer to the paths in $\mathcal{W}$ as the *columns* of $\hat{G}$, and denote them by $W_1, \ldots, W_{\ell/2}$ in this left-to-right order. Paths $W_1, W_{\ell/2}, R_1$ and R_h are called the left, right, top and bottom boundary edges of $\hat{G}$, respectively, and the union of these paths is the boundary of $\hat{G}$. Given a wall $\hat{G}$, a consecutive subset $\mathcal{R}'$ of its rows, and a consecutive subset $\mathcal{W}'$ of its columns, the sub-wall of $\hat{G}$ spanned by the rows of $\mathcal{R}'$ and the columns of $\mathcal{W}'$ is the subgraph of $\hat{G}$ induced by the set $\{v \mid \exists R \in \mathcal{R}', W \in \mathcal{W}' : v \in R \cap W\}$ of vertices. The first and the last columns of $\mathcal{W}'$ serve as the left and the right boundary edges of the sub-wall, and the top and the bottom rows of $\mathcal{R}'$ serve as its top and bottom boundary edges.

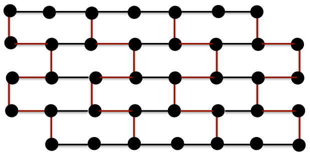

Figure 11: A wall of height 5 and length 4; the columns of the wall are shown in red.

We perform a reduction from the 3SAT(5) problem. Assume that we are given an instance φ of 3SAT(5) on n variables and $m = 5n/3$ clauses. As before, our construction has $\Theta(\log n)$ levels. For every level $i \ge 0$, we define a family of instances of EDP. In order to construct a level-i instance $\mathcal{I}$, we define the parameters H_i, L_i and L'_i exactly as before, a path $Z(\mathcal{I})$ and a box $B(\mathcal{I})$ (which is a subgraph of the grid G'_i of length L'_i and height H_i), a collection $\mathcal{M}$ of demand pairs, and the mappings of the vertices of $S(\mathcal{M})$ to the vertices of $Z(\mathcal{I})$, and of the vertices of $T(\mathcal{M})$ to distinct vertices of the middle row of $B(\mathcal{I})$ exactly as before. In order to instantiate this instance, we select an arbitrary grid G_i of length $\ell \ge 2L_i + 2L'_i + 4H_i$, where ℓ is an even integer, and height $h \ge 3H_i$, map the vertices of $Z(\mathcal{I})$ to the vertices of the first row of G_i, and map the vertices of $B(\mathcal{I})$ to the vertices of a sub-grid G''_i of G_i exactly as before, obtaining an instantiation $(G, \mathcal{M})$ of the level-i instance $\mathcal{I}$ of NDP. Our final step is to delete from G every edge of $E^*(G_i) \cap E(G)$, and then to delete all vertices that have degree 1 in the resulting graph. We also delete every other edge on the top row of G_i, and all horizontal edges that are incident to the vertices of $T(\mathcal{M})$, to ensure that the degree of every terminal is at most 2. The final graph, denoted by $\hat{G}$, is a subgraph of a wall of length $\ell/2$ and height h. We denote by $\hat{B}(\mathcal{I})$ the intersection of the image of $B(\mathcal{I})$ in G_i and the graph $\hat{G}$. This concludes the definition of the reduction. Since the resulting graph $\hat{G}$ is a subgraph of G, the following observation is immediate.

Observation 7.1. *If* φ *is a* No-Instance*, then for every level i, for every instantiation* $(G, \mathcal{M})$ *of the level-i instance* $\mathcal{I}$ *of* NDP*, every solution to the corresponding instance* $(\hat{G}, \mathcal{M})$ *of* EDP *has value at most* N'_i.

It is now enough to show that, if φ is a Yes-Instance, then for every level i, for every instantiation $(G, \mathcal{M})$ of the level-i instance $\mathcal{I}$ of NDP, the corresponding instance $(\hat{G}, \mathcal{M})$ of EDP has a solution of value at least $N_i/2$. Before we do so, we need several definitions.

Suppose we are given some set $\mathcal{P}$ of node-disjoint paths in some wall $\hat{H}$, and assume that every path in $\mathcal{P}$ connects some vertex on a row R of $\hat{H}$ to a vertex on a row R' of $\hat{H}$, where $R \ne R'$.

A subset U of the vertices lying on a row R of a wall $\hat{H}$ is called *well-spread* iff U does not contain a pair of vertices connected by an edge in $\hat{H}$. Notice that if U is well-spread, then no two vertices of U may lie on the same column of $\hat{H}$.

We now define an analogue of box-respecting paths. Consider some level-i instance $\mathcal{I}$ of NDP, for $i \ge 0$, and an instantiation $(G, \mathcal{M})$ of this instance. Let $(\hat{G}, \mathcal{M})$ be the corresponding instance of EDP, and let $\mathcal{P}$ be a set of node-disjoint paths routing some subset of the demand pairs in $\hat{G}$. Let A be the top boundary of $\hat{B}(\mathcal{I})$. We say that set $\mathcal{P}$ of paths is *canonical with respect to the box* $\hat{B}(\mathcal{I})$ iff for every path $P \in \mathcal{P}$, $P \cap A$ is a single edge, and the following holds. Denote $\mathcal{P} = (P_1, \dots, P_r)$, and denote, for every path P_i, its source vertex by s_i, and the unique edge of $P_i \cap A$ by e_i, such that $s_1, \dots, s_r$ appear on the top row of $\hat{G}$ in this left-to-right order. Then the edges of $e_1, \dots, e_r$ must appear in this left-to-right order on A, and for each $1 \leq j \leq r$, e_j is the $(2j)$th edge of A from the left.

Assume that φ is a YES-INSTANCE. Recall that for each $i \geq 0$, for every level-i instance $\mathcal{I}$ of NDP, we have defined a collection $\mathcal{M}^*(\mathcal{I})$ of demand pairs, such that for every instantiation of $\mathcal{I}$, there is a set $\mathcal{P}$ of node-disjoint paths, that respect the box $B(\mathcal{I})$, and route the set $\mathcal{M}^*(\mathcal{I})$ of demand pairs. Denote the pairs in $\mathcal{M}^*(\mathcal{I})$ by $(s_1, t_1), \dots, (s_p, t_p)$, and assume that the vertices $s_1, \dots, s_p$ appear in this left-to-right order on $Z(\mathcal{I})$. We partition $\mathcal{M}^*(\mathcal{I})$ into two subsets: set $\mathcal{M}_1^*(\mathcal{I})$ contains all demand pairs (s_j, t_j) where j is odd, and set $\mathcal{M}_2^*(\mathcal{I})$ contains all remaining demand pairs. Notice that each of the sets $S(\mathcal{M}_1^*(\mathcal{I})), S(\mathcal{M}_2^*(\mathcal{I}))$ is well-spread, for any instantiation of $\mathcal{I}$ and its corresponding instance of EDP. The proof of the following lemma appears in the full version of the paper, and it completes the proof of Theorem 1.2.

LEMMA 7.2. *For all $i \geq 0$, for every level-i instance $\mathcal{I}$ of* NDP*, for every instance $(\hat{G}, \mathcal{M})$ of* EDP *corresponding to an instantiation $(G, \mathcal{M})$ of $\mathcal{I}$, there is a set of node-disjoint paths in $\hat{G}$ that route all demand pairs in $\mathcal{M}_1^*(\mathcal{I})$, such that the paths are canonical with respect to $\hat{B}(\mathcal{I})$, and the same holds for $\mathcal{M}_2^*(\mathcal{I})$.*

Hardness of NDP *on Sub-Cubic Planar Graphs.* Consider the instances of EDP constructed above. Each such instance is defined on a sub-cubic planar graph, where the degree of every terminal is at most 2. It is easy to see that, if we are given a graph G with the above properties, and any set $\mathcal{P}$ of paths whose endpoints are distinct terminals, the paths in $\mathcal{P}$ are mutually edge-disjoint iff they are mutually node-disjoint. Therefore, the number of the demand pairs that can be routed in the YES-INSTANCE and the NO-INSTANCE via node-disjoint paths remains the same as for edge-disjoint paths. This completes the proof of Theorem 1.1.

REFERENCES

[1] Matthew Andrews. 2010. Approximation Algorithms for the Edge-Disjoint Paths Problem via Raecke Decompositions. In *Proceedings of IEEE FOCS*. 277–286. DOI: https://doi.org/10.1109/FOCS.2010.33

[2] Matthew Andrews, Julia Chuzhoy, Venkatesan Guruswami, Sanjeev Khanna, Kunal Talwar, and Lisa Zhang. 2010. Inapproximability of Edge-Disjoint Paths and low congestion routing on undirected graphs. *Combinatorica* 30, 5 (2010), 485–520.

[3] Matthew Andrews and Lisa Zhang. 2005. Hardness of the undirected edge-disjoint paths problem. In *STOC*. ACM, 276–283.

[4] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. 1998. Proof verification and the hardness of approximation problems. *Journal of the ACM (JACM)* 45, 3 (1998), 501–555.

[5] Sanjeev Arora and Shmuel Safra. 1998. Probabilistic checking of proofs: A new characterization of NP. *Journal of the ACM (JACM)* 45, 1 (1998), 70–122.

[6] Yonatan Aumann and Yuval Rabani. 1995. Improved bounds for all optical routing. In *Proceedings of the sixth annual ACM-SIAM symposium on Discrete algorithms (SODA '95)*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 567–576. http://dl.acm.org/citation.cfm?id=313651.313820

[7] Chandra Chekuri and Julia Chuzhoy. 2016. Half-Integral All-or-Nothing Flow. (2016). Personal Communication.

[8] Chandra Chekuri and Alina Ene. 2013. Poly-logarithmic Approximation for Maximum Node Disjoint Paths with Constant Congestion. In *Proc. of ACM-SIAM SODA*.

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

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

[11] Chandra Chekuri, Sanjeev Khanna, and F. Bruce Shepherd. 2006. An $O(\sqrt{n})$ Approximation and Integrality Gap for Disjoint Paths and Unsplittable Flow. *Theory of Computing* 2, 1 (2006), 137–146.

[12] Julia Chuzhoy. 2016. Routing in Undirected Graphs with Constant Congestion. *SIAM J. Comput.* 45, 4 (2016), 1490–1532. DOI: https://doi.org/10.1137/130910464

[13] Julia Chuzhoy and David H. K. Kim. 2015. On Approximating Node-Disjoint Paths in Grids. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2015, August 24-26, 2015, Princeton, NJ, USA (LIPIcs)*, Naveen Garg, Klaus Jansen, Anup Rao, and José D. P. Rolim (Eds.), Vol. 40. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 187–211. DOI: https://doi.org/10.4230/LIPIcs.APPROX-RANDOM.2015.187

[14] Julia Chuzhoy, David H. K. Kim, and Shi Li. 2016. Improved Approximation for Node-disjoint Paths in Planar Graphs. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing (STOC 2016)*. ACM, New York, NY, USA, 556–569. DOI: https://doi.org/10.1145/2897518.2897538

[15] Julia Chuzhoy and Shi Li. 2016. A Polylogarithmic Approximation Algorithm for Edge-Disjoint Paths with Congestion 2. *J. ACM* 63, 5 (2016), 45:1–45:51. http://dl.acm.org/citation.cfm?id=2893472

[16] Shimon Even, Alon Itai, and Adi Shamir. 1976. On the Complexity of Timetable and Multicommodity Flow Problems. *SIAM J. Comput.* 5, 4 (1976), 691–703. DOI: https://doi.org/10.1137/0205048

[17] R. Karp. 1972. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, R. Miller and J. Thatcher (Eds.). Plenum Press, 85–103.

[18] Ken-Ichi Kawarabayashi and Yusuke Kobayashi. 2013. An O(log n)-Approximation Algorithm for the Edge-Disjoint Paths Problem in Eulerian Planar Graphs. *ACM Trans. Algorithms* 9, 2, Article 16 (March 2013), 13 pages. DOI: https://doi.org/10.1145/2438645.2438648

[19] Jon Kleinberg. 2005. 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)*. IEEE Computer Society, Washington, DC, USA, 627–636. DOI: https://doi.org/10.1109/SFCS.2005.18

[20] Jon M. Kleinberg and Éva Tardos. 1995. Disjoint Paths in Densely Embedded Graphs. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science*. 52–61.

[21] Jon M. Kleinberg and Éva Tardos. 1998. Approximations for the Disjoint Paths Problem in High-Diameter Planar Networks. *J. Comput. Syst. Sci.* 57, 1 (1998), 61–73.

[22] Stavros G. Kolliopoulos and Clifford Stein. 2004. Approximating disjoint-path problems using packing integer programs. *Mathematical Programming* 99 (2004), 63–87. DOI: https://doi.org/10.1007/s10107-002-0370-6

[23] MR Kramer and Jan van Leeuwen. 1984. The complexity of wire-routing and finding minimum area layouts for arbitrary VLSI circuits. *Advances in computing research* 2 (1984), 129–146.

[24] James F. Lynch. 1975. The Equivalence of Theorem Proving and the Interconnection Problem. *SIGDA Newsl.* 5, 3 (Sept. 1975), 31–36. DOI: https://doi.org/10.1145/1061425.1061430

[25] Harald Räcke. 2002. Minimizing Congestion in General Networks. In *Proc. of IEEE FOCS*. 43–52.

[26] Prabhakar Raghavan and Clark D. Tompson. 1987. Randomized rounding: a technique for provably good algorithms and algorithmic proofs. *Combinatorica* 7 (December 1987), 365–374. Issue 4. DOI: https://doi.org/10.1007/BF02579324

[27] Satish Rao and Shuheng Zhou. 2010. Edge Disjoint Paths in Moderately Connected Graphs. *SIAM J. Comput.* 39, 5 (2010), 1856–1887.

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

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

[30] Loïc Seguin-Charbonneau and F. Bruce Shepherd. 2011. 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)*. IEEE Computer Society, Washington, DC, USA, 200–209. DOI: https://doi.org/10.1109/FOCS.2011.30