THE UNIVERSITY OF CHICAGO

APPROXIMATION ALGORITHMS AND HARDNESS OF ROUTING ON DISJOINT PATHS

A DISSERTATION SUBMITTED TO
THE FACULTY OF THE DIVISION OF THE PHYSICAL SCIENCES
IN CANDIDACY FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE

BY

HONG KYUN KIM

CHICAGO, ILLINOIS

JUNE 2018

To my family

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

I am sincerely grateful to my principal advisor, Julia Chuzhoy, for guiding me through my graduate studies with kind patience and for introducing me to the area of approximation algorithms and graph routing. I feel extremely fortunate to have had her as my advisor. Her passion, brilliance, and enthusiasm in research were immense source of motivation and encouragement during the past years. I am thankful for all the discussions and advices she gave me throughout, not only in research but life in general. The way of thinking about problems from various aspects, constantly learning new ideas and techniques, and persistently tackling difficult problems in the long term are invaluable lessons that I still wish to learn more from her.

I am also extremely thankful to my adivsor, Laci Babai, for his help during my graduate studies. He introduced me to many areas of theory in the first few years that I was searching for a research topic. He actively encouraged me to read papers and talk to others. I was able to discuss any topic and problem with him, and his advice and encouragements have helped me immensely many times.

I am also sincerely grateful to Hank Hoffmann, who has been a mentor and supervisor during my time here. He helped me develop a small research topic in his computer architecture class to a paper, which then led me to explore a whole new area in scheduling algorithms. He is never afraid to explore new areas and challenge conventional thought, and he encouraged me to do the same. I feel extremely lucky to have met Hank during my graduate studies.

I would like to thank Janos Simon for kindly being on the committee for all of my master's thesis, candidacy exam, and dissertation defense. I would like to thank all of my co-authors, especially Rachit Nimavat, who was always helpful and available for discussions. I also want to thank all of my friends in the department and at TTI-C, especially Gokalp Demirci and Liwen Zhang, who were always there for random coffee breaks and made my time in Chicago enjoyable.

Lastly, I would like to thank my beloved parents and my sister. Their unconditional love, support, and trust gave me courage and strength in my times of difficulty, and I am forever grateful.

ABSTRACT

In the classical node-disjoint paths problem, one is given as input an n-vertex graph G and a collection $\mathcal{M} = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ of pairs of vertices of G called *source-destination* or *demand pairs*, and the goal is to find a maximum-cardinality set of pairwise vertex-disjoint paths connecting the demand pairs. While the node-disjoint paths problem is one of the most basic routing problems, there has been a wide gap in our understanding of its approximability status. Currently, the best upper bound of $O(\sqrt{n})$ is achieved by a simple, greedy algorithm, while until very recently, the best known hardness of approximation result was a $\Omega(\log^{1/2-\delta} n)$ lower bound for any constant δ, under standard complexity assumptions. Even for special cases of the problem, including when the input graph G is constrained to be planar, and surprisingly, even when G is a square grid, no better upper bound was known, while only NP-hardness was known on the negative side.

This thesis studies the approximability of the node-disjoint paths problem and three of its special cases. The first part of the thesis presents approximation algorithms for the special cases of the problem. In the first chapter, we investigate when the input graph is constrained to be a square grid. We show an $\tilde{O}(n^{1/4})$-approximation algorithm for this case and extend the approximation algorithm to show that the same upper bound holds for the closely related edge-disjoint paths problem in wall graphs.

In the second chapter, we focus on two cases of node-disjoint paths when the input graph G is planar. In the first case, we assume that we are given a planar graph G embedded into a disc, where all of the terminals participating in the demand pairs lie on the boundary of the disc. In the second case, we assume that we are given a cylinder obtained by removing two disjoint, open discs from it. We assume that G can be embedded into the cylinder such that all source terminals lie on the boundary of one of the discs and all destination terminals on the boundary of the other. We present an $O(\log k)$-approximation algorithm for both problems of routing node-disjoint paths on a disc and a cylinder.

The third chapter of the thesis presents an improved inapproximability result for the node-disjoint paths problem which holds for planar graphs. We show that the problem is $2^{\Omega(\sqrt{\log n})}$-hard to approximate, unless all problems in NP have deterministic, quasi-polynomial time algorithms. The result holds even when the underlying graph is a vertex-induced sub-graph of a grid with all sources of the demand pairs lying on the boundary of the outer face.

CHAPTER 1

INTRODUCTION

The main focus of this thesis 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, find a maximum-cardinality set of paths $\mathcal{P}$ connecting the demand pairs that are pairwise node-disjoint. More specifically, a path $P \in \mathcal{P}$ is said to *route* a demand pair (s_i, t_i) iff the endpoints of P are s_i and t_i. The goal of the problem is to compute the largest subset $\mathcal{M}' \subseteq \mathcal{M}$ and a set of paths $\mathcal{P}$ routing the demand pairs in $\mathcal{M}'$, such that no two paths in $\mathcal{P}$ intersect at a vertex.

NDP is a fundamental graph routing problem that has been extensively studied in different contexts. The study of NDP-Grid, A special case of NDP where the underlying graph G is a square grid, dates back to the 1970's, initially motivated by VLSI design. Other technical problems studied in this context included obtaining upper and lower bounds on the area required for routing the demand pairs [22, 1] and finding disjoint paths connecting the demand pairs given a prescribed homotopy, or a rough routing of the paths connecting the terminals [21].

NDP has also played an important role in the famous Graph Minors Series of Robertson and Seymour [38, 41]. They showed an efficient algorithm for solving the problem if the number k of the demand pairs is bounded by a constant. More specifically, they showed that NDP can be solved in time $O(f(k)n^3)$ for some function f depending on k, and currently, the best running time is obtained by an $O(f(k)n^2)$-time algorithm [26]. However, when k is part of the input, the problem becomes NP-hard [24, 23], even on planar graphs [33] and grid graphs [31].

The approximability status of NDP has had a wide gap in the best known upper and lower bounds, until very recently. Currently, the best known approximation factor for NDP is

$O(\sqrt{n})$, obtained by a simple, greedy approximation algorithm [30]: while G contains any path connecting a demand pair, choose the shortest such path P, add P to the solution, and delete all vertices of P from G. Surprisingly, the upper bound of $O(\sqrt{n})$ achieved by this simple, greedy algorithm was the best known even for NDP-Planar, the special case of NDP when the input graph is a planar graph, and even for NDP-Grid.

One of the difficulties to obtaining better than an $O(\sqrt{n})$-approximation for NDP, even for NDP-Planar and NDP-Grid, is that the integrality gap of the standard multi-commodity flow LP-relaxation for NDP is $\Omega(\sqrt{n})$, even in grid graphs. In the multi-commodity flow LP-relaxation, one is allowed to send fractional flows between the demand pairs, where at most a unit of flow is sent per demand pair, under the constraint that the total flow carried at each vertex is at most 1. The objective of the LP-relaxation is to maximize the total flow sent between all of the demand pairs. It turns out that the simple, greedy algorithm above can be cast as an LP-rounding algorithm of the multi-commodity flow LP-relaxation, implying that the $O(\sqrt{n})$ approximation factor of the algorithm is tight, even for NDP-Grid.

Until recently, the best known lower bound for NDP was an $\Omega(\log^{1/2-\epsilon} n)$-hardness of approximation result, for any constant ϵ, unless $\mathsf{NP} \subseteq \mathsf{ZPTIME}(n^{\mathrm{polylog(n)}})$ [6, 5]. However, the hardness result did not hold for NDP-Planar and NDP-Grid, and it was only known that both problems are NP-hard [33, 31].

NDP-Planar and NDP-Grid now have better approximation algorithms beating the $\Omega(\sqrt{n})$ integrality gap. An $\tilde{O}(n^{1/4})$-approximation algorithm was presented in joint work with Chuzhoy [16], and the full approximation algorithm is also presented in the first chapter of this thesis.

Subsequently, an $\tilde{O}(n^{9/19})$-approximation algorithm was shown for NDP-Planar in joint work with Chuzhoy and Li [17]. At a very high level, the $\tilde{O}(n^{9/19})$-approximation algorithm for NDP-Planar defines a new LP-relaxation for the problem and employs the ellipsoid algorithm. The approximation algorithm either returns a good LP-rounding solution, or reduces the problem to an instance of NDP-Disc or NDP-Cylinder. In the latter case, the approximation algorithm computes an $O(\log k)$-approximate solution to the optimal integral solution of the NDP-Disc or NDP-Cylinder instance. Since the approximate solution does not depend on the solution to the LP-relaxation, the algorithm is able to use it in serving as a separation oracle for the new LP-relaxation. The approximation algorithms for NDP-Disc and NDP-Cylinder from [17] are presented in the second chapter of this thesis.

Significantly improved inapproximability results for NDP have been proven in recent years. In addition to the $\tilde{O}(n^{1/4})$-approximation for NDP-Grid, it was also shown in the same paper that NDP-Grid is APX-hard [16]. In joint work with Chuzhoy and Nimavat [19], it was recently shown that NDP is hard to approximate to within a $2^{\Omega(\sqrt{\log n})}$ factor, unless $\mathsf{NP} \subseteq \mathsf{DTIME}(n^{O(\log n)})$, even if the underlying graph is a planar graph with maximum vertex degree at most 3, and all source vertices lie on the boundary of a single face. This result holds even when the input graph G is a vertex-induced subgraph of a grid, with all sources lying on the outer face. This thesis presents the hardness proof for when the instance is a sub-graph of a grid with maximum vertex degree at most 4, with all source vertices lying on the boundary of the outer face. Even more recently, it was shown that NDP-Grid is $2^{\Omega(\log^{1-\epsilon} n)}$-hard to approximate for any constant ϵ unless $\mathsf{NP} \subseteq \mathsf{RTIME}(n^{\text{poly}\log n})$, and $n^{\Omega(1/(\log\log n)^2)}$-hard unless $\mathsf{NP} \subseteq \mathsf{RTIME}(2^{n^\delta})$ for some constant $\delta > 0$ [18].

A problem closely related to NDP is the Edge-Disjoint Paths (EDP) problem. The problem is very similar to NDP, except that the paths in the solution are allowed to share vertices as long as they are pairwise edge-disjoint. More precisely, in EDP, one is given as input an undirected n-vertex graph G, together with $\mathcal{M} = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ of demand pairs. The goal is to find $\mathcal{M}' \subseteq \mathcal{M}$ and a set of paths $\mathcal{P}$ routing the demand pairs in $\mathcal{M}'$, such that no two paths in $\mathcal{P}$ share an edge. In general, EDP can be seen as a special case of NDP: given an EDP instance $(G, \mathcal{M})$, computing the line graph of the input graph G transforms it into an equivalent instance of NDP. However, the transformation may inflate the number of the graph vertices, and so approximation factors that depend on n may no longer be preserved.

Moreover, this transformation does not preserve planarity, and no such relationship is known between NDP and EDP in planar graphs.

The approximation status of EDP is similar to that of NDP. The best upper bound on the approximation factor is $O(\sqrt{n})$ [14]. Until recently, the best lower bound for EDP has also been an $\Omega(\log^{1/2-\epsilon} n)$-hardness of approximation result, for any constant ϵ, unless NP $\subseteq$ ZPTIME$(n^{\text{polylog(n)}})$ [6, 5]. However, unlike NDP, there are constant-factor approximation for the special case of EDP on grid graphs [9, 29, 28]. The analogue of grids for EDP is the wall graph, where the integrality gap of the standard multicommodity flow LP-relaxation is $\Omega(\sqrt{n})$. We let EDP-Wall denote the special case of EDP where the input graph is restricted to walls. Until recently, the best approximation factor achieved by an efficient algorithm for EDP-Wall has also been $O(\sqrt{n})$. In this thesis, we extend the $\tilde{O}(n^{1/4})$-approximation algorithm for NDP-Grid to EDP-Wall. However, the best upper bound known for EDP on planar graphs still remains $O(\sqrt{n})$.

The recent hardness of approximation results for NDP also hold for EDP. EDP-Wall was shown to be APX-hard in [16]. In joint work with Chuzhoy and Nimavat [19], it was shown that EDP is hard to approximate to within a $2^{\Omega(\sqrt{\log n})}$ factor, unless NP $\subseteq$ DTIME$(n^{O(\log n)})$, even if the underlying graph is a subgraph of a wall graph, with all sources lying on the wall boundary. The recent, almost-polynomial hardness of approximation results for NDP-Grid also hold for EDP-Wall. That is, EDP-Wall is $2^{\Omega(\log^{1-\epsilon} n)}$-hard to approximate for any constant ϵ unless NP $\subseteq$ RTIME$(n^{\text{poly}\log n})$, and $n^{\Omega(1/(\log\log n)^2)}$-hard unless NP $\subseteq$ RTIME(2^{n^δ}) for some constant $\delta > 0$ [18].

We note that many special cases of EDP have been studied extensively, and better approximation algorithms are known for them: Kleinberg and Tardos [29, 28] showed $O(\log n)$-approximation algorithms for nearly-Eulerian uniformly high-diameter planar graphs, and nearly-Eulerian densely embedded graphs. Building on the work of Chekuri, Khanna and Shepherd [13, 12], Kleinberg [27] has shown an $O(\log^2 n)$-approximation LP-rounding algorithm for even-degree planar graphs. Extending on this work, Kawarabayashi and Kobayashi [25] gave an $O(\log n)$-approximation algorithm for EDP when the input graph is either 4-edge-connected planar or Eulerian planar.

A variation of NDP and EDP that has also been extensively studied is the NDPwC and EDPwC problems, where small congestion is allowed in the routing. In both problems, the input is the same as in the original problems, and a non-negative parameter c is given additionally. In NDPwC, the solution paths are allowed to have congestion at most c on the vertices. That is, every vertex may have at most c paths containing it in the routing solution. Similarly, in EDPwC, the paths are allowed to have congestion at most c on the edges. A constant-factor approximation algorithm with congestion $\Theta(\log n/\log\log n)$ for both NDPwC and EDPwC follows from Raghavan and Thompson's randomized rounding technique [35]. Many developments since [13, 34, 4, 36, 15, 20, 11, 10] have led to a $O(\text{poly}\log k)$-approximation algorithm for both problems with congestion 2. A constant approximation with congestion 2 is known for EDPwC in planar graphs [42]. On the negative side, it is known that for any $1 \leq c \leq O(\frac{\log\log n}{\log\log\log n})$, there is no $O((\log n)^{\frac{1-\epsilon}{c+1}})$-approximation algorithm for EDPwC with congestion c unless $\mathsf{NP} \subseteq \mathsf{ZPTIME}(n^{\text{poly}\log n})$ [5].

This thesis represents the author's contributions in the line of papers [16, 17, 19] that led to improvements in understanding the approximability of NDP. The improved approximation algorithm for [16] also appears in the authors master's paper and was joint work with and under guidance from Julia Chuzhoy. The second result of approximation algorithms for $\mathsf{NDP\text{-}Disc}$ and $\mathsf{NDP\text{-}Cylinder}$ were joint work with Julia Chuzhoy and Shi Li in [17]. The author's main contribution was the development of a constant approximation algorithm for the Demand Pair Selection Problem (DPSP) which both problems are reduced to. Lastly, the new hardness result [19] was joint work with Julia Chuzhoy and Rachit Nimavat. The author developed the level-1 instance and the technique to boost the gap in higher level instances by removing vertices from the grid and replacing each demand pair with a previous level instance.

CHAPTER 2

IMPROVED APPROXIMATION FOR NODE-DISJOINT PATHS IN GRIDS

2.1 Introduction

In this chapter we show an $\tilde{O}(n^{1/4})$-approximation for NDP-Grid. Our algorithm distinguishes between two types of the demand pairs: an (s_i, t_i) pair is *bad* if both s_i and t_i are close to the grid boundary, and it is good otherwise. Interestingly, the standard integrality gap examples for the multicommodity flow LP relaxation usually involve a grid graph, and bad demand pairs. Our algorithm deals with bad and good demand pairs separately, and in particular it shows that if all demand pairs are good, then the integrality gap of the LP relaxation becomes $O(n^{1/4} \cdot \log n)$ (but unfortunately it still remains polynomial in n - see Section 2.5).

NDP in grid graphs has been studied in the past, often in the context of VLSI layout. Aggarwal, Kleinberg and Williamson [2] consider a special case, where the set of the demand pairs is a permutation — that is, every vertex of the grid participates in exactly one demand pair. They show that for any permutation, one can route $\Omega(\sqrt{n}/\log n)$ demand pairs. They also show that with spacing d, every permutation contains a set of $\Omega(\sqrt{nd}/\log n)$ pairs that can be routed on node-disjoint paths. Our algorithm for routing on grids is inspired by their work.

Cutler and Shiloach [22] studied NDP in grids in the following three settings. They assume that all source vertices appear on the top row R_1 of the grid, and all destination vertices appear on some other row R_ℓ of the grid, sufficiently far from the top and the bottom rows (for example, $\ell = \lceil n/2 \rceil$). In the packed-packed setting, the sources are a set of k consecutive vertices of R_1, and the destinations are a set of k consecutive vertices of R_ℓ. They show a necessary and a sufficient condition for when all demand pairs can be routed in the packed-packed instance. The second setting is the packed-spaced setting. Here, the sources are again a set of k consecutive vertices of R_1, but the distance between every consecutive pair of the destination vertices on R_ℓ is at least d. For this setting, the authors show that if $d \geq k$, then all demand pairs can be routed. We extend their algorithm to a more general setting, where the destination vertices may appear anywhere in the grid, as long as the distance between any pair of the destination vertices, and any destination vertex and the boundary of the grid, is at least $\Omega(k)$. This extension of the algorithm of [22] is used as a basic building block in both our algorithm, and the APX-hardness proof. We note that Robertson and Seymour [40] provided sufficient conditions for the existence of node-disjoint routing of a given set of demand pairs in the more general setting of graphs drawn on surfaces, and they provide an algorithm whose running time is $\text{poly}(n) \cdot f(k)$ for finding the routing, where $f(k)$ is at least exponential in k. Their result implies the existence of the routing on grids, when the destination vertices are sufficiently spaced from each other and from the grid boundaries. However, we are not aware of an algorithm for finding the routing, whose running time is polynomial in n and k, and we provide such an algorithm here. The third setting studied by Cutler and Shiloach is the spaced-spaced setting, where the distance between any pair of source vertices, and any pair of destination vertices is at least d. The authors note that they could not come up with a better algorithm for this setting, than the one provided for the packed-spaced case.

2.2 Preliminaries

We consider the NDP problem in two-dimensional grids: The input is an $(N \times N)$-grid graph $G = (V, E)$, and a collection $\mathcal{M} = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ of demand, or source-destination, pairs. The goal is to find a largest-cardinality collection $\mathcal{P}$ of paths, where each path in $\mathcal{P}$ connects some demand pair (s_i, t_i), and every vertex participates in at most one path in $\mathcal{P}$.

The vertices in the set $\{s_1, t_1, \ldots, s_k, t_k\}$ are called *terminals*. By convention, we denote $n = |V|$, so $n = N^2$.

We assume that the grid rows are indexed $R_1, \ldots, R_N$ in the top-to-bottom order, and the columns are indexed $C_1, \ldots, C_N$ in the left-to-right order. We denote by $v(i,j)$ the unique vertex in $R_i \cap C_j$. Given a vertex $v \in V$, let $\mathrm{col}(v)$ denote the column, and $\mathrm{row}(v)$ denote the row in which v lies. The boundary of the grid is $\Gamma(G) = R_1 \cup R_N \cup C_1 \cup C_N$. We call R_1, R_N, C_1, C_N the *boundary edges* of the grid. Given any integers $1 \leq i \leq i' \leq N$, $1 \leq j \leq j' \leq N$, we denote by $G[i : i', j : j']$ the sub-graph of G, induced by the set $\{v(i'', j'') \mid i \leq i'' \leq i', j \leq j'' \leq j'\}$ of vertices. We sometimes say that $G[i : i', j : j']$ is the sub-grid of G, spanned by rows $R_i, \ldots, R_{i'}$ and columns $C_j, \ldots, C_{j'}$.

Given a path P in G, and a set S of vertices of G, we say that P is *internally disjoint* from S, if no vertex of S serves as an inner vertex of P.

Observation 2.2.1. *Let G be a $(h \times w)$-grid, with $w, h > 2$, and let $k \leq \min\{w-2, h-2\}$ be an integer. Then for any pair $L \neq L'$ of distinct boundary edges of G, for any pair $S \subseteq V(L)$, $T \subseteq V(L')$ of vertex subsets on these boundary edges, with $|S| = |T| = k$, there is a set $\mathcal{P}$ of k node-disjoint paths, connecting the vertices of S to the vertices of T in G, such that all paths in $\mathcal{P}$ are internally disjoint from $V(L \cup L')$. Moreover, the path set $\mathcal{P}$ can be found efficiently.*

Proof. Let G' be the sub-graph of G, obtained by deleting all vertices of $(L \cup L') \setminus (S \cup T)$ from G. It is enough to show that there is a set $\mathcal{P}$ of k disjoint paths connecting the vertices of S to the vertices of T in G'.

Assume first that L and L' are opposite boundary edges, e.g. L is the top and L' is the bottom boundary edge of G. Assume for contradiction that such a set $\mathcal{P}$ of paths does not exist. Then from Menger's theorem, there is a set Z of at most $k-1$ vertices, such that in $G' \setminus Z$, there is no path from a vertex of $S \setminus Z$ to a vertex of $T \setminus Z$. However, the vertices of S lie on k distinct columns of G, so at least one such column, say C, does not contain a vertex of Z. Similarly, there is some column C' of G that contains a vertex of T, and $V(C') \cap Z = \emptyset$. Finally, since there are at least $k+2$ rows in G, there is some row $R \neq R_1, R_h$, that contains no vertex of Z. Altogether, $(C \cup R \cup C') \cap G'$ lie in the same connected component of $G' \setminus Z$, and this connected component contains a vertex of S and a vertex of T, a contradiction. The set $\mathcal{P}$ of paths can be found efficiently by computing the maximum single-commodity flow between the vertices of S and the vertices of T in G', and using the integrality of flow.

The proof for when L and L' are neighboring boundary edges is similar. $\square$

Consider the input grid graph G. The L_∞-distance between two vertices $v(i,j)$ and $v(i',j')$ is defined as

$$d_\infty(v(i,j), v(i',j')) = \max(|i-i'|, |j-j'|).$$

The distance between a set $S \subseteq V(G)$ of vertices and a vertex $v \in V(G)$ is

$$d_\infty(v, S) = \min_{u \in S} \{d_\infty(v,u)\}.$$

Multicommodity Flow LP Relaxation. For each demand pair $(s_i, t_i) \in \mathcal{M}$, let $\mathcal{P}_i$ be the set of all paths connecting s_i to t_i in G, and let $\mathcal{P} = \bigcup_{i=1}^k \mathcal{P}_i$. In order to define the multicommodity flow LP-relaxation of NDP, every path $P \in \mathcal{P}$ is assigned a variable $f(P)$ representing the amount of flow that is sent on P, and for each demand pair (s_i, t_i), we introduce variable x_i, whose value is the total amount of flow sent from s_i to t_i. The LP-relaxation is then defined as follows.

$$\begin{array}{lll} \text{(LP-flow)} & \max & \sum_{i=1}^k x_i \\ & \text{s.t.} & \sum_{P \in \mathcal{P}_i} f(P) = x_i \quad \forall 1 \le i \le k \\ & & \sum_{P: v \in P} f(P) \le 1 \quad \forall v \in V \\ & & f(P) \ge 0 \qquad \forall 1 \le i \le k, \forall P \in \mathcal{P}_i \end{array}$$

Even though this LP-relaxation has exponentially many variables, it can be efficiently solved by standard techniques, e.g. by using an equivalent polynomial-size edge-based formulation.

It is well known that the integrality gap of (LP-flow) is $\Omega(\sqrt{n})$ even in grid graphs. Indeed, let G be an $(N \times N)$-grid, and let $k = N-2$. We let the sources $s_1, \ldots, s_k$ appear consecutively on row R_1, starting from $v(1,1)$ in this order, and the destinations appear consecutively on row R_N starting from $v(N,1)$, in the opposite order: $t_k, \ldots, t_1$ (see Figure 2.1). It is easy to see that there is a solution to (LP-flow) of value $k/3 = \Omega(N)$: for each pair (s_i, t_i), we send $1/3$ flow unit on the path P_i, where P_i is an s_i–t_i path lying in the union of columns C_i, C_{N-i-1} and row R_i+1. On the other hand, it is easy to see that the value of any integral solution is 1, since any pair of paths connecting the demand pairs have to cross. Since the number of vertices in G is $n = N^2$, this gives a lower bound of $\Omega(\sqrt{n})$ on the integrality gap of (LP-flow).

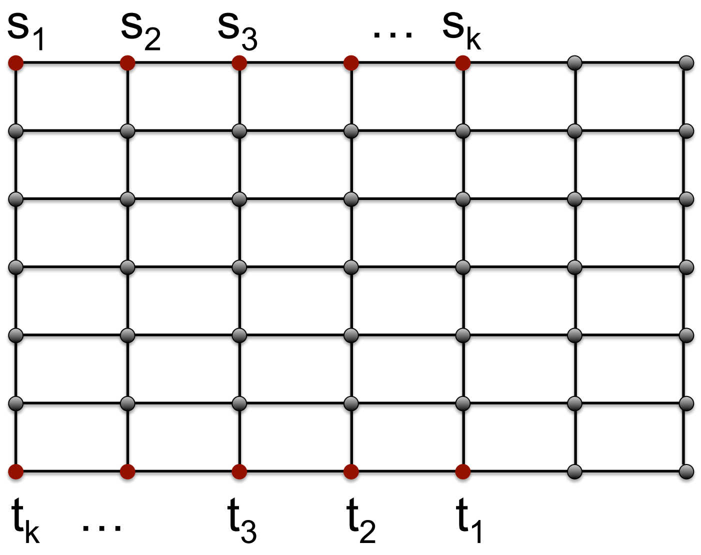


Figure 2.1: Integrality gap example

2.3 Routing with Well-Separated Destinations

In this section we generalize the results of Cutler and Shiloach [22], by proving the following theorem.

Theorem 2.3.1. *Let H be the $(N \times N)$-grid, and let $\mathcal{M} = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ be a set of $k \geq 4$ demand pairs in H, such that: (i) $s_1, \ldots, s_k$ are all distinct, and they appear on the first row of H; (ii) for all $1 \leq i \neq j \leq k$, $d_\infty(t_i, t_j) > 4k+4$; and (iii) for all $1 \leq i \leq k$, $d_\infty(t_i, V(\Gamma(H))) > 4k+4$. Then there is an efficient algorithm that routes all demand pairs in $\mathcal{M}$ in graph H.*

The rest of this section is devoted to proving Theorem 2.3.1. For each destination vertex t_j, we define a sub-grid B_j of H of size $((2k+3) \times (2k+3))$, centered at t_j, that is, if $t_j = v(i, i')$, then B_j is a sub-grid of G spanned by rows $R_{i-(k+1)}, \ldots, R_{i+(k+1)}$ and columns $C_{i'-(k+1)}, \ldots, C_{i'+(k+1)}$ of H.

We call the resulting sub-grids $B_1, \ldots, B_k$ *boxes*. Notice that all boxes are disjoint from each other, due to the spacing of the destination terminals. We start with a high-level intuitive description of our algorithm. For each box B_j, we can associate an interval $I(B_j) \subseteq (1, N)$ with B_j, as follows: If C_{i_1}, C_{i_2} are the columns of H containing the first and the last columns of B_j, respectively, then $I(B_j) = (i_1, i_2)$. We say that the resulting set $\mathcal{I} = \left\{I(B_j)\right\}_{j=1}^{k}$ of intervals is aligned, if for all $i \neq j$, either $I(B_i) = I(B_j)$, or $I(B_i) \cap I(B_j) = \emptyset$. For simplicity, assume first that all intervals in $\mathcal{I}$ are aligned, and let $\{I_1, I_2, \ldots, I_r\}$ be the set of all distinct intervals in $\mathcal{I}$, ordered in their natural left-to-right order. For each $1 \leq h \leq r$, let $\mathcal{B}_h$ be the set of all boxes B_j with $I(B_j) = I_h$, and let $\mathcal{B} = \left\{B_j \mid 1 \leq j \leq k\right\}$. We define a "snake-like" ordering of the boxes in $\mathcal{B}$ as follows. For all $1 \leq h < h' \leq r$, the boxes of $\mathcal{B}_h$ appear before all boxes of $\mathcal{B}_{h'}$ in this ordering. Within each set $\mathcal{B}_h$, if h is odd, then the boxes of $\mathcal{B}_h$ are ordered in the order of their position in H from top to bottom, and otherwise they are ordered in the order of their position in H from bottom to top. We then define a set $\mathcal{P}$ of k paths, that start from the sources $s_1, \ldots, s_k$, and visit all boxes in $\mathcal{B}$ in this order (see Figure 2.2). We will make sure that when the paths of $\mathcal{P}$ traverse any box B_j, the path $P_j \in \mathcal{P}$ that originates at s_j visits the vertex t_j. In order to accomplish this, we need the following lemma.

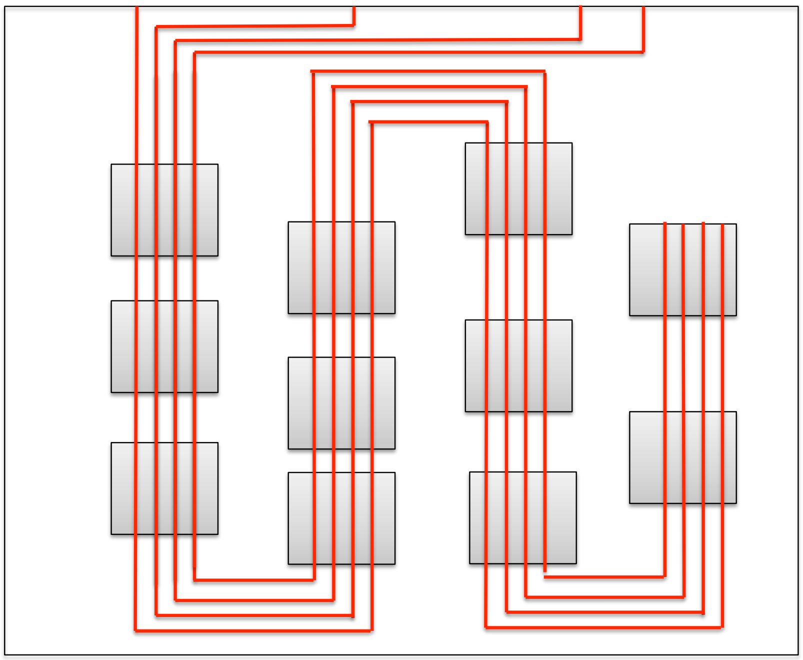


Figure 2.2: Traversing the boxes

Lemma 2.3.2. *Let B be the $((2k+3)\times(2k+3))$ grid, $t = v(k+2, k+2)$ the vertex in the center of the grid, and $1 \leq j \leq k$ any integer. Let $X = \{x_1, \ldots, x_k\}$ be any set of k vertices on the top boundary edge L of B and $Y = \{y_1, \ldots, y_k\}$ any set of k vertices on the bottom boundary edge L' of B, both sets ordered from left to right. Then we can efficiently find k disjoint paths $P'_1, \ldots, P'_k$ in B, such that for $1 \leq i \leq k$, path P'_i connects x_i to y_i; all paths are internally disjoint from $V(L \cup L')$; and path P'_j contains t.*

Proof. Let $U = \{u_1, \ldots, u_k\}$ be any set of k vertices on row R_{k+2} of B, ordered from left to right, such that $u_j = t$. Let $B' \subseteq B$ be the grid spanned by the top $k+2$ rows of B, and $B'' \subseteq B$ the grid spanned by the bottom $k+2$ rows of B. Note that $B' \cap B'' = R_{k+2}$.

From Observation 2.2.1, there is a set $\mathcal{P}_1$ of k node-disjoint paths in B', connecting the vertices of X to the vertices of U, and there is a set $\mathcal{P}_2$ of k node-disjoint paths in B'', connecting the vertices of U to the vertices of Y. Moreover, the paths in $\mathcal{P}_1 \cup \mathcal{P}_2$ are internally disjoint from $V(R_{k+2} \cup L \cup L')$. By concatenating the paths in $\mathcal{P}_1$ and $\mathcal{P}_2$, we obtain a set $\mathcal{P}'$ of k node-disjoint paths in B, connecting the vertices of X to the vertices of Y, such that the paths in $\mathcal{P}'$ are internally disjoint from $L \cup L'$. The intersection of each path in $\mathcal{P}'$ with the row R_{k+2} is exactly one vertex. Since graph B is planar, the paths in $\mathcal{P}'$ cross the row R_{k+2} in the same left-to-right order in which their endpoints appear on L and L'. Therefore, for $1 \leq i \leq k$, the ith path connects x_i to y_i, and the jth path contains the vertex t. $\square$

Since in general the intervals in $\mathcal{I}$ may not be aligned, we need to define the ordering between the boxes, and the set of paths traversing them more carefully. We start by defining an ordering of the destination vertices $\{t_j\}_{j=1}^{k}$, which will define an ordering of their corresponding boxes.

We draw vertical lines in the grid at every column whose index is an integral multiple of $(3k+2)$, and let $\{V_1, V_2, \dots\}$ denote the sets of vertices of the resulting vertical strips of width $3k+2$, that is, for $1 \leq m \leq \lceil N/(3k+2) \rceil$,

$$V_m = \left\{ v(j,j') \mid (m-1)(3k+2) < j' \leq \min\left\{m(3k+2), N\right\}; 1 \leq j \leq N \right\}.$$

Notice that $\left\{V_1, \dots, V_{\lceil N/(3k+2) \rceil}\right\}$ partition $V(H)$. We assign every terminal t_j to the unique set V_m containing t_j. We then define a collection $\mathcal{S}$ of vertical strips of H as follows: For each set V_m, such that at least one terminal is assigned to V_m, we add $H[V_m]$ to $\mathcal{S}$. We assume that the set of strips $\mathcal{S} = \left\{S_1, \dots, S_p\right\}$ is indexed in their natural left-to-right order. Abusing the notation, we will denote $V(S_m)$ by V_m, for $1 \leq m \leq p$.

Consider some vertical strip S_m, and let $t_i, t_j \in V_m$, for $j \neq i$. Then the horizontal distance between t_i and t_j, $|\operatorname{col}(t_i) - \operatorname{col}(t_j)| \leq 3k+2$, and since $d_\infty(t_i, t_j) > 4k+4$, t_i and t_j must be at vertical distance at least $4k+4$. Therefore, we can order the destination terminals assigned to the same vertical strip in the increasing or decreasing row index. We define the ordering of all destination terminals as follows:

1. for every $1 \leq m < m' \leq p$, every terminal $t_i \in V_m$ precedes every terminal $t_j \in V_{m'}$; and
2. for $t_i, t_j \in V_m$, with $\mathrm{row}(t_j) > \mathrm{row}(t_i)$, if m is odd then t_i precedes t_j, and if m is even, then t_j precedes t_i.

Let $\mathcal{B} = \{B_j \mid 1 \leq j \leq k\}$ be the set of boxes corresponding to the destination vertices. The ordering of the destination vertices now imposes an ordering on $\mathcal{B}$. We re-index the boxes B_j according to this ordering, and we denote by $t(B_j)$ the unique destination terminal lying in B_j. We will say that a box B_j belongs to strip S_m iff the corresponding terminal $t(B_j) \in V_m$. (Note that B_j is not necessarily contained in S_m). The following observation is immediate.

Observation 2.3.3. *If box B_j belongs to strip S_m, then at least $k+2$ vertices from the top boundary of B_j, and at least $k+2$ vertices from the bottom boundary of B_j belong to V_m.*

In order to complete the construction of the set $\mathcal{P}$ of paths routing all demand pairs, we define, for $1 \leq i \leq k$, a set $\mathcal{P}_i$ of k disjoint paths, with the following properties:

P1. Paths in $\mathcal{P}_1$ connect $\{s_i\}_{i=1}^k$ to some set of k vertices on the top boundary of B_1;

P2. For $i > 1$:

- if B_{i-1} and B_i belong to the same strip S_m, and m is odd, then paths in $\mathcal{P}_i$ connect k vertices on the bottom row of B_{i-1} to k vertices on the top row of B_i;
- if B_{i-1} and B_i belong to the same strip S_m, and m is even, then paths in $\mathcal{P}_i$ connect k vertices on the top row of B_{i-1} to k vertices on the bottom row of B_i;

- if B_{i-1} belongs to strip S_m and B_i to strip S_{m+1}, and m is odd, then paths in $\mathcal{P}_i$ connect k vertices on the bottom row of B_{i-1} to k vertices on the bottom row of B_i;
- if B_{i-1} belongs to strip S_m and B_i to strip S_{m+1}, and m is even, then paths in $\mathcal{P}_i$ connect k vertices on the top row of B_{i-1} to k vertices on the top row of B_i; and

P3. All paths in $\bigcup_{i=1}^{k} \mathcal{P}_i$ are disjoint from each other, and each path is internally disjoint from $\bigcup_{B\in\mathcal{B}} V(B)$.

Theorem 2.3.4. *There is an efficient algorithm to find the collections $\mathcal{P}_1, \ldots, \mathcal{P}_k$ of paths with properties (P1)–(P3).*

Proof. For each box B_j, for $1 \leq j \leq k$, we define four sub-graphs of H, Z_j^t, Z_j^b, Z_j^{tt}, Z_j^{bb}, that will be used in order to route the sets $\mathcal{P}_j$, $\mathcal{P}_{j+1}$ of paths.

Consider some box B_j, and assume that it belongs to strip S_m. Let C_ℓ, C_r be the columns of H that serve as the left and the right boundaries of S_m, respectively. Let R_t, R_b be the rows of H containing the top and the bottom row of B_j, respectively. Intuitively, Z_j^t is the sub-grid of strip S_m, containing the $k+1$ rows immediately above row R_t, in addition to the row R_t, and Z_j^b is defined similarly below B_j. Formally, Z_j^t is the sub-grid of H spanned by columns $C_\ell, \ldots, C_r$, and rows $R_{t-k-1}, \ldots, R_t$, so Z_j^t contains $k+2$ rows and $3k+2$ columns. Similarly, Z_j^b is the sub-grid of H spanned by columns $C_\ell, \ldots, C_r$, and rows $R_b, \ldots, R_{b+k+1}$, so Z_j^b contains $k+2$ rows and $3k+2$ columns (see Figure 2.3).

We now turn to define the grids Z_j^{tt} and Z_j^{bb}. Graph Z_j^{tt} is defined as follows. Assume w.l.o.g. that m is odd (recall that S_m is the strip containing $t(B_j)$). If B_j is not the topmost box that belongs to S_m, then let R_a be the row of H containing the bottom row of Z_{j-1}^b; otherwise let $R_a = R_{2k+1}$ if $j > 1$ and $R_a = R_{k+1}$ if $j = 1$. Let $R_{a'}$ be the row of H containing the top row of Z_j^t. We would like Z_j^{tt} to be the grid containing the segments of

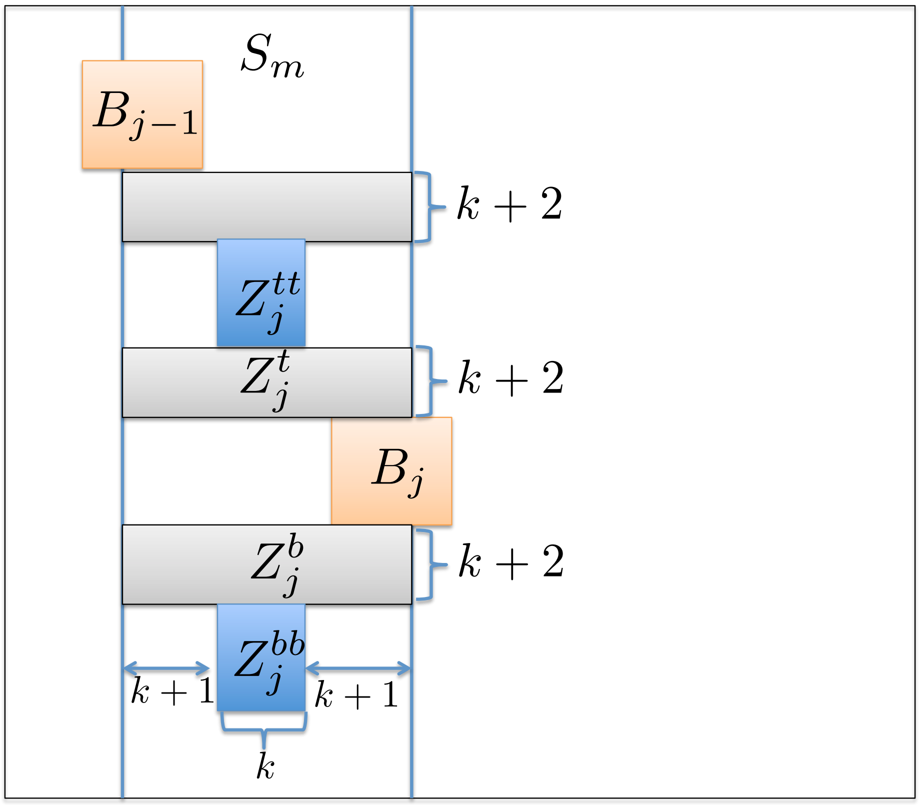


Figure 2.3: Graphs Z_j^b, Z_j^t, Z_j^{bb} and Z_j^{tt}

the middle k columns of S_m, between rows R_a and $R_{a'}$. Formally, we let Z_j^{tt} be the sub-grid of H spanned by rows $R_a, \ldots, R_{a'}$, and columns $C_{\ell+k+2}, \ldots, C_{\ell+2k+1}$.

We define the graph Z_j^{bb} similarly. If B_j is not the bottommost box of S_m, then let R_c be the row of H containing the top row of Z_{j+1}^t, and otherwise let $R_c = R_{N-k-1}$. Let $R_{c'}$ be the row of H containing the bottom row of Z_j^b. Graph Z_j^{bb} is the sub-grid of H spanned by rows $R_{c'}, \ldots, R_c$, and columns $C_{\ell+k+2}, \ldots, C_{\ell+2k+1}$.

Notice that if B_j is not the topmost box of S_m, then $Z_j^{tt} = Z_{j-1}^{bb}$, and if B_j is not the bottommost box of B_m, then $Z_j^{bb} = Z_{j+1}^{tt}$. We need the following observation.

Observation 2.3.5. *For all $1 \leq q \leq k$, $B_q \cap Z_j^{tt}, B_q \cap Z_j^{bb} = \emptyset$. Moreover, if $q \neq j$, then additionally $B_q \cap Z_j^b, B_q \cap Z_j^t = \emptyset$.*

Proof. We prove for Z_j^t and Z_j^{tt}. The proofs for Z_j^b and Z_j^{bb} are symmetric.

Consider some box B_q with $q \neq j$, and assume for contradiction that $B_q \cap Z_j^t \neq \emptyset$. Then the vertical distance between $t(B_q)$ and $t(B_j)$ is less than $4k+4$, and so the horizontal distance between them must be at least $4k+4$. However, $t(B_j)$ lies in the strip S_m, and, since B_q intersects Z_j^t, the horizontal distance between $t(B_q)$ and the left or the right column of S_m is at most $k+1$, and so the total horizontal distance between $t(B_q)$ and $t(B_j)$ is at most $4k+4$, a contradiction.

Consider now some box B_q, for $1 \leq q \leq k$, and assume for contradiction that $B_q \cap Z_j^{tt} \neq \emptyset$. If B_j is the topmost box in S_m, then B_q cannot belong to S_m. If B_j is not the topmost box of S_m, then B_q cannot belong to S_m due to the definition of Z_j^{tt}. Therefore, $t(B_q)$ lies in either S_{m+1} or S_{m-1}. But since B_q is a box of width $2k+3$, with $t(B_q)$ lying in $(k+2)$th column of B_q, it is impossible for B_q to intersect Z_j^{tt}. $\square$

We are now ready to define the sets $\mathcal{P}_i$ of paths. In order to do so, we define a collection $\{H_1, \ldots, H_k\}$ of disjoint sub-graphs of H, and each such sub-graph H_i will be used to route the set $\mathcal{P}_i$ of paths. We start by letting H_1 be the union of three graphs, Z_1^t, Z_1^{tt}, and the sub-grid of H spanned by the top $k+1$ rows of H. We denote this latter graph by H_1'. Recall that the terminal $t(B_1)$ lies in strip S_1. Let A_1 be the set of k vertices on the top boundary of Z_1^{tt}, A_2 the set of k vertices on the bottom row of Z_1^{tt}, and let A_3 be any set of k vertices on the top row of B_1, that lie in S_1 (from Observation 2.3.3, such a set exists). From Observation 2.2.1, we can construct three sets of paths: set $\mathcal{P}_1'$ in H_1', connecting each source vertex to some vertex of A_1; set $\mathcal{P}_1''$ in Z_1^{tt} connecting the vertices of A_1 to the vertices of A_2 (the paths in $\mathcal{P}_1''$ are just the columns of Z_1^{tt}), and set $\mathcal{P}_1'''$ in Z_1^t, connecting the vertices of A_2 to the vertices of A_3. We let $\mathcal{P}_1$ be obtained by concatenating the paths in $\mathcal{P}_1', \mathcal{P}_1''$, and $\mathcal{P}_1'''$.

Consider now some index $1 < j \leq k$, and assume that B_{j-1} belongs to some strip S_m. We assume w.l.o.g. that m is odd (the case where m is even is dealt with similarly), and we show how to construct the set $\mathcal{P}_j$ of paths. We consider two cases. The first case is when

B_j also lies in S_m. We then let H_j be the union of Z^b_{j-1}, Z^{bb}_{j-1} and Z^t_j. The set $\mathcal{P}_j$ of paths will be contained in H_j, and it is defined as follows. Let A_1 be any set of k vertices on the bottom row of B_{j-1}, that lie in V_m (this set exists due to Observation 2.3.3); let A_2 and A_3 be the vertices of the top and the bottom rows of Z^{bb}_{j-1}, respectively, and let A_4 be any set of k vertices on the top row of B_j that lie in V_m. As before, using Observation 2.2.1, we can construct three sets of paths: set $\mathcal{P}'_j$ in Z^b_{j-1}, connecting each vertex of A_1 to some vertex of A_2; set $\mathcal{P}''_j$ in Z^{bb}_{j-1} connecting the vertices of A_2 to the vertices of A_3 (the paths in $\mathcal{P}''_j$ are just the columns of Z^{bb}_{j-1}), and set $\mathcal{P}'''_j$ in Z^t_j, connecting the vertices of A_3 to the vertices of A_4. We let $\mathcal{P}_j$ be obtained by concatenating the paths in $\mathcal{P}'_j, \mathcal{P}'_j$, and $\mathcal{P}'''_j$.

Finally, assume that B_j belongs to S_{m+1}. Let C_ℓ and C_r be the columns of H that serve as the left boundary of S_m and the right boundary of S_{m+1}, respectively. Let H'_j be the sub-grid of H, spanned by columns $C_\ell, \dots, C_r$, and rows $R_{N-k-1}, \dots, R_N$. We let H_j be the union of $Z^b_{j-1}, Z^{bb}_{j-1}, H'_j, Z^b_j$ and Z^{bb}_j. Using methods similar to those described above, it is easy to find a set $\mathcal{P}_j$ of k disjoint paths in H_j, connecting k vertices on the bottom row of B_{j-1} to k vertices on the bottom row of B_j.

The case where m is even is dealt with similarly. The only difference is that in the case where B_j belongs to S_{m+1}, we use rows $R_{k+2}, \dots, R_{2k+1}$ to define H'_j, instead of rows $R_{N-k+1}, \dots, R_N$, to avoid collision with the graph H'_1.

From the construction of the graphs H_i, it is easy to see that all such graphs are mutually disjoint, and therefore we obtain the desired sets $\mathcal{P}_1, \dots, \mathcal{P}_k$ of paths with properties (P1)–(P3).

□

Given the path sets $\mathcal{P}_1, \dots, \mathcal{P}_k$ with properties (P1)–(P3), it is now easy to complete the proof of Theorem 2.3.1. For each box B_j, let $X_j \subseteq V(B_j)$ be the set of k vertices that serve as endpoints of the paths of $\mathcal{P}_j$, and let $Y_j \subseteq V(B_j)$ be the set of k vertices that serve as endpoints of the paths in $\mathcal{P}_{j+1}$. (For $j = k$, we choose the set Y_k of k vertices on the top or the bottom boundary of B_k (opposing the boundary edge where the vertices of X_k lie) arbitrarily).

We construct the set $\mathcal{P}$ of paths gradually, by starting with $\mathcal{P} = \mathcal{P}_1$, and performing k iteration. We assume that at the beginning of iteration i, set $\mathcal{P}$ contains k disjoint paths, connecting the k source vertices to the vertices of X_i. This is clearly true at the beginning of the first iteration. The ith iteration is executed as follows. Assume that $t(B_i) = t_r$, and let $u \in X_i$ be the vertex where the path of $\mathcal{P}$ originating at s_r terminates. From Lemma 2.3.2, we can find a set $\mathcal{Q}_i$ of paths inside B_i, connecting the vertices of X_i to the vertices of Y_i, that are internally disjoint from the top and the bottom boundary edges of B_i, such that the path originating at u contains the vertex t_r. We then concatenate the paths in $\mathcal{P}$ with the paths in $\mathcal{Q}_i$, and, if $i < k$, with the paths in $\mathcal{P}_{i+1}$, to obtain the new set $\mathcal{P}$ of paths, and continue to the next iteration. After k iterations, we obtain a collection of k node-disjoint paths that traverse all boxes B_j, such that for each $1 \leq i \leq k$, the path originating from s_i contains the vertex t_i.

2.4 An $\tilde{O}(n^{1/4})$-Approximation Algorithm

We assume that we are given the $(N \times N)$ grid graph $G = (V, E)$, so $n = |V| = N^2$, and a collection $\mathcal{M} = \{(s_i, t_i)\}_{i=1}^{k}$ of demand pairs. We say that a demand pair (s_i, t_i) is *bad* if both $d_\infty(s_i, \Gamma(G)), d_\infty(t_i, \Gamma(G)) < 4\sqrt{N} + 4$, and we say that it is good otherwise. Let $\mathcal{M}', \mathcal{M}'' \subseteq \mathcal{M}$ denote the sets of the good and the bad demand pairs in $\mathcal{M}$, respectively. We find an approximate solution to each of the two sub-problems, defined by $\mathcal{M}'$ and $\mathcal{M}''$, separately, and take the better of the two solutions.

2.4.1 Routing the Good Pairs

Let (f, x) be the optimal solution to the linear program (LP-flow) on instance $(G, \mathcal{M}')$, and let OPT_{LP} be its value. We show an algorithm that routes $\Omega(\mathsf{OPT}_{LP}/(n^{1/4} \cdot \log n))$ demand pairs. The algorithm consists of two steps. In the first step, we reduce the problem to routing between two square sub-grids of G. We note that a similar reduction has been used in prior work, e. g. by Aggarwal et al. [2]. In the second step, we show an approximation algorithm for the resulting sub-problem.

Reduction to the 2-Square Problem. In this step, we reduce the problem of routing on G with a general set $\mathcal{M}'$ of good demand pairs, to a problem where we are given two disjoint sub-grids (or squares) Q_1, Q_2 of G, and every demand pair (s_j, t_j) has $s_j \in Q_1$ and $t_j \in Q_2$, or vice versa.

We start by partitioning the set $\mathcal{M}'$ of the demand pairs into $\lceil \log N \rceil$ subsets. We denote the subsets by $\mathcal{M}_1, \ldots, \mathcal{M}_{\lceil \log N \rceil}$, where

$$\mathcal{M}_h = \left\{(s_j, t_j) \in \mathcal{M}' \mid 2^{h-1} \le d_\infty(s_j, t_j) < 2^h\right\}.$$

For each $1 \le h \le \lceil \log N \rceil$, let

$$F_h = \sum_{(s_j, t_j) \in \mathcal{M}_h} x_j,$$

where x_j is the amount of flow sent from s_j to t_j in the solution to (LP-flow). We let h^* be the index maximizing F_{h^*}, so

$$F_{h^*} \ge \mathsf{OPT}_{LP} / \lceil \log N \rceil .$$

From now on, we focus on routing the pairs in $\mathcal{M}_{h^*}$, and we will route $\Omega(F_{h^*}/n^{1/4})$ such pairs.

Assume first that $h^* \leq 6$. In this case, we partition the grid into sub-grids of size at most (256×256) with a random offset, as follows. Select an integer $0 \leq z < 256$ uniformly at random, and use the set $\mathcal{C} = \{C_{z+256i}\}_{i=0}^{\lfloor (N-z)/256 \rfloor}$ of columns and the set $\mathcal{R} = \{R_{z+256i}\}_{i=0}^{\lfloor (N-z)/256 \rfloor}$ of rows to partition the grid into sub-grids. Let $\mathcal{Q}$ be the resulting collection of sub-grids.

We define a new LP-solution as follows, starting with the original LP-solution:

- for every demand pair $(s_j, t_j) \notin \mathcal{M}_{h^*}$, set $x_j = 0$, and $f(P) = 0$ for all paths $P \in \mathcal{P}_j$;
- for every demand pair $(s_j, t_j) \in \mathcal{M}_{h^*}$, if s_j or t_j lie on a row of $\mathcal{R}$ or a column of $\mathcal{C}$, or if they belong to different sub-grids in $\mathcal{Q}$, set $x_j = 0$ and $f(P) = 0$ for all paths $P \in \mathcal{P}_j$.

Since for each pair $(s_j, t_j) \in \mathcal{M}_{h^*}$, $d_\infty(s_j, t_j) < 64$, it is easy to see that the expected value of the resulting LP-solution is

$$C = \Omega(F_{h^*}) = \Omega(\mathsf{OPT}_{LP}/\log N) = \Omega(\mathsf{OPT}_{LP}/\log n).$$

By trying all possible values $0 \leq z < 256$, we can find a partition $\mathcal{Q}$ of G, and a corresponding LP-solution, whose value is at least C. Notice that for each sub-grid $Q \in \mathcal{Q}$, the number of vertices in Q is bounded by 256^2, and so the total amount of flow routed between the demand pairs contained in Q is bounded by 256^2. For each sub-grid $Q \in \mathcal{Q}$, if there is any demand pair $(s_j, t_j) \in \mathcal{M}_{h^*}$ with $s_j, t_j \in Q$, and a non-zero value x_j in the current LP-solution, we select any such pair and route it via any path P contained in Q, which is disjoint from the boundary of Q. It is easy to see that the total number of the demand pairs routed is $\Omega(C) = \Omega(\mathsf{OPT}_{LP}/\log n)$. From now on, we assume that $h^* > 6$.

For convenience, we denote h^* by h from now on. Let $m = 2^h/16$. We partition the grid into a collection

$$\mathcal{Q} = \left\{Q_{p,q} \mid 1 \leq p \leq \lfloor N/m \rfloor, 1 \leq q \leq \lfloor N/m \rfloor\right\}$$

of disjoint sub-grids, or squares, as follows:

1. First, partition G into $\lfloor N/m \rfloor$ disjoint vertical strips $V_1, \ldots, V_{\lfloor N/m \rfloor}$, each containing m consecutive columns of G, except for the last strip, that may contain between m and $2m-1$ columns.

2. Next, partition each vertical strip V_p into $\lfloor N/m \rfloor$ disjoint sub-grids, where each sub-grid contains m consecutive rows of V_p, except possibly for the last sub-grid, that may contain between m and $2m-1$ rows.

The width and the height of each such sub-grid is then between m and $2m-1$, where $m \leq N/16$. Notice that for each such grid $Q_{p,q} \in \mathcal{Q}$, if L is the left boundary edge of $Q_{p,q}$, and L' is the left boundary edge of G, then either $L \subseteq L'$, or L and L' are separated by at least $m-1$ columns. The same holds for the other three boundary edges. We need the following observation.

Observation 2.4.1. *Let $(s_j, t_j) \in \mathcal{M}_h$ be a demand pair, and assume that $s_j \in Q_{p,q}$ and $t_j \in Q_{p',q'}$. Then:*

$$5 \leq |p - p'| + |q - q'| \leq 34.$$

Proof. We first show that $|p - p'| + |q - q'| \geq 5$. Indeed, assume otherwise. Then both the horizontal and the vertical distance between s_j and t_j is less than $8m = 8 \cdot 2^h/16 = 2^{h-1}$, while $d_\infty(s_j, t_j) \geq 2^{h-1}$, a contradiction.

Assume now for contradiction that $|p - p'| + |q - q'| > 34$. Then $d_\infty(s_j, t_j) > 16m = 2^h$, contradicting the fact that $d_\infty(s_j, t_j) < 2^h$. $\square$

We say that a pair $(Q_{p,q}, Q_{p',q'})$ of squares in $\mathcal{Q}$ is *interesting* iff

$$5 \leq |p - p'| + |q - q'| \leq 34.$$

Let $\mathcal{Z}$ be the set of all interesting pairs of squares in $\mathcal{Q}$. We associate an NDP instance with each such pair $Z = (Q_{p,q}, Q_{p',q'})$, as follows.

Let $\mathcal{M}(Z) \subseteq \mathcal{M}_h$ be the set of all demand pairs $(s_j, t_j) \in \mathcal{M}_h$ where $s_j \in Q_{p,q}$ and $t_j \in Q_{p',q'}$, or vice versa. We also define a box $A(Z)$, that contains $Q_{p,q} \cup Q_{p',q'}$, and adds a margin of m around them, if possible. More precisely, let ℓ be the smallest integer, such that

$$R_\ell \cap (Q_{p,q} \cup Q_{p',q'}) \neq \emptyset,$$

and let ℓ' be the largest integer, such that

$$R_{\ell'} \cap (Q_{p,q} \cup Q_{p',q'}) \neq \emptyset.$$

Similarly, let b and b' be the smallest and the largest integers, respectively, such that

$$C_b \cap (Q_{p,q} \cup Q_{p',q'}), C_{b'} \cap (Q_{p,q} \cup Q_{p',q'}) \neq \emptyset.$$

We then let $A(Z)$ be the sub-grid of G spanned by rows $R_{\max\{1,\ell-m\}}, \ldots, R_{\min\{\ell'+m,N\}}$, and by columns $C_{\max\{1,b-m\}}, \ldots, C_{\min\{b'+m,N\}}$.

For every interesting pair of squares $Z \in \mathcal{Z}$, we now define an instance of the NDP problem on graph $A(Z)$, with the set $\mathcal{M}(Z)$ of demand pairs. Let $F(Z)$ be the total amount of flow routed between the demand pairs in $\mathcal{M}(Z)$ in the current LP-solution F_h to our original problem (notice that in our LP-solution, the fractional routing of the demand pairs in $\mathcal{M}(Z)$ is not necessarily contained in $A(Z)$). From the above discussion, $\sum_{Z \in \mathcal{Z}} F(Z) = \Omega(\mathsf{OPT}_{LP}/\log N)$. We will show an algorithm that routes, for each $Z \in \mathcal{Z}$, $\Omega(F(Z)/n^{1/4})$ demand pairs in $\mathcal{M}(Z)$ integrally, in graph $A(Z)$. However, it is possible that for two pairs $Z, Z' \in \mathcal{Z}$, $A(Z) \cap A(Z') \neq \emptyset$, and the two routings may interfere with each other. We resolve this problem in the following step.

From Observation 2.4.1, it is easy to see that for each interesting pair of squares $Z \in \mathcal{Z}$, the number of pairs $Z' \in \mathcal{Z}$ with $A(Z) \cap A(Z') \neq \emptyset$ is bounded by some constant c. We construct a graph H, whose vertex set is $V(H) = \{v_Z \mid Z \in \mathcal{Z}\}$, and there is an edge $(v_Z, v_{Z'})$ iff $A(Z) \cap A(Z') \neq \emptyset$. As observed above, the maximum vertex degree in this graph is bounded by some constant c, and so we can color H with $c+1$ colors.

Let $U_i \subseteq V(H)$ be the set of vertices of color i. We select a color class i^*, maximizing the value $F^{i^*} = \sum_{v_Z \in U_{i^*}} F(Z)$. Clearly,

$$F^{i^*} = \Omega(\mathsf{OPT}_{LP}/\log N).$$

For every pair $v_Z, v_{Z'}$ of vertices in U_{i^*}, we now have $A(Z) \cap A(Z') = \emptyset$. In order to obtain an $O(n^{1/4}\log n)$-approximation algorithm for the special case where all demand pairs are good, it is now enough to prove the following theorem.

Theorem 2.4.2. *There is an efficient algorithm, that, for every interesting pair $Z \in \mathcal{Z}$ of squares, routes $\Omega(F(Z)/n^{1/4})$ demand pairs of $\mathcal{M}(Z)$ inside the grid $A(Z)$.*

The Rounding Algorithm. From now on we focus on proving Theorem 2.4.2. We assume that we are given an interesting pair $Z = (Q, Q')$ of squares, where the width and the height of each square is bounded by $2m-1$. We are also given a collection $\mathcal{M}(Z)$ of demand pairs, that, for convenience, we denote by $\mathcal{M}$ from now on.

For each demand pair $(s_j, t_j) \in \mathcal{M}$, we can assume without loss of generality that $s_j \in Q$ and $t_j \in Q'$. We are also given a fractional solution (f, x) that routes $F^* = F(Z)$ flow units between the demand pairs in $\mathcal{M}$, in the grid G. Additionally, we are given a square $A = A(Z)$, containing Q and Q', as defined above. Recall that for any pair $v \in Q$, $v' \in Q'$ of vertices, $d_\infty(v, v') \geq 5m$.

Recall that from our definition of good demand pairs, it is possible that for a pair $(s_j, t_j) \in \mathcal{M}$, $d_\infty(s_j, \Gamma(G)) < 4\sqrt{N}+4$, or $d_\infty(t_j, \Gamma(G)) < 4\sqrt{N}+4$, but not both. We say that (s_j, t_j) is a type-1 pair if $d_\infty(s_j, \Gamma(G)) < 4\sqrt{N}+4$, and we say that it is a type-2 demand pair otherwise.

Let F_1 be the total flow in the LP-solution between the type-1 demand pairs, and F_2 the total flow between type-2 demand pairs. We assume without loss of generality that $F_1 \leq F_2$, so $F_2 \geq F^*/2$. From now on we focus on routing type-2 demand pairs. Abusing the notation, we use $\mathcal{M}$ to denote the set of all type-2 demand pairs.

We next define a sub-grid Q^+ of A, obtained by adding a margin of m around the grid Q, if possible. Specifically, let $R_\ell, R_{\ell'}$ be the rows of G, containing the top and the bottom rows of Q, respectively. Similarly, let $C_b, C_{b'}$ be the columns of G, containing the left and the right columns of Q, respectively. We let Q^+ be the sub-grid of G, spanned by rows $R_{\max\{1,\ell-m\}}, \ldots, R_{\min\{N,\ell'+m\}}$ and columns $C_{\max\{1,b-m\}}, \ldots, C_{\min\{N,b'+m\}}$. From our definition of A, $Q^+ \subseteq A$. Moreover, since $m \leq N$, and since we have assumed that all demand pairs are type-2 good pairs, all source vertices corresponding to the demand pairs in $\mathcal{M}$ are within L_∞ distance at least $4\sqrt{m}+5$ from the boundary of Q^+. We start with the following simple observation.

Observation 2.4.3. *Let L' be a boundary edge of Q', such that $L' \not\subseteq \Gamma(G)$, and let $Y \subseteq V(L')$ be any set of its vertices. Then there is a boundary edge L of Q^+, and a set $\mathcal{P}$ of $|Y|$ disjoint paths in graph A, connecting every vertex of Y to a distinct vertex of L, such that the paths in $\mathcal{P}$ are internally disjoint from $Q^+ \cup Q'$.*

Proof. If the top boundary edge $\tilde{L}$ of Q^+ is separated by at least m rows from the top boundary edge of G, then set $L = \tilde{L}$; otherwise, let L be the bottom boundary edge of Q^+ - notice that it must be separated by at least m rows from the bottom boundary edge of G. Let $X \subseteq V(L)$ be any set of $|Y|$ vertices, and let A' be the graph obtained from A, by deleting all vertices in $Q^+ \setminus X$ and $Q' \setminus Y$ from it. It is enough to show that there is a set $\mathcal{P}$ of $|X| = |Y|$ disjoint paths in A', connecting the vertices of X to the vertices of Y. Let $z = |X|$. From Menger's theorem, if such a set of paths does not exist, then there is a set J of at most $z-1$ vertices, such that in $A' \setminus J$ there is no path from a vertex of $X \setminus J$ to a vertex of $Y \setminus J$. But from our definition of Q^+, Q', and A, it is clear that no such set of vertices exists. $\square$

Let r be the smallest integral power of 2 greater than $4\sqrt{m} + 4$, so $r = \Theta(\sqrt{m})$. Our next step is to partition Q into a collection $\mathcal{X}$ of disjoint sub-grids of size at most $(r \times r)$ each. For $1 \leq p, q \leq m/r$, we let $X_{p,q}$ be the sub-grid of Q, spanned by rows $R_{(p-1)r+1}, \ldots, R_{pr}$ and columns $C_{(q-1)r+1}, \ldots, C_{qr}$ of Q. We then let

$$\mathcal{X} = \left\{ X_{p,q} \mid 1 \leq p, q \leq m/r \right\}.$$

The next theorem is key to finding the final routing.

Theorem 2.4.4. *There is a subset $\mathcal{M}_1 \subseteq \mathcal{M}$ of $\Omega(F^*/n^{1/4})$ demand pairs, such that every vertex of $Q \cup Q'$ participates in at most one demand pair. Moreover, if S_1 and T_1 denote the sets of all source and all destination vertices of the pairs in $\mathcal{M}_1$, respectively, then:*

- *for every square $X_{p,q} \in \mathcal{X}$, at most one vertex of $X_{p,q}$ belongs to S_1; and*
- *there is a boundary edge L' of Q', with $L' \not\subseteq \Gamma(G)$, and a set $\mathcal{P}_1$ of node-disjoint paths in graph Q', connecting every vertex of T_1 to a distinct vertex of L'.*

Proof. Let U be the union of the boundary edges L' of Q', with $L' \not\subseteq \Gamma(G)$. We build a flow network $\mathcal{N}$, starting with the graph Q'. We add a source vertex a, that connects to every vertex in U with a directed edge. Let $S \subseteq Q$ be the set of all vertices participating in the demand pairs in $\mathcal{M}$ as sources. Observe that each vertex $s \in S$ may participate in several demand pairs in $\mathcal{M}$. We add every vertex $s \in S$ to graph $\mathcal{N}$, and for each demand pair $(s,t) \in \mathcal{M}$, we connect t to s with a directed edge. Next, for each square $X_{p,q} \in X$, we add a vertex $u_{p,q}$, and we connect every vertex $s \in S \cap X_{p,q}$ to $u_{p,q}$ with a directed edge. Finally, we add a destination vertex b, and connect every vertex $u_{p,q}$ for $1 \leq p, q \leq m/r$ to b with a directed edge. We set all vertex-capacities (except for those of a and b) to 1.

We claim that there is a valid flow of value $\Omega(F^*/\sqrt{m})$ from a to b in $\mathcal{N}$. Indeed, consider the multicommodity flow between the demand pairs in $\mathcal{M}$, given by our current LP-solution. For each (s_j, t_j)-pair in $\mathcal{M}$, we send $x_j/4r$ flow units on the edge (t_j, s_j) in $\mathcal{N}$. For each flow-path $P \in \mathcal{P}_j$, notice that P must contain some vertex of U. Let v be the last such vertex on P (where we view P as directed from s_j to t_j), and let P' be the sub-path of P from v to t_j. We send $f(P)/4r$ flow units on every edge in P'. For every vertex $v \in U$, we set the flow on the edge (a, v) to be the total flow leaving the vertex v; for each vertex $s \in S$, with $s \in X_{p,q}$, we set the flow on the edge $(s, u_{p,q})$ to be the total amount of flow entering s. The flow on edge $(u_{p,q}, b)$ is then set to the total amount of flow entering $u_{p,q}$. Notice that for each square $X_{p,q}$, every flow-path originating at a vertex of $S \cap X_{p,q}$ must cross the boundary $\Gamma(X_{p,q})$ of $X_{p,q}$, that contains at most $4r$ vertices. Therefore, the total amount of flow in the original LP-solution leaving the vertices in $S \cap X_{p,q}$ is at most $4r$. It is now easy to see that we have defined a valid a-b flow of value $\tilde{F} = \Omega(F^*/\sqrt{m})$.

From the integrality of flow, there is an integral flow of the same value in $\mathcal{N}$. Let $\mathcal{P}$ be the set of paths carrying one flow unit in the resulting flow. Then there is a boundary edge L' of Q', such that $L' \not\subseteq \Gamma(G)$, with at least $\tilde{F}/4$ of the paths in $\mathcal{P}$ containing a vertex of L'. Let $\mathcal{P}' \subseteq \mathcal{P}$ be this set of paths.

We are now ready to define the final set $\mathcal{M}_1$ of the demand pairs, and the corresponding set $\mathcal{P}_1$ of paths. Consider some path $P \in \mathcal{P}'$, and let (t, s) be the unique edge with $(s, t) \in \mathcal{M}$ on this path. We then add (s, t) to $\mathcal{M}_1$. Let P' be the sub-path of P, starting from the last vertex on P that belongs to L', to vertex t. We add P' to $\mathcal{P}_1$. This finishes the definition of the subset $\mathcal{M}_1$ of demand pairs, and the corresponding set $\mathcal{P}_1$ of paths. $\square$

If $|\mathcal{M}_1| > \sqrt{m}$, then we discard pairs from $\mathcal{M}_1$, until $|\mathcal{M}_1| = \sqrt{m}$ holds, and we update the sets S_1, T_1, and $\mathcal{P}_1$ accordingly.

For $w, w' \in \{0, 1\}$, let $S_{w,w'}$ be a subset containing all vertices $s \in S_1$ lying in the squares $X_{p,q}$, where $p = w \mod 2$ and $q = w' \mod 2$. Then there is some choice of $w, w' \in \{0, 1\}$, so that $|S_{w,w'}| \geq |S_1|/4$. We let $S_2 = S_{w,w'}$ for this choice of w, w', and we define $\mathcal{M}_2 = \{(s,t) \in \mathcal{M}_1 \mid s \in S_2\}$, and T_2 as the set of all destination vertices for the pairs in $\mathcal{M}_2$.

Let $\mathcal{P}_2 \subseteq \mathcal{P}_1$ be the set of paths originating from the vertices of T_2. Let Y be the set of endpoints of the paths in $\mathcal{P}_2$ that lie on the boundary edge L' of Q'. Finally, from Observation 2.4.3, there is a boundary edge L of Q^+, a set Y' of $|Y|$ vertices of L, and a set $\mathcal{P}_2'$ of disjoint paths in A, connecting every vertex in Y to a distinct vertex of Y', so that the paths in $\mathcal{P}_2'$ are internally disjoint from $Q^+ \cup Q'$. By concatenating the paths in $\mathcal{P}_2$ and $\mathcal{P}_2'$, we obtain a new set $\mathcal{P}^*$ of paths, connecting every vertex of T_2 to a distinct vertex of Y'. Denote $\mathcal{M}_2 = \left\{(s_j, t_j)\right\}_{j=1}^{|\mathcal{M}_2|}$, and let $u_j \in Y'$ be the vertex where the path $P_j \in \mathcal{P}^*$, originating at vertex t_j, terminates. Notice that all vertices in S_2 are now at the L_∞-distance at least $r > 4\sqrt{m} + 4$ from each other, and at distance at least $4\sqrt{m} + 5$ from the boundaries of Q^+, and $|\mathcal{M}_1| \leq \sqrt{m}$. From Theorem 2.3.1, we can efficiently find a set $\mathcal{Y}$ of disjoint paths in graph Q^+, connecting every vertex $s_j \in S_2$ to the corresponding vertex $u_j \in Y'$. By concatenating the paths in $\mathcal{P}^*$ and $\mathcal{Y}$, we obtain a set of paths routing all pairs in $\mathcal{M}_2$.

Notice that from the above discussion,

$$|\mathcal{M}_2| = \min\left\{\Omega(\sqrt{m}), \Omega(F^*/\sqrt{m})\right\}.$$

It is easy to see that $F^* \leq 4m$, since every flow-path routing a pair in $\mathcal{M}$ must cross the boundary of Q'. Therefore,

$$|\mathcal{M}_2| = \Omega(F^*/\sqrt{m}).$$

Since $m \leq N = \sqrt{n}$, our algorithm routes $\Omega(F^*/n^{1/4})$ demand pairs.

2.4.2 Routing the Bad Pairs

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

Theorem 2.4.5. *Let $d^* \geq 1$ be a parameter, and let $(G, \mathcal{M})$ be an instance of the* NDP *problem, where G is an $(N \times N)$ grid, and $\mathcal{M} = \{(s_1, t_1), \ldots, (s_k, t_k)\}$. Assume further that for each demand pair (s_j, t_j), both $d_\infty(s_j, \Gamma(G)), d_\infty(t_j, \Gamma(G)) < d^*$. Then there is an efficient algorithm that finds an $O(d^*)$-approximate solution to the* NDP *instance $(G, \mathcal{M})$.*

Notice that by setting $d^* = 4\sqrt{N} + 4$, so that $d^* = \Theta(n^{1/4})$, we obtain an $O(n^{1/4})$-approximate solution for NDP instances on grid graphs, where all demand pairs are bad.

The rest of this section is dedicated to proving Theorem 2.4.5. Let T be the set of all vertices participating in the bad demand pairs. We call the vertices in T *terminals*. Let L_1, L_2, L_3, L_4 be the four boundary edges of the grid G. Notice that a terminal $t \in T$ may be within distance d^* from up to two boundary edges.

For each terminal $t \in T$, we let $L(t)$ be any boundary edge of G, such that

$$d_\infty(t, V(L(t))) < d^*.$$

We now partition all bad demand pairs into 16 subsets: for $1 \leq p, q \leq 4$, set $\mathcal{M}_{p,q}$ contains all pairs (s_j, t_j), where $L(s_j) = L_p$ and $L(t_j) = L_q$.

Let OPT be the optimal solution to the NDP instance. For every possible choice of $1 \leq p, q \leq 4$, let $\mathsf{OPT}_{p,q}$ be the optimal solution restricted to the pairs in $\mathcal{M}_{p,q}$. Clearly, there is a choice of p and q, such that at least $|\mathsf{OPT}|/16$ of the demand pairs routed in OPT belong to $\mathcal{M}_{p,q}$, and so $|\mathsf{OPT}_{p,q}| \geq \mathsf{OPT}/16$. For each choice of values $1 \leq p, q \leq 4$, we show an algorithm that routes $\Omega(\mathsf{OPT}_{p,q}/d^*)$ demand pairs in $\mathcal{M}_{p,q}$. We then take the best of these solutions, thus obtaining an $O(d^*)$-approximation algorithm.

Fix some $1 \leq p, q \leq 4$. We consider three cases.

The first case happens when L_p and L_q are two distinct opposing boundary edges of G. We assume without loss of generality that L_p is the top, and L_q is the bottom boundary of G. We say that a subset $\mathcal{M}' \subseteq \mathcal{M}_{p,q}$ of demand pairs is a *monotone matching*, if the following holds. Let S' be the set of all source vertices, and T' the set of all destination vertices, participating in the pairs in $\mathcal{M}'$. Then:

- All vertices of S' lie in distinct columns of G;
- All vertices of T' lie in distinct columns of G;
- Every vertex of $S' \cup T'$ participates in exactly one demand pair; and
- For any two distinct pairs $(s_i, t_i), (s_j, t_j) \in \mathcal{M}'$, $\text{col}(s_i) < \text{col}(s_j)$ iff $\text{col}(t_i) < \text{col}(t_j)$.

The following observation is immediate.

Observation 2.4.6. *Let $\mathcal{M}' \subseteq \mathcal{M}_{p,q}$ be any monotone matching with $|\mathcal{M}'| \leq N/2$. Then there is an efficient algorithm to route all pairs in $\mathcal{M}'$ in graph G.*

Our algorithm then simply computes the largest monotone matching $\mathcal{M}' \subseteq \mathcal{M}_{p,q}$, using standard dynamic programming.

We maintain a dynamic programming table Π, that contains, for all $0 \leq x, y \leq N$, an entry $\Pi(x, y)$, whose value is the size of the largest monotone matching $\mathcal{M}(x, y) \subseteq \mathcal{M}_{p,q}$, such that every source vertex s participating in pairs in $\mathcal{M}(x, y)$ has $1 \leq \text{col}(s) \leq x$, and every destination vertex t participating in pairs in $\mathcal{M}(x, y)$ has $1 \leq \text{col}(t) \leq y$. We fill the entries of the table from smaller to larger values of $x + y$, initializing $\Pi(x, 0) = 0$ and $\Pi(0, y) = 0$ for all x and y.

Entry $\Pi(x, y)$ is computed as follows: If there is a pair $(s, t) \in \mathcal{M}_{p,q}$, with $\text{col}(s) = x$ and $\text{col}(t) = y$, then we let $\Pi(x, y)$ be the maximum of $\Pi(x-1, y-1) + 1$, $\Pi(x-1, y)$, and $\Pi(x, y-1)$. Otherwise, $\Pi(x, y)$ is the maximum of $\Pi(x-1, y)$, and $\Pi(x, y-1)$. The size of the largest monotone matching $\mathcal{M}' \subseteq \mathcal{M}_{p,q}$ is then stored in $\Pi(N, N)$, and we can use standard techniques to compute the matching itself.

Finally, we show that there is a large enough monotone matching $\mathcal{M}' \subseteq \mathcal{M}_{p,q}$.

Lemma 2.4.7. *There is a monotone matching $\mathcal{M}' \subseteq \mathcal{M}_{p,q}$ of cardinality $\Omega(\mathsf{OPT}_{p,q}/d^*)$.*

Proof. For every source vertex s of a demand pair in $\mathcal{M}_{p,q}$, let $P(s)$ denote the segment of the column in which s lies, from the first row of G to s itself. Similarly, for each destination vertex t of a demand pair in $\mathcal{M}_{p,q}$, let $P(t)$ denote the segment of the column in which t lies, from t to the last row of G.

Consider the solution $\mathsf{OPT}_{p,q}$, and let $\mathcal{M}^* \subseteq \mathcal{M}_{p,q}$ be the set of the demand pairs routed in it. For each pair $(s_i, t_i) \in \mathcal{M}^*$, let $P_i \in \mathsf{OPT}_{p,q}$ be the path routing this demand pair in the solution. We say that two terminal pairs (s_i, t_i) and (s_j, t_j) in $\mathcal{M}^*$ *have a conflict* iff either P_i contains a vertex of $P(s_j) \cup P(t_j)$, or P_j contains a vertex of $P(s_i) \cup P(t_i)$.

Let H be a directed graph, that contains a vertex v_i for every pair $(s_i, t_i) \in \mathcal{M}^*$, and a directed edge (v_i, v_j) iff path P_i intersects $P(s_j)$ or $P(t_j)$. Notice that the length of every path $P(s_j)$ or $P(t_j)$ is bounded by d^*, and so every vertex of H has in-degree bounded by $2d^*$. Therefore, any vertex-induced sub-graph H' of H with z vertices has at most $2d^*z$ edges, and contains at least one vertex whose degree (including the incoming and the outgoing edges) is at most $4d^*$.

We now construct the set $\mathcal{M}'$ of demand pairs as follows. Start with $\mathcal{M}' = \emptyset$. While H is non-empty, let v_i be any vertex of degree at most $4d^*$. Delete v_i and all its neighbors from H, and add the pair (s_i, t_i) to $\mathcal{M}'$.

When this procedure terminates, it is easy to see that $\mathcal{M}'$ contains at least

$$|\mathsf{OPT}_{p,q}|/(4d^* + 1) = \Omega(\mathsf{OPT}_{p,q}/d^*)$$

demand pairs. Moreover, if (s_i, t_i) and (s_j, t_j) are distinct pairs in $\mathcal{M}'$, then there is no conflict between (s_i, t_i) and (s_j, t_j). In particular, this means that $\mathrm{col}(s_i) \neq \mathrm{col}(s_j)$ and $\mathrm{col}(t_i) \neq \mathrm{col}(t_j)$. Moreover, if we assume that $\mathrm{col}(s_i) < \mathrm{col}(s_j)$, then $\mathrm{col}(t_i) < \mathrm{col}(t_j)$ must hold: this is since the union of $P_i, P(s_i)$ and $P(t_i)$ partitions the face defined by $\Gamma(G)$ into two sub-faces, and both s_j and t_j must be contained in a single sub-face, as the path P_j cannot intersect the paths $P_i, P(s_i)$ and $P(t_i)$. $\square$

This concludes the analysis of the algorithm for the case where L_p and L_q are two distinct opposing boundary edges of G. The case where L_p and L_q are two adjacent boundary edges of G is dealt with very similarly.

Finally, we consider the case where $L_p = L_q$. Assume without loss of generality that L_p is the bottom boundary of the grid. We say that a subset $\mathcal{M}' \subseteq \mathcal{M}_{p,q}$ is a *nested matching*, if the following holds. Let S' be the set of all source vertices, and T' the set of all destination vertices, participating in the pairs in $\mathcal{M}'$. Then:

- All vertices of S' lie in distinct columns of G;
- All vertices of T' lie in distinct columns of G;
- Every vertex of $S' \cup T'$ participates in exactly one demand pair; and
- For any two distinct pairs $(s_i, t_i), (s_j, t_j) \in \mathcal{M}'$, with $\mathrm{col}(s_i)$ lying to the left of $\mathrm{col}(s_j)$, either both $\mathrm{col}(s_i), \mathrm{col}(t_i)$ lie to the left of both $\mathrm{col}(s_j), \mathrm{col}(t_j)$, or both $\mathrm{col}(s_j), \mathrm{col}(t_j)$ lie between $\mathrm{col}(s_i)$ and $\mathrm{col}(t_i)$, or both $\mathrm{col}(s_i), \mathrm{col}(t_i)$ lie between $\mathrm{col}(t_j)$ and $\mathrm{col}(s_j)$.

It is immediate to see that any nested matching $\mathcal{M}' \subseteq \mathcal{M}_{p,q}$, with $|\mathcal{M}'| \leq N/2$ can be routed efficiently in G. As before, we can find a largest-cardinality nested matching $\mathcal{M}' \subseteq \mathcal{M}_{p,q}$ using standard dynamic programming techniques. The following claim will then finish the proof.

Claim 2.4.8. *There is a nested matching $\mathcal{M}' \subseteq \mathcal{M}_{p,q}$ of cardinality $\Omega(\mathsf{OPT}_{p,q}/d^*)$.*

Proof. We construct the paths $P(s), P(t)$, the graph H', and the matching $\mathcal{M}'$ corresponding to an independent set in H' exactly as in the proof of Lemma 2.4.7. As before,

$$|\mathcal{M}'| = \Omega(\mathsf{OPT}_{p,q}/d^*).$$

Moreover, if (s_i,t_i) and (s_j,t_j) are distinct pairs in $\mathcal{M}'$, then there is no conflict between (s_i,t_i) and (s_j,t_j). As before, this means that $\mathrm{col}(s_i) \neq \mathrm{col}(s_j)$ and $\mathrm{col}(t_i) \neq \mathrm{col}(t_j)$.

Assume now $\mathrm{col}(s_i)$ lies to the left of $\mathrm{col}(s_j)$. Then the union of $P_i, P(s_i)$ and $P(t_i)$ partitions the face defined by $\Gamma(G)$ into two sub-faces, and both s_j and t_j must be contained in a single sub-face, as before. In this case, this means that either both $\mathrm{col}(s_i), \mathrm{col}(t_i)$ lie to the left of both $\mathrm{col}(s_j), \mathrm{col}(t_j)$, or both $\mathrm{col}(s_j), \mathrm{col}(t_j)$ lie between $\mathrm{col}(s_i)$ and $\mathrm{col}(t_i)$, or both $\mathrm{col}(s_i), \mathrm{col}(t_i)$ lie between $\mathrm{col}(t_j)$ and $\mathrm{col}(s_j)$. □

2.4.3 Putting Everything Together

Our algorithm for an input NDP instance $(G, \mathcal{M})$, where G is an $(N \times N)$ grid, applies the algorithm from Section 2.4.1 to the set $\mathcal{M}'$ of the good demand pairs, and the algorithm from Section 2.4.2 to the set $\mathcal{M}''$ of the bad demand pairs, and returns the better of the two solutions. Since each of the two algorithms achieves an $O(n^{1/4} \log n)$-approximation to the corresponding problem, and since at least half of the demand pairs routed in the optimal solution are either all good pairs, or all bad pairs, we obtain an $O(n^{1/4} \log n)$-approximation overall.

2.5 Integrality Gap of (LP-flow) for Good Pairs

We prove that the integrality gap of (LP-flow) is $\Omega(n^{1/8})$ even when all of the terminals are far from the grid boundary. We note that the family of instances that we construct here was previously used by Cutler and Shiloach [22], to provide a lower bound on the size of permutation layouts. Our analysis also closely follows theirs.

Given any integer $p > 10$, let $k = p^2$ and $N = 6k$. We show that the integrality gap of (LP-flow) on the $(N \times N)$ grid G, where all terminals are within distance at least $N/6$ from $\Gamma(G)$ is $\Omega(k^{1/4}) = \Omega(n^{1/8})$.

In order to define the demand pairs, we let S be any set of k consecutive vertices on row R_{2k} of G, where all vertices are at distance at least $2k$ from both the left and the right boundary of G, and define a set T of k consecutive vertices on row R_{4k} similarly.

We partition the set S into p subsets $S_1, \dots, S_p$ of p consecutive vertices each, where for $1 \le i, j \le p$, the jth vertex in set S_i is denoted by $s_{i,j}$. Similarly, we partition T into p subsets $T_1, \dots, T_p$ of p consecutive vertices each, and for $1 \le i, j \le p$, the jth vertex in set T_i is denoted by $t_{i,j}$. The set $\mathcal{M}$ of the demand pairs is then:

$$\mathcal{M} = \left\{(s_{i,j}, t_{j,i}) \mid 1 \le i, j \le p\right\}.$$

It is easy to see that there is a solution to (LP-flow) of value $k/3$: for each pair $(s_{i,j}, t_{j,i})$, we send $1/3$ flow unit on the path P, lying in the union $\mathrm{col}(s_{i,j}), \mathrm{col}(t_{j,i})$ and R_{ip+j}, that connects $s_{i,j}$ to $t_{j,i}$. We next show that the value of any integral solution is $O(k^{3/4})$, thus establishing the integrality gap of $\Omega(k^{1/4})$.

In our analysis we use the notions of graph drawing and graph crossing number. A drawing of a graph H in the plane is a mapping, in which every vertex of H is mapped into a point in the plane, and every edge into a continuous curve connecting the images of its endpoints, such that no three curves meet at the same point, and no curve contains an image of any vertex other than its endpoints. A *crossing* in such a drawing is a point where the images of two edges intersect, and the *crossing number* of a graph H, denoted by $\mathsf{OPT}_{\mathsf{cr}}(H)$, is the smallest number of crossings achievable by any drawing of H in the plane. We use the following well-known theorem [3, 32].

Theorem 2.5.1. *For any graph $H = (V, E)$ with $|E| > 7|V|$, $\mathsf{OPT}_{\mathsf{cr}}(G) \ge \frac{|E|^3}{29|V|^2}$.*

Let OPT denote the optimal integral solution for the instance $(G, \mathcal{M})$, $\mathcal{M}^* \subseteq \mathcal{M}$ the set of the demand pairs routed by OPT, and let $x = |\mathsf{OPT}|$. We define two bipartite graphs. The first bipartite graph, $H = (S, T, E^*)$ is defined over the sets S and T of the source and the destination vertices of $\mathcal{M}$, and it contains an edge $e = (s,t)$ for every pair $(s,t) \in \mathcal{M}^*$. The second graph is $H' = (A, B, E')$, where $A = \{v_1, \ldots, v_p\}$, $B = \{u_1, \ldots, u_p\}$, and E' contains all edges (v_i, u_j), where $(s_{i,j}, t_{j,i}) \in \mathcal{M}^*$. The following claim is central to our analysis.

Claim 2.5.2. *There is a drawing of H' with at most $2px$ crossings.*

If $|E'| < 14p$, then $|\mathsf{OPT}| = O(\sqrt{k})$ and we are done, so we assume that $|E'| \geq 14p$. Then from Theorem 2.5.1,

$$\mathsf{OPT}_{\mathsf{cr}}(H') \geq \frac{x^3}{29p^2},$$

while from Claim 2.5.2,

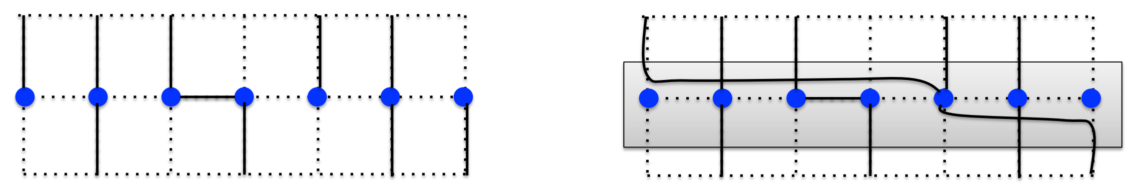
$$\mathsf{OPT}_{\mathsf{cr}}(H') \leq 2px.$$

Therefore, $x = O(p^{3/2}) = O(k^{3/4})$. It now remains to prove Claim 2.5.2.

Proof of of Claim 2.5.2. Notice that the natural drawing of the grid G, together with the solution OPT to the NDP instance gives a planar drawing φ of the graph H in the plane.

For each $1 \leq i \leq p$, let $S'_i \subseteq S_i$ be the set of the sources that have an edge incident to them in E^*, and define $T'_i \subseteq T_i$ similarly. Let $x_i = |S'_i|$ and $y_i = |T'_i|$. For each $1 \leq i \leq p$, if $x_i = 0$, then the vertex v_i of H', corresponding to S_i is an isolated vertex, and we can draw it anywhere. Otherwise, let $s_{i,j} \in S'_i$ be any vertex. We draw v_i at $\varphi(s_{i,j})$. Let $I(i)$ be the segment of row R_{2k} containing the vertices of S_i, and no other vertices. Let L_i be a very thin strip (of height 1/10) around the segment $I(i)$ (see Figure 2.4). We alter the drawings of all edges in E^*, originating at the vertices of S'_i, so that they now originate at $\varphi(s_{i,j})$, by re-routing them inside the strip L_i.

Since the number of paths in OPT containing the vertices of S_i is bounded by p, it is easy to do so, by introducing at most px_i crossings. We perform the same transformation for the sets T_i of destination vertices, and obtain a drawing of the graph H' with at most $p\sum_{i=1}^{p}(x_i + y_i) \leq 2px$ crossings. □


(a)
t

(b)
t

Figure 2.4: Altering the drawing around S_i.

2.6 Approximation Algorithm for EDP on Wall Graphs

In this section we show that the algorithm from Section 2.4 can be adapted to give an $O(n^{1/4} \cdot \log n)$-approximation on wall graphs of width and height $N = \Omega(\sqrt{n})$ for EDP. In order to construct a wall W of height h and width r (or an $(h \times r)$- wall), we start from a grid of height h and width $2r$. Consider some column C_j of the grid, for $1 \leq j \leq r$, and let $e_1^j, e_2^j, \ldots, e_{h-1}^j$ be the edges of C_j, in the order of their appearance on C_j, where e_1^j is incident on $v(1, j)$. If j is odd, then we delete from the graph all edges e_i^j where i is even. If j is even, then we delete from the graph all edges e_i^j where i is odd. We process each column C_j of the grid in this manner, and in the end delete all vertices of degree 1. The resulting graph is a wall of height h and width r, that we denote by W (See Figure 4.2).

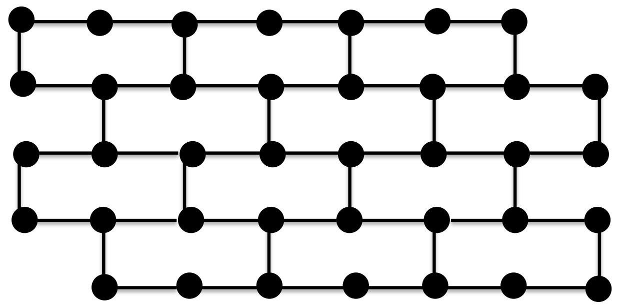


Figure 2.5: A wall graph.

Let E_1 be the set of edges of W that correspond to the horizontal edges of the original grid, and let E_2 be the set of the edges of W that correspond to the vertical edges of the original grid. The sub-graph of W induced by E_1 is a collection of h node-disjoint paths, that we refer to as the rows of W. We denote these rows by $R_1, \ldots, R_h$, where for $1 \leq i \leq h$, R_i is incident on $v(i,1)$. Let V_1 denote the set of all vertices in the first row of W, and V_h the set of vertices in the last row of W. There is a unique set $\mathcal{C}$ of r node-disjoint paths, where each path $C \in \mathcal{C}$ starts at a vertex of V_1, terminates at a vertex of V_h, and is internally disjoint from $V_1 \cup V_h$. We refer to these paths as the columns of W. We order these columns from left to right, and denote by C_j the jth column in this ordering, for $1 \leq j \leq r$. The sub-graph $\Gamma(W) = R_1 \cup C_1 \cup R_h \cup C_r$ of W is a simple cycle, that we call the boundary of W.

For every vertex $v \in V(W)$, we let $\mathrm{col}(v)$ and $\mathrm{row}(v)$ denote the column and the row of W to which v belongs. As before, for a pair $u, v \in V(W)$ of vertices, we define:

$$d_\infty(u,v) = \max\left\{|\,\mathrm{col}(v) - \mathrm{col}(u)|, |\,\mathrm{row}(v) - \mathrm{row}(u)|\right\},$$

and for a vertex v and a subset $U \subseteq V(W)$ of vertices, we let

$$d_\infty(v,U) = \min_{u \in U}\left\{d_\infty(u,v)\right\}.$$

Assume now that we are given an $(N \times N)$-wall graph $G = (V,E)$, so $n = |V| = \Theta(N^2)$, and a collection $\mathcal{M} = \{(s_i,t_i)\}_{i=1}^k$ of demand pairs. As before, we say that a demand pair (s_i,t_i) is bad if both $d_\infty(s_i,\Gamma(G)), d_\infty(t_i,\Gamma(G)) < 4\sqrt{N}+4$, and we say that it is good otherwise. Let $\mathcal{M}', \mathcal{M}'' \subseteq \mathcal{M}$ denote the sets of the good and the bad demand pairs in $\mathcal{M}$, respectively. We find an approximate solution to each of the two sub-problems, defined by $\mathcal{M}'$ and $\mathcal{M}''$, separately, and take the better of the two solutions.

The algorithm for the bad pairs remains exactly the same as the algorithm from Section 2.4.2. We now focus on the problem defined by the set $\mathcal{M}'$ of the good pairs. Let G' be the $(N \times N)$-grid obtained from G, by contracting, for each $1 \leq i, j \leq N$, the unique edge $e \in R_i \cap C_j$, and consider the NDP problem instance $(G', \mathcal{M}')$. Any collection $\mathcal{P}'$ of node-disjoint paths in G', routing a subset $\tilde{\mathcal{M}} \subseteq \mathcal{M}'$ of the demand pairs immediately gives a collection $\mathcal{P}''$ of edge-disjoint paths in G, routing the same subset of the demand pairs. Moreover, it is easy to see that there is an LP-solution to (LP-flow) on instance $(G', \mathcal{M}')$ of value $\mathsf{OPT}'/2$, where OPT' is the optimal solution for the EDP instance $(G, \mathcal{M}')$. Indeed, for every path $P \in \mathsf{OPT}'$, we simply set $f(P') = 1/2$, where P' is the path of G' corresponding to the path P of G, and for every demand pair (s_j, t_j) routed by OPT', we set $x_j = 1/2$. It is immediate to verify that this is a feasible solution to (LP-flow) on NDP instance $(G', \mathcal{M}')$, of value $\mathsf{OPT}'/2$. We then use the algorithm from Section 2.4.1 to find an $O(n^{1/4} \log n)$-approximation solution to $(G', \mathcal{M}')$, which in turn gives an $O(n^{1/4} \log n)$-approximation solution to the EDP instance $(G, \mathcal{M}')$.

CHAPTER 3

ROUTING ON A DISC AND A CYLINDER

3.1 Introduction

In this chapter, we present approximation algorithms for NDP-Disc and NDP-Cylinder, two special cases of NDP-Planar. NDP-Disc is the NDP problem on a disc, where the input graph G can be drawn in a disc, such that all terminals lie on the boundary of the disc. Likewise, NDP-Cylinder denotes the NDP problem on a cylinder, obtained from the sphere by removing two disjoint open discs from it. We assume that the input graph G can be drawn on the cylinder with all of the source terminals appearing on the boundary of one of the open discs and all of the destination terminals on the boundary of the other.

Theorem 3.1.1. *There is an efficient $O(\log k)$-approximation algorithm for* NDP-Disc *and* NDP-Cylinder.

We note that Robertson and Seymour [39] showed an algorithm that, given an instance of NDP-Disc or NDP-Cylinder, decides whether all of the demand pairs in $\mathcal{M}$ can be routed simultaneously via node-disjoint paths, and efficiently find the routing if it exists. In fact, they gave exact characterizations of instances of NDP-Disc and NDP-Cylinder for which all demand pairs can be routed, respectively. A linear time algorithm for NDP-Cylinder is shown in [37], while an $O(n \log n)$-time algorithm is presented in [43] for the more general Steiner forest problem for planar graphs with the terminals lying on two face boundaries.

However, for the optimization version of NDP-Disc and NDP-Cylinder, it is not known whether the two problems are even NP-hard. The $O(\log k)$-approximation algorithm for the two problems was developed as a subroutine for an $\tilde{O}(n^{9/19})$-approximation algorithm for NDP-Planar and appears in the joint work with Chuzhoy and Li [17].

3.2 Preliminaries

Routing on a disc. Assume we are given an instance $(G, \mathcal{M})$ of NDP-Disc, where G is drawn in a disc D whose boundary is denoted by C. We start with the following two definitions.

Definition 1. *We say that two demand pairs $(s,t),(s',t') \in \mathcal{M}$ cross iff either $\{s,t\} \cap \{s',t'\} \neq \emptyset$, or (s,s',t,t') appear on C in this circular order. We say that the set $\mathcal{M}$ of demand pairs is non-crossing if no two demand pairs in $\mathcal{M}$ cross.*

Definition 2. *Let C be a closed simple curve and $\mathcal{M}$ a set of demand pairs with all vertices of $\mathcal{T}(\mathcal{M})$ lying on C. We say that $\mathcal{M}$ is an* r-split collection of demand pairs *with respect to C, iff there is a partition $\mathcal{M}_1, \ldots, \mathcal{M}_r$ of the demand pairs in $\mathcal{M}$, and there is a partition $\{\sigma_1, \sigma_2, \ldots, \sigma_{2r}\}$ of C into disjoint segments, such that $\sigma_1, \ldots, \sigma_{2r}$ appear on C in this circular order, and for each $1 \leq i \leq r$, for every demand pair $(s,t) \in \mathcal{M}_i$, either $s \in \sigma_{2i-1}$ and $t \in \sigma_{2i}$, or vice versa.*

The following lemma allows us to partition any set of demand pairs into a small collection of split sets.

Lemma 3.2.1. *There is an efficient algorithm, that, given a closed simple curve C in the plane and a set $\mathcal{M}$ of κ demand pairs, whose corresponding terminals lie on C, computes a partition $\mathcal{M}^1, \ldots, \mathcal{M}^{4\lceil \log \kappa \rceil}$ of $\mathcal{M}$, such that for each $1 \leq i \leq 4\lceil \log \kappa \rceil$, set $\mathcal{M}^i$ is r_i-split with respect to C for some integer $r_i \geq 0$.*

Proof. We denote by $\mathcal{T}$ the set of all vertices participating in the demand pairs in $\mathcal{M}$, and we refer to them as terminals. Consider any demand pair $(s,t) \in \mathcal{M}$, and let $\sigma(s,t), \sigma'(s,t)$ be the two segments of C whose endpoints are s and t. We assume without loss of generality that $|\sigma(s,t) \cap \mathcal{T}| \leq |\sigma'(s,t) \cap \mathcal{T}|$, and we denote $\delta(s,t) = |\sigma(s,t) \cap \mathcal{T}| - 1$. By possibly renaming the terminals s and t, we assume that s appears before t on $\sigma(s,t)$ as we traverse it in counter-clock-wise direction along C.

Our first step is to partition the demand pairs in $\mathcal{M}$ into $\lceil \log \kappa \rceil$ subsets $\mathcal{N}_1, \ldots, \mathcal{N}_{\lceil \log \kappa \rceil}$, as follows. For each $1 \leq i \leq \lceil \log \kappa \rceil$, $\mathcal{N}_i$ contains all demand pairs (s, t) with $2^{i-1} \leq \delta(s, t) < 2^i$. In order to complete the proof of the lemma, it is enough to show that for each $1 \leq i \leq \lceil \log \kappa \rceil$, we can partition $\mathcal{N}_i$ into four sets of demand pairs, each of which is r-split, for some integer r.

Fix some $1 \leq i \leq \lceil \log \kappa \rceil$, and assume that $\mathcal{T} = \{v_1, \ldots, v_{2\kappa}\}$, where the vertices are indexed in the circular order of their appearance on C. We let A contain all vertices v_j, where $j = 1$ modulo 2^{i-1}. For convenience, we denote $A = \{a_1, \ldots, a_z\}$, and we assume that the vertices of A appear in this circular order on C, with $a_1 = v_1$.

If $|A| = 1$, then $|\mathcal{T}| \leq 2^{i-1}$, and so $\delta(s, t) \leq 2^{i-2}$ for all $(s, t) \in \mathcal{M}$, and $\mathcal{N}_i = \emptyset$.

If $|A| = 2$, then let β, β' be the two segments of C between a_1 and a_2, where β contains a_1 but not a_2, and β' contains a_2 but not a_1. Then for every pair $(s, t) \in \mathcal{N}_i$, one of the two terminals lies on β and the other on β', and so $\mathcal{N}_i$ is 1-split. We assume from now on that $z \geq 3$. For each $1 \leq j \leq z - 1$, let β_j be the segment of C from a_j and a_{j+1}, as we traverse C in the counter-clock-wise order, so that β_j includes a_j but excludes a_{j+1}. We let β_z be the segment of C from a_z to a_1, as we traverse C in the counter-clock-wise order, so that β_z includes a_z, but not a_1. Then for every segment β_j with $1 \leq j < z$, $|\beta_j \cap \mathcal{T}| = 2^{i-1}$, while $|\beta_z \cap \mathcal{T}| \leq 2^{i-1}$. Notice that for every demand pair $(s, t) \in \mathcal{N}_i$, one of the following must happen:

1. $s \in \beta_j$, $t \in \beta_{j+1}$ for some $1 \leq j \leq z$, where we treat $z + 1$ as 1; or
2. $s \in \beta_z$, $t \in \beta_2$; or
3. $s \in \beta_{z-1}$, $t \in \beta_1$.

We are now ready to partition $\mathcal{N}_i$ into four subsets. The first subset, $\mathcal{N}_i^1$, contains all pairs $(s, t) \in \mathcal{N}_i$ with $s \in \beta_{z-1}$, $t \in \beta_1$. The second set, $\mathcal{N}_i^2$, contains all pairs $(s, t) \in \mathcal{N}_i$, with $s \in \beta_z$, $t \in \beta_1 \cup \beta_2$. It is immediate to verify that each of these two sets is 1-split. The third set, $\mathcal{N}_i^3$, contains all pairs $(s,t) \in \mathcal{N}_i \setminus (\mathcal{N}_i^1 \cup \mathcal{N}_i^2)$, where $s \in \beta_j$, for an odd index $1 \le j < z$. This set is $\lfloor z/2 \rfloor$-split, with the segments $\beta_1, \ldots, \beta_z$ giving the splitting. The last set, $\mathcal{N}_i^4$, contains all pairs $(s,t) \in \mathcal{N}_i \setminus (\mathcal{N}_i^1 \cup \mathcal{N}_i^2 \cup \mathcal{N}_i^3)$. This set is similarly $\lfloor z/2 \rfloor$-split, since for all $(s,t) \in \mathcal{N}_i^4$, $s \in \beta_j$ for an even index $1 \le j < z$. Overall, we obtain a partition of $\mathcal{M}$ into $4 \lceil \log \kappa \rceil$ sets, each of which is r-split for some integer r. □

Routing on a cylinder. Assume we are given an instance $(G, \mathcal{M})$ of NDP-Cylinder, where we are given a cylinder Σ, obtained from the sphere by removing two disjoint open discs (caps) from it. We denote the boundaries of the two discs by Γ_1 and Γ_2, respectively, and also call them the *cuffs* of the cylinder Σ. We assume that G can be drawn on Σ with all source terminals appearing on Γ_1 and all destination terminals appearing on Γ_2.

Definition 3. *We say that a set $\mathcal{M}' \subseteq \mathcal{M}$ of demand pairs is* non-crossing *if there is an ordering $(s_{i_1}, t_{i_1}), \ldots, (s_{i_r}, t_{i_r})$ of the demand pairs in $\mathcal{M}'$, such that $s_{i_1}, s_{i_2}, \ldots, s_{i_r}$ are all distinct and appear in this counter-clock-wise order on Γ_1, and $t_{i_1}, t_{i_2}, \ldots, t_{i_r}$ are all distinct and appear in this counter-clock-wise order on Γ_2.*

It is immediate to verify that if we are given any instance $(G, \mathcal{M})$ of NDP-Cylinder, and any set $\mathcal{M}' \subseteq \mathcal{M}$ of demand pairs that can all be routed via node-disjoint paths in G, then set $\mathcal{M}'$ is non-crossing.

Tight Concentric Cycles. We start with the following definition.

Definition 4. *Given a planar graph H drawn in the plane and a vertex $v \in V(H)$ that is not incident to the infinite face,* min-cycle(H,v) *is the cycle C in H, such that: (i) $v \in D^{\circ}(C)$; and (ii) among all cycles satisfying (i), C is the one for which $D(C)$ is minimal inclusion-wise.*

It is easy to see that min-cycle(H, v) is uniquely defined. Indeed, consider the graph $H \setminus v$, and the face F in the drawing of $H \setminus v$ where v used to reside. Then the boundary of F

contains exactly one cycle C with $D(C)$ containing v, and $C = \text{min-cycle}(H, v)$. We next define a family of tight concentric cycles.

Definition 5. *Suppose we are given a planar graph H, an embedding of H in the plane, a simple closed H-normal curve C, and an integral parameter $r \geq 1$. A family of r tight concentric cycles around C is a sequence $Z_1, Z_2, \ldots, Z_r$ of disjoint simple cycles in H, with the following properties:*

- *$D(C) \subsetneq D(Z_1) \subsetneq D(Z_2) \subsetneq \cdots \subsetneq D(Z_r)$;*
- *if H' is the graph obtained from H by contracting all vertices lying in $D(C)$ into a super-node a, then $Z_1 = \text{min-cycle}(H', a)$; and*
- *for every $1 < h \leq r$, if H' is the graph obtained from H by contracting all vertices lying in $D(Z_{h-1})$ into a super-node a, then $Z_h = \text{min-cycle}(H', a)$.*

We will sometimes allow C to be a simple cycle in H. The family of tight concentric cycles around C is then defined similarly.

Monotonicity of Paths and Cycles. Suppose we are given a planar graph H, embedded into the plane, a simple H-normal curve C in H, and a family $(Z_1, \ldots, Z_r)$ of tight concentric cycles around C. Assume further that we are given a set $\mathcal{P}$ of κ node-disjoint paths, originating at the vertices of C, and terminating at some vertices lying outside of $D(Z_r)$. We would like to re-route these paths to ensure that they are monotone with respect to the cycles, that is, for all $1 \leq h \leq r$, and for all $P \in \mathcal{P}$, $P \cap Z_h$ is a path. We first discuss re-routing to ensure monotonicity with respect to a single cycle, and then extend it to monotonicity with respect to a family of concentric cycles.

Definition 6. *Given a graph H, a cycle C and a path P in H, we say that P is monotone with respect to C, iff $P \cap C$ is a path.*

We prove the following lemma below.

Lemma 3.2.2. *Let H be a planar graph embedded into the plane, C a simple cycle in H, and $\mathcal{P}$ a collection of κ simple internally node-disjoint paths between two vertices: vertex s lying in $D^\circ(C)$, and vertex $t \notin D(C)$, that is incident on the outer face. Assume further that H is the union of C and the paths in $\mathcal{P}$, and that $C = \text{min-cycle}(H, s)$. Then there is an efficient algorithm to compute a set $\mathcal{P}'$ of κ internally node-disjoint paths connecting s to t in H, such that every path in $\mathcal{P}'$ is monotone with respect to C.*

Proof. Consider any path $P \in \mathcal{P}$. A sub-path Q of P is called a bump, if the two endpoints of Q lie on C, and all the intermediate vertices of Q do not lie on C (notice that Q may be simply an edge of C). Since path P is simple, a bump Q cannot contain s.

We now define a shadow of a bump Q. If Q is an edge of C, then the shadow of Q is Q itself. Assume now that Q is not an edge of C. Let $u, v \in V(C)$ be the two endpoints of Q, and let σ, σ' be the two segments of C with endpoints u and v. Let C_1 be the union of σ and Q, and C_2 the union of σ' and Q. Then one of the two corresponding discs, $D(C_1)$ and $D(C_2)$ contains s - we assume that it is the latter disc. We then let σ be the *shadow* of Q on C (see Figure 3.1).

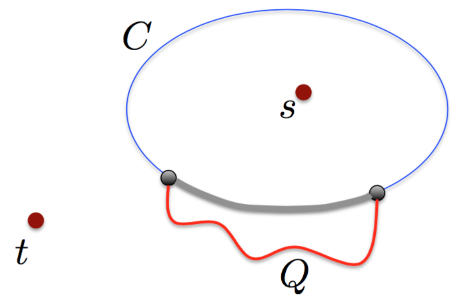


Figure 3.1: A bump Q and its shadow

We note that all inner points on the image of Q must lie outside $D^\circ(C)$, as otherwise, we can find a cycle C' in H with $D(C')$ containing s and $D(C') \subsetneq D(C)$, contradicting the fact that $C = \text{min-cycle}(H, s)$. We obtain the following two simple observations.

Observation 3.2.3. *Let $P \in \mathcal{P}$ be any path, and let R be the longest segment of P, starting from s and terminating at a vertex of C, such that R does not contain any vertex of C as an inner vertex. Let $v \in V(C)$ be the endpoint of R, and let R' be the sub-path of P from v to t. Then every point p on the image of R' lies outside $D^{\circ}(C)$.*

Proof. If any such point p lies in $D^{\circ}(C)$, then for some bump Q of P, some inner point on the image of Q lies in $D^{\circ}(C)$, and this is impossible, as observed above. □

Observation 3.2.4. *Let $P, P' \in \mathcal{P}$ be two distinct paths, let Q be a bump on P, and let Q' be a bump on P'. Then the shadows of Q and Q' are node-disjoint.*

Proof. Let σ be the shadow of Q, and let a, b be its endpoints. Let σ' be the shadow of Q', and let a', b' be its endpoints. Since the inner points of the images of both Q and Q' lie outside $D^{\circ}(C)$, and their endpoints are all distinct, if σ and σ' are not disjoint, then either $\sigma \subsetneq \sigma'$, or $\sigma' \subsetneq \sigma$.

Assume without loss of generality that the latter is true. Let A be the disc whose boundary is $\sigma \cup Q$. Then $t \notin A$, but some vertex of P' lies in A. Let R' be the longest sub-path of P' starting from s and terminating at a vertex of C, such that R' is internally disjoint from C. Then there is a vertex $v' \in V(Q')$, that does not lie on R'. Let R'' be the sub-path of P' from v' to t. Then path R'' originates in disc A and terminates outside A. But it can only leave A by traveling inside $D^{\circ}(C)$, which is impossible from Observation 3.2.3. □

We now re-route each path $P \in \mathcal{P}$, as follows. Let u and v be the first and the last vertex of P that belong to C, respectively. Let $\tilde{P}$ be the union of the shadows of all bumps of P. Then $\tilde{P}$ is a path, contained in C, that contains u and v. Let $\tilde{P}'$ be a simple sub-path of $\tilde{P}$ connecting u to v, and let P' be obtained by concatenating the segment of P from s to u, $\tilde{P}'$, and the segment of P from v to t. Notice that path P' is monotone with respect to C, and all paths in the resulting set $\mathcal{P}' = \left\{P' \mid P \in \mathcal{P}\right\}$ are internally node-disjoint by Observation 3.2.4.

□

We now define monotonicity with respect to a family of cycles.

Definition 7. *Let H be a graph, $\mathcal{Z} = (Z_1, \ldots, Z_r)$ a collection of r disjoint cycles, and $\mathcal{P}$ a collection of node-disjoint paths in H. We say that the paths in $\mathcal{P}$ are* monotone *with respect to $\mathcal{Z}$, iff for every $1 \le h \le r$, every path in $\mathcal{P}$ is monotone with respect to Z_h.*

The following theorem allows us to re-route sets of paths so they become monotone with respect to a given family of tight concentric cycles. Its proof is a simple application of Lemma 3.2.2.

Theorem 3.2.5. *Let H be a planar graph embedded in the plane, C any simple closed H-normal curve or a simple cycle in H, and $\mathcal{Z} = (Z_1, \ldots, Z_r)$ a family of r tight concentric cycles in H around C. Let $Y \subsetneq H$ be any connected subgraph of H lying completely outside of $D(Z_r)$, and let $\mathcal{P}$ be a set of κ node-disjoint paths, connecting a subset $A \subseteq V(C)$ of κ vertices to a subset $B \subseteq V(Y)$ of κ vertices, so that the paths of $\mathcal{P}$ are internally disjoint from $V(C) \cup V(Y)$. Let $H' = \left(\bigcup_{h=1}^{r} V(Z_h)\right) \cup \mathcal{P}$. Then there is an efficient algorithm to compute a collection $\mathcal{P}'$ of κ node-disjoint paths in H', connecting the vertices of A to the vertices of B, so that the paths in $\mathcal{P}'$ are monotone with respect to $\mathcal{Z}$, and they are internally node-disjoint from $V(C) \cup V(Y)$.*

Proof. We perform r iterations. At the beginning of the hth iteration, for $1 \le h \le r$, we assume that we are given a set $\mathcal{P}_{h-1}$ of κ node-disjoint paths in H', connecting the vertices of A to the vertices of B, so that the paths in $\mathcal{P}_{h-1}$ are internally disjoint from $V(C) \cup V(Y)$, and they are monotone with respect to $Z_1, \ldots, Z_{h-1}$. The output of iteration h is a set $\mathcal{P}_h$ of κ node-disjoint paths in H', connecting the vertices of A to the vertices of B, so that the paths in $\mathcal{P}_h$ are internally disjoint from $V(C) \cup V(Y)$, and they are monotone with respect to $Z_1, \ldots, Z_h$. The output of the algorithm is the set $\mathcal{P}_r$ of paths computed in the last iteration. For simplicity of notation, we denote $Z_0 = C$, even though C is not a cycle of H.

We start with $\mathcal{P}_0 = \mathcal{P}$. It is immediate to see that this is a valid input to iteration 1. Assume now that we are given a set $\mathcal{P}_{h-1}$ of paths, which is a valid input to iteration h. The iteration is then executed as follows. Let H'' be the graph obtained by starting with $H'' = \left(\bigcup_{h=0}^{r} V(Z_h)\right) \cup \mathcal{P}_{h-1}$, and then contracting all vertices lying in $D(Z_{h-1})$ into a source vertex s', and all vertices of B into a destination vertex t'. Since $B \subseteq V(Y)$, where Y is a connected sub-graph of G, $Y \cap D(Z_r) = \emptyset$, and the paths in $\mathcal{P}_{h-1}$ are internally disjoint from $V(Y)$, graph H'' is a planar graph. Moreover, $Z_h = \text{min-cycle}(H'', s')$ from the definition of tight concentric cycles. We consider the drawing of H'' in the plane where t' is incident on the outer face. It is easy to see that $Z_h = \text{min-cycle}(H'', s')$ still holds with respect to this new drawing of H''.

From Lemma 3.2.2, there is a set $\mathcal{Q}$ of κ internally node-disjoint paths in H'', connecting s' to t', that are monotone with respect to Z_h. In order to construct the set $\mathcal{P}_h$ of paths, let $P \in \mathcal{P}_{h-1}$ be any path, and let v_P be the last vertex of P lying on Z_{h-1}. Let P' be the sub-path of P starting from v_P and terminating at the endpoint of P lying in B. Notice that from the monotonicity of the paths in $\mathcal{P}_{h-1}$ with respect to $Z_1, \ldots, Z_{h-1}$, there are exactly κ edges leaving the vertex s' in H'', each edge lying on a distinct path $P \in \mathcal{P}_{h-1}$. If edge e is leaving s' in H'', and e lies on P, then it is incident on v_P, and so exactly one path of $\mathcal{Q}$ originates at v_P. We denote this path by Q_P. For each path $P \in \mathcal{P}_{h-1}$, we let P^* be the path obtained from P by replacing P' with Q_P. Notice that, since the paths in $\mathcal{P}_{h-1}$ are internally disjoint from B, exactly κ edges are incident on t' in graph H', each of which is incident on a distinct vertex of B in H. It is now easy to verify that the set $\mathcal{P}_h$ contains κ node-disjoint paths, connecting the vertices of A to the vertices of B, and the paths in $\mathcal{P}_h$ are internally disjoint from $V(C) \cup V(Y)$ and monotone with respect to $Z_1, \ldots, Z_h$. We return $\mathcal{P}_r$ as the output of the algorithm. $\square$

3.3 Demand Pair Selection Problem

In order to prove Theorem 3.1.1, we define a new problem, called Demand Pair Selection Problem (DPSP). We first define the problem and show an 8-approximation algorithm for it. We then show that both NDP-Disc and NDP-Cylinder reduce to DPSP.

We assume that we are given two disjoint directed paths, σ and σ', and a collection $\mathcal{M} = \{(s_1,t_1),\ldots,(s_k,t_k)\}$ of pairs of vertices of $\sigma \cup \sigma'$ that are called demand pairs, where all vertices of $S = \{s_1,\ldots,s_k\}$ lie on σ, and all vertices of $T = \{t_1,\ldots,t_k\}$ lie on σ' (not necessarily in this order). We refer to the vertices of S and T as the *source* and the *destination* vertices, respectively. Note that the same vertex of σ may participate in several demand pairs, and the same is true for the vertices of σ'. Given any pair a, a' of vertices of σ, with a lying before a' on σ, we sometimes denote by (a, a') the sub-path of σ between a and a' (that includes both these vertices), and we will sometimes refer to it as an interval. We define intervals of σ' similarly.

For every pair $v, v' \in V(\sigma)$ of vertices, we denote $v \prec v'$ if v lies strictly before v' on σ, and we denote $v \preceq v'$, if $v \prec v'$ or $v = v'$ hold. Similarly, for every pair $v, v' \in V(\sigma')$ of vertices, we denote $v \prec v'$ if v lies strictly before v' on σ', and we denote $v \preceq v'$, if $v \prec v'$ or $v = v'$ hold. We need the following definitions.

Definition 8. *Suppose we are given two pairs (a,b) and (a',b') of vertices of $\sigma \cup \sigma'$, with $a, a' \in \sigma$ and $b, b' \in \sigma'$. We say that (a,b) and (a',b')* cross *iff one of the following holds: either (i) $a = a'$; or (ii) $b = b'$; or (iii) $a \prec a'$ and $b' \prec b$; or (iv) $a' \prec a$ and $b \prec b'$.*

Definition 9. *We say that a subset $\mathcal{M}' \subseteq \mathcal{M}$ of demand pairs is* non-crossing *iff for all distinct pairs $(s,t), (s',t') \in \mathcal{M}'$, (s,t) and (s',t') do not cross.*

Our goal is to select the largest-cardinality non-crossing subset $\mathcal{M}' \subseteq \mathcal{M}$ of demand pairs, satisfying a collection $\mathcal{K}$ of constraints. Set $\mathcal{K}$ of constraints is given as part of the problem input, and consists of four subsets, $\mathcal{K}_1, \ldots, \mathcal{K}_4$, where constraints in set $\mathcal{K}_i$ are called *type-*

i constraints. Every constraint $K \in \mathcal{K}$ is specified by a quadruple (i, a, b, w), where $i \in \{1, 2, 3, 4\}$ is the constraint type, $a, b \in V(\sigma \cup \sigma')$, and $1 \leq w \leq |\mathcal{M}|$ is an integer.

For every type-1 constraint $K = (1, a, b, w) \in \mathcal{K}_1$, we have $a, b \in V(\sigma)$ with $a \prec b$. The constraint is associated with the sub-path $I = (a, b)$ of σ. We say that a subset $\mathcal{M}' \subseteq \mathcal{M}$ of demand pairs *satisfies* K iff the total number of the source vertices participating in the demand pairs of $\mathcal{M}'$ that lie on I is at most w.

Similarly, for every type-2 constraint $K = (2, a, b, w) \in \mathcal{K}_2$, we have $a, b \in V(\sigma')$ with $a \prec b$, and the constraint is associated with the sub-path $I = (a, b)$ of σ'. A set $\mathcal{M}' \subseteq \mathcal{M}$ of demand pairs satisfies K iff the total number of the destination vertices participating in the demand pairs in $\mathcal{M}'$ that lie on I is at most w.

For each type-3 constraint $K = (3, a, b, w) \in \mathcal{K}_3$, we have $a \in V(\sigma)$ and $b \in V(\sigma')$. The constraint is associated with the sub-path L_a of σ between the first vertex of σ and a (including both these vertices), and the sub-path R_b of σ' between b and the last vertex of σ' (including both these vertices). We say that a demand pair $(s, t) \in \mathcal{M}$ *crosses* K iff $s \in L_a$ and $t \in R_b$. A set $\mathcal{M}' \subseteq \mathcal{M}$ of demand pairs satisfies K iff the total number of pairs $(s, t) \in \mathcal{M}'$ that cross K is bounded by w.

Finally, for each type-4 constraint $K = (4, a, b, w) \in \mathcal{K}_4$, we also have $a \in V(\sigma)$ and $b \in V(\sigma')$. The constraint is associated with the sub-path R_a of σ between a and the last vertex of σ (including both these vertices), and the sub-path L_b of σ' between the first vertex of σ' and b (including both these vertices). We say that a demand pair $(s, t) \in \mathcal{M}$ crosses K iff $s \in R_a$ and $t \in L_b$. A set $\mathcal{M}' \subseteq \mathcal{M}$ of demand pairs satisfies K iff the total number of pairs $(s, t) \in \mathcal{M}'$ that cross K is bounded by w.

Given the paths σ, σ', the set $\mathcal{M}$ of the demand pairs, and the set $\mathcal{K}$ of constraints as above, the goal in the DPSP problem is to select a maximum-cardinality non-crossing subset $\mathcal{M}' \subseteq \mathcal{M}$ of demand pairs, such that all constraints in $\mathcal{K}$ are satisfied by $\mathcal{M}'$. We now focus on proving the following theorem.

Theorem 3.3.1. *There is an efficient* 8*-approximation algorithm for* DPSP*.*

The following definition and observation allow us to slightly relax the problem.

Definition 10. *Let $c \geq 1$ be an integer, and let $\mathcal{M}' \subseteq \mathcal{M}$ be a subset of the demand pairs. Given a constraint $K = (i, a, b, w)$ of type* 1 *or* 2*, we say that $\mathcal{M}'$ violates K by a factor of at most c, iff the number of the demand pairs $(s, t) \in \mathcal{M}'$ with either s or t lying in (a, b) is at most cw. Likewise, given a constraint $K = (i, a, b, w)$ of type* 3 *or* 4*, we say that $\mathcal{M}'$ violates K by a factor of at most c iff the number of the demand pairs in $\mathcal{M}'$ crossing K is at most cw.*

Observation 3.3.2. *There is an efficient algorithm, that, given a* DPSP *instance $(\sigma, \sigma', \mathcal{M}, \mathcal{K})$ and a non-crossing set $\mathcal{M}' \subseteq \mathcal{M}$ of demand pairs that violates every constraint in $\mathcal{K}$ by a factor of at most c, for any integer $c > 1$, computes a subset $\mathcal{M}'' \subseteq \mathcal{M}'$ of at least $|\mathcal{M}'|/c$ demand pairs, satisfying all constraints in $\mathcal{K}$.*

Proof. Assume that

$$\mathcal{M}' = \left\{(s_{j_1}, t_{j_1}), \ldots, (s_{j_z}, t_{j_z})\right\},$$

where $s_{j_1} \prec \cdots \prec s_{j_z}$ and $t_{j_1} \prec \cdots \prec t_{j_z}$. Let

$$\mathcal{M}'' = \left\{(s_{j_\ell}, t_{j_\ell}) \mid 1 \leq \ell \leq z \text{ and } \ell \equiv 1 \mod c\right\}.$$

Clearly, $\mathcal{M}'' \subseteq \mathcal{M}'$ and $|\mathcal{M}''| \geq |\mathcal{M}'|/c$. We now claim that all constraints in $\mathcal{K}$ are satisfied by $\mathcal{M}''$.

Indeed, consider any constraint $K = (i, a, b, w) \in \mathcal{K}$. Assume first that K is a type-1 constraint. Then at most cw demand pairs in $\mathcal{M}'$ have a source vertex in the interval (a, b). It is easy to see that the number of the demand pairs of $\mathcal{M}''$ that have a source vertex in the interval (a, b) is at most w. If K is a type-2 constraint, the argument is similar. Assume now that K is a type-3 constraint (the case where it is a type-4 constraint is symmetric).

Let $\mathcal{M}_K \subseteq \mathcal{M}'$ be the set of all demand pairs of $\mathcal{M}'$ crossing K. The key observation is that the demand pairs of $\mathcal{M}_K$ appear consecutively in the ordered set $\mathcal{M}'$. Since $|\mathcal{M}_K| \leq cw$, it is easy to see that $|\mathcal{M}'' \cap \mathcal{M}_K| \leq w$, and so set $\mathcal{M}''$ satisfies the constraint K. $\square$

Let $r = \lceil \log |\mathcal{M}| \rceil$, and for $1 \leq j \leq r+1$, set $W_j = 2^j$. We partition the constraints of $\mathcal{K}$ into r levels, where for $1 \leq j \leq r$, the jth level contains all constraints (i, a, b, w) with

$$W_{j-1} \leq w < 2 \cdot W_{j-1} = W_j.$$

For all $1 \leq i \leq 4$ and $1 \leq j \leq r$, we denote by $\mathcal{S}_j^{(i)}$ the set of all type-i constraints that belong to level j. Let $\mathcal{S}_j = \bigcup_{i=1}^4 \mathcal{S}_j^{(i)}$ be the set of all level-j constraints.

Our algorithm employs dynamic programming. It is convenient to view the algorithm as constructing r dynamic programming tables - one for each level. Consider some level $1 \leq j \leq r$. Let $I = (x, y) \subseteq \sigma$, $I' = (x', y') \subseteq \sigma'$ be a pair of intervals. We say that it is a *good level-j* pair if the following conditions hold for every level-j constraint $K = (i, a, b, w) \in \mathcal{S}_j$:

C1. if K is a type-1 constraint, then I is not contained in (a, b);

C2. if K is a type-2 constraint, then I' is not contained in (a, b);

C3. if K is a type-3 constraint, then either I is not contained in L_a, or I' is not contained in R_b; and

C4. if K is a type-4 constraint, then either I is not contained in R_a, or I' is not contained in L_b.

For each $1 \leq j \leq r$, let $\mathcal{P}_j$ denote the set of all good level-j pairs of intervals. The level-j dynamic programming table, Π_j contains an entry $\Pi_j[I, I']$ for every good level-j pair $(I, I') \in \mathcal{P}_j$ of intervals. The entry will either remain empty, or it will contain a collection of non-crossing demand pairs from $\mathcal{M}$ of cardinality exactly W_j, whose sources lie in I and destinations lie in I'.

We now describe an efficient algorithm that computes the entries of the dynamic programming tables. We start with $j = 1$. For every pair $(I, I') \in \mathcal{P}_1$ of good level-1 intervals, if there are two distinct non-crossing demand pairs $(s,t),(s',t') \in \mathcal{M}$ with $s, s' \in I$ and $t, t' \in I'$, then we set

$$\Pi_1[I, I'] = \left\{(s,t),(s',t')\right\}.$$

Otherwise, we set

$$\Pi_1[I, I'] = \emptyset.$$

Assume now that we have constructed the tables for levels $1, \ldots, j-1$, and consider the level-j table, for some $1 < j \le r$, and its entry $\Pi_j(I, I')$ for some good level-j pair $(I, I') \in \mathcal{P}_j$ of intervals, where $I = (x, y)$ and $I' = (x', y')$. If there exist two pairs of vertices $u, v \in I$ and $u', v' \in I'$ such that $u \prec v$, $u' \prec v'$, and both $\Pi_{j-1}[(x,u),(x',u')]$ and $\Pi_{j-1}[(v,y),(v',y')]$ are non-empty, then we set

$$\Pi_j[I, I'] = \Pi_{j-1}[(x,u),(x',u')] \cup \Pi_{j-1}[(v,y),(v',y')].$$

Otherwise, we set

$$\Pi_j[I, I'] = \emptyset.$$

This completes the description of the algorithm that computes the entries of the dynamic programming tables. We now proceed to analyze it, starting with the following easy observations.

Observation 3.3.3. *For all $1 \le j \le r$ and $(I, I') \in \mathcal{P}_j$, either $|\Pi_j[I, I']| = 0$ or $|\Pi_j[I, I']| = W_j$.*

Proof. The proof is by induction on j. Clearly, the claim holds for $j = 1$ and all $(I, I') \in \mathcal{P}_1$ by our construction. Consider now some $j > 1$, and assume that the claim holds for all values $j' < j$. Consider some entry $\Pi_j(I, I')$ of the level-j dynamic programming table. Our algorithm either sets $\Pi_j[I, I'] = \emptyset$ or $\Pi_j[I, I'] = \Pi_{j-1}[(x, u), (x', u')] \cup \Pi_{j-1}[(v, y), (v', y')]$. The latter only happens when both $\Pi_{j-1}[(x, u), (x', u')], \Pi_{j-1}[(v, y), (v', y')]$ are non-empty, and so by the induction hypothesis,

$$|\Pi_{j-1}[(x, u), (x', u')]|, |\Pi_{j-1}[(v, y), (v', y')]| = W_{j-1},$$

giving us

$$|\Pi_j[I, I']| = 2 \cdot W_{j-1} = W_j.$$

□

Observation 3.3.4. *For all $1 \leq j \leq r$ and $(I, I') \in \mathcal{P}_j$, $\Pi_j[I, I']$ is a non-crossing subset of the demand pairs in $\mathcal{M}$, where every $(s, t) \in T_j[I, I']$ has $s \in I$ and $t \in I'$.*

Proof. The proof is again by induction on j. Clearly, the claim holds for $j = 1$ and all $(I, I') \in \mathcal{P}_1$ from our construction.

Consider now some level $j > 1$ and assume that the claim holds for levels $1, \ldots, (j-1)$. Let $(I, I') \in \mathcal{P}_j$ be a good level-j pair of intervals, with $I = (x, y)$ and $I' = (x', y')$. The claim trivially holds when $\Pi_j[I, I'] = \emptyset$, so we assume that

$$\Pi_j[I, I'] = \Pi_{j-1}[(x, u), (x', u')] \cup \Pi_{j-1}[(v, y), (v', y')],$$

where $x \preceq u \prec v \preceq y$ and $x' \preceq u' \prec v' \preceq y'$.

From the induction hypothesis, no two demand pairs from $\Pi_{j-1}[(x, u), (x', u')]$ can cross, and the same holds for the demand pairs of $\Pi_{j-1}[(v, y), (v', y')]$. Since every demand pair $(s, t) \in \Pi_{j-1}[(x, u), (x', u')]$ has $s \in (x, u), t \in (x', u')$, and every demand pair $(s', t') \in \Pi_{j-1}[(v, y), (v', y')]$ has $s' \in (v, y), t' \in (v', y')$, it is immediate to verify that no pair of demands $(s, t) \in \Pi_{j-1}[(x, u), (x', u')]$ and $(s', t') \in \Pi_{j-1}[(v, y), (v', y')]$ can cross, and for every demand pair $(s, t) \in T_j[I, I']$, $s \in I$ and $t \in I'$ holds. □

Observation 3.3.5. *For all $1 \leq j \leq r$ and $(I, I') \in \mathcal{P}_j$, the set $\Pi_j[I, I']$ of demand pairs violates every constraint in $\mathcal{K}$ by at most factor 4.*

Proof. Fix some $1 \leq j \leq r$ and $(I, I') \in \mathcal{P}_j$, and consider the corresponding table entry $\Pi_j[I, I']$. The claim holds trivially when $|\Pi_j[I, I']| = 0$, so we assume that $|\Pi_j[I, I']| = W_j$. Consider some constraint $K = (i, a, b, w) \in \mathcal{S}^{(i)}_{j'}$, for some level $1 \leq j' \leq r$. if $j' \geq j$, then $w \geq W_{j-1}$ must hold, while $|\Pi_j[I, I']| = W_j$, so the constraint is violated by the factor of at most 4.

Consider now the case where $j' < j$. Assume first that $i = 1$. Note that $\Pi_j[I, I']$ is the union of exactly $2^{j-j'}$ non-empty level-j' table entries, each of which contains a set of demand pairs of cardinality exactly $W_{j'}$. Let $\mathcal{R}$ be the set of all these level-j' table entries. We claim that there are at most two table entries $\Pi_{j'}(\hat{I}, \hat{I}') \in \mathcal{R}$ with $\hat{I} \cap (a, b) \neq \emptyset$. Indeed, assume for contradiction that there are 3 such distinct entries in $\mathcal{R}$, say $\Pi_{j'}[I_1, I'_1], \Pi_{j'}[I_2, I'_2]$, and $\Pi_{j'}[I_3, I'_3]$, with $I_1 \cap (a, b), I_2 \cap (a, b), I_3 \cap (a, b) \neq \emptyset$. Then at least one of the intervals I_1, I_2, I_3 must be contained in (a, b), violating our condition for good level-j' pairs of intervals. We conclude that the number of the demand pairs in $\Pi_j[I, I']$ with a source vertex in (a, b) is bounded by $2 \cdot W_{j'} = 4 \cdot W_{j'-1} \leq 4 \cdot w$. The proof for the case where $i = 2$ is analogous.

Assume now that K is a type-3 constraint. As before, $\Pi_j[I, I']$ is the union of exactly $2^{j-j'}$ non-empty level-j' table entries, each of which stores a set of demand pairs of cardinality exactly $W_{j'}$. Let $\mathcal{R}$ be the set of these level-j' table entries. We claim that there are at most two entries $\Pi_{j'}(\hat{I}, \hat{I}') \in \mathcal{R}$, with $\hat{I} \cap L_a \neq \emptyset$ and $\hat{I}' \cap R_b \neq \emptyset$. Indeed, assume otherwise, and let $\Pi_{j'}[I_1, I'_1], \Pi_{j'}[I_2, I'_2], \Pi_{j'}[I_3, I'_3]$ be three distinct entries in $\mathcal{R}$, such that for each $1 \leq \ell \leq 3$, $I_\ell \cap L_a \neq \emptyset$ and $I'_\ell \cap R_b \neq \emptyset$. Assume that I_1, I_2, I_3 appear on σ in this order, and recall that from our construction they are disjoint. Then it is easy to see that $I_2 \subseteq L_a$ and $I'_2 \subseteq R_b$ must hold, contradicting our definition of good level-j' pairs of terminals. Therefore, there are at most two entries $\Pi_{j'}(\hat{I}, \hat{I}') \in \mathcal{R}$, with $\hat{I} \cap L_a \neq \emptyset$ and $\hat{I}' \cap R_b \neq \emptyset$. Demand pairs participating in solutions corresponding to other entries in $\mathcal{R}$ cannot cross K, and so the number of the demand paris crossing K is at most $2 \cdot W_{j'} = 4 \cdot W_{j'-1} \leq 4 \cdot w$. The case where K is a type-4 constraint is treated similarly. $\square$

From the discussion so far, every entry of every dynamic programming table contains a non-crossing set of demand pairs, that violates every constraint of $\mathcal{K}$ by at most factor 4. Let $\tilde{\mathcal{M}}$ be the largest-cardinality set of demand pairs stored in any entry any of the tables, and let OPT be the value of the optimal solution to the DPSP problem instance. The following theorem is central to our analysis.

Theorem 3.3.6. *If* $\mathsf{OPT} \geq 2$*, then* $|\tilde{\mathcal{M}}| \geq \mathsf{OPT}/2$.

We can now apply Observation 3.3.2 to compute a subset $\tilde{\mathcal{M}}' \subseteq \tilde{\mathcal{M}}$ of at least $|\tilde{\mathcal{M}}|/4 \geq \mathsf{OPT}/8$ non-crossing demand pairs satisfying all constraints in $\mathcal{K}$ (if $|\tilde{\mathcal{M}}| = 0$, then $\mathsf{OPT} \leq 1$ must hold, and finding an optimal solution is trivial). In order to complete the proof of Theorem 3.3.1, it now remains to prove Theorem 3.3.6.

Proof of Theorem 3.3.6. Denote $\kappa = \mathsf{OPT}$, and let $\mathcal{M}^* = \{(s_1, t_1), \ldots, (s_\kappa, t_\kappa)\}$ be the optimal solution to the DPSP instance, where $s_1 \prec \ldots \prec s_\kappa$ and $t_1 \prec \ldots \prec t_\kappa$. Let r' be the largest value for which

$$\kappa / W_{r'} \geq 1$$

(this is well-defined since we have assumed that $\mathsf{OPT} \geq 2$).

Let $\mathcal{M}^{**} \subseteq \mathcal{M}^*$ contain the first $W_{r'}$ demand pairs of $\mathcal{M}^*$, so $|\mathcal{M}^{**}| \geq |\mathcal{M}^*|/2$. We will show that some entry of the level-r' dynamic programming table stores a solution whose cardinality is at least $W_{r'}$.

For every level $1 \leq j \leq r'$, we define a partition $\mathcal{S}_j$ of $\mathcal{M}^{**}$ into $2^{r'-j}$ subsets, each containing exactly W_j consecutive demand pairs from $\mathcal{M}^{**}$. For every set $S \in \mathcal{S}_j$ of the partition, we define a pair $(I(S), I'(S))$ of intervals, with $I(S) \subseteq \sigma$ and $I'(S) \subseteq \sigma'$, as follows. Let $(s, t), (s', t')$ be the first and the last demand pairs of S, respectively (this is well-defined since the demand pairs are non-crossing). Then $I(S) = (s, s')$ and $I'(S) = (t, t')$. We denote by $\mathcal{Q}_j$ the resulting collection of $2^{r'-j}$ pairs of intervals. Note that for every pair $(I_1, I'_1), (I_2, I'_2) \in \mathcal{Q}_j$, $I_1 \cap I_2 = \emptyset$ and $I'_1 \cap I'_2 = \emptyset$. We need the following claim.

Claim 3.3.7. *For every $1 \leq j \leq r'$, every pair $(I, I') \in \mathcal{Q}_j$ of intervals is a good level-j pair.*

Proof. Fix some $1 \leq j \leq r'$ and some pair $(I, I') \in \mathcal{Q}_j$ of intervals. From our definition of the pairs of intervals in $\mathcal{Q}_j$, there are exactly W_j demand pairs (s,t) in $\mathcal{M}^*$ with $s \in I$ and $t \in I'$. Consider any level-j constraint $K = (i, a, b, w) \in \mathcal{S}_j$, and recall that $W_{j-1} \leq w < W_j$ must hold.

Assume first that K is a type-1 constraint. Then I cannot be contained in (a, b), since then $\mathcal{M}^*$ would have $W_j > w$ demand pairs whose sources lie in I, and hence in (a, b). The analysis for type-2 constraints is similar.

Assume now that K is a type-3 constraint, and assume for contradiction that $I \subseteq L_a$ and $I' \subseteq R_b$. Then $\mathcal{M}^*$ has $W_j > w$ demand pairs (s,t) with $s \in I$ and $t \in I'$, each of which crosses the constraint K, a contradiction. The case where K is a type-4 constraint is proved similarly. □

From the above claim, for all $1 \leq j \leq r'$ and $(I, I') \in \mathcal{Q}_j$, there is an entry $\Pi_j[I, I']$ in the level-j dynamic programming table. It is now enough to show that each such entry contains a solution of cardinality W_j.

Claim 3.3.8. *For each $1 \leq j \leq r'$, for every pair $(I(S), I'(S)) \in \mathcal{Q}_j$ of intervals, entry $\Pi_j[I(S), I'(S)]$ contains a solution of value W_j.*

Proof. The proof is by induction on j. The claim clearly holds for $j = 1$, since there are two distinct non-crossing demand pairs $(s,t), (s',t')$ with $s, s' \in I(S)$, $t, t' \in I'(S)$ - the two demand pair lying in S. Assume now that the claim holds for levels $1, \ldots, j-1$, for some $1 < j \leq r'$, and we would like to prove it for j.

Our definition of the partitions $\mathcal{S}_\ell$ of $\mathcal{M}^{**}$ ensures that there are exactly two distinct sets $S_1, S_2 \in \mathcal{S}_{j-1}$, with $S_1, S_2 \subseteq S$ (and in fact $S_1 \cup S_2 = S$). From the induction hypothesis, the entries of Π_{j-1} corresponding to pairs $(I(S_1), I'(S_1))$ and $(I(S_2), I'(S_2))$ each contain W_{j-1} demand pairs, and so $\Pi_j[I(S), I'(S)]$ must contain W_j demand pairs. $\square$

Recall that $\mathcal{S}_{r'}$ contains a single set of demand pairs - the set $\mathcal{M}^{**}$. We conclude that the corresponding entry of the level-r' dynamic programming table contains $W_{r'} \geq \mathsf{OPT}/2$ demand pairs. $\square$

3.4 Approximating Node-Disjoint Paths on a Disc

In this section we prove Theorem 3.1.1 for $\mathsf{NDP\text{-}Disc}$. The proof builds on the work of Robertson and Seymour [39], who gave a precise characterization of the instances $\mathsf{NDP\text{-}Disc}$, where all demand pairs can be routed simultaneously. Many of the definitions below are from [39]. Let Σ be a disc, whose boundary is denoted by Γ, and let G be any graph drawn on Σ. Suppose we are given a set $\mathcal{M} = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ of pairs of vertices of G, called demand pairs, and let $\mathcal{T}$ be the set of all vertices participating in the demand pairs in $\mathcal{M}$, that we refer to as terminals. We identify the graph G with its drawing. A *region* of G is a connected component of $\Sigma \setminus G$. We say that the drawing of G is *semi-proper* if no inner point on an image of an edge of G lies on Γ, and we say that it is *proper* with respect to $\mathcal{M}$, if additionally $V(G) \cap \Gamma = \mathcal{T}$.

Suppose we are given a planar graph G, together with a set $\mathcal{M}$ of demand pairs, such that G is drawn properly (with respect to $\mathcal{M}$) on Σ. Let W be a set of points on Γ, constructed as follows. First, we add to W all points corresponding to the vertices of $\mathcal{T}$. Next, for every segment β of $\Gamma \setminus \mathcal{T}$, we add one arbitrary point $p \in \beta$ to W. A vertex v of G is *peripheral* if $v \in \Gamma$, and a region of G is peripheral, if it contains a segment of $\Gamma \setminus \mathcal{T}$. Given two points $x, y \in W$, we denote by $\Delta_{\mathcal{M}}(x, y)$ the total number of the demand pairs $(s, t) \in \mathcal{M}$, where either $\{s, t\} \cap \{x, y\} \neq \emptyset$, or s and t belong to different segments of $\Gamma \setminus \{x, y\}$. We need the following definition.

Definition 11. *Given G, $\mathcal{M}$ and W as above, for any $x, y \in W$, an (x, y)-chain is a sequence $A_1, A_2, \ldots, A_r$, such that:*

- *for all $1 \leq i < r$, one of A_i, A_{i+1} is a vertex of G, the other is a region, and they are incident;*
- *if A_1 is a vertex then $A_1 = x$; if A_1 is a region then $x \in A_1$;*
- *similarly, if A_r is a vertex then $A_r = y$, and if A_r is a region then $y \in A_r$; and*
- *for all $1 \leq i \leq r$, A_i is peripheral if and only if $i = 1$ or $i = r$.*

The length of an (x, y)-chain is the number of its terms that are vertices. The redundancy of an (x, y)-chain is its length minus $\Delta_{\mathcal{M}}(x, y)$.

The following theorem, proved by Roberston and Seymour [39] characterizes routable sets of demand pairs for the NDP-Disc problem.

Theorem 3.4.1 (Theorem 3.6 in [39])**.** *Let G be a planar graph properly drawn on a disc Σ, with respect to a set $\mathcal{M}$ of demand pairs, and let W be defined as above. Then there is a set of node-disjoint paths, routing all demand pairs in $\mathcal{M}$ if and only if: (i) $\mathcal{M}$ is a non-crossing set of demand pairs; and (ii) for all $x, y \in W$, every (x, y)-chain has a non-negative redundancy. Moreover, there is an efficient algorithm to determine whether these conditions hold, and if so, to find a routing of the demand pairs in $\mathcal{M}$.*

Notice that if G is a graph drawn on a disc Σ, such that all terminals appear on the boundary Γ of the disc, then, by slightly altering Γ, we can ensure that the drawing of G is proper with respect to $\mathcal{M}$. Therefore, we assume from now on that we are given a proper drawing φ of G on Σ with respect to $\mathcal{M}$. We prove Theorem 3.1.1 for NDP-Disc in three steps. In the first step, we prove a stronger version of the theorem for the case where G is connected and the set $\mathcal{M}$ of terminals is 1-split. In the second step, we prove the theorem for the case where G is 2-connected, and we make no assumptions on the demand pairs. In the third step we prove the theorem without any additional assumptions.

3.4.1 Special Case: G is Connected and $\mathcal{M}$ is 1-Split

In this section, we prove the following theorem.

Theorem 3.4.2. *There is an efficient algorithm, that, given a connected planar graph G with a set $\mathcal{M}$ of demand pairs, and a proper drawing of G on the disc Σ with respect to $\mathcal{M}$, where $\mathcal{M}$ is 1-split with respect to the disc boundary, computes a routing of $\mathsf{OPT}(G,\mathcal{M})/8$ demand pairs of $\mathcal{M}$ via node-disjoint paths in G.*

Recall that if $\mathcal{M}$ is 1-split with respect to the boundary Γ of the disc Σ, then we can partition Γ into two disjoint segments, σ and σ', such that for every demand pair $(s,t) \in \mathcal{M}$, one of the vertices s, t lies on σ, and the other on σ'. We assume without loss of generality that all source vertices of the demand pairs in $\mathcal{M}$ lie on σ, and all destination vertices lie on σ'. Let $\mathcal{T}$ be the set of all vertices participating in the demand pairs in $\mathcal{M}$, that we refer to as terminals. We denote by S and T the sets of the source and the destination vertices of the demand pairs in $\mathcal{M}$, respectively. As before, we construct a set W of points: start with $W = \mathcal{T}$; then for every segment of $\Gamma \setminus \mathcal{T}$, add one arbitrary point of that segment to W.

The idea is to reduce this problem to the DPSP problem. We let $\tilde{\sigma}$ be a directed path, whose vertices correspond to the points of $W \cap \sigma$, ordered in their counter-clock-wise order on Γ. We let $\tilde{\sigma}'$ be a directed path, whose vertices correspond to the points of $W \cap \sigma'$, ordered in their clock-wise order on Γ. Therefore, all vertices of S lie on $\tilde{\sigma}$, and all vertices of T lie on $\tilde{\sigma}'$. Moreover, $\mathcal{M}'$ is a non-crossing set of demand pairs with respect to $\tilde{\sigma}$ and $\tilde{\sigma}'$ iff it is non-crossing with respect to Γ.

Suppose we are given any pair (x, y) of vertices of W. Let $\gamma(x, y)$ denote the shortest G-normal curve connecting x to y in the drawing of G on Σ, so that $\gamma(x, y)$ is contained in the disc Σ. Curve $\gamma(x, y)$ can be found efficiently by considering the graph G' dual to G, deleting the vertex corresponding to the outer face of G from it, and computing shortest paths between appropriately chosen vertices of G' (that correspond to the faces of G incident on x and y). Let $\ell(x, y)$ be the length of $\gamma(x, y)$. Notice that, since G is connected, if $\Delta_{\mathcal{M}}(x, y) \geq 1$, then $\ell(x, y) \geq 1$. We now construct a collection $\mathcal{K}$ of constraints of the DPSP probem, as follows:

- For every pair x, y of vertices of $W \cap \tilde{\sigma}$, with $\Delta_{\mathcal{M}}(x, y) \geq 1$, we add a type-1 constraint $(1, x, y, \ell(x, y))$ to $\mathcal{K}$.
- Similarly, for every pair x, y of vertices of $W \cap \tilde{\sigma}'$, with $\Delta_{\mathcal{M}}(x, y) \geq 1$, we add a type-2 constraint $(2, x, y, \ell(x, y))$ to $\mathcal{K}$.
- For every pair $x \in W \cap \tilde{\sigma}$, $y \in W \cap \tilde{\sigma'}$ of vertices, with $\Delta_{\mathcal{M}}(x, y) \geq 1$, we add a type-3 constraint $(3, x, y, \ell(x, y))$ and a type-4 constraint $(4, x, y, \ell(x, y))$ to $\mathcal{K}$.

This finishes the definition of the DPSP problem instance. The following observation is immediate.

Observation 3.4.3. *Let $\mathcal{M}' \subseteq \mathcal{M}$ be any set of demand pairs that can be routed via disjoint paths in G. Then $\mathcal{M}'$ is a valid solution to the* DPSP *problem instance.*

We apply the 8-approximation algorithm for the DPSP problem, to obtain a set $\mathcal{M}'$ of non-crossing demand pairs (with respect to $\tilde{\sigma}$ and $\tilde{\sigma}'$) satisfying all constraints in $\mathcal{K}$, with $|\mathcal{M}'| \geq |\mathsf{OPT}(G, \mathcal{M})|/8$. As observed above, the pairs in $\mathcal{M}'$ are non-crossing with respect to Γ. It now only remains to show that all demand pairs in $\mathcal{M}'$ can be routed in G. The algorithm from Theorem 3.4.1 can then be used to find the routing. The following theorem will finish the proof of Theorem 3.4.2.

Theorem 3.4.4. *Let $\mathcal{M}' \subseteq \mathcal{M}$ be a set of non-crossing demand pairs (with respect to $\tilde{\sigma}$ and $\tilde{\sigma}'$), that satisfy all constraints in $\mathcal{K}$. Then all demand pairs in $\mathcal{M}'$ can be routed in G.*

Proof. Let $\mathcal{T}' = \mathcal{T}(\mathcal{M}')$, and let φ be the current proper drawing of G with respect to $\mathcal{M}$. Notice that φ is not necessarily a proper drawing of G with respect to $\mathcal{M}'$, since the vertices of $\mathcal{T} \setminus \mathcal{T}'$ may lie on Γ. We can obtain a proper drawing φ' of G with respect to $\mathcal{M}'$ by moving all such terminals inside the disc, so they no longer lie on Γ. It is immediate to verify that the demand pairs in $\mathcal{M}'$ remain non-crossing with respect to Γ in the new drawing.

As before, we construct a set W' of points of Γ, by first adding all points corresponding to the terminals of $\mathcal{T}'$ to W'. Additionally, for every segment β of $\Gamma \setminus \mathcal{T}'$, we add an arbitrary point of β to W'. It now remains to show that for every pair x, y of points in W', for every (x, y)-chain in φ', the redundancy of the chain (with respect to $\mathcal{M}'$) is non-negative.

Assume otherwise. Let $x, y \in W'$ be any pair of points, and let $\mathcal{A} = (A_1, \ldots, A_r)$ be an x-y chain in the drawing φ', such that the redundancy of $\mathcal{A}$ is negative. We modify the chain $\mathcal{A}$, by replacing A_1 and A_r with elements A'_1 and A'_r, as follows.

If A_1 is a vertex of $\mathcal{T}'$, then we let $A'_1 = A_1$, and we let $x' = A'_1$. Otherwise, A_1 is a region of G in the drawing φ'. Let $v = A_2$, so $v \in V(G)$. Then v must lie on a boundary of some peripheral region R of G in the drawing φ. Let $A'_1 = R$, let $\beta(R)$ be the segment of Γ that serves as part of the boundary of the region R, and let $x' \in W$ be the point lying on the interior of $\beta(R)$. Similarly, if A_r is a vertex of $\mathcal{T}'$, then we let $A'_r = A_r$, and we let $y' = A'_r$. Otherwise, A_r is a region of G in the drawing φ'. Let $v' = A_{r-1}$, so $v' \in V(G)$. Then v' must lie on a boundary of some peripheral region R' of G of the drawing φ. Let $A'_r = R'$, let $\beta(R')$ be the segment of Γ that serves as part of the boundary of the region R', and let $y' \in W$ be the point lying on the interior of $\beta(R)$. We then obtain a sequence $\mathcal{A}' = (A'_1, A_2, \ldots, A_{r-1}, A'_r)$ (it may not be a valid chain for the drawing φ, since some of the elements A_i, for $1 < i < r'$, may be peripheral with respect to φ). We can then construct a G-normal curve γ' in the original drawing φ of G, connecting x' to y', such that $\gamma' \cap V(G)$ only contains the vertices that participate in $\mathcal{A}'$, and γ' is contained in Σ. Let ℓ denote the length of the chain $\mathcal{A}$, and let $\Delta = |\Delta_{\mathcal{M}'}(x, y)|$. We can assume that $\Delta \geq 1$, since otherwise it is immediate to verify that $\Delta \leq \ell$. Then $\ell(x', y') \leq \ell$, from the definition of $\ell(x', y')$. Notice that one of the segments of $\Gamma \setminus \{x, x'\}$ contains no terminals of $\mathcal{T}'$, and the same holds for one of the segments of $\Gamma \setminus \{y, y'\}$. We now consider three cases.

Assume first that both x' and y' lie on $\tilde{\sigma}$. Then pair $(s, t) \in \Delta_{\mathcal{M}'}(x, y)$ iff s belongs to the sub-path (x', y') of $\tilde{\sigma}$. Since we assumed that $\Delta \geq 1$, we get that $\Delta_{\mathcal{M}}(x', y') \geq 1$, so constraint $(1, x', y', \ell(x', y'))$ belongs to $\mathcal{K}$, and we get that $\Delta \leq \ell(x', y') \leq \ell$, a contradiction.

The case where $x', y' \in \tilde{\sigma}'$ is dealt with similarly.

Assume now that $x' \in \tilde{\sigma}$ and $y' \in \tilde{\sigma}'$. As before, since we assumed that $\Delta \geq 1$, we get that $\Delta_{\mathcal{M}}(x', y') \geq 1$. Consider the two corresponding type-3 and type-4 constraints $K = (3, x', y', \ell(x', y')), K'' = (4, x', y', \ell(x', y')) \in \mathcal{K}$. Let $L_{x'}, R_{x'}$ be the segments of $\tilde{\sigma}$ from its first endpoint to x', and from x' to its last endpoint, respectively, where both segments include x'. Define segments $L_{y'}, R_{y'}$ of $\tilde{\sigma}'$ similarly. Recall that a demand pair (s, t) crosses K iff $s \in L_{x'}$ and $t \in R_{y'}$, and it crosses K' iff $s \in R_{x'}$ and $t \in L_{y'}$. Let $\mathcal{M}_1 \subseteq \mathcal{M}'$ be the set of the demand pairs crossing $\mathcal{K}$, and let $\mathcal{M}_2 \subseteq \mathcal{M}'$ be the set of the demand pairs crossing $\mathcal{K}'$. Since $\mathcal{M}'$ is a non-crossing set of demand pairs, it is easy to verify that either $\mathcal{M}_1 \setminus \{(x', y')\} = \emptyset$, or $\mathcal{M}_2 \setminus \{(x', y')\} = \emptyset$. We assume without loss of generality that it is the latter. Notice that if $(x', y') \in \mathcal{M}'$, then it belongs to both $\mathcal{M}_1$ and $\mathcal{M}_2$. Constraint $(3, x', y', \ell(x', y'))$ then ensures that $|\mathcal{M}_1 \cup \mathcal{M}_2| \leq |\mathcal{M}_1| \leq \ell(x', y') \leq \ell$. It is easy to verify that $\Delta_{\mathcal{M}'}(x, y) \subseteq \mathcal{M}_1 \cup \mathcal{M}_2$, and so $\Delta \leq \ell$, a contradiction. $\square$

3.4.2 Special Case: G is 2-Connected

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

Theorem 3.4.5. *There is an efficient algorithm, that, given any 2-connected planar graph G, and a set $\mathcal{M}$ of k demand pairs for G, such that G is properly drawn on a disc with respect to $\mathcal{M}$, returns an $O(\log k)$-approximate solution to problem $(G, \mathcal{M})$.*

Proof. Let $|\mathcal{M}| = k$, and let $z = 4\lceil \log k \rceil$. We use Lemma 3.2.1 to compute a partition $\mathcal{M}^1, \ldots, \mathcal{M}^z$ of $\mathcal{M}$ into z disjoint subsets, so that for each $1 \le i \le z$, $\mathcal{M}^i$ is r_i-split for some integer r_i. We prove the following lemma.

Lemma 3.4.6. *There is an efficient algorithm, that, given an index $1 \le i \le z$, computes a solution to instance $(G, \mathcal{M}^i)$ routing $\Omega(\mathsf{OPT}(G, \mathcal{M}^i))$ demand pairs.*

Theorem 3.4.5 then easily follows from Lemma 3.4.6, since there is some index $1 \le i \le z$, for which

$$\mathsf{OPT}(G, \mathcal{M}^i) = \Omega(\mathsf{OPT}(G, \mathcal{M})/\log k).$$

In the rest of this section, we focus on proving Lemma 3.4.6.

For simplicity, we denote $\mathcal{M}^i$ by $\mathcal{M}$ and r_i by r. We assume that we are given a partition $\{\mathcal{M}_1, \ldots, \mathcal{M}_r\}$ of $\mathcal{M}$, and a collection $\sigma_1, \ldots, \sigma_{2r}$ of disjoint segments of Γ, such that for all $1 \le j \le r$, for every demand pair $(s,t) \in \mathcal{M}_j$, $s \in \sigma_{2j-1}$ and $t \in \sigma_{2j}$.

Clearly, for each $1 \le j \le r$, set $\mathcal{M}_j$ is 1-split. Therefore, we can use the algorithm from Section 3.4.1 in order to compute a set $\mathcal{P}_j$ of disjoint paths, routing a subset $\mathcal{M}'_j \subseteq \mathcal{M}_j$ of demand pairs, with $|\mathcal{M}'_j| \ge \Omega(\mathsf{OPT}(G, \mathcal{M}_j))$. Since

$$\sum_{j=1}^{r} \mathsf{OPT}(G, \mathcal{M}_j) \ge \mathsf{OPT}(G, \mathcal{M}),$$

we get that

$$\sum_{j=1}^{r} |\mathcal{M}'_j| \ge \Omega(\mathsf{OPT}(G, \mathcal{M})).$$

For every $1 \le j \le r$, we compute a subset $\mathcal{M}''_j \subseteq \mathcal{M}'_j$ of demand pairs, as follows. If $|\mathcal{M}'_j| \le 3$, then we let $\mathcal{M}''_j$ contain any demand pair from $\mathcal{M}'_j$. Otherwise, we assume that $\mathcal{M}'_j = \left((s^j_1, t^j_1), \ldots, (s^j_{q_j}, t^j_{q_j})\right)$, where the vertices $s^j_1, s^j_2, \ldots, s^j_{q_j}$ appear in this counter-clock-wise order on σ_{2j-1}. We then add to $\mathcal{M}''_j$ all demand pairs (s^j_a, t^j_a), where $a = 1$ modulo 3.

Notice that

$$\sum_{j=1}^{r} |\mathcal{M}''_j| \geq \sum_{j=1}^{r} \Omega(|\mathcal{M}'_j|) \geq \Omega(\mathsf{OPT}(G, \mathcal{M})).$$

It is now enough to prove that all demand pairs in $\mathcal{M}'' = \bigcup_{j=1}^{r} \mathcal{M}''_j$ can be routed in G. It is easy to verify that the demand pairs in $\mathcal{M}''$ are non-crossing, since for each $1 \leq j \leq r$, the demand pairs in $\mathcal{M}'_j$ are non-crossing, and pairs that belong to different sets $\mathcal{M}_j$ cannot cross. Let $\mathcal{T}'' = \mathcal{T}(\mathcal{M}'')$, and let φ'' is a proper drawing of G in Σ with respect to $\mathcal{M}''$, obtained from the original drawing φ by moving all terminals of $\mathcal{T} \setminus \mathcal{T}''$ to the interior of the disc. We define a set W of points on Γ as before: it contains all terminals of $\mathcal{T}''$, and for every segment of $\Gamma \setminus \mathcal{T}''$, W contains an arbitrary point in the interior of the segment.

Let $(x, y) \in W$ be any pair of points, and let $\mathcal{A} = (A_1, A_2, \ldots, A_p)$ be any (x, y)-chain. Let ℓ denote the length of $\mathcal{A}$. It is now enough to prove that $|\Delta_{\mathcal{M}''}(x, y)| \leq \ell$. Assume first that for some $1 \leq j \leq z$, $x, y \in \sigma_{2j-1} \cup \sigma_{2j}$. Then $\Delta_{\mathcal{M}''}(x, y)$ only contains demand pairs from $\mathcal{M}''_j$. Since all demand pairs in $\mathcal{M}''_j$ can be routed in G via node-disjoint paths, $|\Delta_{\mathcal{M}''}(x, y)| \leq \ell$.

Assume now that $x \in \sigma_{2j-1} \cup \sigma_{2j}$ and $y \in \sigma_{2j'-1} \cup \sigma_{2j'}$ for some $j \neq j'$. Then $\Delta_{\mathcal{M}''}(x, y)$ only contains demand pairs from $\mathcal{M}''_j \cup \mathcal{M}''_{j'}$. Let

$$\mathcal{N}_1 = \Delta_{\mathcal{M}''}(x, y) \cap \mathcal{M}''_j$$
$$\mathcal{N}_2 = \Delta_{\mathcal{M}''}(x, y) \cap \mathcal{M}''_{j'}.$$

Also, let

$$\mathcal{N}'_1 = \Delta_{\mathcal{M}'}(x, y) \cap \mathcal{M}'_j,$$
$$\mathcal{N}'_2 = \Delta_{\mathcal{M}'}(x, y) \cap \mathcal{M}'_{j'}.$$

Since both $\mathcal{M}'_j$ and $\mathcal{M}'_{j'}$ can be routed in G via node-disjoint paths,

$$|\mathcal{N}'_1|, |\mathcal{N}'_2| \leq \ell.$$

From our selection of the sets $\mathcal{M}''_j$, $\mathcal{M}''_{j'}$ of demand pairs,

$$|\mathcal{N}_1| \leq \max\left\{|\mathcal{N}'_1|/2, 1\right\},$$

and

$$|\mathcal{N}_2| \leq \max\left\{|\mathcal{N}'_2|/2, 1\right\}.$$

Since graph G is 2-vertex connected, we can assume that either $|\Delta_{\mathcal{M}''}(x, y)| \leq 1$, or $\ell \geq 2$. In the former case, $\ell \geq |\Delta_{\mathcal{M}''}(x, y)|$ since the graph is connected. In the latter case, we are now guaranteed that

$$|\Delta_{\mathcal{M}''}(x, y)| = |\mathcal{N}'_1| + |\mathcal{N}'_2| \leq \ell.$$

□

3.4.3 The General Case

In this section, we complete the proof of Theorem 3.1.1 for NDP-Disc. We assume that the input graph G is connected, since otherwise we can solve the problem separately on each connected component of G.

We use a block-decomposition of G. Recall that a block of G is a maximal 2-node-connected component of G. A block-decomposition of G is a tree τ, whose vertex set consists of two subsets, $V(\tau) = V_1 \cup V_2$, where V_1 is the set of all cut-vertices of G, and V_2 contains a vertex v_B for every block B of G. There is an edge between a vertex $u \in V_1$ and $v_B \in V_2$ iff $u \in v_B$. We choose an arbitrary vertex $r \in V_1$ as the root of τ. We assume that the value of the optimal solution is at least 10, as otherwise we can route a single demand pair. We then discard from $\mathcal{M}$ all demand pairs in which r participates – this changes the value of the optimal solution by at most 1.

Over the course of the algorithm, we will gradually change the tree τ, by deleting vertices from it. Given any vertex u in the current tree τ, we let τ_u denote the subtree of τ rooted at u, and we let G_u be the subgraph of G induced by the union of all blocks B with $v_B \in V(\tau_u)$. For every block B, we denote by $p(B)$ the unique vertex of B that serves as the parent of the vertex v_B in τ. As the tree τ changes, so do the trees τ_u and the graphs G_u.

If any block B of G contains all the terminals, then we can apply the algorithm from Theorem 3.4.5 to instance $(B, \mathcal{M})$ of NDP-Disc to obtain an $O(\log k)$-approximate solution. Otherwise, for every block B, if we denote by $\Gamma(B)$ the set of all cut vertices of G that belong to B, and by $\mathcal{T}(B)$ the set of all terminals lying in B, then we can draw B inside a disc, so that the vertices of $\Gamma(B) \cup \mathcal{T}(B)$ lie on its boundary.

We gradually construct a solution $\mathcal{P}$ to the NDP-Disc instance $(G, \mathcal{M})$, starting from $\mathcal{P} = \emptyset$. Throughout the algorithm, we ensure that all paths in $\mathcal{P}$ are disjoint from the vertices of G_r, where r is the root vertex of τ, and G_r is computed with respect to the current tree τ. Clearly, the invariant holds at the beginning of the algorithm.

We maintain a collection $\mathcal{B}$ of blocks, that is initialized to $\mathcal{B} = \emptyset$. For every $B \in \mathcal{B}$, we will define two subsets, $\mathcal{M}_B, \mathcal{M}'_B \subseteq \mathcal{M}$ of demand pairs, so that eventually, $\{\mathcal{M}_B, \mathcal{M}'_B \mid B \in \mathcal{B}\}$ is a partition of $\mathcal{M}$.

In every iteration, we consider all vertices $v_B \in V_2$ that belong to the current tree τ, such that $G_{v_B} \setminus p(B)$ contains at least one demand pair $(s,t) \in \mathcal{M}$. Among all such vertices, we choose one that is furthest from the root of the tree, breaking ties arbitrarily. Let v_B be the selected vertex. We add B to $\mathcal{B}$, and we define two new subsets of demand pairs $\mathcal{M}_B, \mathcal{M}'_B$, as follows. Set $\mathcal{M}_B$ contains all demand pairs (s,t) with $s,t \in G_{v_B} \setminus p(B)$. Set $\mathcal{M}'_B$ contains all demand pairs (s,t) with exactly one of s,t lying in $G_{v_B} \setminus p(B)$, while the other terminal must belong to G_r. Notice that any path connecting any demand pair in $\mathcal{M}'_B$ must use the vertex $p(B)$.

Next, we construct a new instance of the NDP-Disc problem, on the graph B. The corresponding set of demand pairs, that we denote by $\mathcal{N}_B$, is defined as follows. Consider any demand pair $(s,t) \in \mathcal{M}_B$. We define a new demand pair (s',t') representing (s,t), with $s',t' \in V(B)$, and define two paths: $Q(s)$ connecting s to s', and $Q(t)$ connecting t to t'. If $s \in V(B)$, then we let $s' = s$, and $Q(s) = \{s\}$. Otherwise, we let $s' \in V_1$ be the unique child of the vertex v_B in τ, such that $s \in G_{s'}$. Notice that s' must be a vertex of B. We then let $Q(s)$ be any simple path connecting s to s' in graph $G_{s'}$. We define the vertex $t' \in B$, and a path $Q(t)$ connecting t to t' similarly. Notice that it is possible that $s' = t'$.

Let $\mathcal{N}_B = \left\{(s',t') \mid (s,t) \in \mathcal{M}_B\right\}$ be the resulting set of demand pairs. All vertices participating in the demand pairs in $\mathcal{N}_B$ belong to B. Consider the NDP instance $(B,\mathcal{N})$. It is immediate to verify that we can draw B in a disc, so that all vertices participating in the demand pairs in $\mathcal{M}(B)$ lie on the boundary of the disc, and clearly B is 2-connected. We need the following immediate observation.

Observation 3.4.7. *$\mathsf{OPT}(B,\mathcal{N}_B) \geq \mathsf{OPT}(G,\mathcal{M}_B)$.*

Proof. Let $\tilde{\mathcal{P}}$ be the optimal solution to instance $\mathsf{OPT}(G,\mathcal{M}_B)$. We can assume that all paths in $\tilde{\mathcal{P}}$ are simple. Let $P \in \tilde{\mathcal{P}}$ be any such path, and assume that P connects some demand pair $(s,t) \in \mathcal{M}_B$. Then it is easy to see that $s',t' \in P$, and moreover, the segment of P between s' and t' is contained in B. By appropriately truncating every path in $\tilde{\mathcal{P}}$, we can obtain a solution to instance $(B,\mathcal{N}_B)$ of the NDP problem of the same value. □

Since B is 2-vertex connected, and can be drawn in a disc with all vertices participating in the demand pairs in $\mathcal{N}_B$ lying on the disc boundary, we can apply the algorithm from Theorem 3.4.5 to instance $(B,\mathcal{N}_B)$, to compute a set $\mathcal{P}(B)$ of node-disjoint paths, routing a subset of at least $\Omega(\mathsf{OPT}(B,\mathcal{N}_B)/\log k)$ demand pairs of $\mathcal{M}_B$ in B. We can assume that $|\mathcal{P}(B)| \geq 1$, since otherwise we can route any demand pair in $\mathcal{N}_B$. We would like to ensure that all paths in $\mathcal{P}(B)$ avoid the vertex $p(B)$. If $|\mathcal{P}(B)| = 1$, then, since B is 2-vertex connected, we can re-route the unique path in $\mathcal{P}(B)$ inside B, so that its endpoints remain the same, but it avoids the vertex $p(B)$ (since $G_{v_B} \setminus p(B)$ contains at least one demand pair, we can ensure that the endpoints of the path are distinct from $p(B)$). Otherwise, we discard from $\mathcal{P}(B)$ the path that uses vertex $p(B)$, if such exists. By concatenating the paths in $\mathcal{P}(B)$ with the paths in $\{Q(s), Q(t) \mid (s,t) \in \mathcal{M}(B)\}$, we obtain a collection $\mathcal{P}'(B)$ of at least $\Omega(\mathsf{OPT}(B, \mathcal{N})/\log k)$ node-disjoint paths, connecting demand pairs in $\mathcal{M}(B)$. We add the paths in $\mathcal{P}'(B)$ to $\mathcal{P}$, and delete from τ all vertices of τ_{v_B}. Since we have ensured that the paths in $\mathcal{P}'(B)$ are disjoint from $p(B)$, the invariant that the paths in $\mathcal{P}$ are disjoint from the new graph G_r continues to hold. The algorithm terminates when no demand pair $(s,t) \in \mathcal{M}$ is contained in G_r.

We claim that the resulting collection $\left\{\mathcal{M}(B), \mathcal{M}'(B) \mid B \in \mathcal{B}\right\}$ of sets of demand pairs partitions $\mathcal{M}$. Indeed, consider any demand pair $(s,t) \in \mathcal{M}$, and consider the last iteration i when both $s, t \in G_r$. Let v_B be the vertex that was processed in the following iteration. If both $s, t \in G_{v_B} \setminus p(B)$, then (s,t) was added to $\mathcal{M}_B$. Otherwise, exactly one of s, t belongs to $G_{v_B} \setminus p(B)$, while the other belongs to G_r, so (s,t) was added to $\mathcal{M}'_B$. We now obtain a set $\mathcal{P}$ of disjoint paths, routing a subset of vertices in $\mathcal{M}$.

We show that $|\mathcal{P}| \geq \Omega(\mathsf{OPT}(G, \mathcal{M})/\log k)$. Let $\mathcal{P}^*$ be the optimal solution to instance $\mathsf{OPT}(G, \mathcal{M})$, and let $\mathcal{M}^*$ be the set of the demand pairs routed by $\mathcal{P}^*$. For every block $B \in \mathcal{B}$, let $\tilde{\mathcal{M}}(B) = \mathcal{M}^* \cap \mathcal{M}_B$, and let $\tilde{\mathcal{M}}'_B = \mathcal{M}^* \cap \mathcal{M}'_B$.

From Observation 3.4.7, set $\mathcal{P}'(B)$ of paths routes at least

$$\Omega(\mathsf{OPT}(B, \mathcal{N}_B)/\log k) \geq \Omega(\mathsf{OPT}(G, \mathcal{M}_B)/\log k) \geq \Omega(|\tilde{\mathcal{M}}_B|/\log k)$$

demand pairs. Therefore,

$$|\mathcal{P}| = \sum_{B\in\mathcal{B}} |\mathcal{P}'(B)| \geq \sum_{B\in\mathcal{B}} \Omega(|\tilde{\mathcal{M}}_B|/\log k).$$

On the other hand, as observed above, for every block $B \in \mathcal{B}$, $|\tilde{\mathcal{M}}'_B| \leq 1$ (since all paths routing the pairs in $\tilde{\mathcal{M}}'_B$ contain vertex $p(B)$), while $|\mathcal{P}'(B)| \geq 1$. Therefore,

$$|\mathcal{P}| \geq \sum_{B\in\mathcal{B}} |\tilde{\mathcal{M}}'_B|.$$

Overall,

$$|\mathcal{P}| \geq \sum_{B\in\mathcal{B}} \Omega((|\tilde{\mathcal{M}}_B| + |\tilde{\mathcal{M}}'_B|)/\log k) = \Omega(|\mathcal{M}^*|/\log k).$$

3.5 Approximating Node-Disjoint Paths on a Cylinder

In this section we prove Theorem 3.1.1 for NDP-Cylinder. Recall that in the NDP-Cylinder problem, we are given a cylinder Σ, obtained from the sphere, by removing two open discs from it. We denote the boundaries of the two discs by Γ_1 and Γ_2, respectively. We assume that we are given a graph G, drawn on Σ, and a set $\mathcal{M}$ of demand pairs. We denote by S and T the sets of all source and all destination vertices participating in the demand pairs in $\mathcal{M}$.

We say that a drawing of G is proper with respect to S and T iff the vertices of S lie on Γ_1, the vertices of T lie on Γ_2, and no other edges or vertices of G intersect Γ_1 or Γ_2. We can assume without loss of generality that we are given a proper drawing of the input graph G on Σ with respect to S and T. We also assume that the graph G is connected, as otherwise we can solve the problem for each connected component of G separately.

We assume that we know the value OPT of the optimal solution to instance $(G, \mathcal{M})$, and a demand pair $(s^*, t^*) \in \mathcal{M}$ that is routed by some optimal solution to instance $(G, \mathcal{M})$. We can make these assumptions by solving the problem for every possible value of OPT between 1 and $|\mathcal{M}|$, and every choice of $(s^*, t^*) \in \mathcal{M}$. It is enough to show that the algorithm returns the desired solution when the value OPT and the pair (s^*, t^*) are guessed correctly. We can also assume that $\mathsf{OPT} > 10$, since otherwise routing a single demand pair gives a desired solution.

We define a set W_1 of points on Γ_1, as follows. First, we add to Γ_1 all points corresponding to the vertices of S. Next, for every segment of $\Gamma_1 \setminus W_1$, we add an arbitrary point on the segment to W_1. We define a set W_2 of points on Γ_2 similarly, using T instead of S. Our first step is to compute the shortest G-normal curve $\gamma^* \subseteq \Sigma$, connecting a point of W_1 to a point of W_2. We consider two cases.

Assume first that the length of γ^* is less than $\mathsf{OPT}/2$. Then we can cut the cylinder Σ along the curve γ^*, deleting from G all vertices lying on γ^*, to obtain a disc Σ', and a drawing of G on Σ', where all terminals of $S \cup T$ lie on the boundary of the disc. It is easy to see that the value of the optimal solution of the resulting problem instance is at least $\mathsf{OPT}/2$. We can now apply the $O(\log k)$-approximation algorithm for NDP-Disc from Section 3.4 to obtain an $O(\log k)$-approximate solution to the new NDP-Disc instance, which in turn gives an $O(\log k)$-approximate solution to the original instance of NDP-Cylinder.

We assume from now on that the length of γ^* is at least $\mathsf{OPT}/2$. We reduce the problem to DPSP. Let σ be cycle, whose vertices are W_1, and they are connected in the order of their appearance on Γ_1. We delete the edge of σ incident on s^*, that appears after s^* in the counter-clock-wise traversal of Γ_1, and direct all edges of the resulting path away from s^*. We define a path σ' similarly - start with the cycle, whose vertices are W_2, and they are connected in the order of their appearance on Γ_2. Delete the edge incident on t^*, that appears after t^* in the counter-clock-wise traversal of Γ_2, and direct all edges of the resulting path away from t^*. Our next step is to define a set $\mathcal{K}$ of constraints for the DPSP problem instance. The instance we construct will only contain type-1 and type-2 constraints.

Let a^* be the last vertex of σ. The first constraint that we add to $\mathcal{K}$ is $(1, s^*, a^*, \mathsf{OPT}/2)$. This constraint ensures that overall we will not attempt to route more than $\mathsf{OPT}/2$ demand pairs.

Consider now any pair $x, y \in W_1$ of points. Let $\beta_1(x, y)$ and $\beta_2(x, y)$ be the two segments of Γ_1 whose endpoints are x and y. For $i \in \{1, 2\}$, we let $\ell_i(x, y)$ be the smallest number of vertices that need to be removed from G, in order to disconnect all vertices of $\beta_i(x, y) \cap S$ from the vertices of T - this value can be computed efficiently using standard minimum cut algorithms. Let $w_i = \ell_i(x, y)$. We assume w.l.o.g. that x lies before y on σ. If s^* is not an inner vertex on $\beta_i(x, y)$, then we add the constraint $(1, x, y, w_i)$ to $\mathcal{K}$. Otherwise, we add two constraints: $(1, s^*, x, w_i)$ and $(1, y, a^*, w_i)$ to $\mathcal{K}$. For every pair of points $(x, y) \in W_1$, we therefore add at most three type-1 constraints to $\mathcal{K}$.

We process all pairs of points $(x, y) \in W_2$, and add corresponding type-2 constraints to $\mathcal{K}$ similarly, except that we use t^* instead of s^*. This finishes the description of the DPSP instance. We start with the following easy observation.

Observation 3.5.1. *Let $\mathcal{P}^*$ be the optimal solution to the NDP instance $(G, \mathcal{M})$, such that (s^*, t^*) is routed by $\mathcal{P}^*$, and let $\mathcal{M}^*$ be the set of the demand pairs routed by $\mathcal{P}^*$. Let $\mathcal{M}^{**} \subseteq \mathcal{M}^*$ be any subset of $\lfloor |\mathcal{M}^*|/2 \rfloor$ demand pairs. Then $\mathcal{M}^{**}$ is a feasible solution to the DPSP instance $(\sigma, \sigma', \mathcal{M}, \mathcal{K})$.*

Proof. Since we assume that the demand pair (s^*, t^*) is routed by $\mathcal{P}^*$, and since the demand pairs in $\mathcal{M}^*$ must be non-crossing with respect to Γ_1 and Γ_2, due to the way in which we have defined the paths σ and σ', set $\mathcal{M}^{**}$ must be non-crossing with respect to σ and σ'.

Recall that we have added the constraint $(1, s^*, a^*, |\mathsf{OPT}|/2)$ to $\mathcal{K}$, where s^* and a^* are the endpoints of σ. Since

$$|\mathcal{M}^{**}| \leq |\mathcal{M}^*|/2 = |\mathsf{OPT}|/2,$$

set $\mathcal{M}^{**}$ satisfies this constraint.

Consider now any pair (x, y) of points in W_1, and fix some $i \in \{1, 2\}$. Since set $\mathcal{M}^*$ of demand pairs is routable in G, the number of the source vertices of the demand pairs in $\mathcal{M}^*$ that lie on $\beta_i(x, y)$ is at most ℓ_i, as the value of the minimum cut separating the vertices of $S \cap \beta_i(x, y)$ from the vertices of T is ℓ_i. It is now easy to verify that all type-1 constraints in $\mathcal{K}$ corresponding to the pair (x, y) are satisfied by $\mathcal{M}^*$, and hence by $\mathcal{M}^{**}$. Type-2 constraints are dealt with similarly. $\square$

Our next step is to use the algorithm from Theorem 3.3.1, in order to compute a set $\mathcal{M}' \subseteq \mathcal{M}$ of non-crossing (with respect to σ and σ') demand pairs, satisfying all constraints in $\mathcal{K}$, with $|\mathcal{M}'| \geq \Omega(\mathsf{OPT}(G, \mathcal{M})/\log k)$. We assume that $\mathcal{M}' = \{(s_1, t_1), \ldots, (s_r, t_r)\}$, where $s_1, \ldots, s_r$ appear in this circular order on Γ_1, and if $(s^*, t^*) \in \mathcal{M}'$, then $s_1 = s^*$. If $|\mathcal{M}'| \leq 10$, then a routing of a single demand pair gives a feasible solution to the NDP problem instance and achieves the desired approximation ratio.

Therefore, we assume that $|\mathcal{M}'| > 10$. We let $\mathcal{M}''$ contain all demand pairs (s_j, t_j), where $j = 0$ modulo 8, and $1 \leq j \leq r$. Notice that $\mathcal{M}''$ excludes the pair (s^*, t^*). Let S'' and T'' be the sets of the source and the destination vertices of the demand pairs in $\mathcal{M}''$. We need the following theorem.

Theorem 3.5.2. *There is a set $\mathcal{P}$ of $|S''|$ node-disjoint paths in G, connecting the vertices of S'' to the vertices of T''.*

We prove Theorem 3.5.2 below, after we complete the proof of Theorem 3.1.1 using it. Denote $|\mathcal{M}''| = \kappa^*$, and recall that from our constraints, $\kappa^* \leq |\mathsf{OPT}|/4$. Our first step is to construct a collection $\mathcal{Z} = (Z_1, \ldots, Z_{\kappa^*})$ of κ^* tight concentric cycles around Γ_1, where we consider a planar drawing of G, whose outer face contains Γ_2. In order to do so, we denote $\Gamma_1 = Z_0$, and perform κ^* iteration, where in the ith iteration we construct the cycle Z_i. In order to execute the ith iteration, for $1 \leq i \leq \kappa^*$, we contract $D(Z_{i-1})$ into a single vertex s, to obtain a new graph H_i. We view the face of H_i containing Γ_2 as the outer face in the planar drawing of H_i, and we then let $Z_i = \text{min-cycle}(H, s)$. Since the length of the shortest

G-normal curve connecting a point of Γ_1 to a point of Γ_2 is at least $|\mathsf{OPT}|/2 > \kappa^*$, it is easy to verify that we can successfully complete the construction of the set $\mathcal{Z}$ of κ^* cycles, so that all cycles are disjoint from the vertices lying on Γ_2.

Our next step is to re-route the paths in $\mathcal{P}$, so that they become monotone with respect to $\mathcal{Z}$. In order to do so, we construct a graph H, as follows. We start with the union of the paths in $\mathcal{P}$ and the cycles in $\mathcal{Z}$. We then add a new cycle Y connecting the vertices of T'' in the order in which they appear on Γ_2. We can now use Theorem 3.2.5 to find a collection $\mathcal{P}'$ of κ^* node-disjoint paths in H, connecting vertices of S'' to vertices of T'', that are monotone with respect to $\mathcal{Z}$. It is easy to see that the paths of $\mathcal{P}'$ are contained in G.

We assume that $\mathcal{P}' = \{P_1, P_2, \ldots, P_{\kappa^*}\}$, and for each $1 \leq i \leq \kappa^*$, we denote by $a_i \in S''$ and $b_i \in T''$ the endpoints of P_i. We assume that $a_1, a_2, \ldots, a_{\kappa^*}$ appear in this circular order on Γ_1. Consider the source vertex $a_1 \in S''$, and let $(a_1, b_{1+z}) \in \mathcal{M}''$ be the demand pair in which a_1 participates. We can assume without loss of generality that $z \leq \kappa^*/2$, since if this is not the case, we can re-order the vertices $a_1, \ldots, a_{\kappa^*}$ in the opposite direction around Γ_1. Observe that for all $1 \leq j \leq \lfloor \kappa^*/2 \rfloor$, pair $(a_j, b_{j+z}) \in \mathcal{M}''$, since the demand pairs in $\mathcal{M}''$ are non-crossing.

We now show how to route all demand pairs in $\left\{(a_j, b_{j+z})\right\}_{1 \leq z \leq \lfloor \kappa^*/2 \rfloor}$. Fix some $1 \leq j \leq \lfloor \kappa^*/2 \rfloor$. We view the paths in $\mathcal{P}'$ as directed from S'' to T''. Let P'_j be the sub-path of P_j from a_j to the first vertex v_j of P_j lying on Z_{κ^*-j+1}. Let P''_{j+z} be the sub-path of P_{j+z}, from the last vertex v'_{j+z} of P_{j+z} lying on Z_{κ^*-j+1} to b_{j+z}. Finally, let Q_j be the segment of Z_{κ^*-j+1} between v_j to v'_{j+z}, that intersects the paths $P_j, P_{j+1}, \ldots, P_{j+z}$, but no other paths of $\mathcal{P}'$. By combining P'_j, P''_{j+z} and Q_j, we obtain a path P^*_j, connecting a_j to b_{j+z}. We then set $\mathcal{P}^* = \left\{P^*_j \mid 1 \leq j \leq z\right\}$. It is immediate to verify that the paths in $\mathcal{P}^*$ are node-disjoint. Therefore, we obtain a solution routing $\Omega(\mathsf{OPT}(G, \mathcal{M}))$ demand pairs in $\mathcal{M}$. It now only remains to prove Theorem 3.5.2.

Proof of Theorem 3.5.2. Assume for contradiction that there is no such set of paths.

Denote $|S''| = \kappa$ and recall that $\kappa < \mathsf{OPT}/2$. Then there is a set Y of at most $\kappa - 1$ vertices, so that $G \setminus Y$ contains no path connecting a vertex of S to a vertex of T. Consider the drawing of G on the sphere Σ'', obtained from the drawing of G on the cylinder Σ, by adding back the two caps with the boundaries Γ_1 and Γ_2. We can then construct a simple closed G-normal curve γ of length at most $\kappa - 1$ in Σ'', so that all vertices of S'' lie in one of the discs of Σ'' with boundary γ, and all vertices of T'' lie on the other disc (but the vertices of S'' and T'' may lie on γ). Notice that γ has to cross Γ_1 or Γ_2. Indeed, otherwise, since there are OPT node-disjoint paths connecting the vertices of Γ_1 to the vertices of Γ_2, all such paths would have to cross γ, and so the length of γ should be at least $\mathsf{OPT} > \kappa$, a contradiction. Moreover, since the length of the shortest G-normal curve connecting a point of Γ_1 to a point of Γ_2 is at least $\mathsf{OPT}/2 > \kappa$, curve γ may not intersect both Γ_1 and Γ_2. We assume without loss of generality that γ crosses Γ_1, and not Γ_2.

Let $\mathcal{R}$ be the set of segments of γ, obtained by deleting all points of γ that lie outside the cylinder Σ (that is, the points that lie in the interior of the cap whose boundary is Γ_1). All curves in $\mathcal{R}$ are mutually disjoint. For each curve $\gamma' \in \mathcal{R}$, let $S(\gamma') \subseteq S''$ be the set of the source vertices that γ' separates from Γ_2 in the cylinder Σ. Then $\bigcup_{\gamma' \in \mathcal{R}} S(\gamma') = S''$ must hold, and so there must be some curve $\gamma^* \in \mathcal{R}$, such that the length of γ^* is less than $|S(\gamma^*)|$. Let ℓ^* denote the length of γ^*.

Let x', y' be the endpoints of the curve γ^*, so $x', y' \in \Gamma_1$. If $x' \in W_1$, then we let $x = x'$. Otherwise, we let x be the closest to x' point of $W_1 \setminus S''$ on Γ_1. We define point y' for y similarly. Consider the two segments $\beta_1(x,y)$ and $\beta_2(x,y)$ of Γ_1, whose endpoints are x and y. One of the segments, say $\beta_1(x,y)$ must contain all points of $S(\gamma^*)$. Since the vertices lying on γ^* separate all vertices of $S(\gamma^*)$ from the vertices of T, $\ell_1(x,y) \leq \ell^*$. Assume without loss of generality that x lies before y on σ.

Assume first that s^* does not lie on $\beta_1(x,y)$, and consider the corresponding constraint $K = (1, x, y, w_1) \in \mathcal{K}$. As observed above, $w_1 \leq \ell^*$. Due to the way we have selected the subset $\mathcal{M}'' \subseteq \mathcal{M}'$ of the demand pairs, we are guaranteed that $|S(\gamma^*)| \leq w_1/2 \leq \ell^*$, a contradiction.

Assume now that s^* lies on $\beta_1(x,y)$. Using the same reasoning as above, $w_1 \leq \ell^*$. Let β_1', β_1'' be the segments of $\beta_1(x,y)$ between x and s^*, and between s^* and y, respectively, where the last segment excludes s^*. Let Δ_1, Δ_1' be the number of the source vertices lying on β_1 and β_1'' respectively, that participate in the demand pairs in $\mathcal{M}'$. Since the constraints $(1, s^*, x, w_1)$ and $(1, y, a^*, w_1)$ belong to $\mathcal{K}$, $\Delta_1 + \Delta_1' \leq 2\ell^*$. Due to the way we have selected the subset $\mathcal{M}'' \subseteq \mathcal{M}'$ of the demand pairs, we are guaranteed that $|S(\gamma^*)| \leq \max\{w_1, 1\} \leq \ell^*$, a contradiction. $\square$

CHAPTER 4

IMPROVED HARDNESS RESULT FOR PLANAR GRAPHS

4.1 Introduction

Our main result of this chapter is the proof of the following theorem.

Theorem 4.1.1. *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 4, where all source vertices lie on the boundary of a single face.*

The proof is based on the joint paper with Chuzhoy and Nimavat [19]. The paper also contains an extension of the above theorem that shows that the same hardness of approximation result holds for planar graphs with maximum degree 3, where the source vertices lie on the boundary of a single face. It also shows that the same lower bound holds for EDP, where the input graph is planar with maximum vertex degree 3, with all source vertices lying on the boundary of a single face. In this thesis, we show the proof for planar graphs with maximum vertex degree 4, with sources on the boundary of a single face.

We first provide an informal high-level overview of the proof of Theorem 4.1.1. 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 [8, 7] shows that, unless $\mathsf{P} = \mathsf{NP}$, no efficient algorithm can distinguish between the YES- and the NO-INSTANCES of 3SAT(5).

We perform $\Theta(\log n)$ iterations, where in iteration i we construct what we call a *level-i* instance. 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 4.1). In other words, we choose a sub-path P of the first row of the grid, of the same length as $Z(\mathcal{I})$, and map the vertices of $Z(\mathcal{I})$ to P in a natural way. 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. We note that box $B(\mathcal{I})$ may be constructed recursively, for example, by placing several boxes $B(\mathcal{I}')$ corresponding to 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.

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 describe our construction of level-1 instances in Section 4.3, together with an intuition for constructing higher-level instances.

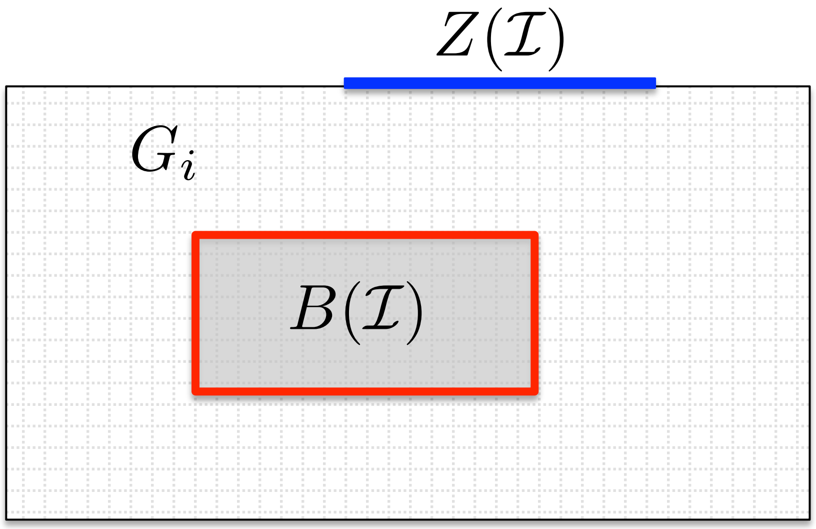


Figure 4.1: A schematic view of a level-i instance $\mathcal{I}$. All source vertices lie on $Z(\mathcal{I})$, whose location can be chosen arbitrarily on the first row of G_i. All destination vertices belong to $B(\mathcal{I})$, that can be located anywhere in G_i, far enough from its boundaries.

4.2 Preliminaries

Grids. 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 $G^{\ell,h}$ is $V(G^{\ell,h}) = \{v(i,j) \mid 1 \le i \le h, 1 \le j \le \ell\}$, and the set of its edges 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 $G^{\ell,h}$ 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 $1 \leq j \leq \ell$, 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 two vertex subsets $X, Y \subseteq V(G^{\ell,h})$, the distance between them is

$$d(X, Y) = \min_{u \in X, u' \in Y} \left\{d(u, u')\right\}.$$

Given a vertex $v = v(i, j)$ of the grid, we denote by $\text{row}(v)$ and $\text{col}(v)$ the row and the column, respectively, that contain v.

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.

Let R', R'' be two distinct rows of some grid G, and let $\mathcal{P}$ be a set of node-disjoint paths, such that every path in $\mathcal{P}$ has one endpoint (called a source) on row R' and another (called a destination) on row R''. We say that the set $\mathcal{P}$ of paths is *order-preserving* iff the source vertices of the paths in $\mathcal{P}$ appear on row R' in exactly the same left-to-right order as their destination vertices on R''.

Walls. 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 $z \neq j \mod 2$, 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 4.2). 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 $\hat{G}$.

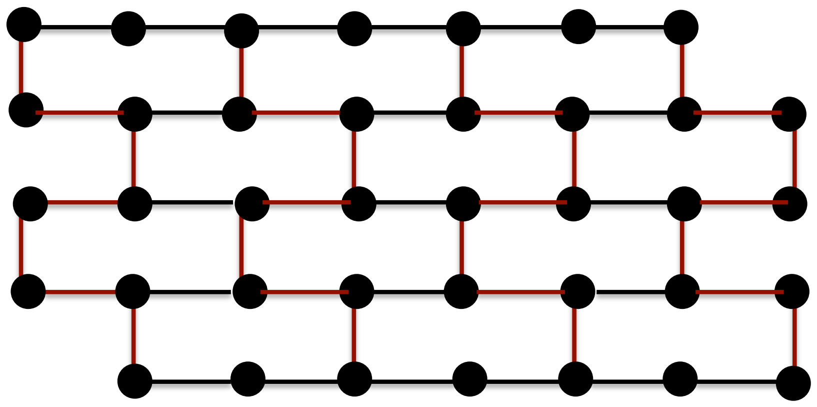


Figure 4.2: A wall of height 5 and length 4; the columns of the wall are shown in red.

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.

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 set $U(B)$ 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 remaining 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. We say that two paths P, P' are *internally disjoint* iff every vertex in $P \cap P'$ is an endpoint of both paths. Given a set $\mathcal{P}$ of paths and some set U of vertices, we say that the paths in $\mathcal{P}$ are internally disjoint from U iff for every path $P \in \mathcal{P}$, every vertex in $P \cap U$ is an endpoint of P. Given a subgraph G' of a graph G and a set $\mathcal{P}$ of paths in G, we say that the paths in $\mathcal{P}$ are internally disjoint from G' iff they are internally disjoint from $V(G')$.

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:

- integral parameters L_i, L'_i and an even integer H_i;
- a path $Z(\mathcal{I})$ of length L_i;
- 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
- 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.

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 $1 \le j \le r$, $P_j \cap A$ 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 $\mathcal{C} = \{C_1,\ldots,C_m\}$ 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 [8, 7].

Theorem 4.2.1. *There is a constant $\epsilon : 0 < \epsilon < \frac{1}{2}$, such that it is NP-hard to distinguish between the* YES-INSTANCES *and the* NO-INSTANCES *(defined with respect to ϵ) of the 3SAT(5) problem.*

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 $\mathsf{NP} \subseteq \mathsf{DTIME}(n^{O(\log n)})$. 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.

4.3 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 4.4 contains all formal definitions and proofs, including those for the level-1 instance, the description here is informal, and we only provide proof sketches.

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 $\mathcal{C} = \{C_1, \ldots, C_m\}$ 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 that represent that variable, and similarly, for every clause $C_q \in \mathcal{C}$ 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

$$\mathcal{M}^C = \bigcup_{C_q \in \mathcal{C}} \mathcal{M}(C_q)$$

clause-pairs.

We now define the parameters we use in the construction:

$$h = 1000/\epsilon$$
$$\delta = 8\epsilon^2/10^{12}$$

where ϵ is the parameter from Theorem 4.2.1. We also set

$$N_1 = (200h/3 + 1)n$$
$$N_1' = (1 - \delta)N_1.$$

Our construction will ensure that, if the input formula φ is a YES-INSTANCE, then for every instantiation of $\mathcal{I}$, there is a collection $\mathcal{P}$ 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

$$L_1 = (80h+2)n$$
$$L'_1 = 20N_1^3$$
$$H_1 = 20N_1.$$

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 4.3).

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 4.3). 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

$$H^C = 3.$$

As before, 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 4.3).

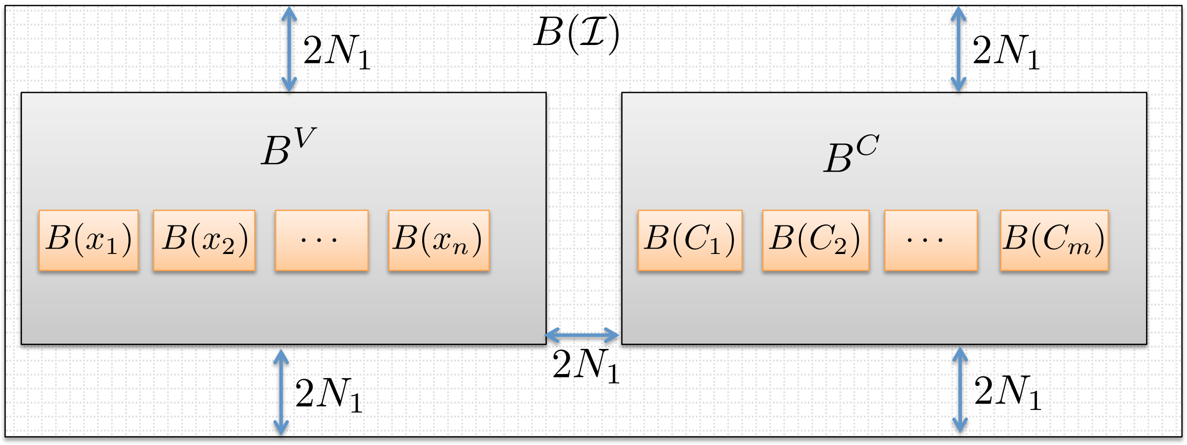


Figure 4.3: High-level view of the level-1 construction. Boxes $B(x_1), \ldots, B(x_n)$ are at a distance at least $2N_1$ from each other and from the boundaries of B^V, and boxes $B(C_1), \ldots, B(C_m)$ are at a distance at least $4N_1$ from each other and from the left and right boundaries of B^C.

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 $1 \leq j \leq n$, 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.

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 4.4(a)). The set $\mathcal{M}(x)$ of demand pairs consists of three subsets:

- **(Extra Pairs).** Let
$$\mathcal{M}^X(x) = \left\{(s^X_y(x), t^X_y(x))\right\}_{y=1}^{60h}$$
be a set of $60h$ demand pairs that we call the Extra pairs for x. The vertices $s^X_1(x), \ldots, s^X_{60h}(x)$ appear on $I^X(x)$ in this order, and the vertices $t^X_1(x), \ldots, t^X_{60h}(x)$ appear on $\hat{I}^X(x)$ in this order.
- **(**True **Pairs).** We denote the vertices appearing on $I^T(x)$ by
$$a^T_1, b^T_1, a^T_2, b^T_2, \ldots, b^T_{5h}, a^T_{5h+1}$$

in this left-to-right order. Let

$$\mathcal{M}^T(x) = \left\{(s_y^T(x), t_y^T(x))\right\}_{y=1}^{5h+1}$$

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

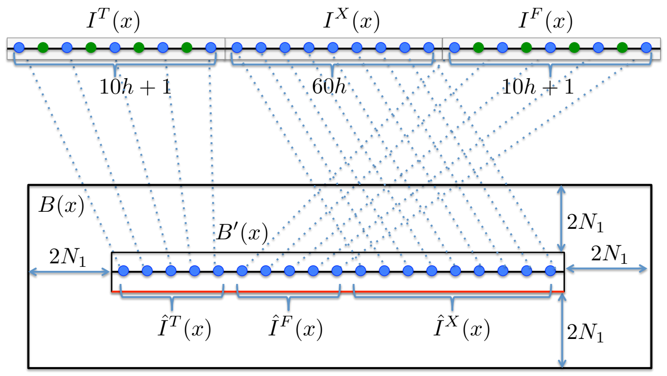
$$\mathcal{M}^F(x) = \left\{(s_y^F(x), t_y^F(x))\right\}_{y=1}^{5h+1}$$

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 $\hat{I}^F(x)$.

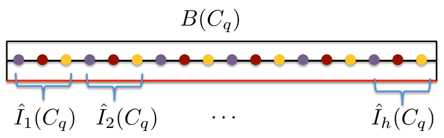
We let $\mathcal{M}(x) = \mathcal{M}^X(x) \cup \mathcal{M}^T(x) \cup \mathcal{M}^F(x)$ be the set of demand pairs representing x.

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 pairs $\left\{(s_j(C_{r'}, x), t_j(C_{r'}, x))\right\}_{j=1}^h$, representing the literal x of $C_{r'}$. Consider the interval $I^F(x)$. We will use its vertices b_j^F as the sources of these demand pairs, where, intuitively, sources corresponding to the same clause-literal pair appear consecutively.

Formally, for each $1 \le r' \le r$ and $1 \le j \le h$, we identify the vertex $s_j(C_{r'}, x)$ with the vertex $b_{(r'-1)h+j}^F$ 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_j^F, thus preventing us from routing demand pairs that represent clause-literal pairs $(C_{r'}, x)$.



(a) A variable gadget. The $(5h)$ green vertices of $I^F(x)$ are partitioned into 5 groups of h consecutive vertices; each group is used by a different clause that contains the literal x. The green intervals of I^T are dealt with similarly, but they are used by clauses containing $\neg x$. The vertices on the bottom boundary of $B'(x)$ are deleted.



(b) A clause gadget. Vertices of different colors correspond to different literals. The vertices on the bottom boundary of $B(C_q)$ are deleted.

Figure 4.4: A variable gadget and a clause gadget

We treat the subset $\mathcal{C}'(x) \subseteq \mathcal{C}$ of clauses containing the literal $\neg x$ similarly, except that we identify their source vertices with the vertices of $\left\{b_1^T, \ldots, b_{5h}^T\right\}$ of $I^T(x)$.

Clause Gadgets. Consider some clause $C_q = (\ell_{q1} \vee \ell_{q2} \vee \ell_{q3})$. For each one of the three literals $\ell \in \left\{\ell_{q1}, \ell_{q2}, \ell_{q3}\right\}$ of C_q, let

$$\mathcal{M}(C_q, \ell) = \left\{(s_j(C_q, \ell), t_j(C_q, \ell)) \mid 1 \le j \le h\right\}$$

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 4.4(b)). Fix some $1 \le j \le h$. We identify the three vertices of $\hat{I}_j(C_q)$ with the destination vertices $t_j(C_q, \ell_{q1})$, $t_j(C_q, \ell_{q2})$, and $t_j(C_q, \ell_{q3})$ in this order. For each $1 \le z \le 3$, the corresponding source vertex $s_j(C_q, \ell_{qz})$ has already been defined as part of the definition of the variable gadget corresponding to the literal ℓ_{qz}. Let $\mathcal{M}(C_q) = \bigcup_{z=1}^{3} \mathcal{M}(C_q, \ell_{qz})$ 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, and then define the routing.

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 $C_q = (\ell_{q1} \vee \ell_{q2} \vee \ell_{q3}) \in \mathcal{C}$, 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

$$\hat{\mathcal{M}}^V = \bigcup_{j=1}^{n} \hat{\mathcal{M}}(x_j)$$
$$\hat{\mathcal{M}}^C = \bigcup_{q=1}^{m} \hat{\mathcal{M}}(C_q).$$

We then set $\hat{\mathcal{M}} = \hat{\mathcal{M}}^V \cup \hat{\mathcal{M}}^C$, so

$$|\hat{\mathcal{M}}| = (65h+1)n + mh = (200h/3+1)n = N_1.$$

We now show that all demand pairs in $\hat{\mathcal{M}}$ can be routed by a set $\mathcal{P}$ of paths that respects the box $B(\mathcal{I})$. We only provide an intuitive description of the routing here; a formal proof appears in Section 4.5.

Consider some variable x, and assume that it is assigned the value TRUE. Let $\hat{\mathcal{M}}'(x) \subseteq \hat{\mathcal{M}}^C$ be the set of all clause-pairs, whose source vertices lie on the interval $I(x)$. Since the current assignment to x must satisfy their corresponding clauses, all vertices of $S(\hat{\mathcal{M}}'(x))$ lie on $I^F(x)$. Similarly, if x is assigned the value FALSE, then all vertices of $S(\hat{\mathcal{M}}'(x))$ lie on $I^T(x)$. Therefore, for every variable x, the sources of the demand pairs in $\hat{\mathcal{M}}(x)$ appear consecutively on $Z(\mathcal{I})$ (relatively to the vertices of $S(\hat{\mathcal{M}})$), and the same holds for the sources of the demand pairs in $\hat{\mathcal{M}}(C_q)$, for every clause C_q.

We build the paths in $\mathcal{P}$ gradually, growing them from the source vertices. We start by selecting, for every variable x, a set $A(x)$ of $|\hat{\mathcal{M}}(x)|$ vertices on the top boundary of $B(x)$, and similarly, for every clause C_q, a set $A(C_q)$ of $|\hat{\mathcal{M}}(C_q)|$ vertices on the top boundary of $B(C_q)$. We discuss this selection later. In the first step, we route the paths from their source vertices to these newly selected vertices, so that for every variable x, all paths originating from the vertices of $S(\hat{\mathcal{M}}(x))$ terminate at the vertices of $A(x)$ and they are order-preserving, and similarly for every clause $C_q \in \mathcal{C}$, all paths originating from the vertices of $S(\hat{\mathcal{M}}(C_q))$

terminate at the vertices of $A(C_q)$ and they are order-preserving.

In order to execute this step, we carefully select a set Γ of $|\hat{\mathcal{M}}|$ vertices on the top boundary of B^V; a set Γ'' of $|\hat{\mathcal{M}}^C|$ vertices on the bottom boundary of B^V; and the set Γ''' of $|\hat{\mathcal{M}}^C|$ left-most vertices on the top boundary of B^C. We first connect all vertices in $S(\hat{\mathcal{M}})$ to the $|\hat{\mathcal{M}}|$ leftmost vertices on the opening of $B(\mathcal{I})$ via a set of order-preserving node-disjoint paths, and then extend them to the vertices of Γ via node-disjoint order-preserving paths. This part of the routing is straightforward.

For every variable x, the paths originating at the vertices of $S(\hat{\mathcal{M}}(x))$ then continue to their corresponding boxes $B(x)$, while for each clause C_q, the paths originating at the vertices of $S(\hat{\mathcal{M}}(C_q))$ continue to the bottom boundary of B^V and terminate at a consecutive set of vertices of Γ'' (see Section 4.5 and Figure 4.11 for more details). We select the vertices Γ and Γ'' in a way that ensures that every path that we route inside the box B^V can be implemented by a sub-path of a column of B^V.

We then connect the vertices of Γ'' to the vertices of Γ''' via a set of node-disjoint order-preserving paths, that are internally disjoint from the boxes B^V and B^C; these paths exploit the spacing between the two boxes. Finally, we complete the routing inside the box B^C, ensuring that for every clause C_q, the paths originating at the vertices of $S(\hat{\mathcal{M}}(C_q))$ terminate at the vertices of $A(C_q)$. This is done via a standard snake-like routing. This routing critically uses the fact that the endpoints of the paths that we have constructed so far, which originate at the vertices of $S(\hat{\mathcal{M}}(C_q))$, appear consecutively on Γ'''.

By appropriately choosing, for every clause $C_q \in \mathcal{C}$, the set $A(C_q)$ of vertices on the top boundary of $B(C_q)$, it is easy to extend the paths originating at the vertices of $S(\hat{\mathcal{M}}(C_q))$, so that they terminate at the vertices in $T(\hat{\mathcal{M}}(C_q))$. Since the resulting paths are order-preserving, we will route all demand pairs in $\hat{\mathcal{M}}(C_q)$.

We now consider some variable x and show how to complete the routing of the demand pairs in $\hat{\mathcal{M}}(x)$ inside the box $B(x)$. Assume first that x is assigned the value TRUE. Then the paths routing the demand pairs in $\hat{\mathcal{M}}(x)$ arrive at the top boundary of $B(x)$ in the same order as the ordering of their source vertices on $Z(\mathcal{I})$. The order of their corresponding destination vertices on the second row of $B'(x)$ is identical, and so it is immediate to extend the paths inside $B(x)$ to complete the routing.

Assume now that x is assigned the value FALSE. Let J and J' be the intervals of the top boundary of $B(x)$ where the paths routing the pairs in $\hat{\mathcal{M}}^F(x)$ and $\hat{\mathcal{M}}^X(x)$ arrive, respectively. Unfortunately, J lies to the right of J', while interval $\hat{I}^F(x)$ lies to the left of the interval $\hat{I}^X(x)$ on the second row of $B'(x)$. Therefore, we need to "switch" the ordering of these two sets of paths before we can complete the routing. It is easy to do so by exploiting the ample spacing between the box $B'(x)$ and the boundaries of the box $B(x)$ (see Figure 4.14).

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 $\mathcal{C}'$ be the set of clauses obtained by adding, for each clause $C_q \in \mathcal{C}$, h copies $C_q^1, \ldots, C_q^h$ of C_q to $\mathcal{C}'$. We will refer to the clauses in $\mathcal{C}$ as the *original clauses*, and the clauses in $\mathcal{C}'$ as the *new clauses*. Notice that $|\mathcal{C}'| = mh$, and it is easy to verify that no assignment to the variables of φ can satisfy more than $(1-\epsilon)hm$ clauses of $\mathcal{C}'$. We will reach a contradiction by defining an assignment to the variables of φ that satisfies more than $(1-\epsilon)hm$ clauses of $\mathcal{C}'$.

For each new clause $C_q^j \in \mathcal{C}'$, we let $\mathcal{M}(C_q^j) \subseteq \mathcal{M}$ be the set of all demand pairs whose destinations lie on interval $\hat{I}_j(C_q)$; we view these demand pairs as representing the clause C_q^j.

For every variable x of φ, let $\tilde{\mathcal{M}}(x) = \tilde{\mathcal{M}} \cap \mathcal{M}(x)$, and let $\tilde{\mathcal{M}}^T(x), \tilde{\mathcal{M}}^F(x), \tilde{\mathcal{M}}^X(x)$ 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 $C_q^j \in \mathcal{C}'$, let $\tilde{\mathcal{M}}(C_q^j) = \tilde{\mathcal{M}} \cap \mathcal{M}(C_q^j)$. We also denote by

$$\tilde{\mathcal{M}}^V = \bigcup_{j=1}^{n} \tilde{\mathcal{M}}(x_j)$$

and by

$$\tilde{\mathcal{M}}^C = \bigcup_{C_q^j \in \mathcal{C}'} \tilde{\mathcal{M}}(C_q^j),$$

the sets of the variable-pairs and the clause-pairs, respectively, that are routed by $\tilde{\mathcal{P}}$. We use the following claim.

Claim 4.3.1. *For each variable x of φ, at least one of the sets $\tilde{\mathcal{M}}^T(x), \tilde{\mathcal{M}}^F(x), \tilde{\mathcal{M}}^X(x)$ is empty.*

The proof is omitted here; we prove a somewhat stronger claim in Section 4.6 (see also Observation 4.6.6 from which the claim follows immediately). 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 $(s_j^T(x), t_j^T(x)) \notin \tilde{\mathcal{M}}(x)$. Otherwise, if $\tilde{\mathcal{M}}^F(x) \neq \emptyset$, then we assign x the value FALSE. In this case, we say that an index $1 \leq j \leq 5h+1$ is bad for x, iff $(s_j^F(x), t_j^F(x)) \notin \tilde{\mathcal{M}}(x)$. We later show that the total number of pairs (x, j), where j is a bad index for variable x, is small.

Consider some new clause C_q^j. We say that it is an *interesting* clause if $|\tilde{\mathcal{M}}(C_q^j)| \geq 1$ (in other words, at least one pair in the set $\left\{(s_j(C_q, \ell_{qz}), t_j(C_q, \ell_{qz}))\right\}_{z=1}^{3}$ is in $\tilde{\mathcal{M}}$), and we say that it is uninteresting otherwise. We say that C_q^j is *troublesome* iff $|\tilde{\mathcal{M}}(C_q^j)| > 1$. The proof of the following simple observation is omitted here; we prove a more general statement in Section 4.6.

Observation 4.3.2. *For each clause C_q, at most three of its copies are troublesome.*

We conclude that

$$|\tilde{\mathcal{M}}^C| \leq m(h+6) = 5n(h+6)/3.$$

Let $\mathcal{C}'_1 \subseteq \mathcal{C}'$ be the set of all interesting new clauses. A simple accounting shows that, if $|\tilde{\mathcal{M}}| \geq (1-\delta)(200h/3+1)n$, then $|\mathcal{C}'_1| \geq (1-\epsilon/10)hm$ must hold.

Notice that for each new clause $C_q^j \in \mathcal{C}'_1$, at least one demand pair from the set

$$\left\{(s_j(C_q, \ell_{qz}), t_j(C_q, \ell_{qz}))\right\}_{z=1}^{3}$$

is in $\tilde{\mathcal{M}}$. We select any terminal $\ell \in \left\{\ell_{q1}, \ell_{q2}, \ell_{q3}\right\}$ such that $(s_j(C_q,\ell), t_j(C_q,\ell)) \in \tilde{\mathcal{M}}$, and we say that clause C_q^j *chooses* the literal ℓ. Let x be the variable corresponding to the literal ℓ. We say that C_q^j 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 $\mathcal{A}(x) = \text{FALSE}$, and if $\ell = \neg x$, then $\mathcal{A}(x) = \text{TRUE}$. Notice that, if C_q^j is an interesting and a non-cheating clause, then the current assignment satisfies C_q^j. Therefore, in order to compete the analysis, it is enough to prove the following claim.

Claim 4.3.3. *The number of cheating clauses in $\mathcal{C}'_1$ is bounded by $\epsilon mh/2$.*

We prove a stronger claim in Section 4.6 (see Lemma 4.6.11), and provide a proof sketch here. Let $C_q^j \in \mathcal{C}'$ be a cheating clause, and suppose it has chosen the literal ℓ, whose corresponding variable is x. We say that C_q^j 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 $\mathcal{A}(x) = \text{TRUE}$ and $(s_j^T(x), t_j^T(x)) \notin \tilde{\mathcal{M}}$, or $\mathcal{A}(x) = \text{FALSE}$ and $(s_j^F(x), t_j^F(x)) \notin \tilde{\mathcal{M}}$). Otherwise, we say that C_q^j 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_q^j and $C_q^{j'}$. 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 $s_j^T(x)$ and $s_{j+1}^T(x)$, while vertex $s_{j'}(C_q, \ell)$ lies on interval $I^T(x)$, between $s_{j'}^T(x)$ and $s_{j'+1}^T(x)$. 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 $(s_j^T(x), t_j^T(x)), (s_{j+1}^T(x), t_{j+1}^T(x))$, and $(s_{j'+1}^T(x), t_{j'+1}^T(x))$ 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 $\mathcal{C}'$, 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, level-1 instances $\mathcal{I}'$, then their 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.

However, 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 $\Theta(\log n)$

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.

4.4 The Full Construction

In this section we provide a full description of the construction of the hardness instance. We start with setting the parameters.

4.4.1 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 4.2.1. 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

$$N_{i+1} = nc_{i+1}(200h/3+1)N_i$$
$$N'_{i+1} = (1-\delta)nc_{i+1}(200h/3+1)N'_i.$$

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 $H_i = 20N_i$. The following bound on H_i follows immediately from the definitions of our parameters, and we use it several times in our analysis:

$$H_i = 20N_i = 20g_iN'_i = \frac{20c_{i+1}N'_i}{10^8h^2} = \frac{2c_{i+1}\epsilon^2N'_i}{10^{13}} \tag{4.1}$$

For all $i \geq 0$, we set $L'_i = 20N_i^3$. Parameter L_i is defined as follows: $L_0 = 1$, and for $i > 0$,

$$L_i = (80h+2)c_iL_{i-1}n \leq (80h+2)c_i \cdot 20N_{i-1}^3n \leq 20N_i^3.$$

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 $i > 0$.

4.4.2 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, 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 $1 \leq j \leq c$, 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 $i \geq 0$, 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 4.5). 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 $B(\mathcal{I})$.

Next, we define smaller sub-grids of the two grids B^V and B^C. For every variable x_j of φ,

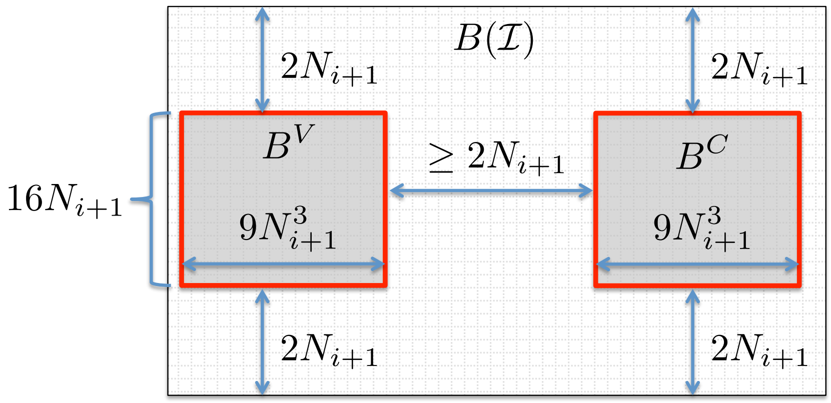


Figure 4.5: High-level view of box $B(\mathcal{I})$

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

$$n \cdot L^V + (n+1) \cdot 2N_{i+1} \leq 7nN_{i+1} + (70h+2)cL'_i n \leq 7nN_{i+1} + 1500hcN_i^3 n < 9N_{i+1}^3,$$

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 4.6).

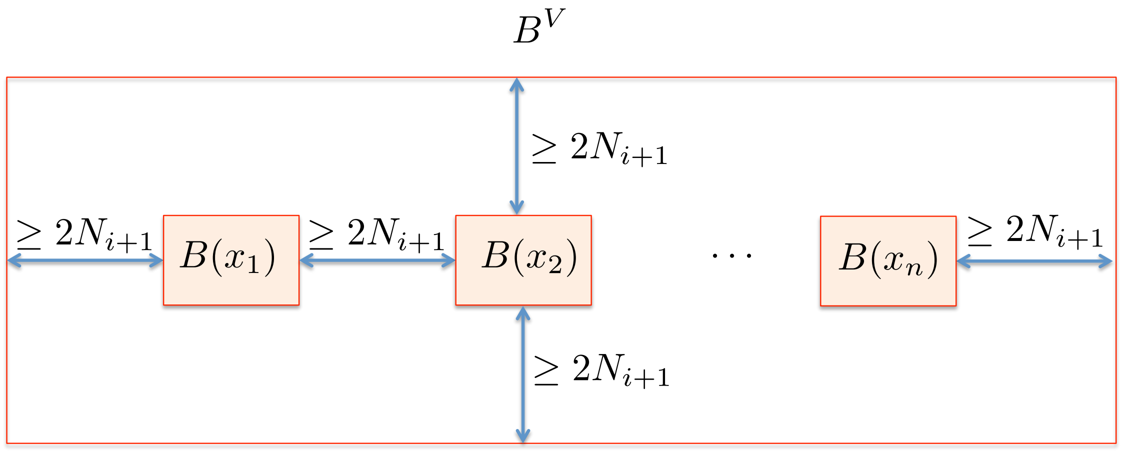


Figure 4.6: Box B^V. Each box $B(x_j)$ has length $L^V = 4N_{i+1} + (70h+2)cL'_i$ and height $H^V = H_i + 2N_{i+1}$.

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 $C_q \in \mathcal{C}$, 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

$$m \cdot L^C + (m+1) \cdot 4N_{i+1} \leq 20nN_{i+1} + 15hcL'_i n \leq 20nN_{i+1} + 300hcN_i^3 n < 9N_{i+1}^3,$$

we can find such grids (see Figure 4.7). 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.

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.

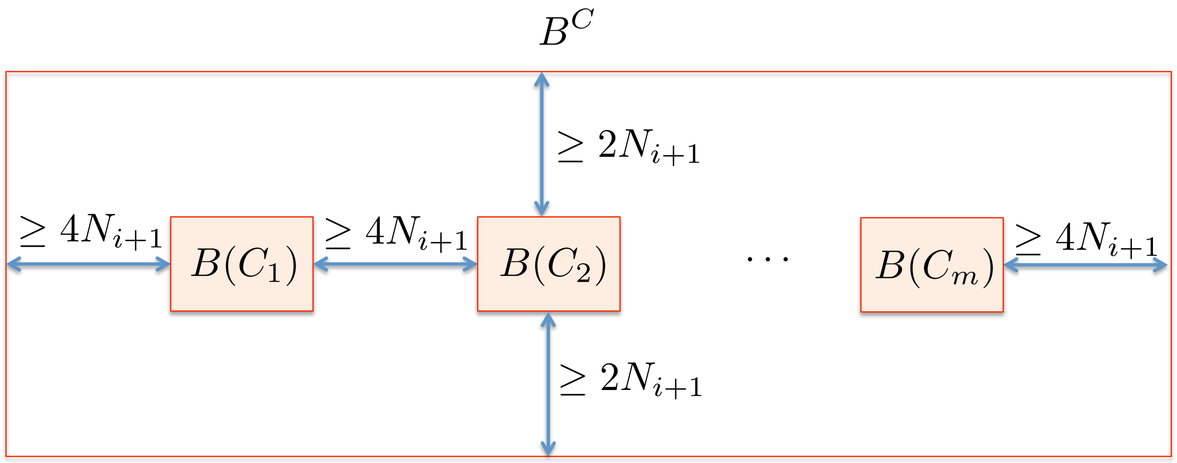


Figure 4.7: Box B^C. Each box $B(C_q)$ has length $L^C = 3chL'_i$ height $H^C = H_i$.

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 $1 \leq j \leq n$, 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 4.8).

- **(Extra Pairs).** We use $60h$ copies of c-wide level-i instances, that we denote by $\mathcal{I}^X_j(x)$, for $1 \leq j \leq 60h$. For each $1 \leq j \leq 60h$, we let $Z(\mathcal{I}^X_j(x))$ be the jth interval of $I^X(x)$. We place the corresponding boxes $B(\mathcal{I}^X_1(x)), \ldots, B(\mathcal{I}^X_{60h}(x))$ 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 $1 \leq j \leq 5h+1$. For each $1 \leq j \leq 5h+1$, 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 $1 \leq j \leq 5h+1$. For each $1 \leq j \leq 5h+1$, 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.

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

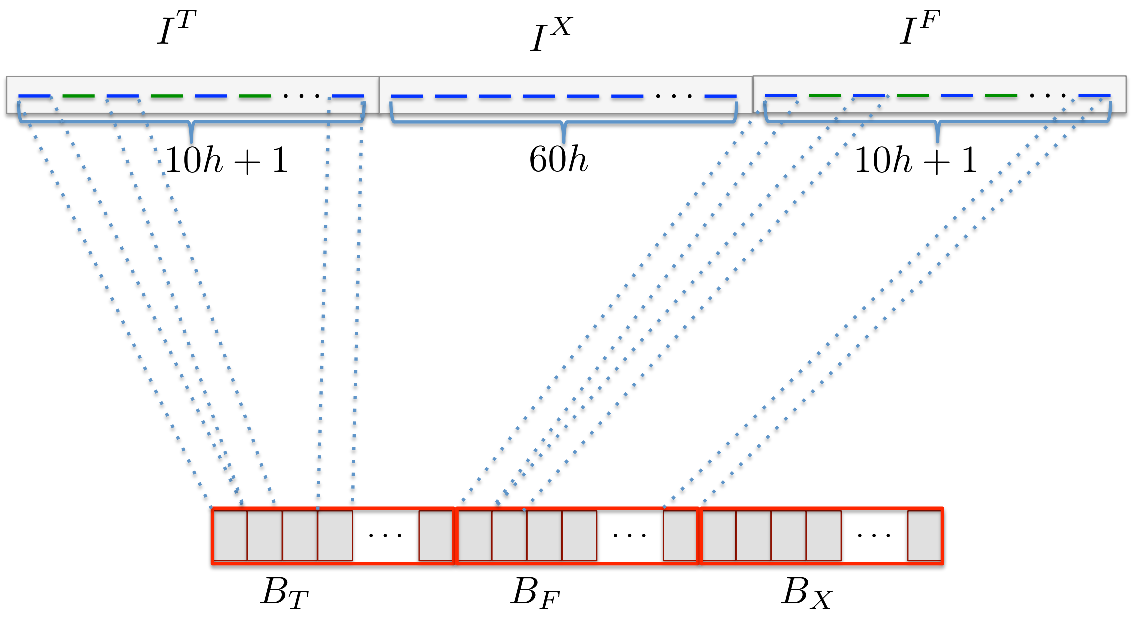


Figure 4.8: Variable gadget for level-$(i+1)$ instance. Index x is omitted for convenience. Blue sub-intervals of I^F belong to instances $\mathcal{I}_j^F$ whose destinations lie in B_F; green sub-intervals belong to instances $\mathcal{I}_j(C,x)$ associated with clauses $C \in \mathcal{C}(x)$. The $(5h)$ green intervals are partitioned into 5 groups of h consecutive intervals each, and each group belongs to a distinct clause. Blue and green intervals of I^T are dealt with similarly.

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 4.9). 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 $B(\mathcal{I})$.

Consider the set $\mathcal{C}(x) \subseteq \mathcal{C}$ of clauses in which variable x appears without negation. Assume without loss of generality that $\mathcal{C}(x) = \{C_1, \ldots, C_r\}$, where $r \leq 5$. For each $1 \leq r' \leq r$, 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 $1 \leq j \leq h$. 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 $1 \leq r' \leq r$, for each $1 \leq j \leq h$, 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

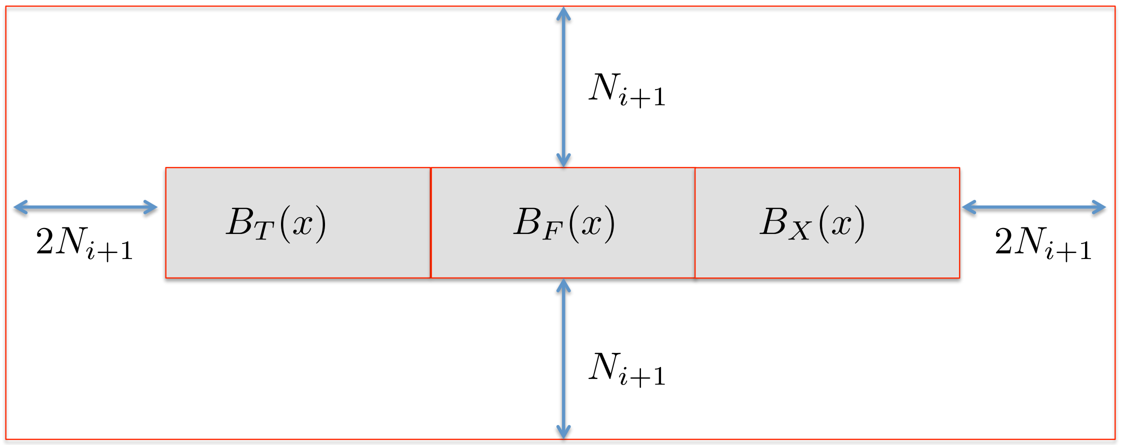


Figure 4.9: Box $B(x)$. Height: $H^V = H_i + 2N_{i+1}$, length: $L^V = 4N_{i+1} + (70h + 2)cL'_i$.

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 $C_{r'} \in \mathcal{C}(x)$.

We treat the subset $\mathcal{C}'(x) \subseteq \mathcal{C}$ 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 $C_q = (\ell_{q1} \vee \ell_{q2} \vee \ell_{q3})$. For each one of the three literals of C_q, we construct h level-i width-c instances, with instances $\{\mathcal{I}_j(C_q, \ell_{qz})\}_{j=1}^h$ representing the literal ℓ_{qz}, 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 $B^1(C_q), \ldots, B^h(C_q)$, each of which has height H_i and length $3cL'_i$. For each $1 \le j \le h$, we place the boxes $B(\mathcal{I}_j(C_q, \ell_{q1})), B(\mathcal{I}_j(C_q, \ell_{q2})), B(\mathcal{I}_j(C_q, \ell_{q3}))$ inside $B^j(C_q)$ side-by-side in this order (see Figure 4.10).

The intervals $Z(\mathcal{I}_j(C_q, \ell_{qz}))$ 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 $\mathcal{M}(C_q, \ell_{qz})$ the set of all demand pairs that belong to instances $\mathcal{I}_j(C_q, \ell_{qz})$, for $1 \le j \le h$,

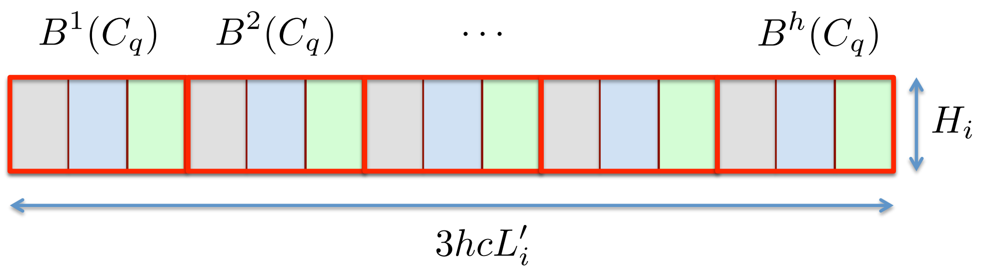


Figure 4.10: Box $B(C_q)$. Different colors show boxes that represent the three different literals.

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.

4.5 Yes-Instance Analysis

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

Theorem 4.5.1. *Assume that the input 3SAT(5) formula φ is a* Yes-Instance. *Then for all $i \geq 0$, 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 $i \geq 0$, 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 $1 \leq j \leq c_{i+1}$, the demand pairs in $\mathcal{M}^*(\mathcal{I}_j)$ are routed via paths that respects 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 some variable, and let

$$\hat{\mathcal{M}}^X(x) = \bigcup_{j=1}^{60h} \mathcal{M}^*(\mathcal{I}_j^X(x))$$

be 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

$$\hat{\mathcal{M}}^T(x) = \bigcup_{j=1}^{5h+1} \mathcal{M}^*(\mathcal{I}_j^T(x))$$
$$\hat{\mathcal{M}}^F(x) = \emptyset.$$

Notice that $|\hat{\mathcal{M}}^T(x)| = (5h+1)c_{i+1}N_i$ in this case. Otherwise, we let

$$\hat{\mathcal{M}}^F(x) = \bigcup_{j=1}^{5h+1} \mathcal{M}^*(\mathcal{I}_j^F(x))$$
$$\hat{\mathcal{M}}^T(x) = \emptyset,$$

so that $|\hat{\mathcal{M}}^F(x)| = (5h+1)c_{i+1}N_i$.

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 $1 \leq j \leq h$, 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 $1 \leq q \leq m$, $|\hat{\mathcal{M}}(C_q)| = hc_{i+1}N_i$, 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

$$|\hat{\mathcal{M}}| = nc_{i+1} \cdot (65h+1)N_i + 5nc_{i+1}hN_i/3 = n \cdot N_i c_{i+1}(200h/3+1) = N_{i+1}.$$

It is now enough to prove the following lemma.

Lemma 4.5.2. *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 $B(\mathcal{I})$.*

For convenience, we denote the first row of the grid G_{i+1} by R. Let $S' \subseteq S(\hat{\mathcal{M}})$ be any subset of the source vertices of the demand pairs in $\hat{\mathcal{M}}$. We say that the sources of S' appear *consecutively* on R, iff there is a sub-path P of R that contains all the vertices of S' and does not contain any vertex of $S(\hat{\mathcal{M}}) \setminus S'$. We let $\mathcal{O}$ be the left-to-right ordering of the vertices of $S(\hat{\mathcal{M}})$ on row R.

Consider some variable x, and denote by $\hat{\mathcal{M}}'(x)$ the set of all demand pairs in $\hat{\mathcal{M}}^C$ whose sources lie on the interval $I(x)$ — these are the demand pairs representing the clauses C_q that chose either x or $\neg x$ as their literal. Assume first that x is assigned the value True. Then all sources of the demand pairs in $\hat{\mathcal{M}}(x)$ lie on the intervals $I^X(x)$ and $I^T(x)$, while all sources in set $S(\hat{\mathcal{M}}'(x))$ lie on $I^F(x)$: indeed, since x is assigned the value True, the corresponding clauses must contain x without negation and so their sources lie on $I^F(x)$. Therefore, the sources of each set $\hat{\mathcal{M}}(x)$ and $\hat{\mathcal{M}}'(x)$ are consecutive on R. If x is assigned the value False, then similarly all sources of the demand pairs in $\hat{\mathcal{M}}(x)$ are consecutive on R, while all sources of the demand pairs in $\hat{\mathcal{M}}'(x)$ appear on $I^T(x)$ and are therefore also consecutive on R. In either case, for every clause $C_q \in \mathcal{C}$, the vertices of $S(\hat{\mathcal{M}}(C_q))$ are consecutive on R.

In order to construct the routing, it is convenient to use special subgraphs of G that we call snakes, in which routing can be done easily. We start by defining a corridor, and then combine several corridors to define a snake.

Recall that our graph G is a subgraph of a grid G_{i+1}. Recall also that, given a set $\mathcal{R}$ of consecutive rows of G_{i+1} and a set $\mathcal{W}$ of consecutive columns of G_{i+1}, we denoted by $\Upsilon(\mathcal{R}, \mathcal{W})$ the subgraph of G_{i+1} 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 a *corridor* iff every vertex of Υ belongs to G. For convenience, we will say that Υ is a corridor spanned by the rows in $\mathcal{R}$ and the columns of $\mathcal{W}$. Let R' and R'' be the first and the last row of $\mathcal{R}$ respectively, and let W' and W'' be the first and the last column of $\mathcal{W}$ respectively. Each of the four paths $\Upsilon \cap R', \Upsilon \cap R'', \Upsilon \cap W'$ and $\Upsilon \cap W''$ is called a *boundary edge* of Υ, and their union is called the *boundary* of Υ. We say that two corridors Υ, Υ' are *internally disjoint*, iff every vertex $v \in \Upsilon \cap \Upsilon'$ belongs to the boundaries of both corridors. We say that two internally disjoint corridors Υ, Υ' are *neighbors* iff $\Upsilon \cap \Upsilon' \neq \emptyset$.

We are now ready to define snakes. A snake $\mathcal{Y}$ of length ℓ is a sequence $\Upsilon_1, \Upsilon_2, \ldots, \Upsilon_\ell$ of ℓ corridors that are pairwise internally disjoint. Moreover, for all $1 \leq \ell', \ell'' < \ell$, $\Upsilon_{\ell'}$ is a neighbor of $\Upsilon_{\ell''}$ iff $|\ell' - \ell''| = 1$. We say that the width of the snake is w iff for each $1 \leq \ell' \leq \ell$, $\Upsilon_{\ell'}$ is spanned by a set of at least w rows and by a set of at least w columns, and for all $1 \leq \ell' < \ell$, $\Upsilon_{\ell'} \cap \Upsilon_{\ell'+1}$ contains at least w vertices. We use the following simple claim for routing in snakes.

Claim 4.5.3. *Let $\mathcal{Y} = (\Upsilon_1, \ldots, \Upsilon_\ell)$ be a snake of width w, and let A, A' be two sets of vertices with $|A| = |A'| \leq w - 2$, such that the vertices of A lie on a single boundary edge of Υ_1 and the vertices of A' lie on a single boundary edge of Υ_ℓ. Then there is a set $\mathcal{Q}$ of node-disjoint paths contained in $\bigcup_{\ell'=1}^{\ell} \Upsilon_{\ell'}$, that connect every vertex of A to a distinct vertex of A'.*

Proof. The proof is by induction on ℓ. For the base case, assume that $\ell = 1$, and let A, A' any pair of vertex sets (that are not necessarily disjoint), such that the vertices of each set lie on a single boundary edge of the corridor $\Upsilon = \Upsilon_1$, and $|A| = |A'| = w' \leq w - 2$. We show that there is a set $\mathcal{Q}$ of node-disjoint paths in Υ, connecting every vertex of A to a vertex of A', with a slightly stronger property: namely, the paths in $\mathcal{Q}$ are internally disjoint from the boundary of Υ. Let Υ' be the graph obtained from Υ, by deleting all vertices lying on the boundary of Υ, except for the vertices of $A \cup A'$. It is enough to show that there is a set $\mathcal{Q}$ of w' node-disjoint paths in Υ', connecting vertices of A to vertices of A'. Assume for contradiction that such a set of paths does not exist. Then from Menger's theorem, there is a set J of $w' - 1$ vertices, such that $\Upsilon' \setminus J$ has no path connecting a vertex of A to the vertex of A'. We consider three cases.

The first case is when A and A' lie on the opposite boundary edges of Υ. Assume without loss of generality that the vertices of A lie on the top boundary edge, and the vertices of A' on the bottom boundary edge of Υ. Let $\mathcal{R}$ and $\mathcal{W}$ be the sets of rows and columns of G_{i+1}, respectively, that span Υ. Let $\mathcal{W}', \mathcal{W}'' \subseteq \mathcal{W}$ be the sets of columns of Υ where the vertices of A and A' lie, respectively, so $|\mathcal{W}'| = |\mathcal{W}''| = w'$. Since $|J| < w'$, there is a column $W' \in \mathcal{W}'$ and a column $W'' \in \mathcal{W}'$, with $W' \cap J, W'' \cap J = \emptyset$. There is also some row $R' \in \mathcal{R}$ of Υ', that is not its top or bottom row, such that $R' \cap J = \emptyset$. But $W' \cup W'' \cup R'$ is a connected subgraph of $\Upsilon' \setminus J$, that contains a vertex of A and a vertex of A', a contradiction.

The other two cases, when A and A' lie on adjacent boundary edges of Υ, and when A and A' lie on the same boundary edge of Υ are analyzed similarly.

Assume now that the claim holds for some value $\ell \geq 0$. We now prove it for $\ell + 1$. Denote $|A| = |A'| = w'$, and let U be any set of w' vertices in $\Upsilon_\ell \cap \Upsilon_{\ell+1}$, so the vertices of U lie on the boundaries of both corridors. Notice that the vertices of U must belong to a single boundary edge of Υ_ℓ, and a single boundary edge of $\Upsilon_{\ell+1}$. Using the induction hypothesis, there is a set $\mathcal{P}_1$ of w' node-disjoint paths in $\Upsilon_1 \cup \cdots \cup \Upsilon_\ell$, connecting every vertex of A to a distinct vertex of U. From our analysis of the base case, there is a set $\mathcal{P}_2$ of w' node-disjoint paths in $\Upsilon_{\ell+1}$, connecting every vertex of U to a distinct vertex of A'. Moreover, the paths in $\mathcal{P}_2$ are internally disjoint from the boundary of $\Upsilon_{\ell+1}$. We obtain the desired set of paths by concatenating the paths in $\mathcal{P}_1$ and the paths in $\mathcal{P}_2$.

□

Our routing consists of two steps. In the first step, we connect each source vertex $s \in S(\hat{\mathcal{M}})$ to the top boundary of the unique box in $\mathcal{B} = \{B(x_1), \ldots, B(x_n), B(C_1) \ldots, B(C_m)\}$, that contains its corresponding destination vertex. We will later select specific vertices on the top boundary of each such box to which the sources are routed. The resulting paths will be internally disjoint from the boxes in $\mathcal{B}$. In the second step, we complete the routing inside each box. The first step is summarized in the following claim.

Claim 4.5.4. *Suppose we are given, for every variable x_j, a set $A(x_j)$ of $|\hat{\mathcal{M}}(x_j)|$ vertices on the top boundary of $B(x_j)$, and for every clause $C_q \in \mathcal{C}$, a set $A(C_q)$ of $|\hat{\mathcal{M}}(C_q)|$ vertices on the top boundary of $B(C_q)$. Then there is a collection $\mathcal{P}'$ of node-disjoint paths in G with the following properties:*

- *the paths of $\mathcal{P}'$ are internally disjoint from all boxes in*

$$\mathcal{B} = \{B(x_1), \ldots, B(x_n), B(C_1) \ldots, B(C_m)\};$$

- *for every variable x_j of φ, there is a subset $\mathcal{P}(x_j) \subseteq \mathcal{P}'$ of paths, that connect every vertex in $S(\hat{\mathcal{M}}(x_j))$ to a vertex of $A(x_j)$, so that the paths in $\mathcal{P}(x_j)$ are order-preserving; and*

- *for every clause $C_q \in \mathcal{C}$, there is a subset $\mathcal{P}(C_q) \subseteq \mathcal{P}'$ of paths, connecting every vertex in $S(\hat{\mathcal{M}}(C_q))$ to a vertex of $A(C_q)$, so that the paths in $\mathcal{P}(C_q)$ are order-preserving.*

Proof. Consider two consecutive variables x_j, x_{j+1}, for $1 \leq j < n$. Recall that the vertices $S(\hat{\mathcal{M}}(x_j))$ appear consecutively on R, and so do the vertices of $S(\hat{\mathcal{M}}(x_{j+1}))$. Let $V_j \subseteq S(\hat{\mathcal{M}}^C)$ be the set of all source vertices that correspond to clause-pairs, and lie between the vertices of $S(\hat{\mathcal{M}}(x_j))$ and the vertices of $S(\hat{\mathcal{M}}(x_{j+1}))$ on R. Notice that the vertices of V_j may only correspond to clauses in which x_j or x_{j+1} participate, so

$$|V_j| \leq 10hN_i c_{i+1} < N_{i+1}.$$

Similarly, we let $V_0, V_n \subseteq S(\hat{\mathcal{M}}^C)$ be the sets of all source vertices that correspond to clause-pairs and lie before the vertices of $S(\hat{\mathcal{M}}(x_1))$ and after the vertices of $S(\hat{\mathcal{M}}(x_n))$ on R, respectively. We still have $|V_0|, |V_n| \leq N_{i+1}$. For all $0 \leq j \leq n$, let $M'_j = |V_j|$. For all $1 \leq j \leq n$, let

$$M_j = |\hat{\mathcal{M}}(x_j)| = c_{i+1}N_i(65h+1) < N_{i+1}.$$

We now select a set Γ of vertices on the top row of B^V, as follows. First, for every variable x_j of φ, for every vertex $a \in A(x_j)$, we select a vertex a' on the top row of B^V, lying in the same column as a, and we let P_a be the sub-path of the column $\mathrm{col}(a)$ between a and a'. Let Γ_j denote the resulting set of vertices that we have selected for x_j.

For $1 \leq j < n$, let $\mathcal{W}_j$ be the set of columns of B^V that lie between the boxes $B(x_j)$ and $B(x_{j+1})$. We also let $\mathcal{W}_0$ be the set of all columns of B^V that lie before $B(x_1)$, and $\mathcal{W}_n$ the set of all columns lying after $B(x_n)$. For all $0 \leq j \leq n$, we select an arbitrary set Γ'_j of M'_j distinct vertices on the top row of B^V that belong to the columns of $\mathcal{W}_j$. Since $|\mathcal{W}_j| \geq N_{i+1}$, while $M'_j \leq N_{i+1}$, such a set of vertices exists. For each selected vertex $v \in \Gamma'_j$, we let P_v be the column of B^V in which v lies, and we let v' be the other endpoint of the column.

Let $\Gamma''_j = \left\{v' \mid v \in \Gamma'_j\right\}$. We denote $\Gamma = \left(\bigcup_{j=1}^n \Gamma_j\right) \cup \left(\bigcup_{j=0}^n \Gamma'_j\right)$, and $\Gamma'' = \bigcup_{j=0}^n \Gamma''_j$. Finally, we let Γ''' be the set of $|\hat{\mathcal{M}}^C|$ consecutive left-most vertices on the top boundary of B^C.

We construct five sets of node-disjoint paths, $\mathcal{P}_0, \ldots, \mathcal{P}_4$. The final set $\mathcal{P}'$ of paths will be obtained by combining these sets of paths.

Set $\mathcal{P}_0$: Let Z' be the set of $|\hat{\mathcal{M}}|$ leftmost vertices on the opening of $B(\mathcal{I})$. Set $\mathcal{P}_0$ consists of $|\hat{\mathcal{M}}|$ node-disjoint paths, connecting every vertex of $S(\hat{\mathcal{M}})$ to a vertex of Z', so that the paths in $\mathcal{P}_0$ are order-preserving and internally disjoint from $B(\mathcal{I})$. Since the distance between $B(\mathcal{I})$ and the boundaries of G_{i+1} is at least $H_{i+1} \geq N_{i+1}$, while $|\hat{\mathcal{M}}| = N_{i+1}$, it is easy to verify that such a set of paths exists.

Set $\mathcal{P}_1$: This set of paths connects every vertex of Z' to a distinct vertex of Γ in a node-disjoint and order-preserving manner. In order to construct it, we use a snake $\mathcal{Y}^1$, consisting of two corridors, Υ^1_1 and Υ^1_2. The first corridor Υ^1_1 is simply the set of the top $N_{i+1} + 2$ rows of $B(\mathcal{I})$. In order to construct the second corridor, Υ^1_2, we denote by $\mathcal{W}^V$ the set of all columns of G_{i+1} that intersect the box B^V, and we denote by $\hat{R}$ the row of G_{i+1} that contains the topmost row of B^V. Let $\hat{\mathcal{R}}$ be a consecutive set of rows of $B(\mathcal{I})$, starting from row $R_{N_{i+1}+2}$ and ending at row $\hat{R}$. Let Υ^1_2 be the corridor spanned by the set $\mathcal{W}^V$ of columns and the set $\hat{\mathcal{R}}$ of rows. By combining the two corridors, we we obtain a snake $\mathcal{Y}^1$ of width at least $N_{i+1} + 2$. From Claim 4.5.3, there is a set $\mathcal{P}_1$ of node-disjoint paths, connecting every vertex of Z' to a distinct vertex of Γ inside the snake. It is immediate to verify that the set $\mathcal{P}_1$ of paths must be order-preserving.

Set $\mathcal{P}_2$: Set $\mathcal{P}_2$ contains, for every vertex $v \in \Gamma$, the path P_v that we have defined above. Note that for each $1 \leq j \leq n$, paths in $\mathcal{P}_2$ connect the vertices of Γ_j to the vertices of $A(x_j)$, lying on the top row of $B(x_j)$, and for each $0 \leq j \leq n$, paths in $\mathcal{P}_2$ connect the vertices of Γ'_j to the vertices of Γ''_j, lying on the bottom row of B^V.

Set $\mathcal{P}_3$: This set contains $|\hat{\mathcal{M}}^C|$ node-disjoint paths, connecting every vertex of Γ'' to a distinct vertex of Γ'''. The paths in $\mathcal{P}_3$ will be internally disjoint from B^V and B^C, and

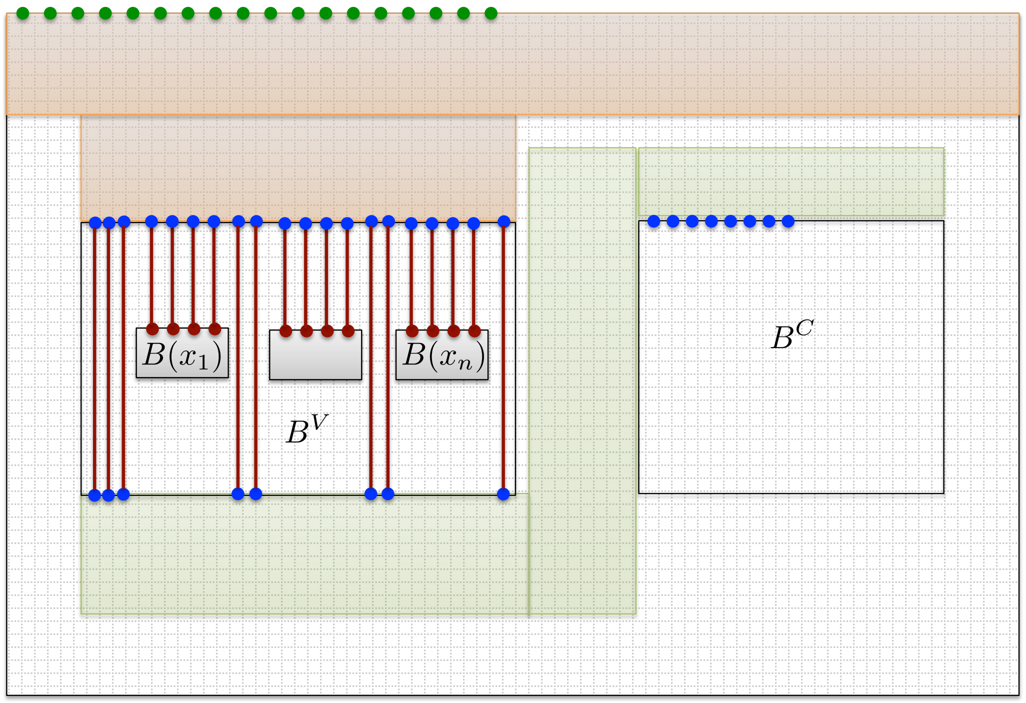


Figure 4.11: Routing the sets $\mathcal{P}_1, \mathcal{P}_2$ and $\mathcal{P}_3$ of paths inside $B(\mathcal{I})$. The paths in $\mathcal{P}_2$ are shown in red; the paths of $\mathcal{P}_1$ are routed inside the orange snake, and the paths of $\mathcal{P}_3$ are routed inside the green snake.

order-preserving. In order to construct the set $\mathcal{P}_3$ of paths, we construct a snake $\mathcal{Y}^2$. Let $\mathcal{W}^V$ and $\mathcal{W}^C$ be the sets of columns of G_{i+1} that intersect B^V and B^C, respectively, and let $\mathcal{W}^M$ be the set of columns lying between these two sets. Let R' be the row of G_{i+1} containing the bottommost row of B^V, and let $\mathcal{R}^V$ be the set of $N_{i+1} - 2$ consecutive rows of G_{i+1} lying below R', including R'. Let R'' be the row of G_{i+1} containing the topmost row of B^C, and let $\mathcal{R}^C$ be the set of $N_{i+1} - 2$ consecutive rows of G_{i+1} lying above R^C, including R^C. Finally, let $\mathcal{R}^M$ be the set of rows lying between the top row of $\mathcal{R}^C$ and the bottom row of $\mathcal{R}^V$, including these two rows. The snake $\mathcal{Y}^2$ consists of three corridors (see Figure 4.11).

The first corridor, Υ_1^2, is spanned by the set $\mathcal{R}^V$ of rows and the set $\mathcal{W}^V$ of columns. The second corridor, Υ_2^2, is spanned by the set $\mathcal{R}^M$ of rows and the set $\mathcal{W}^M$ of columns. The third and the final corridor, Υ_3^2, is spanned by the set $\mathcal{R}^C$ of rows and the set $\mathcal{W}^C$ of columns.

We extend corridor Υ_1^2 by one column to the right, so that it intersects the boundary of Υ_2^2, and similarly we extend corridor Υ_3^2 by one column to the left. It is now easy to verify that we obtain a snake of width at least $N_{i+1} - 2$, and so from Claim 4.5.3, there is a set $\mathcal{P}_3$ of node-disjoint paths, connecting every vertex of Γ'' to a distinct vertex of Γ'''. It is immediate to verify that the paths in $\mathcal{P}_3$ are order-preserving.

By combining the paths of $\mathcal{P}_0, \ldots, \mathcal{P}_3$, we obtain a set $\mathcal{P}''$ of node-disjoint paths, that are internally disjoint from B^C. For every variable x_j of φ, set $\mathcal{P}''$ contains a set $\mathcal{P}(x_j)$ of paths connecting every vertex of $S(\hat{\mathcal{M}}(x_j))$ to a vertex of $A(x_j)$, so that the paths in $\mathcal{P}(x_j)$ are order-preserving. For every clause $C_q \in \mathcal{C}$, set $\mathcal{P}''$ contains a set $\mathcal{P}'(C_q)$ of node-disjoint order-preserving paths, connecting every vertex of $S(\hat{\mathcal{M}}(C_q))$ to a distinct vertex of Γ'''. We denote by $\Gamma(C_q) \subseteq \Gamma'''$ the set of vertices that serve as endpoints of the paths in $\mathcal{P}'(C_q)$. Note that for each clause $C_q \in \mathcal{C}$, the vertices of $\Gamma(C_q)$ appear consecutively in Γ'''. We define an ordering $\mathcal{O}'$ of the clauses in $\mathcal{C}$ as follows: clause C_q appears before clause $C_{q'}$ in this ordering iff the vertices of $\Gamma(C_q)$ appear to the left of the vertices of $\Gamma(C_{q'})$ on the top row of B^C.

Set $\mathcal{P}_4$: In our final step, we construct, for each clause $C_q \in \mathcal{C}$, a set $\mathcal{P}''(C_q)$ of paths, connecting the vertices of $\Gamma(C_q)$ to the vertices of $A(C_q)$, so that the paths in $\mathcal{P}''(C_q)$ are order-preserving and the paths in $\bigcup_{C_q \in \mathcal{C}} \mathcal{P}''(C_q)$ are node-disjoint. We then let $\mathcal{P}_4$ be the union of these sets of paths.

Claim 4.5.5. *For each clause $C_q \in \mathcal{C}$, there is a set $\mathcal{P}''(C_q) \subseteq B^C$ of paths, connecting every vertex of $\Gamma(C_q)$ to a distinct vertex of $A(C_q)$, such that the paths in set $\mathcal{P}_4 = \bigcup_{C_q \in \mathcal{C}} \mathcal{P}''(C_q)$ are mutually node-disjoint and internally disjoint from the boxes $\{B(C_1), \ldots, B(C_m)\}$. Moreover, the paths in $\mathcal{P}_4$ are internally disjoint from the top boundary of B^C, and for each $1 \leq q \leq m$, the paths in $\mathcal{P}(C_q)$ are order-preserving.*

We obtain the final set $\mathcal{P}'$ of paths by combining the paths of $\mathcal{P}_0, \ldots, \mathcal{P}_4$. In order to complete the proof of Claim 4.5.4, it now remains to prove Claim 4.5.5.

Proof. Consider some clause $C_q \in \mathcal{C}$, and let $\mathcal{W}(C_q)$ be the set of columns of G_{i+1} that intersect $B(C_q)$. We let $\mathcal{W}^L(C_q)$ be the set of N_{i+1} columns lying immediately to the left of $\mathcal{W}(C_q)$, and similarly we let $\mathcal{W}^R(C_q)$ be the set of N_{i+1} columns lying immediately to the right of C_q. Let $\mathcal{R}$ be the set of all rows of G_{i+1} that intersect the box $B(C_q)$ (this set is the same for all clauses C_q). We let $B'(C_q)$ be the sub-grid of G_{i+1} spanned by the set $\mathcal{W}^L(C_q) \cup \mathcal{W}(C_q) \cup \mathcal{W}^R(C_q)$ of columns and the set $\mathcal{R}$ of rows.

The idea is to define a snake-like routing, where the paths visit the boxes $B'(C_1), \ldots, B'(C_m)$ in turn. For each clause C_q, only the paths corresponding to the clauses $C_q, C_{q+1}, \ldots, C_m$ will visit the box $B'(C_q)$. The paths corresponding to clause C_q will terminate on the top boundary of $B(C_q)$. For each $q+1 \leq q' \leq m$, if $C_{q'}$ appears before C_q in the ordering $\mathcal{O}'$, then the paths corresponding to $C_{q'}$ will traverse the box $B'(C_q)$ to the left of $B(C_q)$; otherwise, they will traverse the box $B'(C_q)$ to the right of $B(C_q)$. In order to implement this, we select a subset of vertices on the top and the bottom boundaries of $B'(C_q)$, through which the paths will enter and leave the box (see Figure 4.12).

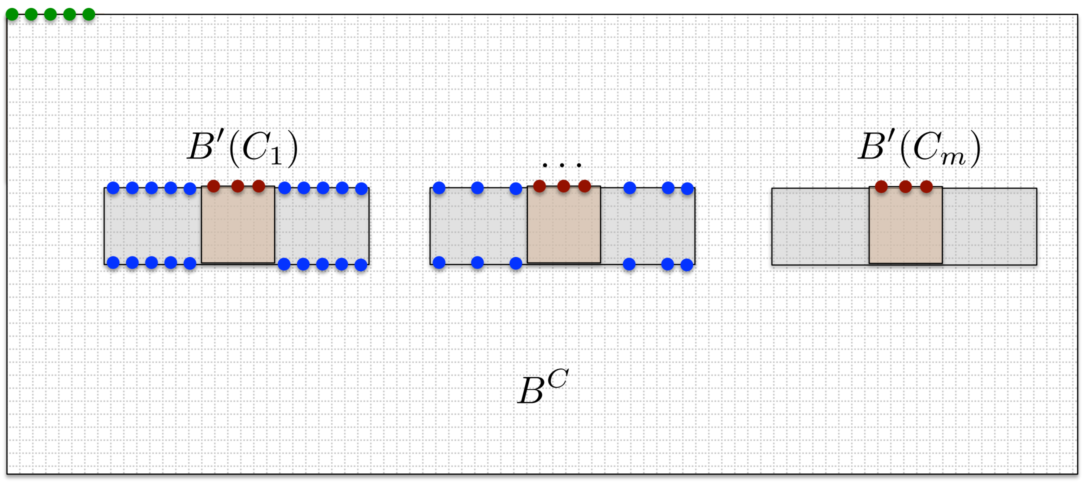


Figure 4.12: Selecting the sets $\Gamma(C_q)$ and $\Gamma'(C_q)$ of vertices for the clauses C_q. The vertices of Γ''' are shown in green.

Fix some clause C_q, for $1 \leq q \leq m$. Let n_q^L be the number of clauses $C_{q'}$, with $q < q' \leq m$, such that $C_{q'}$ appears before C_q in the ordering $\mathcal{O}'$, and let n_q^R be the number of such clauses with $C_{q'}$ appearing after C_q in $\mathcal{O}'$. Recall that for each clause $C_{q'}$, $|\hat{\mathcal{M}}(C'_q)| = hc_{i+1}N_i$, and $mhc_{i+1}N_i < N_{i+1} - 2$. We select an arbitrary set $\Gamma^L(C_q)$ of $n_q^L \cdot hc_{i+1}N_i$ vertices on the top boundary of $B'(C_q)$, strictly to the left of $B(C_q)$, and similarly, we select an arbitrary set $\Gamma^R(C_q)$ of $n_q^R \cdot hc_{i+1}N_i$ vertices on the top boundary of $B'(C_q)$, strictly to the right of $B(C_q)$. For each vertex $v \in \Gamma^L(C_q) \cup \Gamma^R(C_q)$, we let P_v be the column of $B'(C_q)$ that contains v, and we let v' be its other endpoint. For each vertex $v \in A(C_q)$, we let P_v be a path consisting of a single vertex v.

We define $\Gamma(C_q) = \Gamma^L(C_q) \cup A(C_q) \cup \Gamma^R(C_q)$, a set of vertices on the top boundary of $B'(C_q)$. We define $\Gamma'(C_q) = \left\{v' \mid v \in \Gamma^L(C_q) \cup \Gamma^R(C_q)\right\}$, a set of vertices on the bottom boundary of $B'(C_q)$. Finally, we let $\tilde{\mathcal{P}}'_q = \left\{P_v \mid v \in \Gamma(C_q)\right\}$. It is easy to verify that for all $1 \le q < m$, $|\Gamma'(C_q)| = |\Gamma(C_{q+1})|$.

In order to compute the routing, we use the following simple observation.

Observation 4.5.6. *For all $1 \le q < m$, there is a set $\tilde{\mathcal{P}}_q \subseteq B^C$ of paths, such that:*

- *Paths in $\tilde{\mathcal{P}}_1$ connect every vertex in Γ''' to a distinct vertex of $\Gamma(C_1)$ and are order-preserving;*
- *For $1 < q < m$, paths in $\tilde{\mathcal{P}}_q$ connect every vertex in $\Gamma'(C_q)$ to a distinct vertex of $\Gamma(C_{q+1})$ and are order-preserving;*
- *The paths in $\bigcup_q \tilde{\mathcal{P}}_q$ are mutually node-disjoint, and they are internally disjoint from all boxes in $\left\{B'(C_1), \ldots, B'(C_m)\right\}$.*

Notice that combining the paths in sets $\bigcup_{q=1}^{m-1} \tilde{\mathcal{P}}_q$ and $\bigcup_{q=1}^{m} \tilde{\mathcal{P}}'_q$ finishes the proof of Claim 4.5.5 and Claim 4.5.4. We now prove the observation.

Proof. We construct, for each $1 \le q \le m$, a snake $\mathcal{Y}(C_q)$, and route the set $\tilde{\mathcal{P}}_q$ of paths inside it (see Figure 4.13 for an illustration). Since $|\hat{\mathcal{M}}^C| < N_{i+1} - 2$, it is enough to ensure that each snake has width at least N_{i+1}.

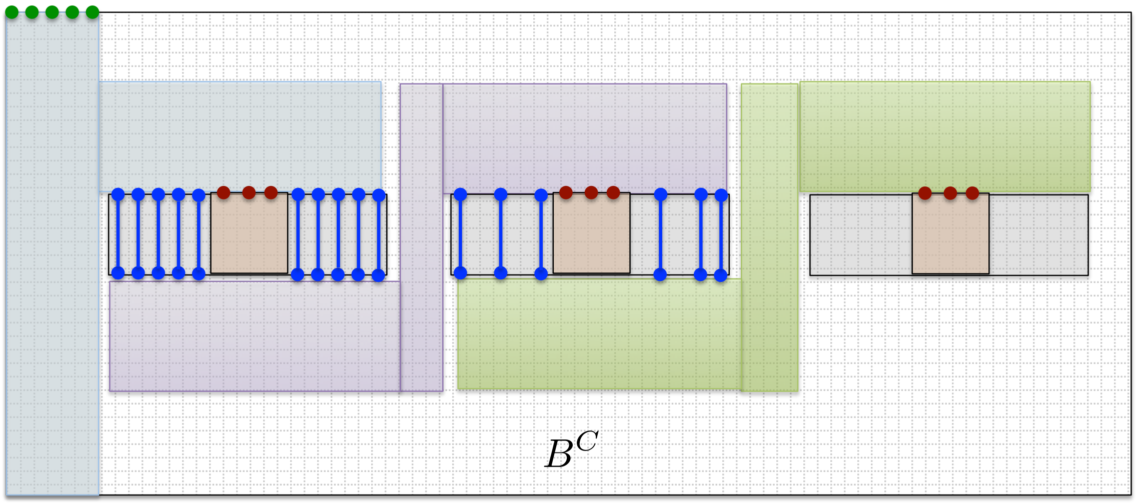


Figure 4.13: The snakes used to route the sets $\tilde{\mathcal{P}}_q$ of paths.

Recall that $\mathcal{R}$ is the set of all rows of G_{i+1} that intersect the boxes $B(C_q)$. Let R' and R'' be the topmost and the bottommost rows of $\mathcal{R}$, respectively. Let $\mathcal{R}'$ be the set of N_{i+1} consecutive rows lying above R' including R', and let $\mathcal{R}''$ be the set of N_{i+1} consecutive rows lying below R'', including R''.

The first snake, $\mathcal{Y}(C_1)$ consists of two corridors. The first corridor, $\Upsilon_1(C_1)$, is spanned by the first N_{i+1} columns of B^C. For the second corridor, $\Upsilon_2(C_1)$, we let W be the last column of $\Upsilon_1(C_1)$, W' the last column of $\mathcal{W}^R(C_1)$, and $\mathcal{W}'$ the set of all columns lying between W and W', including these two columns. We then let $\Upsilon_2(C_1)$ be the corridor spanned by the columns of $\mathcal{W}'$ and the rows of $\mathcal{R}'$.

For each $1 < q \leq m$, we construct a snake $\mathcal{Y}(C_q)$, that consists of three corridors. The first corridor, $\Upsilon_1(C_q)$ is spanned by the rows of $\mathcal{R}''$ and the columns of $\mathcal{W}^L(C_{q-1}) \cup \mathcal{W}(C_{q-1}) \cup \mathcal{W}^R(C_{q-1})$. The third corridor, $\Upsilon_3(C_q)$ is spanned by the rows of $\mathcal{R}'$ and the columns of $\mathcal{W}^L(C_q) \cup \mathcal{W}(C_q) \cup \mathcal{W}^R(C_q)$. For the second corridor, $\Upsilon_2(C_q)$, we let $\mathcal{R}^*$ be the set of consecutive rows starting from the top row of $\mathcal{R}'$ and terminating at the bottom row of $\mathcal{R}''$, including these two rows, and we let $\mathcal{W}'(C_q)$ be the set of all columns lying between

$B'(C_{q-1})$ and $B'(C_q)$. We then let $\Upsilon_2(C_q)$ be the corridor spanned by the rows in $\mathcal{R}^*$ and the columns in $\mathcal{W}'(C_q)$. In order to obtain a valid snake, we extend $\Upsilon_1(C_q)$ by one column to the right and $\Upsilon_3(C_q)$ by one column to the left (see Figure 4.13).

It is immediate to verify that for each $1 \leq q \leq m$, we obtain a valid snake $\mathcal{Y}(C_q)$ of width at least N_{i+1}; the resulting snakes are mutually disjoint from each other, and each snake $\mathcal{Y}(C_q)$ intersects the boxes in $\{B(C_1), \ldots, B(C_m)\}$ only on their boundaries. For each $1 \leq q < m$, we can now find the desired set $\tilde{\mathcal{P}}(C_q)$ of paths inside the snake $\mathcal{Y}(C_q)$, using Claim 4.5.3. It is immediate to verify that the paths are order-preserving. $\square$

$\square$

$\square$

We now define the sets $\left\{A(x_j) \mid 1 \leq j \leq n\right\} \cup \left\{A(C_q) \mid C_q \in \mathcal{C}\right\}$ of vertices and complete the routing.

Consider first some clause $C_q \in \mathcal{C}$, and assume that it has chosen the literal ℓ_q. Fix some index $1 \leq j \leq h$, and consider the width-c_{i+1} level-i instance $\mathcal{I}' = \mathcal{I}_j(C_q, \ell_q)$. Instance $\mathcal{I}'$ consists of c_{i+1} level-i instances, that we denote by $\mathcal{I}'_1, \ldots, \mathcal{I}'_{c_{i+1}}$. Then set $\hat{\mathcal{M}}(C_q)$ contains $c_{i+1}N_i$ demand pairs that belong to instance $\mathcal{I}'$, with exactly N_i pairs from each instance $\mathcal{I}'_r$.

For each such instance $\mathcal{I}'_r$, we select N_i leftmost vertices on the opening of the box $B(\mathcal{I}'_r)$. We then let $A(C_q)$ be the set of all such vertices we have selected, for all $1 \leq j \leq h$ and $1 \leq r \leq c_{i+1}$, so

$$|A(C_q)| = hc_{i+1}N_i = |\hat{\mathcal{M}}(C_q)|.$$

Notice that, if we are given any set $\mathcal{P}(C_q)$ of $|\hat{\mathcal{M}}(C_q)|$ node-disjoint order-preserving paths, that connect the vertices of $S(\hat{\mathcal{M}}(C_q))$ to the vertices of $A(C_q)$, such that the paths in $\mathcal{P}(C_q)$ are internally disjoint from the box $B(C_q)$, then we can extend the paths in $\mathcal{P}(C_q)$ inside

$B(C_q)$, so that they route the set $\hat{\mathcal{M}}(C_q)$ of demand pairs. In order to do so, for each c_{i+1}-wide level-i instance $\mathcal{I}' = \mathcal{I}_j(C_q, \ell_q)$, for each level-$i$ instance $\mathcal{I}'_r$ that was used to construct $\mathcal{I}'$, we use the routing that respects the box $B(\mathcal{I}'_r)$, which is guaranteed by the induction hypothesis and by our choice of the demand pairs to route, in order to connect the vertices we have selected on the opening of $B(\mathcal{I}'_r)$ to the corresponding destination vertices.

Consider now some variable x_j, for $1 \le j \le n$. We now show how to select a set $A(x_j)$ of $|\hat{\mathcal{M}}(x_j)|$ vertices on the top boundary of box $B(x_j)$. We start with $A(x_j) = \emptyset$ and then add vertices to it.

Assume first that x_j is assigned the value TRUE. Let $\mathcal{I}'$ be any of the $(65h+1)c_{i+1}$ level-i instances whose box $B(\mathcal{I}')$ is contained in $B_X(x) \cup B_T(x)$. Recall that $\hat{\mathcal{M}}(x_j)$ contains N_i demand pairs that belong to this instance. Let $A'(\mathcal{I}')$ be the set of N_i leftmost vertices on the opening of the box $B(\mathcal{I}')$. For each vertex $v \in A'(\mathcal{I}')$, we add to $A(x_j)$ the vertex v' that belongs to the top boundary of $B(x_j)$ and lies on the same column W as v. We let P_v be the sub-path of W between v and v'. Notice that $|A(x_j)| = |\hat{\mathcal{M}}(x_j)|$, and, given any set $\mathcal{P}(x_j)$ of node-disjoint order-preserving paths connecting the vertices of $S(\hat{\mathcal{M}}(x_j))$ to the vertices of $A(x_j)$, such that the paths in $\mathcal{P}(x_j)$ are internally disjoint from $B(x_j)$, we can extend these paths inside the box $B(x_j)$, so that they route the set $\hat{\mathcal{M}}(x_j)$ of demand pairs. This is done using the box-respecting routing of each level-i instance as before, together with the set $\left\{P_v \mid v' \in A(x_j)\right\}$ of paths.

Finally, assume that x_j is assigned the value FALSE. Notice that the sources of the EXTRA demand pairs for x_j appear to the left of the sources of the FALSE demand pairs of x_j, while the box $B_F(x_j)$ appears to the left of the box $B_X(x_j)$. Therefore, the straightforward routing as above cannot be employed here and we need to "switch" the two sets of paths. In order to do so, we exploit the spacing between the boxes $B_T(x_j), B_F(x_j), B_X(x_j)$ and the boundaries of $B(x_j)$ (see Figure 4.9).

For each of the $(5h+1)c_{i+1}$ level-i instances whose box $B(\mathcal{I}')$ is contained in $B_F(x)$, we define the set $A'(\mathcal{I}')$ of N_i vertices on the opening of the box $B(\mathcal{I}')$, and for each such vertex v, the corresponding path P_v and vertex v' that is added to $A(x_j)$ exactly as before. Let J denote the set of vertices added to $A(x_j)$ so far. We also add to $A(x_j)$ the set J' of $60hc_{i+1}N_i$ leftmost vertices on the top boundary of $B(x_j)$. This ensures that

$$|A(x_j)| = (65h+1)hc_{i+1}N_i = |\hat{\mathcal{M}}(x_j)|.$$

Assume now that we are given any set $\mathcal{P}(x_j)$ of node-disjoint order-preserving paths connecting the vertices of $S(\hat{\mathcal{M}}(x_j))$ to the vertices of $A(x_j)$, such that the paths in $\mathcal{P}(x_j)$ are internally disjoint from $B(x_j)$. We now show how to extend these paths inside the box $B(x_j)$, so that they route the set $\hat{\mathcal{M}}(x_j)$ of demand pairs. We partition the set $\mathcal{P}(x_j)$ into two subsets: set $\mathcal{P}^X$ contains all paths that originate at the sources of the demand pairs in $\hat{\mathcal{M}}^X(x_j)$ — that is, the Extra demand pairs of x_j, and $\mathcal{P}^F$ contains all remaining demand pairs, that must originate at the source vertices of the False demand pairs for x_j. Since the set $\mathcal{P}(x_j)$ of paths is order-preserving, the paths in $\mathcal{P}^X$ terminate at the vertices of J', while the paths in $\mathcal{P}^F$ terminate at the vertices of J. We extend the paths in set $\mathcal{P}^F$ exactly as before, using the paths in set $\left\{P_v \mid v' \in J\right\}$, and then using the box-preserving routing of each corresponding level-i instance to connect each source vertex of $S(\hat{\mathcal{M}}^F(x_j))$ to its destination.

In order to extend the paths in $\mathcal{P}^X$ we do the following. For each level-i instance $\mathcal{I}'$ whose box $B(\mathcal{I}')$ is contained in $B_X(x_j)$, we let $A'(\mathcal{I}')$ be the set of N_i left-most vertices on the opening of $B(\mathcal{I}')$. Let A' be the set of all such vertices, for all such instances $\mathcal{I}'$, so $|A'| = |J'|$. It is now enough to construct a set $\mathcal{Q}$ of order-preserving node-disjoint paths contained in $B(x_j)$ that connect every vertex in J to a distinct vertex of A', such that the paths in $\mathcal{Q}$ are internally disjoint from $B_T(x_j) \cup B_F(x_j) \cup B_X(x_j)$ and are completely disjoint from $\left\{P_v \mid v' \in J\right\}$. We show the existence of the set $\mathcal{Q}$ of paths by constructing a snake $\mathcal{Y}$, that consists of four corridors. The first corridor, Υ_1, is the set of the first N_{i+1} columns of $B(x_j)$.

The third corridor, Υ_3, is the set of the last N_{i+1} columns of $B(x_j)$. Let W' be the last column of Υ_1 and let W'' be the first column of Υ_3. The second corridor, Υ_2, is spanned by the bottom N_{i+1} rows of $B(x_j)$ and the columns of $B(x_j)$ from W' to W'', including these two columns. Let W''' be the leftmost column of G_{i+1} that intersects $B_X(x_j)$; let R' be the topmost row of G_{i+1} that intersects $B_X(x_j)$, and let $\mathcal{R}$ be the set of N_{i+1} consecutive rows lying above R', including R'. The last corridor, Υ_4, is spanned by the set $\mathcal{R}$ of rows, and the set of all columns from W''' to W'' (see Figure 4.14). Using Claim 4.5.3, it is immediate to verify that the desired set $\mathcal{Q}$ of paths exists inside the resulting snake. We then extend the routing inside each box $B(\mathcal{I}')$ of each corresponding level-i instance $\mathcal{I}'$ exactly as before.

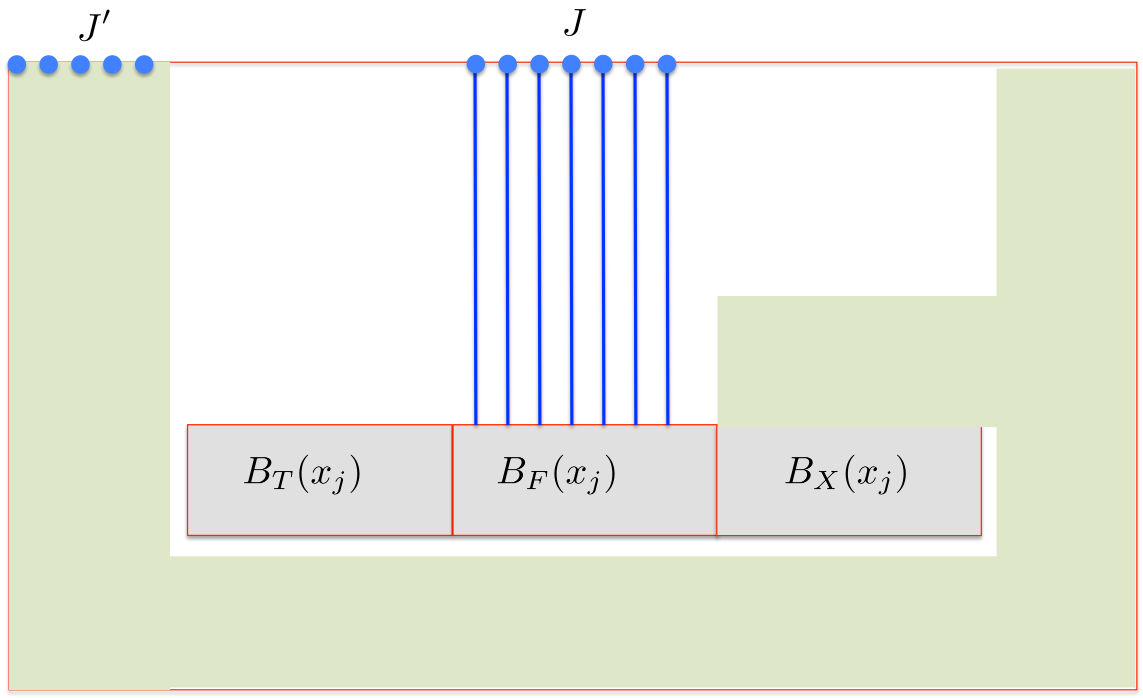


Figure 4.14: Routing inside box $B(x_j)$ if x_j is assigned value F.

Using Claim 4.5.4 with our definitions of the sets $\{A(x_j) \mid 1 \leq j \leq n\} \cup \{A(C_q) \mid 1 \leq q \leq m\}$ of vertices, and the above discussion, we can route all demand pairs in $\hat{\mathcal{M}}$ via a set $\mathcal{P}$ of node-disjoint paths that respects the box $B(\mathcal{I})$.

4.6 No-Instance Analysis

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

Theorem 4.6.1. *Assume that φ is a* No-Instance*. Then for every integer $i \geq 0$, for every instantiation of the level-i instance $\mathcal{I}$, and for every solution $\mathcal{P}$ to this instance, $|\mathcal{P}| \leq N'_i$.*

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 $i \geq 0$ 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 a 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.

Observation 4.6.2. $|\hat{\mathcal{M}}| \leq \delta(200h/3+1)nc_{i+1}N'_i$.

Proof. As observed above, there are at most $75nh+2n$ uninteresting instances, each of which contributes at most $25H_i \leq \frac{50c_{i+1}\epsilon^2 N'_i}{10^{13}}$ excess demand pairs. Therefore, it is enough to show that:

$$(75nh+2n)\cdot\frac{50c_{i+1}\epsilon^2 N'_i}{10^{13}} \leq \delta\left(\frac{200h}{3}+1\right)nc_{i+1}N'_i,$$

which is immediate to verify, substituting $\delta = 8\epsilon^2/10^{12}$. □

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 4.6.2,

$$|\tilde{\mathcal{M}}| \geq (1-2\delta)nc_{i+1}(200h/3+1)N'_i$$

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, and 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

$$\mathcal{C}' = \left\{C_q^j \mid 1 \leq q \leq m, 1 \leq j \leq h\right\}$$

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 $1 \leq q \leq m$, $1 \leq j \leq h$, we call C_q^j the *jth 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: $B(\mathcal{I}_j(C_q, \ell_{q1})), B(\mathcal{I}_j(C_q, \ell_{q2}))$ and $B(\mathcal{I}_j(C_q, \ell_{q3}))$ (see Figure 4.10). 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, $\tilde{\mathcal{M}}(C_q^j, \ell_{q1}), \tilde{\mathcal{M}}(C_q^j, \ell_{q2}), \tilde{\mathcal{M}}(C_q^j, \ell_{q3})$, each of which contains demand pairs from the instances $\mathcal{I}_j(C_q, \ell_{q1}), \mathcal{I}_j(C_q, \ell_{q2})$, and $\mathcal{I}_j(C_q, \ell_{q3})$ respectively. The following observation is immediate:

Observation 4.6.3. *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.

Lemma 4.6.4. *Let $S_1, \ldots, S_r$ be a collection of disjoint subsets of $\tilde{\mathcal{M}}$, such that for all $1 \leq j \leq r$, $|S_j| \geq r^2 H_i/2$. Then there is a collection $\mathcal{M}' = \{(s_1,t_1), \ldots, (s_r,t_r)\}$ of demand pairs, such that for all $1 \leq j \leq r$, $(s_j,t_j) \in S_j$, and for all distinct $(s,t),(s',t') \in \mathcal{M}'$, pair (s',t') does not encircle pair (s,t).*

Proof. We perform $(r-1)$ iterations. At the beginning of iteration ℓ, we are given a set $\mathcal{M}' = \{(s_1,t_1), \ldots, (s_{\ell-1}, t_{\ell-1})\}$ of demand pairs, where for all $1 \leq j \leq \ell-1$, $(s_j,t_j) \in S_j$, and no two pairs in $\mathcal{M}'$ encircle each other. Additionally, for each $\ell \leq j \leq r$, we are given a subset $S'_j \subseteq S_j$ of $(r^2-r(\ell-1))H_i/2$ demand pairs, such that no pair in S'_j encircles a pair in $\mathcal{M}'$, and no pair in S'_j is encircled by a pair in $\mathcal{M}'$. Therefore, at the end of iteration $(r-1)$, set $\mathcal{M}'$ contains one pair from each set $S_1, \ldots, S_{r-1}$, and $S_r \neq \emptyset$. We add an arbitrary pair of S_r to $\mathcal{M}'$ to obtain the desired output.

At the beginning of the algorithm, $\mathcal{M}' = \emptyset$, and for each $1 \leq j \leq r$, set S'_j contains any subset of $r^2H_i/2$ demand pairs of S_j. We now describe an execution of some iteration ℓ. Let $\mathcal{M}'' = \bigcup_{j=\ell+1}^{r} S'_j$, and let U be the set of all vertices appearing on lines Q_t, where t is a destination vertex of a demand pair in $\mathcal{M}''$. Then

$$|\mathcal{M}''| \leq (r-1)(r^2 - r(\ell-1))H_i/2$$

and

$$|U| \leq (r-1)(r^2 - r(\ell-1))H_i^2/4,$$

while S_ℓ contains $(r^2 - r(\ell-1))H_i/2$ demand pairs. Therefore, there is some demand pair $(s_\ell, t_\ell) \in S_\ell$ that contains at most $(r-1)H_i/2$ vertices of U. We add (s_ℓ, t_ℓ) to $\mathcal{M}'$. Consider now some set S'_j, for some $\ell+1 \leq j \leq r$. Then pair (s_ℓ, t_ℓ) may encircle at most $(r-1)H_i/2$ pairs of S'_j, and at most $H_i/2$ pairs in S'_j may encircle (s_ℓ, t_ℓ). We discard from S'_j all pairs that either encircle (s_ℓ, t_ℓ) or are encircled by it. At the end of this procedure,

$$|S'_j| \geq (r^2 - r(\ell-1))H_i/2 - rH_i/2 \geq (r^2 - r\ell)H_i/2.$$

If $|S'_j| > (r^2 - r\ell)H_i/2$, then we discard arbitrary pairs from S'_j until the equality holds. $\square$

Variable Gadget Analysis We fix some variable x and consider its corresponding gadget. We start with the following lemma.

Lemma 4.6.5. *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.

Proof. Assume otherwise. Since we have discarded all excess pairs, each one of the three sets $\tilde{\mathcal{M}}_X, \tilde{\mathcal{M}}_T, \tilde{\mathcal{M}}_F$ contains at least $25H_i$ demand pairs. From Lemma 4.6.4, we can find three pairs, $(s_X, t_X) \in \tilde{\mathcal{M}}_X$, $(s_T, t_T) \in \tilde{\mathcal{M}}_T$ and $(s_F, t_F) \in \tilde{\mathcal{M}}_F$, with neither pair encircling the other. We now claim that it is impossible to route all three pairs simultaneously via node-disjoint paths. In order to do so, we use the following simple observation.

Observation 4.6.6. *Let Σ be the sphere, and let D, D' be two disjoint closed simple discs on Σ, whose boundaries are denoted by Γ and Γ', respectively. Let Σ' be the cylinder obtained from Σ by removing the open discs $D \setminus \Gamma$, $D' \setminus \Gamma'$ from it. Assume further that we have three distinct points a_1, a_2, a_3 appearing on Γ in this circular order, and three distinct points a'_1, a'_3, a'_2 appearing on Γ' in this circular order (see Figure 4.15). Let $\gamma_1, \gamma_2, \gamma_3$ be three simple curves on Σ', where for each $1 \leq j \leq 3$, γ_j connects a_j to a'_j. Then the three curves cannot be pairwise disjoint.*

Proof. Assume otherwise. We can twist the cylinder so that the curve γ_1 becomes a straight vertical line, and then cut the cylinder along this line, obtaining a square, with a_2 appearing to the left of a_3 on the top of the square, and a'_3 appearing to the left of a'_2 on the bottom of the square. It is now immediate to see that it is impossible to connect a_2 to a'_2 and a_3 to a'_3 via disjoint curves.

□

Assume for contradiction that there are three node-disjoint paths, routing demand pairs (s_X, t_X), (s_T, t_T), and (s_F, t_F). We embed the grid G_{i+1} onto a sphere in a natural way, and then construct a curve γ_X, by concatenating the image of the path $P(s_X, t_X)$ with the line Q_{t_X}, denoting by b_X the endpoint of this curve that is distinct from s_X. If this curve is not simple, we delete all loops from it until it becomes simple. We define curves γ_T, γ_F, and their endpoints b_T and b_F for the pairs (s_T, t_T), and (s_F, t_F), respectively, in a similar way.

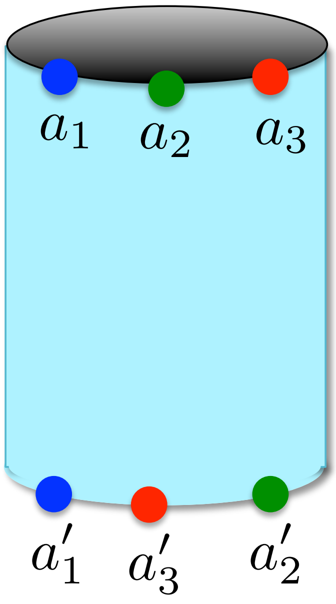


Figure 4.15: Illustration to Observation 4.6.6.

Let Γ be the top boundary of the grid, and let Γ' be the union of the bottom boundaries of the boxes $B_T(x), B_F(x)$ and $B_X(x)$. By slightly thickening Γ and Γ', we can create two disjoint discs D and D' in the plane, with Γ being the boundary of D and Γ' the boundary of D', so that the curves γ_X, γ_T and γ_F are internally disjoint from D and D'. Vertices s_T, s_X and s_F appear on Γ in this circular order, while vertices b_T, b_F and b_X appear on Γ' in this circular order. But then we obtain a collection of three disjoint curves on a cylinder that contradicts Observation 4.6.6. □

The following corollary is now immediate.

Corollary 4.6.7. $|\tilde{\mathcal{M}}^V| \leq (65h+1) \cdot n \cdot c_{i+1} N'_i$.

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 $1 \leq j \leq 5h+1$. We say that index j is bad for variable x if either

1. x is assigned the value True, and instance $\mathcal{I}_j^T(x)$ is uninteresting; or

2. x is assigned the value False, and instance $\mathcal{I}_j^F(x)$ is uninteresting.

We will later show that the total number of pairs (x, j) where j is a bad index for x is bounded by $12n$.

Clause Gadget Analysis Consider a new clause $C_q^j \in \mathcal{C}'$ and its three literals $\ell_{q1}, \ell_{q2}, \ell_{q3}$. 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 $1 \leq z < z' \leq 3$, for which instances $\mathcal{I}_j(C_q, \ell_{q_z})$, $\mathcal{I}_j(C_q, \ell_{q_{z'}})$ are both interesting.

Lemma 4.6.8. *For every original clause $C_q \in \mathcal{C}$, at most three of its copies are troublesome.*

Proof. Assume otherwise, and let $C_q \in \mathcal{C}$ be a clause, such that at least four of its copies are troublesome. Then there are two indices $1 \leq z < z' \leq 3$, and two indices $1 \leq j < j' \leq h$, such that each one of the four instances $\mathcal{I}_j(C_q, \ell_{q_z}), \mathcal{I}_j(C_q, \ell_{q_{z'}}), \mathcal{I}_{j'}(C_q, \ell_{q_z}), \mathcal{I}_{j'}(C_q, \ell_{q_{z'}})$ is interesting, and so each of the four sets $\tilde{\mathcal{M}}(C_q^j, \ell_{q_z}), \tilde{\mathcal{M}}(C_q^j, \ell_{q_{z'}}), \tilde{\mathcal{M}}(C_q^{j'}, \ell_{q_z}), \tilde{\mathcal{M}}(C_q^{j'}, \ell_{q_{z'}})$ contains at least $25H_i$ demand pairs.

From Lemma 4.6.4, we can find demand pairs $(s_1, t_1) \in \tilde{\mathcal{M}}(C_q^j, \ell_{q_z})$, $(s_2, t_2) \in \tilde{\mathcal{M}}(C_q^j, \ell_{q_{z'}})$, $(s_1', t_1') \in \tilde{\mathcal{M}}(C_q^{j'}, \ell_{q_z})$, and $(s_2', t_2') \in \tilde{\mathcal{M}}(C_q^{j'}, \ell_{q_{z'}})$, such that none of the four resulting pairs encircles another. Notice that the sources of these pairs appear on the top boundary of the grid in one of the following two orders: (s_1, s_1', s_2, s_2'), if the variable corresponding to ℓ_{q_z} appears before the variable corresponding to $\ell_{q_{z'}}$ in the ordering $(x_1, x_2, \ldots, x_n)$ of the variables, or (s_2, s_2', s_1, s_1') otherwise.

We draw a curve γ_1 by concatenating the image of the path $P(s_1, t_1)$ and the line Q_{t_1}. We draw curves γ_1', γ_2 and γ_2' for the remaining pairs similarly. On the one hand, none of the the resulting curves may cross each other, while on the other hand the vertices lying on the bottom boundary of the box $B(C_q)$ have been deleted, which is impossible. □

Corollary 4.6.9. *For each original clause $C_q \in \mathcal{C}$, $|\tilde{\mathcal{M}}(C_q)| \leq (6+h)N'_i c_{i+1}$, and overall $|\tilde{\mathcal{M}}^C| \leq 5n(6+h)N'_i c_{i+1}/3$.*

Proof. Consider some original clause $C_q \in \mathcal{C}$. Each of its non-troublesome copies C_q^j contributes at most $c_{i+1}N'_i$ demand pairs to $\tilde{\mathcal{M}}^C$, while a troublesome copy may contribute at most $3c_{i+1}N'_i$ demand pairs. Since at most three copies are troublesome, overall $|\tilde{\mathcal{M}}(C_q)| \leq (6+h)N'_i c_{i+1}$ and

$$|\tilde{\mathcal{M}}^C| \leq m(6+h)N'_i c_{i+1} = 5n(6+h)N'_i c_{i+1}/3,$$

since $m = 5n/3$. □

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, $\mathcal{I}_j(C_q, \ell_{q1}), \mathcal{I}_j(C_q, \ell_{q2})$, and $\mathcal{I}_j(C_q, \ell_{q3})$. 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.

Claim 4.6.10. *$|\mathcal{C}'_0| \leq 12n$.*

Proof. Assume otherwise. From Corollary 4.6.7, $|\tilde{\mathcal{M}}^V| \leq (65h+1)nc_{i+1}N'_i$, while from

Corollary 4.6.9, it is easy to see that

$$\begin{aligned}|\tilde{\mathcal{M}}^C| &\le \frac{5n(6+h)N_i'c_{i+1}}{3} - |\mathcal{C}_0'|N_i'c_{i+1}\\ &\le \frac{5n(6+h)N_i'c_{i+1}}{3} - 12nN_i'c_{i+1}\\ &= nN_i'c_{i+1}(\frac{5h}{3}-2).\end{aligned}$$

But then:

$$\begin{aligned}|\tilde{\mathcal{M}}| &\le (65h+1)nc_{i+1}N_i' + \left(\frac{5h}{3}-2\right)nN_i'c_{i+1}\\ &\le \left(\frac{200h}{3}-1\right)nN_i'c_{i+1}\\ &< (1-2\delta)\left(\frac{200h}{3}+1\right)nN_i'c_{i+1},\end{aligned}$$

since $h = 1000/\epsilon$ and $\delta = 8\epsilon^2/10^{12}$, a contradiction. □

Consider some clause $C_q^j \in \mathcal{C}'$ 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_q^j is troublesome), then we choose one of them arbitrarily. We say that clause C_q^j *chooses* the literal ℓ_{q_z}. We say that C_q^j 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 $\mathcal{C}_1' \subseteq \mathcal{C}' \setminus \mathcal{C}_0'$ be the set of all the cheating clauses. In the following lemma we bound the number of the cheating clauses.

Lemma 4.6.11. *There are at most* $24n$ *bad cheating clauses, and at most* $3m$ *good cheating clauses.*

We provide the proof of the lemma below, after we complete the analysis of the NO-INSTANCE using it. Notice that if a clause C_q^j is an interesting non-cheating clause, then the current assignment must satisfy it. From Claim 4.6.10 and Lemma 4.6.11, there are at least

$$hm - 12n - 24n - 3m = hm - 123m/5 = (1 - 123\epsilon/5000)hm > (1-\epsilon)hm$$

such clauses, contradicting Observation 4.6.3. It now remains to complete the proof of Lemma 4.6.11.

Proof of Lemma 4.6.11. In order to bound the number of bad cheating clauses, we first bound the total number of bad indices for variables. Let β denote the total number of pairs (x, j), where x is a variable of φ and $1 \leq j \leq 5h+1$ is an index, such that j is a bad index for x. Recall that, assuming that x is assigned the value TRUE, this means that instance $\mathcal{I}_j^T(x)$ is uninteresting, and if x is assigned the value FALSE, then instance $\mathcal{I}_j^F(x)$ is uninteresting. For every variable x, let $\beta(x)$ be the total number of indices $1 \leq j \leq 5h+1$ that are bad for x. Then

$$|\tilde{\mathcal{M}}(x)| \leq (65h+1)N_i' c_{i+1} - \beta(x) \cdot N_i' c_{i+1}.$$

Therefore, overall,

$$\begin{aligned} |\tilde{\mathcal{M}}| &\leq n \cdot (65h+1)N_i' c_{i+1} - \beta \cdot N_i' c_{i+1} + |\tilde{\mathcal{M}}^C| \\ &\leq n \cdot (65h+1)N_i' c_{i+1} - \beta \cdot N_i' c_{i+1} + 5n(h+6)N_i' c_{i+1}/3 \\ &= (200h/3 + 11)nN_i' c_{i+1} - \beta \cdot N_i' c_{i+1}. \end{aligned}$$

Assume now for contradiction that $\beta > 12n$. Then

$$|\tilde{\mathcal{M}}| < (200h/3 - 1)nN_i' c_{i+1} < (1-2\delta)(200h/3+1)nN_i' c_{i+1},$$

a contradiction. It is immediate to verify that each pair (x, j) where j is a bad index for variable x may be responsible for at most two bad cheating clauses, and so the total number of bad cheating clauses is bounded by $24n$.

We now turn to bound the number of good cheating clauses. In order to do so, it is enough to prove the following claim:

Claim 4.6.12. *For each original clause $C_q \in \mathcal{C}$ and index $z \in \{1, 2, 3\}$, at most one copy of C_q that chooses the literal ℓ_{q_z} is a good cheating clause.*

Notice that from the above claim, for each original clause $C_q \in \mathcal{C}$, there are at most 3 copies of C_q that are good cheating clauses, and so overall there are at most $3m$ good cheating clauses in $\mathcal{C}' \setminus \mathcal{C}'_0$. It now remains to prove Claim 4.6.12.

Proof of Claim 4.6.12. Let $C_q \in \mathcal{C}$ be an original clause, and let $\ell \in \left\{\ell_{q1}, \ell_{q2}, \ell_{q3}\right\}$ be one of its literals. Assume for contradiction that two distinct copies of C_q, that we denote by C_q^j and $C_q^{j'}$ are good cheating clauses and that they both select the literal ℓ of C_q. Let x be the variable corresponding to ℓ. We assume without loss of generality that x appears in C_q without negation (so $\ell = x$), but we assigned x the value FALSE; the other case is dealt with similarly. We also assume without loss of generality that $j < j'$.

Since both $C_q^j, C_q^{j'}$ are good cheating clauses, indices $j, j+1$ and $j'+1$ are good indices for x. Therefore, each of the instances $\mathcal{I}_j^F(x), \mathcal{I}_{j+1}^F(x)$ and $\mathcal{I}_{j'+1}^F(x)$ is an interesting instance, and $\tilde{\mathcal{M}}$ contains at least $25H_i$ demand pairs that belong to each of the three instances. Moreover, $\tilde{\mathcal{M}}$ contains at least $25H_i$ demand pairs that belong to each of the two instances $\mathcal{I}_j(C_q, \ell)$ and $\mathcal{I}_{j'}(C_q, \ell)$, from the choice of the literal ℓ by each corresponding copies of C_q.

We let $\mathcal{M}_1$ be the set of all demand pairs of $\mathcal{I}_j(C_q, \ell)$ that belong to $\tilde{\mathcal{M}}$, and we define $\mathcal{M}_2$ for instance $\mathcal{I}_{j'}(C_q, \ell)$ similarly. We also let $\mathcal{M}_3, \mathcal{M}_4$ and $\mathcal{M}_5$ be the sets of the demand pairs from instances $\mathcal{I}_j^F(x)$, $\mathcal{I}_{j+1}^F(x)$, and $\mathcal{I}_{j'+1}^F(x)$ that belong to $\tilde{\mathcal{M}}$, respectively. From the above discussion, each of the five sets contain at least $25H_i$ demand pairs. From Lemma 4.6.4, we can find, for each $1 \leq y \leq 5$, a demand pair $(s_y, t_y) \in \mathcal{M}_y$, such that none of the five resulting demand pairs encircle each other.

Consider the plane into which the grid G_i is embedded with only the following curves. First, we add the boundary of the grid. We also add the bottom boundary of the box $B_F(x)$, and for every $3 \leq y \leq 5$, a curve γ_y, obtained by concatenating the image of $P(s_y, t_y)$, with the line Q_{t_y}. Notice that the three curves cannot intersect. Let f and f' be the faces of the resulting drawing, that are distinct from the outer face, and are incident to the intervals Y_j^F and $Y_{j'}^F$, respectively. Let Γ be the bottom boundary of the box $B(C_q)$. Then Γ must lie in the same face of the drawing as t_1 and t_2, since none of the five pairs encircles the other. Assume without loss of generality that this face is $f^* \neq f$. Then the source of the pair (s_1, t_1) lies on the part Y_j^F of boundary of f that is also incident to the infinite face, while its destination lies strictly inside another face (and this other face is disjoint from Y_j^F), which is impossible. $\square$

$\square$

REFERENCES

[1] Alok Aggarwal, Maria Klawe, David Lichtenstein, Nathan Linial, and Avi Wigderson. A lower bound on the area of permutation layouts. *Algorithmica*, 6(1-6):241–255, June 1991.

[2] Alok Aggarwal, Jon Kleinberg, and David P. Williamson. Node-disjoint paths on the mesh and a new trade-off in VLSI layout. *SIAM J. Comput.*, 29(4):1321–1333, February 2000.

[3] M. Ajtai, V. Chvtal, M.M. Newborn, and E. Szemerdi. Crossing-free subgraphs. In Gert Sabidussi Peter L. Hammer, Alexander Rosa and Jean Turgeon, editors, *Theory and Practice of Combinatorics A collection of articles honoring Anton Kotzig on the occasion of his sixtieth birthday*, volume 60 of *North-Holland Mathematics Studies*, pages 9 – 12. North-Holland, 1982.

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

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

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

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

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

[9] Yonatan Aumann and Yuval Rabani. Improved bounds for all optical routing. In *Proceedings of the sixth annual ACM-SIAM symposium on Discrete algorithms*, SODA '95, pages 567–576, Philadelphia, PA, USA, 1995. Society for Industrial and Applied Mathematics.

[10] Chandra Chekuri and Julia Chuzhoy. Half-integral all-or-nothing flow. Unpublished Manuscript.

[11] Chandra Chekuri and Alina Ene. Poly-logarithmic approximation for maximum node disjoint paths with constant congestion. In *Proc. of ACM-SIAM SODA*, 2013.

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

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

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

[15] Julia Chuzhoy. Routing in undirected graphs with constant congestion. *SIAM J. Comput.*, 45(4):1490–1532, 2016.

[16] Julia Chuzhoy and David H. K. Kim. On approximating node-disjoint paths in grids. In Naveen Garg, Klaus Jansen, Anup Rao, and José D. P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2015, August 24-26, 2015, Princeton, NJ, USA*, volume 40 of *LIPIcs*, pages 187–211. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015.

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

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

[19] Julia Chuzhoy, David H. K. Kim, and Rachit Nimavat. New hardness results for routing on disjoint paths. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 86–99, 2017.

[20] Julia Chuzhoy and Shi Li. A polylogarithmic approximation algorithm for edge-disjoint paths with congestion 2. *J. ACM*, 63(5):45:1–45:51, 2016.

[21] Richard Cole and Alan Siegel. River routing every which way, but loose (extended abstract). In *25th Annual Symposium on Foundations of Computer Science, West Palm Beach, Florida, USA, 24-26 October 1984*, pages 65–73, 1984.

[22] M. Cutler and Y. Shiloach. Permutation layout. *Networks*, 8:253–278, 1978.

[23] Shimon Even, Alon Itai, and Adi Shamir. On the complexity of timetable and multicommodity flow problems. *SIAM J. Comput.*, 5(4):691–703, 1976.

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

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

[26] Ken-ichi Kawarabayashi, Yusuke Kobayashi, and Bruce Reed. The disjoint paths problem in quadratic time. *J. Comb. Theory Ser. B*, 102(2):424–435, March 2012.

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

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

[29] Jon M. Kleinberg and Éva Tardos. Approximations for the disjoint paths problem in high-diameter planar networks. *J. Comput. Syst. Sci.*, 57(1):61–73, 1998.

[30] Stavros G. Kolliopoulos and Clifford Stein. Approximating disjoint-path problems using packing integer programs. *Mathematical Programming*, 99:63–87, 2004.

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

[32] Frank Thomson Leighton. *Complexity Issues in VLSI: Optimal Layouts for the Shuffle-exchange Graph and Other Networks.* MIT Press, Cambridge, MA, USA, 1983.

[33] James F. Lynch. The equivalence of theorem proving and the interconnection problem. *SIGDA Newsl.*, 5(3):31–36, September 1975.

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

[35] Prabhakar Raghavan and Clark D. Thompson. Randomized rounding: a technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7:365–374, December 1987.

[36] Satish Rao and Shuheng Zhou. Edge disjoint paths in moderately connected graphs. *SIAM J. Comput.*, 39(5):1856–1887, 2010.

[37] Heike Ripphausen-Lipa, Dorothea Wagner, and Karsten Weihe. Linear-time algorithms for disjoint two-face paths problems in planar graphs. *International Journal of Foundations of Computer Science*, 07(02):95–110, 1996.

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

[39] Neil Robertson and Paul D Seymour. Graph minors. vi. disjoint paths across a disc. *Journal of Combinatorial Theory, Series B*, 41(1):115–138, 1986.

[40] Neil Robertson and Paul D. Seymour. Graph minors. VII. disjoint paths on a surface. *J. Comb. Theory, Ser. B*, 45(2):212–254, 1988.

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

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

[43] Hitoshi Suzuki, Takehiro Akama, and Takao Nishizeki. Finding steiner forests in planar graphs. In *Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '90, pages 444–453, Philadelphia, PA, USA, 1990. Society for Industrial and Applied Mathematics.