ETH-based Hardness for Approximation via L-Reductions from Matrix Tiling with $\leq$

Rahul Goraniya

Supervisor: Dr. Rajesh Chitnis

School of Computer Science

College of Engineering and Physical Sciences

University of Birmingham

2024-25

Abstract

Existing frameworks for establishing lower bounds on parameterized problems often utilize reductions from the Grid Tiling problem. While direct parameterized reductions from Grid Tiling are complex, the introduction of an intermediate problem, Grid Tiling with $\leq$, provides a simpler parameterized reduction to the desired problems while preserving the same bounds. In this work, we extend this approach to optimization problems. Specifically, we L-reduce the Matrix Tiling problem to its intermediate variant, Matrix Tiling with $\leq$, and demonstrate how this can be further reduced to derive PTAS lower bounds for other optimization problems. Our results refine the existing methodology, making it more accessible for establishing PTAS lower bounds.

Acknowledgements

I would like to thank my parents for their constant support throughout this project and my entire degree. I am also deeply grateful to my supervisor Dr. Rajesh Chitnis for introducing me to this interesting field, offering steady support during the project, and always being patient with me. Finally, I would like to thank my project inspector, Dr. Matthew Leeke, for ensuring that the project demonstration went smoothly.

Preface

This report consists of five chapters.

- Chapter 1 provides the necessary background and literature review.
- Chapter 2 introduces the motivation and specific problems studied in this project.
- Chapter 3 presents the main contribution, where we give an L-reduction from MATRIX TILING to MATRIX TILING WITH $\leq$ deriving PTAS lower bound, inspired by the known reduction from GRID TILING to GRID TILING WITH $\leq$.
- Chapters 4 revisit the Independent Set problem. Although PTAS lower bounds for this is already known, we rederive it using our reduction, thereby reinforcing the broader utility of the approach.
- Chapter 5 discusses future work that can be done based on this project

Readers already familiar with the definitions and context may choose to skip Chapters 1 and 2. Similarly, those primarily interested in the new contributions may focus on Chapter 3 and skip the final two chapters.

Contents

List of Figures

CHAPTER 1

Introduction and Overview

In this chapter, we introduce the basic terminology and notations used throughout the report. Unless otherwise stated, the definitions and theorems are adapted from [9].

1.1 NP hardness and P vs NP

In theoretical computer science there are classes of problems which are defined based on the time it takes for algorithms to solve them, below we provide some of the basic definitions.

Definition 1. ***P****: the class of problems which we can solve in polynomial time in the size od the input instance of the problem.*

Definition 2. ***NP****: the class of problems where we can verify a potential solution/answer in polynomial time in the size od the input instance of the problem.*

Clearly, $\mathbf{P} \subseteq \mathbf{NP}$, meaning solving is a (hard) way of verifying. But what about the other direction $\mathbf{P} \subseteq^? \mathbf{NP}$ i.e., Is $\mathbf{P} = \mathbf{NP}$, or $\mathbf{P} \neq \mathbf{NP}$?. We don't know the answer at this moment.

Definition 3. *Polytime reduction: We say that $Y \leq_P X$ if*

- *The reduction taked polynomial time*
- *Y can be solved using (a black-box which solves) X.*

We say that Y is polytime reducible to X.

Definition 4. ***NP-hard****: A problem X is* ***NP-hard*** *if for all problems $Y \in NP$ we have $X \leq_P Y$.*

Intuitively, the above definition means that a problem is **NP-hard** if it is at least as hard as any other problem in NP. Assuming $P \neq NP$, a problem X being NP-hard implies that we cannot have an algorithm ALG for it which satisfies both of the following properties:

- ALG is correct
- ALG runs in polynomial time

This has led to development of new algorithmic paradigms such as:

- (Exact) Exponential Algorithms
- (Polytime) Approximation Algorithms
- (Polytime) Randomization Algorithms
- Parameterized Algorithms

1.2 Introduction to Parameterized Complexity and Why It Is Useful

In classical complexity, we can think of it as analysis in 1-D, we analyze in terms of input size n. But in Parameterized complexity we analyze complexity in terms on input size n as well as solution size k, i.e., we can think of this as analysis in 2-D, as we not only analyze in terms of input size n, but also in terms of size of input k.
In classical complexity the problems: Vertex Cover and Independent Set are equivalent: If S is an independent set, $V \setminus S$ is a vertex cover. But in Parameterized complexity the problems: K-Vertex Cover and K-Independent Set are not equivalent

- There is an **FPT** $\left(f(k) \cdot n^{O(1)}\right)$ algorithm for k-VC.
- But, there is no function f such that the k-IS problem has an algorithm which runs in $f(k) \cdot n^{O(1)}$ time.

The above example is a basic example which shows that the parameterized complexity is much more refined way of analyzing computational complexity of problems. Now we formally define parameterized complexity and parameterized algorithms.
Algorithms with running time $f(k) \cdot n^c$, for a constant c independent of both n and k, are called *fixed-parameter algorithms*, or FPT algorithms. Typically the goal in parameterized algorithmics is to design FPT algorithms, trying to make both the $f(k)$ factor and the constraint c in the bound on the running time as small as possible. FPT algorithms can be put in contrast with less efficient XP algorithms (for *slice-wise polynomial*), where the running time is of the form $f(k) \cdot n^{g(k)}$, for some functions f, g. There is a tremendous difference in the running times $f(k) \cdot n^{g(k)}$ and $f(k) \cdot n^c$.

In parameterized algorithmics, k is simply a *relevant secondary measurement* that encapsulates some aspect of the input instance, be it the size of the solution sought after, or a number describing how "structured" the input instance is.

Any algorithmic theory is incomplete without an accompanying complexity theory that establishes intractability of certain problems. There is such a complexity theory providing lower bounds on the running time required to solve parameterized problems which we will describe later sections.

For the same problem there can be multiple choices of parameters. Selecting the right parameter(s) for a particular problem is an art.

Definition 5 ([9])**.** *A parameterized problem is a language $L \subseteq \sum^* \times \mathbb{N}$, where $\sum$ is a fixed, finite alphabet. For an instance $(x, k) \in \sum \times \mathbb{N}$, k is called the parameter.*

Definition 6 ([9])**.** *A parameter problem $L \subseteq \sum^* \times \mathbb{N}$ is called fixed-parameter tractable (FPT) if there exists an algorithm $\mathcal{A}$ (called a fixed-parameter algorithm), a computable function $f : \mathbb{N} \to \mathbb{N}$, and a constant c such that, given $(x, k) \in \sum^* \times \mathbb{N}$, the algorithm $\mathcal{A}$ correctly decides whether $(x, k) \in L$ in time bounded by $f(k) \cdot |(x, k)|^c$. The complexity class containing all fixed-parameter tractable problems is called FPT.*

Definition 7 ([9])**.** *A parameterized problem $L \subseteq \sum^* \times \mathbb{N}$ is called slice-wise polynomial (XP) if there exists an algorithm $\mathcal{A}$ correctly decided whether $(x, k) \in L$ in time bounded by $f(k) \cdot |(x, k)|^{g(k)}$. The complexity class containing all slice-wise polynomial problems is called XP.*

1.3 W[1] hardness

The theory of NP-completeness at least shows that there is one common underlying reason for the lack of polynomial-time algorithms, and therefore conditional lower bounds based on NP-completeness are the best we can have at this point of time.

Similar to NP-completeness theory of polynomial-time computation, a lower bound theory for parameterized problems has also been developed, which helps algorithm designers to present evidence for as many problems as possible that algorithms with certain specifications do not exist. Being aware of and being able to produce such negative results saves algorithm designers countless hours trying to prove results that contradict commonly accepted assumptions.

As we have no proof of P $\neq$ NP, we cannot rule out the possibility that problems such as CLIQUE and DOMINATING SET are polynomial-time solvable and hence FPT. Therefore, the lower bound theory for parameterized problems has to be conditional: we are proving the statements of the form "if problem A has a certain type of algorithm, then problem B has a certain type of algorithm as well". If we have accepted as a working hypothesis that B has no such algorithms (or we have already proved that such an algorithm for B would contradict our working hypothesis), then this provides evidence that problem A does not have this kind of algorithm either.

The standard notion of polynomial-time reduction used in NP-completeness theory is not sufficient for our purposes. There a notion of parameterized reductions has been developed, which has slightly different flavor that NP-hardness proofs.

If we accept as a working hypothesis that CLIQUE is not fixed-parameter tractable, then the reductions from CLIQUE to other problems provide practical evidence that these problems are not fixed-parameter tractable either.

The remarkable aspect of NP-completeness is that there are literally thousands of natural hard problems that are equally hard in the sense that they are reducible to each other. The situation is different in the case of parameterized problems: there seems to be different levels of hardness in the case of parameterized problems: there seem to be different levels of hardness and even basic problems such as Clique and Dominating set seem to occupy different levels.

Downey and Fellows introduced the W-hierarchy in an attempt to classify parameterized problems according to their hardness. The Clique problem is W[1]-complete, that is, complete for the first level of the W-hierarchy. Therefore, Clique not being fixed-parameter tractable is equivalent to FPT $\neq$ W[1]. This is the basic assumption of parameterized complexity; we interpret W[1]-hardness as evidence that a problem is not fixed-parameter tractable.

Definition 8 (**Parameterized reduction**, [9]). *Let $A, B \subseteq \sum^* \times \mathbb{N}$ be two parameterized problems. A parameterized reduction form A to B is an algorithm that, given an instance (n, k) of A, outputs an instance (x', k') of B such that*

1. *(x, k) is a yes-instance of A if and only if (x', k') is an yes-instance of B,*
2. *$k' \leq g(k)$ for some computable function g, and*
3. *the running time is $f(k) \cdot |x|^{\mathcal{O}(1)}$ for some computable function f.*

A *Boolean circuit* is a directed acyclic graph where the nodes are labeled in the following way:

- every node of indegree 0 is an *input node*,
- every node of indegree 1 is a *negation node*,
- every node of indegree ≥ 2 is either an *and-node* or an *or-node.*

Additionally, exactly one of the nodes with outdegree 0 is labeled as *output node* (in addition to being, for example, an and-node). The *depth* of the circuit is the maximum length of a path from an input node to the output node.

Deciding if a circuit has a satisfying assignment is clearly an NP-complete problem: for example, 3-SAT is its special case. The parameterized version of this problem is defined in the following way. The *weight* of an assignment is the number of input gates receiving value 1. In the Weighted Circuit Satisfiability (WCS) problem, we are given a circuit C and an integer k, the task is to decide if C has a satisfying assignment of weight exactly k and then checking whether it satisfies C. This problem does not seem to be fixed-parameter tractable.

The levels of the W-hierarchy are defined by restricting Weighted Circuit Satisfiability to various classes of circuits. Formally, if $\mathcal{C}$ is a class of circuits, then we define WCS[$\mathcal{C}$] to be the restriction of the problem where the input circuit C belongs to $\mathcal{C}$. To define what kind of restriction we are going to use, we first distinguish between *small nodes,* which have indegree at most 2, and *large nodes*, which have indegree > 2. The $weft$ of a circuit is the maximum number of large nodes on a path from an input node to the output node. We denote by $\mathcal{C}_{t,d}$ the class of circuits with weft at most t and depth at most d.

Definition 9 (W-hierarchy [9]). *For $t \geq 1$, a parameterized problem P belongs to the class $W[t]$ if there is a parameterized reduction from P to $WCS[\mathcal{C}_{t,d}]$ for some $d \geq 1$.*

1.4 E.T.H. and classical complexity

The Exponential Time Hypothesis (ETH) is a conjecture stating that, roughly speaking, 3-SAT has no algorithms subexponential in the number of variables. This conjecture implies that FPT≠W[1], hence it can also be used to give conditional evidence that certain problems are not fixed-parameter tractable. We can for example prove results saying that (assuming ETH) a problem cannot be solved in time $2^{o(n)}$, or a parameterized problem can not be solved in time $f(k)n^{o(k)}$, or a fized-parameter tractable problem does not admit a $2^{o(n)}n^{\mathcal{O}(1)}$-time algorithm. In many cases, the lower bounds obtained this way match (up to small factors) the best known algorithm.

For $q \geq 3$, let δ_q be the infinimum of the set of constants c for which there exists an algorithm solving q-SAT in time $\mathcal{O}^*(2^{cn})$. The *Exponential-Time Hypothesis* is then defined as follows.

Conjecture 1 (Exponential-Time Hypothesis [9]).

$$\delta_3 > 0$$

Intuitively, ETH states that any algorithm for 3-SAT needs to search through an exponential number of alternatives. Note that ETH implies that 3-SAT cannot be solved in time $2^{o(n)}$.

Theorem 1 ([9]). *Unless ETH fails, there exists a constant $c > 0$ such that no algorithm for 3-SAT can achieve running time $\mathcal{O}^*(2^{c(n+m)})$. In particular, 3-SAT cannot be solved in time $2^{o(n+m)}$.*

The CNF-SAT and 3-SAT problems lie at the very foundations of the theory of NP-completeness. The problems around satisfiability of propositional formulas were the first problems whose NP-completness has been settled, and the standard approach to prove NP-hardness of a given problem is to try to find a polynomial-time reduction from 3-SAT, or from some problem whose NP-hardness is already known. Therefore, it is not surprising that by making a stronger assumption about the complexity of 3-SAT, we can infer stronger corollaries about all the problems that can be reached via polynomial time reductions from 3-SAT.

Consider, for instance, a problem A that admits admits a *linear* reduction from 3-SAT, i.e., a polynomial-time algorithm that takes an instance of 3-SAT on n variables and m clauses, and outputs an equivalent instance of A whose size is bounded by $\mathcal{O}(n + m)$. Then if A admitted an algorithm with running time $2^{o(|x|)}$, where $|x|$ is the size of the input instance , then composing the reduction with such an algorithm would yield an algorithm for 3-SAT running in time $2^{o(n+m)}$, which contradicts ETH by Theorem 1.
There are many problems that are proved to be W[1]-hard based on ETH. Some examples are presented in the book Theorem 14.11, Theorem 14.12, Theorem 14.14 etc. in [9, Section 14.4].

CHAPTER 2

Lower bounds based on ETH and motivation of this project

2.1 Framework for lower bounds

Clique was shown to be a W[1]-*hard*, which appears as Theorem 14.21 in [9, Section 14.4]. But it is well known fact that planar graphs do not contain a clique of size 5 or more. So reductions from this problem is not that useful to show W[1]-*hardness* in planar graph problems.

In his 2007 paper, Marx proposed a new problem called the Matrix Tiling problem [12].

Note: We denote by Z_D the set $\{0, 1, ..., D-1\}$throughout the paper.

> Matrix Tiling
>
> ***Input:*** Integers k, D, and k^2 nonempty sets $S_{i,j} \subseteq Z_D \times Z_D$ for $1 \leq i, j \leq k$.
>
> ***Find:*** For each $1 \leq i, j \leq k$, a value $s_{i,j} \in S_{i,j} \cup \{\star\}$ such that:
>
> 1. If $s_{i,j} = (a_1, a_2)$ and $s_{i,j+1} = (b_1, b_2)$, then $a_1 = b_1$.
> 2. If $s_{i,j} = (a_1, a_2)$ and $s_{i+1,j} = (b_1, b_2)$, then $a_2 = b_2$.
>
> ***Goal:*** Maximize the number of pairs (i, j) with $s_{i,j} \neq \star$.

$S_{1,1}$:	$S_{1,2}$:	$S_{1,3}$:
(1,1) (3,1) (2,4)	(5,1) (1,4) (5,3)	(1,1) (2,5) (3,3)
$S_{2,1}$: (2,2) (1,4)	$S_{2,2}$: (3,1) (2,2)	$S_{2,3}$: (3,2) (2,3)
$S_{3,1}$: (1,3) (2,3) (3,3)	$S_{3,2}$: (1,1) (2,3)	$S_{3,3}$: (5,4) (3,4)

Figure 2.1: An instance of Matrix Tiling with $\leq$ with $k = 3$ and $n = 5$. The red pairs form a solution [9, Figure 14.4]

In the same paper, Marx derived a PTAS lower bound for the Matrix Tiling problem [12].

Definition 10 ([15], Definition 1.2)**.** *A polynomial-time approximation scheme (PTAS) is a family of algorithms $\{A_\epsilon\}$, where there is an algorithm for each $\epsilon > 0$, such that A_ϵ is a $(1+\epsilon)$-approximation algorithm (for minimization problems) or a $(1-\epsilon)$-approximation algorithm (for maximization problems).*

Theorem 2 ([12], Theorem 2.3)**.** *If there are constants $\delta, d > 0$ such that* Matrix Tiling *has a PTAS with the running time $2^{O(1/\epsilon)^d} \cdot n^{O(1/\epsilon)^{1-\delta}}$, then ETH fails.*

We can use the above Theorem 2 to derive PTAS lower bounds for new other problems using the following a reduction known as L-reduction and the lemma accompanying it.

> L-reduction:
>
> Let A *and* B be optimization problems and c_A *and* c_B their respective cost functions. A pair of *polynomial time-computable functions* R *and* S *is an* L-reduction if all of the following conditions are met:
>
> 1. If x is an instance of problem A, then $R(x)$ is an instance of problem B,
> 2. If y is a solution to $R(x)$, then $S(y)$ is a solution to x,
> 3. There exists a constant $\alpha > 0$ such that $OPT(R(x)) \leq \alpha\, OPT(x)$,
> 4. There exists a constant $\beta > 0$ such that $|OPT(x) - c_A(S(y))| \leq \beta|OPT(R(x)) - c_B(y)|$.

Lemma 1 ([12], Lemma 2.8 (i))**.** *If there is an L-reduction from* Matrix Tiling *to Problem X, then there are no $d, \delta > 0$ such that Problem X admits a PTAS with running time $2^{O(1/\epsilon)^d} \cdot n^{O(1/\epsilon)^{1-\delta}}$, unless ETH fails.*

This problem turns out to be very useful especially for planar graphs in $\mathbb{R}^2$, because of its planar structure. So Marx then modified this to be a yes/no problem [14], which is known as GRID TILLNG problem.

> GRID TILING
>
> ***Input:*** Integers k, n, and k^2 nonempty sets $S_{i,j} \subseteq Z_n \times Z_n (1 \leq i, j \leq k)$.
> ***Find:*** For each $1 \leq i, j \leq k$, a value $s_{i,j} \in S_{i,j}$ such that:
>
> 1. If $s_{i,j} = (a_1, a_2)$ and $s_{i,j+1} = (b_1, b_2)$, then $a_1 = b_1$.
> 2. If $s_{i,j} = (a_1, a_2)$ and $s_{i+1,j} = (b_1, b_2)$, then $a_2 = b_2$.

But in this problem we have for rows and columns constraint of equality so for reductions we have to check two conditions $\leq$ and $\geq$ for two things to be equal.

So to make reductions simpler Marx and Sidiropoulos simplified the GRID TILING to GRID TILING WITH $\leq$. Now we only have to check one inequality. GRID TILING WITH $\leq$ is very useful in deriving lower bounds for many graph theory problems [3, 10, 1, 13, 7, 6, 2, 11, 8, 5, 4].

> GRID TILING WITH $\leq$
>
> ***Input:*** Integers k, n, and k^2 nonempty sets $S_{i,j} \subseteq Z_n \times Z_n (1 \leq i, j \leq k)$.
> ***Find:*** For each $1 \leq i, j \leq k$, a value $s_{i,j} \in S_{i,j}$ such that:
>
> 1. If $s_{i,j} = (a_1, a_2)$ and $s_{i,j+1} = (b_1, b_2)$, then $a_1 \leq b_1$.
> 2. If $s_{i,j} = (a_1, a_2)$ and $s_{i+1,j} = (b_1, b_2)$, then $a_2 \leq b_2$.

2.2 Motivation of the project

The question we explored for this project is that can we do the same thing for optimization version MATRIX TILING and derive the PTAS lower bound for MATRIX TILING WITH $\leq$, which would make L-reductions to derive Lower bounds for other problems comparatively easier.

In next chapter, we prove the PTAS lower bound for MATRIX TILING WITH $\leq$, which is inspired from the exact reduction Theorem 14.30 in [9, Section 14.4] and in chapter 4, we explore the problem: MAXIMUM INDEPENDENT SET ON UNIT DISK GRAPHS.

> MAXIMUM INDEPENDENT SET ON UNIT DISK GRAPH
> We are given a set S of unit- diameter disks in the plane(described by the coordinates of their centers). The goal is to find a maximum cardinality subset $S' \in S$ of disks, such that the disks in S' are pairwise disjoint.

The Lower bound for the above the problem is already derived but in this project we derive same lower bound using L-reduction from MATRIX TILING WITH $\leq$ which are inspired from the exact reduction Theorem 14.34 in [9, Section 14.4], providing evidence that deriving lower bounds using this framework requires simpler reductions compared to existing techniques.

CHAPTER 3

Lower bound for Matrix Tiling with $\leq$

3.1 Introduction

We begin by formally defining both the variants of Matrix Tiling.

Matrix Tiling:

Input: Integers k, n, and k^2 nonempty sets $\mathcal{S}_{i,j} \subseteq [n] \times [n]$, for $1 \leq i, j \leq k$.
Goal: For each $1 \leq i, j \leq k$, a value $s_{i,j} \in \mathcal{S}_{i,j} \cup \{\star\}$ such that:

- If $s_{i,j} = (a_1, a_2)$ and $s_{i,j+1} = (b_1, b_2)$, then $a_1 = b_1$.
- If $s_{i,j} = (a_1, a_2)$ and $s_{i+1,j} = (b_1, b_2)$, then $a_2 = b_2$.

The objective is to maximize the number of pairs $s_{i,j} \neq \star$.

Matrix Tiling with $\leq$:

Input: Integers k, n, and k^2 nonempty sets $G_{i,j} \subseteq [n] \times [n]$ for each $1 \leq i, j \leq k$.
Find: For each $1 \leq i, j \leq k$, a value $g_{i,j} \in G_{i,j} \cup \{\star\}$ such that:

- If $g_{i,j} = (a_1, a_2)$ and $g_{i,j+1} = (b_1, b_2)$, then $a_1 \leq b_1$.
- If $g_{i,j} = (a_1, a_2)$ and $g_{i+1,j} = (b_1, b_2)$, then $a_2 \leq b_2$.

The objective is to maximize the number of pairs $g_{i,j} \neq \star$.

We prove the following ETH-based PTAS lower bound for Matrix Tiling with $\leq$:

Theorem 3. *If there are constants $\delta, d > 0$ such that* Matrix Tiling with $\leq$ *has a PTAS with the running time $2^{O(1/\epsilon)^d} \cdot n^{O(1/\epsilon)^{1-\delta}}$, then ETH fails.*

Our approach is based on an L-reduction from MATRIX TILING problem to the MATRIX TILING WITH $\leq$. An L-reduction between problems A and B, with respective cost functions c_A and c_B, is a pair of polynomial-time computable functions R and S satisfying the following:

1. If x is an instance of problem A, then $R(x)$ is an instance of problem B,
2. If y is a solution to $R(x)$, then $S(y)$ is a solution to x,
3. There exists a constant $\alpha > 0$ such that $OPT(R(x)) \leq \alpha\, OPT(x)$,
4. There exists a constant $\beta > 0$ such that $|OPT(x) - c_A(S(y))| \leq \beta |OPT(R(x)) - c_B(y)|$.

Note: In the above theorem n is not the range for the coordinated, but the input size of the problem instance.

3.1.1 Conventions and Helper Function:

- We denote $[n] = \{1, \ldots, n\}$,
- Throughout this reduction, the origin $(0, 0)$ lies at the top-left corner of the matrix,
- For a pair (a, b), we use $\text{first}(a, b) = a$, and $\text{second}(a, b) = b$,
- Given a cell $\mathcal{S}_{i,j}$, we refer to its neighboring cell as:
  1. Right: $\mathcal{S}_{i,j+1}$,
  2. Left: $\mathcal{S}_{i,j-1}$,
  3. Above: $\mathcal{S}_{i-1,j}$,
  4. Below: $\mathcal{S}_{i+1,j}$.

$\mathcal{S}_{i,j}$	$\mathcal{S}_{i,j+1}$
$\mathcal{S}_{i+1,j}$	$\mathcal{S}_{i+1,j+1}$

- Let the following helper functions compute min/max coordinate values in neighboring cells:
  - Vertical (first coordinate):
    * $a_{\text{min}}^{\text{below}(i,j)} = \min\{\text{first}(s_{i+1,j}) \mid s_{i+1,j} \in \mathcal{S}_{i+1,j}\}$,
    * $a_{\text{max}}^{\text{below}(i,j)} = \max\{\text{first}(s_{i+1,j}) \mid s_{i+1,j} \in \mathcal{S}_{i+1,j}\}$,
    * $a_{\text{min}}^{\text{above}(i,j)}$, and $a_{\text{max}}^{\text{above}(i,j)}$ defined analogously from $\mathcal{S}_{i-1,j}$.
  - Horizontal (second coordinate):
    * $b_{\text{min}}^{\text{left}(i,j)}$, $b_{\text{max}}^{\text{left}(i,j)}$ from $\mathcal{S}_{i,j-1}$,
    * $b_{\text{min}}^{\text{right}(i,j)}$, $b_{\text{max}}^{\text{right}(i,j)}$ from $\mathcal{S}_{i,j+1}$.
- Let the pair $(a_{i,j}^{\text{max}}, b_{i,j}^{\text{max}})$ be the pair in the set $\mathcal{S}_{i,j}$ with the largest first coordinate. Similarly, let the pair $(a_{i,j}^{\text{min}}, b_{i,j}^{\text{min}})$ be the pair in the set $\mathcal{S}_{i,j}$ with the smallest first coordinate.

We now proceed to defining the functions R, and S and analyze their properties to prove that it is an L-reductions with respect to all four conditions of an L-reduction.

3.2 Constructing the instance of Matrix Tiling with $\leq$ (Definition of R):

We now describe a polynomial-time L-reduction from Matrix Tiling to Matrix Tiling with $\leq$. Let $\mathcal{I} = (k, n, \{\mathcal{S}_{i,j}\})$ be an instance of the Matrix Tiling problem. We will construct an instance $\mathcal{M} = (k', n', \{G_{i',j'}\})$ of Matrix Tiling with $\leq$ such that an approximate solution to $\mathcal{M}$ can be efficiently transformed into an approximate solution to $\mathcal{I}$, satisfying the four L-reduction conditions.

3.2.1 Shifting Coordinates (Step 1):

To allow room for inserting auxiliary pairs, we first apply a uniform shift to all coordinate values.

Let each $\mathcal{S}_{i,j} \subseteq [n] \times [n]$. We define a new set:

$$\mathcal{S}'_{i,j} = \{(x + k, y + k) \mid (x, y) \in \mathcal{S}_{i,j}\}. \tag{3.1}$$

We update the domain size to $n \leftarrow n + 2k$, so that coordinates now lie in $[n + 2k]$. This shift ensures:

- The minimum coordinate is at least k, allowing insertion of values less than any existing coordinate (down to 0),
- The maximum coordinate is at most $n + k$, allowing insertion of values greater than any existing coordinate (up to $n + 2k$).

This transformation preserves all original pair relations, prepares the instance for the addition auxiliary pairs in order to satisfy the four L-reduction conditions.

3.2.2 Auxiliary Coordinate Values (Step 2):

To keep our reduction "approximation preserving" and to satisfy all four conditions L-reduction conditions, we introduce seven auxiliary values for each cell $\mathcal{S}'_{i,j}$, derived from adjacent cells.

For each cell $\mathcal{S}_{i,j}$, we define the following eight new values, collectively called as NEW-I:

For Vertical constraints:	For Horizontal constraints:
• $a^{\text{d}+}_{i,j} = a^{\text{below}(i,j)}_{\max} + 1$, • $a^{\text{d}-}_{i,j} = a^{\text{below}(i,j)}_{\min} - 1$, • $a^{\text{u}+}_{i,j} = \max\{a^{\text{above}(i,j)}_{\max}, a^{\text{d}+}_{i-1,j}, a^{\text{d}-}_{i-1,j}\} + 1$, • $a^{\text{u}-}_{i,j} = \min\{a^{\text{below}(i,j)}_{\min}, a^{\text{d}+}_{i-1,j}, a^{\text{d}-}_{i-1,j}\} - 1$.	• $b^{\text{r}+}_{i,j} = b^{\text{right}(i,j)}_{\max} + 1$, • $b^{\text{r}-}_{i,j} = b^{\text{right}(i,j)}_{\min} - 1$, • $b^{\text{l}+}_{i,j} = \max\{b^{\text{left}(i,j)}_{\max}, b^{\text{r}+}_{i,j-1}, b^{\text{r}-}_{i,j-1}\} + 1$, • $b^{\text{l}-}_{i,j} = \min\{b^{\text{left}(i,j)}_{\min}, b^{\text{r}+}_{i,j-1}, b^{\text{r}-}_{i,j-1}\} - 1$.

3.2.3 Constructing the instance of Matrix Tiling with $\leq$ (Step 3):

We now construct $\mathcal{M} = (n', k', G_{i',j'})$ of MATRIX TILING WITH $\leq$, with

$$n' = 3n^2(k+1) + n^2 + 3n, \quad \text{and} \quad k' = 4k$$

Set $N = 4n^2$, and define a encoding function $\iota(a,b) = n \cdot a + b$, let $z[i,j] = \{\iota(a,b) \mid (a,b) \in \mathcal{S}'_{i,j}\}$, and define:

$$z^{+}_{i,j} = \iota((a^{\max}_{i,j} + 2), b^{\max}_{i,j}), \quad \text{and} \quad z^{-}_{i,j} = ((a^{\min}_{i,j} - 2), b^{\min}_{i,j}).$$

For each cell $\mathcal{S}'_{i,j}$ we construct a gadget which is a 4×4 grid of sets $G_{i',j'}$, indexed by $(4i - 3 \leq i' \leq 4i)$, and $(4j - 3 \leq j' \leq 4j)$ (see Figure 3.1). These can be categorized into two groups:

- 4 **inner sets**: $(G_{4i-2,4j-2}, G_{4i-2,4j-1}, G_{4i-1,4j-2}, G_{4i-1,4j-1})$ are dummy sets and they have one only pairs for each of them. These sets are place-holders and do not depend on pairs from $S^{3n}_{i,j}$.
- 12 **outer sets**: are populated using a mapping function $\iota(a_{i,j}, b_{i,j})$ and N. For each $(a_{i,j}, b_{i,j}) \in \mathcal{S}'_{i,j}$, we call them encoded pairs.

$G_{4i-3,4j-3}$: $(iN-z, jN+z)$	$G_{4i-3,4j-2}$: $(iN+a, jN+z)$	$G_{4i-3,4j-1}$: $(iN-a, jN+z)$	$G_{4i-3,4j}$: $(iN+z, jN+z)$
$G_{4i-2,4j-3}$: $(iN-z, jN+b)$	$G_{4i-2,4j-2}$: $((i+1)N, (j+1)N)$	$G_{4i-2,4j-1}$: $(iN, (j+1)N)$	$G_{4i-2,4j}$: $(iN+z, (j+1)N+b)$
$G_{4i-1,4j-3}$: $(iN-z, jN-b)$	$G_{4i-1,4j-2}$: $((i+1)N, jN)$	$G_{4i-1,4j-1}$: (iN, jN)	$G_{4i-1,4j}$: $(iN+z, (j+1)N-b)$
$G_{4i,4j-3}$: $(iN-z, jN-z)$	$G_{4i,4j-2}$: $((i+1)N+a, jN-z)$	$G_{4i,4j-1}$: $((i+1)N-a, jN-z)$	$G_{4i,4j}$: $(iN+z, jN-z)$

Figure 3.1: The 16 sets of the constructed Matrix Tiling with $\leq$ instance representing a set $S_{i,j}$ of the Matrix Tiling in the reduction in the proof of together with the pairs corresponding to a pair $(a,b) \in S'_{i,j}$(with $z = \iota(a,b)$)

Now we add some pairs to the specific cells in each gadget $G_{i',j'}$ which are created using the NEW-I values introduced in the previous section in the following way, we call these 13 pairs as New-A pairs :

1. Add new pairs to the *corner* cells of the gadget as follows:
   (a) $G_{4i-3,4j-3} = (iN - z^{+}_{i,j}, jN + z^{+}_{i,j})$,
   (b) $G_{4i-3,4j} \quad = (iN + z^{+}_{i,j}, jN + z^{+}_{i,j})$,
   (c) $G_{4i,4j-3} \quad = (iN - z^{+}_{i,j}, jN - z^{+}_{i,j})$,
   (d) $G_{4i,4j} \quad\;\; = (iN + z^{+}_{i,j}, jN - z^{+}_{i,j})$
2. Use the values $\left(b^{\mathrm{r}+}_{i,j}, b^{\mathrm{r}-}_{i,j}, b^{\mathrm{l}-}_{i,j}\right)$, to construct the pairs and add them to the cells as mentioned below:
   (a) $G_{4i-1,4j-3} = (iN - z^{+}_{i,j}, jN - b^{\mathrm{l}-})$.
   (b) $G_{4i-2,4j} \quad = (iN + z^{+}_{i,j}, (j+1)N + b^{\mathrm{r}-})$,
   (c) $G_{4i-1,4j} \quad = (iN + z^{+}_{i,j}, (j+1)N - b^{\mathrm{r}+})$,
3. Use $\left(a^{\mathrm{u}+}_{i,j}, a^{\mathrm{d}+}_{i,j}, a^{\mathrm{u}-}_{i,j}, a^{\mathrm{d}-}_{i,j}\right)$, to construct the pairs and add them to the cells as mentioned below:
   (a) $G_{4i-3,4j-2} = (iN + a^{\mathrm{u}+}, jN + z^{+}_{i,j})$,
   (b) $G_{4i-3,4j-1} = (iN - a^{\mathrm{u}-}, jN + z^{+}_{i,j})$,
   (c) $G_{4i,4j-2} \quad = ((i+1)N + a^{\mathrm{d}-}, jN - z^{+}_{i,j})$,
   (d) $G_{4i,4j-1} \quad = ((i+1)N - a^{\mathrm{d}+}, jN - z^{+}_{i,j})$,

4. Finally, we add a pair fp $= (iN - z^-_{i,j}, jN + b^{l+})$ to the cell $G_{4i-2,4j-3}$, we call this pair **FORBIDDEN PAIR**.

We call the first 11 pairs i.e., NEW-A \ fp (all the pairs from NEW-A excluding the fp pair) as NEW-M pairs.

Claim 1. *For all $z \in z[i,j]$, we have the following relation:*

$$z < z^+_{i,j}$$

Proof. We know that:

- $a_{i,j} \leq a^{\max}_{i,j}$, since $a^{\max}_{i,j} = \max\{\text{first}(s_{i,j}) \mid s_{i,j} \in \mathcal{S}'_{i,j}\}$,
- $z = n \cdot a_{i,j} + b_{i,j}$,
- $z^+_{i,j} = n \cdot (a^{\max}_{i,j} + 2) + b^{\max}_{i,j}$.

Let us upper-bound the largest value in the set $z[i,j] : z_{a_{i,j},b_{i,j}}$. Since $a_{i,j} \leq a^{\max}_{i,j}$, the worst case is $a_{i,j} = a^{\max}_{i,j}$, and since $b_{i,j} \leq b^{\max}_{i,j}$. Then:

$$z_{a_{i,j},b_{i,j}} \leq n \cdot a^{\max}_{i,j} + b^{\max}_{i,j}.$$

On the other hand:

$$z^+_{i,j} = n \cdot (a^{\max}_{i,j} + 2) + b^{\max}_{i,j} = n \cdot a^{\max}_{i,j} + 2n + b^{\max}_{i,j}.$$

Therefore,

$$\begin{aligned} z^+_{i,j} - z_{a_{i,j},b_{i,j}} &\geq 2n > 0 \\ \implies z^+_{i,j} &> z \quad \text{for all} \quad z \in z[i,j]. \end{aligned}$$

□

Claim 2. *For all $z \in z[i,j]$, we have the following relation:*

$$z, z^+_{i,j} > z^-_{i,j}$$

Proof. We know that:

- $a_{i,j} \geq a^{\min}_{i,j}$, since $a^{\min}_{i,j} = \min\{\text{first}(s_{i,j}) \mid s_{i,j} \in \mathcal{S}'_{i,j}\}$,
- $z = n \cdot a_{i,j} + b_{i,j}$,
- $z^-_{i,j} = n \cdot (a^{\min}_{i,j} - 2) + b^{\min}_{i,j}$.

Let us lower-bound the smallest value in the set $z[i,j] : z_{a_{i,j},b_{i,j}}$. Since $a_{i,j} \geq a^{\min}_{i,j}$, the worst case is $a_{i,j} = a^{\min}_{i,j}$, and since $b_{i,j} \geq b^{\min}_{i,j}$. Then:

$$z_{a_{i,j},b_{i,j}} \geq n \cdot a^{\min}_{i,j} + b^{\min}_{i,j}.$$

On the other hand:

$$z^-_{i,j} = n \cdot (a^{\min}_{i,j} - 2) + b^{\min}_{i,j} = n \cdot a^{\min}_{i,j} - 2n + b^{\min}_{i,j}.$$

Therefore,

$$z_{a_{i,j},b_{i,j}} - z^{-}_{i,j} \geq 2n > 0$$
$$\implies z > z^{-}_{i,j} \quad \text{for all} \quad z \in z[i,j].$$

Now, since $z^{+}_{i,j} > z$ for all $z \in z[i,j]$ (from 1), it follows that:

$$z(a_{i,j}, b_{i,j}), z^{+}_{i,j} > z^{-}_{i,j}$$

. □

For all $z \in z[i,j]$, $(1 \leq i,j \leq k)$ we have:

$$z, z^{+}_{i,j}, z^{-}_{i,j} \leq N = 4n^2 \tag{3.2}$$

Claim 3. *For any $1 \leq i,j \leq k$, suppose the assigned pair for the cell $G_{4i-2,4j-3}$ is*

$$(iN - z^{-}_{i,j},\ jN - b^{l+}_{i,j}).$$

Then, the cell $G_{4i-1,4j-3}$ must be assigned $\star$ in any feasible solution.

Proof. Since $G_{4i-2,4j-3}$ is above $G_{4i-1,4j-3}$, the MATRIX TILING WITH $\leq$ constraint requires:

$$\text{first}(g_{4i-1,4j-3}) \geq iN - z^{-}_{i,j}.$$

Since $\text{first}(g_{4i-1,4j-3})$ is of the form $iN - z_{i,j}$, it follows that $z_{i,j}$ less than $z^{-}_{i,j}$.

However by Claim 2, no pair $z \in z[i,j]$ satisfies $z < z^{-}_{i,j}$. Hence, no pair with a feasible first coordinate exists for that cell, and it must be assigned $\star$. □

Lemma 2. *Suppose a NEW-M pair is assigned to any cell of the gadget $G_{i,j}$. Then, the total number of non-$\star$ assignments in $G_{i,j}$ is at most 15.*

Proof. Assume that in the $G_{i,j}$, one of the selected pairs is a NEW-M pair. Without loss of generality, suppose the selected pair appears in cell $G_{4i-3,4j-2}$ and is $(iN + a^{\text{u}+}_{i,j}, jN + z^{+}_{i,j})$.

To satisfy the $\leq$ constraint in MATRIX TILING WITH $\leq$, the pair selected in the next cell in the row, $G_{4i-3,4j-1}$, must have its second coordinate at least $jN + z^{+}_{i,j}$. Since all values $z \in z[i,j]$ satisfy $z < z^{+}_{i,j}$ by 1 Hence,the only feasible option for this cell is the NEW-M pair $(iN - a^{\text{u}-}_{i,j}, jN + z^{+}_{i,j})$.

Proceeding clockwise around the outer sets of the gadget, each cell is similarly forced to be assigned a NEW-M pair to maintain feasibility under the $\leq$ constraints.

Eventually, this propagation reaches a cell (namely $G_{4i-2,4j-3}$) that cannot satisfy the inequality unless it also receives a matching NEW-M pair (because the first coordinate of the pair for this cell must be at most $iN - z^{+}_{i,j}$). However in our construction, no such pair was added to that cell. This cell only received one additional pair why has the first coordinate $iN - z^{-}_{i,j}$, which is strictly greater $iN - z^{+}_{i,j}$ (by claim 2), and therefore there are no pairs which have the first coordinate which is exactly $iN - z^{+}_{i,j}$, which is required to satisfy the condition from the cells above and below it. Thus, it must be assigned $\star$.

Hence, any gadget where a NEW-M pair is selected must contain at least one $\star$, and therefore at most 15 non-$\star$'s. □

From 2 and 2, we can conclude than if all the 16 cells of a gadget $G_{i',j'}$ are non-$\star$ pairs, then none of them is a "NEW-A" pair.

Lemma 3. *Let $\mathcal{S}'_{i,j} = \star$ in a feasible solution to $\mathcal{I}$. Then, the corresponding 4×4 gadget $G_{i',j'}$ in $\mathcal{M}$ admits a feasible assignment with at least 15 non-$\star$ entries.*

Proof. We explicitly construct a feasible assignment for the gadget $G_{i',j'}$ with 15 non-$\star$ pairs, and show that all constraints are satisfied.

Gadget Construction: Select the 15 NEW-M pairs corresponding to the non-$\star$ cells in $G_{i',j'}$. Each pair is chosen to satisfy the local constraints within the gadget:

- In columns 1 and 4, the first coordinates of selected pairs are equal.
- In columns 2 and 3, the vertical consistency is ensured by the ordering of coordinates: for example, in the third row of column 2, we have $iN + a$ above $iN + N$ (as $N = 4n^2$ and $a \leq n$), satisfying the constraint. Horizontal constraints follow similarly.

Inter-Gadget Constraints: We verify that the "$\leq$" constraints hold across gadgets for both horizontal and vertical adjacencies. Each direction is handled via four exhaustive cases depending on whether each adjacent cell is $\star$ or not.

Horizontal Cases:

(i) Both $\mathcal{S}'_{i,j}$ and $\mathcal{S}'_{i,j+1}$ are non-$\star$: As shown in Figure 3.2 (a), the second coordinates of all pairs selected in each cell satisfy the constraints using the fact that $N > z_{i,j}$ (from Equation 3.2), and $b'_{i,j} = b'_{i,j+1}$.

(ii) $\mathcal{S}'_{i,j}$ is non-$\star$, $\mathcal{S}'_{i,j+1}$ is $\star$: See Figure 3.2 (b), the second coordinates of all pairs selected in each cell satisfy the constraints using the fact that $N > z'_{i,j}$ (from Equation 3.2), and $b'_{i,j} > b^{l-}_{i,j+1}$.

(iii) $\mathcal{S}'_{i,j}$ is $\star$, $\mathcal{S}'_{i,j+1}$ is non-$\star$: See Figure 3.2 (c), the second coordinates of all pairs selected in each cell satisfy the constraints using the fact that $N > z^{+}_{i,j}$ (from Equation 3.2), $b^{r-}_{i,j} < b'_{i,j+1}$, and $b^{r+}_{i,j} > b'_{i,j+1}$.

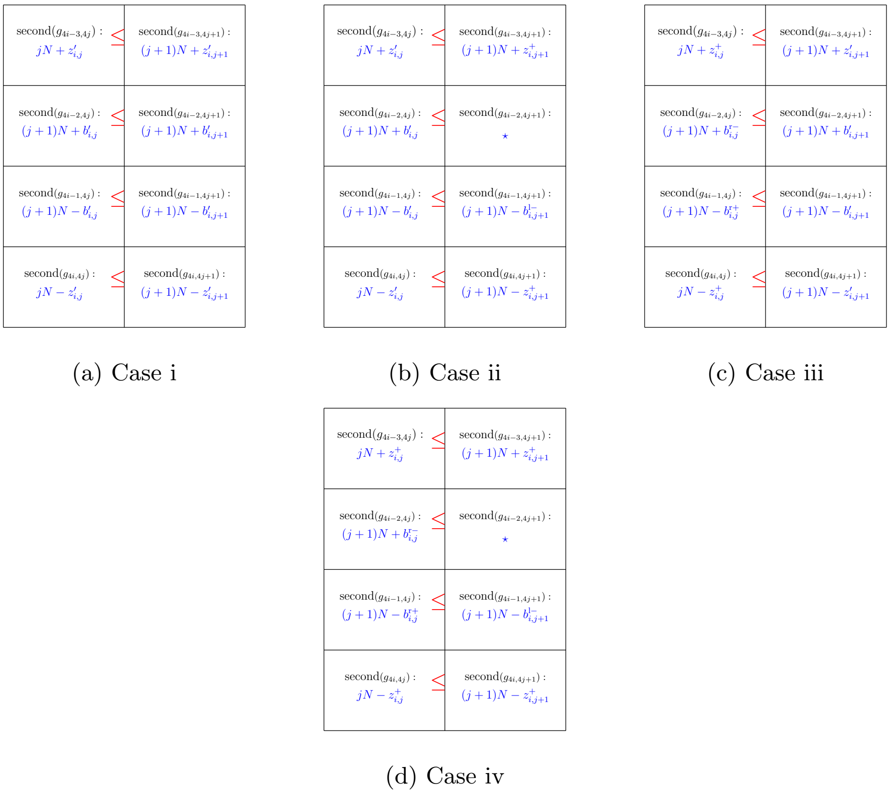
(iv) Both are $\star$: See Figure 3.2 (d), the second coordinates of all pairs selected in each cell satisfy the constraints using the fact that $N > z^{+}_{i,j}$ (from Equation 3.2), and $b^{r+}_{i,j} > b^{l-}_{i,j+1}$.

Vertical Cases:

(i) Both $\mathcal{S}'_{i,j}$ and $\mathcal{S}'_{i+1,j}$ are non-$\star$: See Figure 3.3 (a), the first coordinates of all pairs selected in each cell satisfy the constraints using the fact that $N > z_{i,j}$ (from Equation 3.2), and $a'_{i,j} = a'_{i,j+1}$.

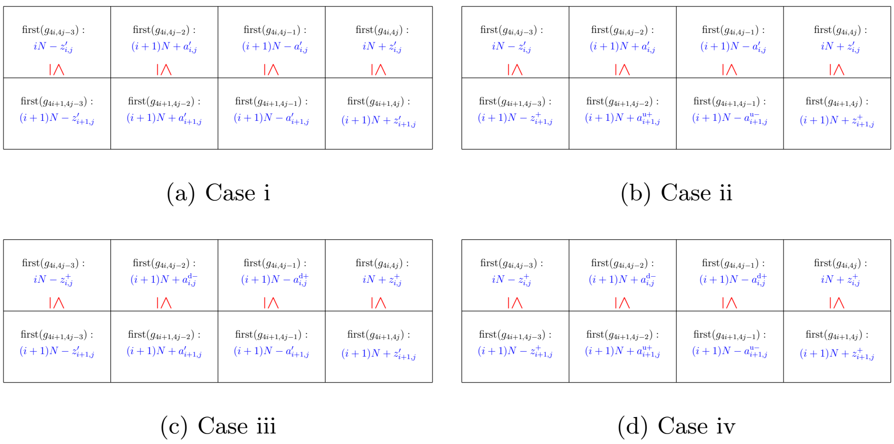
(ii) $\mathcal{S}'_{i,j}$ is non-$\star$, $\mathcal{S}'_{i+1,j}$ is $\star$: See Figure 3.3(b). Constraints are satisfied using the inequalities: $N > z_{i,j}$, $a^{u+}_{i+1,j} > a'_{i,j}$ and $a^{u-}_{i+1,j} < a'_{i,j}$.

(iii) $\mathcal{S}'_{i,j}$ is $\star$, $\mathcal{S}'_{i+1,j}$ is non-$\star$: See Figure 3.3(c). Constraints are satisfied using the inequalities: $N > z^{+}_{i,j}$, $a^{d-}_{i,j} < a'_{i+1,j}$ and $a^{d+}_{i,j} > a'_{i+1,j}$.



(a) Case i (b) Case ii (c) Case iii (d) Case iv

Figure 3.2: All four cases for the horizontal constraint.



(a) Case i (b) Case ii (c) Case iii (d) Case iv

Figure 3.3: All four cases for the vertical constraint.

(iv) Both are $\star$: See Figure 3.3(d). Constraints are satisfied using the inequalities: $N >^+ z_{i,j}$, $a^{d-}_{i,j} < a^{u+}_{i+1,j}$ and $a^{d+}_{i,j} > a^{u-}_{i+1,j}$.

In each case, constraints across the gadgets are satisfied due to the definitions and inequalities involving parameters like z^+, $a^{u\pm}$, $a^{d\pm}$, b^{l-}, $b^{r\pm}$, ensuring feasibility.

Remark. *Here we point out that we have also added the pair* fp $= (iN - z^-_{i,j}jN + b^{l+})$ *to the cell* $G_{4i-2,4j-3}$. *Similar to all the other pairs it satisfies the inter-gadget constraint in all the cases, both it doesn't satisfy the vertical constraint within the gadget that is why it cannot be picked. But this pair can be useful whenever we are doing L-reductions from and using this constructed instance of* Matrix Tiling with $\leq$ *as an intermediate gadget, because this pair satisfies one of the two constraints of the cell (horizontal), hence during L-reductions to minimization problems, whenever there is a* $\star$ *in the optimal solution of , in the corresponding intermediate gadget of* Matrix Tiling with $\leq$, *we only need to worry about one directional constraint (vertical).*

□

3.3 Constructing a solution of $\mathcal{I}$ given any solution of Matrix Tiling with $\leq$ (Definition of S):

Lemma 4 (Uniform Encoding in Gadgets)**.** *Suppose a gadget* $G_{i',j'}$ *in* $R(x)$ *contains* 16 *non-$\star$ values in a feasible solution* y*, Then all* 12 *outer cells of the gadget encode the same pair* $(a, b) \in \mathcal{S}'_{i,j}$.

Proof. Since the gadget has no $\star$ assignment, none of the selected values are NEW-A (see observation 3.2.3). Therefore, all selected values come from the encoding of some original pair $(a, b) \in \mathcal{S}'_{i,j}$.

Let the pairs selected in the solution from these sets define 12 values z, denoted as $z_{4i-3,4j-3}, z_{4i-3,4j-2}, \ldots$, representing the values selected from these sets. We claim that all these 12 values are equal.

To see this, let us first consider the second coordinate of the pairs selected from the set $G_{4i-3,4j-3}$ which is $jN + z_{4i-3,4j-3}$, and $G_{4i-3,4j-2}$ which is $jN + z_{4i-3,4j-2}$. By the $\leq$ constraint of Matrix Tiling with $\leq$, it follows that:

$$z_{4i-3,4j-3} \leq z_{4i-3,4j-2}.$$

Continuing this reasoning for the other sets, we obtain the following chain of inequalities:

$$\begin{array}{rrrrr}
z_{4i-3,4j-3} \leq & z_{4i-3,4j-2} \leq & z_{4i-3,4j-1} \leq & z_{4i-3,4j} & \text{(first row)} \\
z_{4i-3,4j} \leq & z_{4i-2,4j} \leq & z_{4i-1,4j} \leq & z_{4i,4j} & \text{(last column)} \\
-z_{4i,4j-3} \leq & -z_{4i,4j-2} \leq & -z_{4i,4j-1} \leq & -z_{4i,4j} & \text{(last row)} \\
-z_{4i-3,4j-3} \leq & -z_{4i-2,4j-3} \leq & -z_{4i-1,4j-3} \leq & -z_{4i,4j-3} & \text{(first column)}
\end{array}$$

Combining all these inequalities results in a cycle of equalities, which implies that all the 12 values are the same.

Let $z^{i,j}$ be this common value and let $s_{i,j} = (a_{i,j}, b_{i,j})$ be the corresponding pair, that is, $\iota(a_{i,j}, b_{i,j}) = z^{i,j}$. The fact that $z^{i,j}$ was defined using the pairs appearing in the gadget of $\mathcal{S}'_{i,j}$ implies that $s_{i,j} \in \mathcal{S}'_{i,j}$. We call this step of retrieving the pairs as *decoding the gadget*.

$\square$

Let y be a feasible solution to the Matrix Tiling with $\leq$ instance $\mathcal{M}$, which is the output of our reduction $R(x)$ applied to an instance x of Matrix Tiling. Define the function S as follows:

Definition 11 (Solution Mapping $S(y)$)**.** *For each gadget $G_{i',j'}$ in $\mathcal{M}$, define:*

- *If **all 16 cells** of the gadget are assigned non-$\star$ values, decode the gadget to retrieve the unique encoded pair $(a, b) \in \mathcal{S}'_{i,j}$ using the gadget decoding step from 4, and assign:*

$$s_{i,j}(a-k, b-k) \in S_{i,j}. \tag{3.3}$$

- *If **any cell** in the gadget is assigned $\star$, set:*

$$s_{i,j}\star$$

Then $S(y) = \{s_{i,j} | 1 \leq i, j \leq k\}$ is a candidate solution to the instance $\mathcal{I}$, of .

Lemma 5 (Feasibility of $S(y)$ for $\mathcal{I}$)**.** *Let y be any feasible solution to $R(x)$. Then $S(y)$ is a feasible solution to x.*

Proof. Let $s_{i,j}$ be the pair extracted from gadget $G_{i',j'}$, and let $s_{i,j} = \star$ if any cell in the gadget is $\star$.

We now verify that the equality constraints of are satisfied for each pair of adjacent cells:

- Notice we do not have to check the constraints where either of the adjacent cells is a $\star$.
- Therefore, suppose both $s_{i,j} \neq \star$ and $s_{i+1,j} \neq \star$, and they were decoded to values $(a_{i,j}, b_{i,j}) \in \mathcal{S}'_{i,j}$ and $(a_{i+1,j}, b_{i+1,j}) \in \mathcal{S}'_{i+1,j}$ respectively.
- The first coordinates of the pairs selected from the cells $G_{4i,4j-2}$ and $G_{4i+1,4j-2}$ are $(i+1)N + a_{i,j}$ and $(i+1)N + a_{i+1,j}$, and by the $\leq$ constraint of Matrix Tiling with $\leq$, we obtain: $a_{i,j} \leq a_{i+1,j}$.
- Similarly, comparing the first coordinates of the pairs selected from the cells $G_{4i,4j-1}$ and $G_{4i+1,4j-1}$ yields $-a_{i,j} \leq -a_{i+1,j}$.
- Comparing the above two equations, we can conclude:

$$a_{i,j} = a_{i+1,j}.$$

- With the similar argument for the horizontal direction, we get $b_{i,j} = b_{i,j+1}$.

Finally, in first step of defining R, we shifted all the coordinates of all pairs by k (see Equation 3.1), that is why we have defined our solution mapping function S to subtract k from both the coordinates of the retrieved pair $(a, b) \in \mathcal{S}'_{i,j}$ (see Equation 3.3) to get the pair which belongs to the original $\mathcal{S}_{i,j}$ set in x. $\square$

3.4 Relation between the optimal solutions of x and $R(x)$ (Deriving α):

Now without analyzing the function R, we trivially bound the ratio between the optimal values of an instance x of Matrix Tiling and its reduced instance $R(x)$ of Matrix Tiling$\leq$, thereby deriving the constant α in the L-reduction.

Lemma 6. *There exists a constant $\alpha = 64$ such that*

$$\mathrm{OPT}(R(x)) \leq \alpha \cdot \mathrm{OPT}(x),$$

where $\mathrm{OPT}(x)$ *and* $\mathrm{OPT}(R(x))$ *denote the optimal values of the respective instances.*

Proof. Since at most one element is assigned per cell, we have $\mathrm{OPT}(x) \leq k^2$. For a lower bound, assign an arbitrary element in cells (i, j) where both i and j are odd, and $\star$ elsewhere. This gives at least $k^2/4$ assignments, so $\mathrm{OPT}(x) \geq k^2/4$ (with similar arguments, same bounds hold for $R(x)$ as well).

In the reduced instance $R(x)$, the grid has size $k' = 4k$, so

$$\mathrm{OPT}(R(x)) \leq (k')^2 = 16k^2 = 64 \cdot (k^2/4) \leq 64 \cdot \mathrm{OPT}(x)$$

Thus, $\alpha = 64$ satisfies the required bound. □

3.5 Relation between the optimal solutions and any approximate solutions of $\mathcal{I}$ and $\mathcal{M}$ (Deriving β):

Let us first analyze the relation between the optimum solutions of both the instances:

Lemma 7. *If $OPT(x) = k^2 - a$, then $OPT(R(x)) = 16k^2 - a$.*

Proof. Assume that the optimal solution for instance x selects $k^2 - a$ cells, implying that exactly a cells are assigned the symbol $\star$. We construct a corresponding solution for instance $R(x)$ as follows.

For each cell $\mathcal{S}_{i,j}$ such that the selected entry in the optimal solution of x is a valid pair $(a_{i,j}, b_{i,j})$, we include in the solution of $R(x)$ all 16 encoding cells within the corresponding gadget $G_{i,j}$ that encode this pair. For each cell $\mathcal{S}_{i,j}$ where the optimal solution of x contains a $\star$, we apply the construction in 3 to select exactly 15 non-$\star$ cells from the corresponding gadget $G_{i,j}$.

This yields a total of

$$(k^2 - a) \cdot 16 + a \cdot 15 = 16k^2 - a$$

non-$\star$ cells in the constructed solution for $R(x)$, thus establishing that $\mathrm{OPT}(R(x)) \geq 16k^2 - a$.

To show optimality, suppose there exists a solution for $R(x)$ with more than $16k^2 - a$ non-$\star$ cells. Then there must exist some gadget $G_{i,j}$, corresponding to a $\star$-cell $\mathcal{S}_{i,j}$ in the optimal solution of x, in which all 16 encoding cells are selected. By 4, this implies that all selected cells correspond to a common pair (a, b), which must satisfy the row and column constraints of x (5). This contradicts the assumption that $\mathcal{S}_{i,j}$ is a $\star$-cell in the optimal solution of x. Hence, no such solution exists, and the constructed solution is indeed optimal. $\square$

Now, based on our definition of the function S, let analyze the relation of the solution to x, based on the definition of our solution mapping function S, given any feasible solution to $R(x)$:

Lemma 8. *If $c_A(y) = 16k^2 - m$ and $c_B(S(y)) = k^2 - n$, then $m \geq n$.*

Proof. By the definition of the mapping function S, an entry $s_{i,j} \neq \star$ only if all 16 vertices in the corresponding gadget $G_{i,j}$ are non-$\star$ in y. Thus, each $\star$ in y can invalidate at most one such gadget

Since $c_A(y) = 16k^2 - m$, the assignment y contains exactly m cell which are $\star$, implying that the number of $\star$ entries in $S(y)$ is at most m. Since $S(y)$ contains exactly n $\star$'s. Therefore, we must have

$$k^2 - n \leq k^2 - m,$$

which implies $m \geq n$, as required. $\square$

Now, based on 7 and 8, it is easy to see that for our L-reduction, the value of β can be 1, which is proved formally below:

Lemma 9. *There exists a constant $\beta = 1$ such that*

$$|OPT(x) - c_A(S(y))| \leq \beta |OPT(R(x)) - c_B(y)|$$

where $\mathrm{OPT}(x)$ *and* $\mathrm{OPT}(R(x))$ *denote the costs of optimal solutions to the respective instances, and $c_A(S(y))$, $c_B(y)$ denote the costs of the mapped and original (possibly non-optimal) solutions respectively.*

Proof. Let $c_A(S(y)) = k^2 - n, c_B(y) = 16k^2 - m$ and $\mathrm{OPT}(x) = k^2 - a$, from 7 $\mathrm{OPT}(R(x)) = 16k^2 - a$, for some $a, n, m \in [0, k^2]$.Substituting into the inequality, we obtain:

$$\begin{aligned} &\mathrm{OPT}(x) - c_A(S(y)) \leq 1 \cdot (\mathrm{OPT}(R(x)) - c_B(y)) \\ \Longrightarrow\ & k^2 - a - k^2 + n \leq 16k^2 - a - 16k^2 + m \\ \Longrightarrow\ & n - a \leq m - a \end{aligned}$$

By 8, we have $n \leq m$, and since a is fixed across both sides, the inequality holds. Hence, the claim holds with $\beta = 1$. $\square$

Note: Because both the problems MATRIX TILING and MATRIX TILING WITH $\leq$ are maximization optimization problems, we have $OPT(x) \geq c_A(S(y))$, and $OPT(R(x)) \geq c_B(y)$. So we can ignore the modulus used in the fourth condition in the L-reduction definition.

3.6 Proof of Theorem 3:

We are now ready to prove our main Theorem 3, which is restated below:

Theorem. *If there are constants $\delta, d > 0$ such that* Matrix Tiling with $\leq$ *has a PTAS with the running time $2^{O(1/\epsilon)^d} \cdot n^{O(1/\epsilon)^{1-\delta}}$, then ETH fails.*

Proof. It is easy to verify that the functions R and S in our reduction are computable in polynomial time with respect to the size of the Matrix Tiling instance. From section 3.4 and section 3.5, we have established that $\alpha = 64$ and $\beta = 1$. Thus, the reduction from Matrix Tiling to Matrix Tiling with $\leq$ is an L-reduction.

Now by [12, Lemma 2.8(1)], if there exists an L-reduction from Matrix Tiling to a problem X (in our case, Matrix Tiling with $\leq$), then X cannot admit a PTAS with running time of the form $2^{O((1/\epsilon)^d)} \cdot n^{O((1/\epsilon)^{1-\delta})}$ for any constants $d, \delta > 0$, unless the ETH fails.

Applying this lemma to our reduction completes the proof.

$\square$

Note: In the above theorem n is not the range for the coordinates, but the input size of the problem instance. Now we have the following lemma:

Lemma 10. *If there is an L-reduction from* Matrix Tiling with $\leq$ *to Problem X, then there are no $d, \delta > 0$ such that Problem X admits a PTAS with running time $2^{O(1/\epsilon)^d} \cdot n^{O(1/\epsilon)^{1-\delta}}$, unless ETH fails.*

CHAPTER 4

Maximum Independent Set on Unit Disk Graph

4.1 Introduction

Maximum Independent Set on Unit Disk Graph:
We are given a set S of unit- diameter disks in the plane(described by the coordinates of their centers). The goal is to find a maximum cardinality subset $S' \in S$ of disks, such that the disks in S' are pairwise disjoint.

Note: When dening the Matrix Tiling problem, we imagined the sets $S_{i,j}$ arranged in a matrix, with $S_{i,j}$ being in row i and column j. When reducing Matrix Tiling to a geometric problem, the natural idea is to represent $S_{i,j}$ with a gadget located around coordinate (i, j). However, this introduces an unnatural 90 degrees rotation compared to the layout of the $S_{i,j}$'s in the matrix, which can be confusing in the presentation of a reduction. Therefore, for geometric problems, it is convenient to imagine that $S_{i,j}$ is located at coordinate (i, j). To emphasize this interpretation, we use the notation $S[x, y]$ to refer to the sets; we imagine that $S[x, y]$ is at location (x, y), hence sets with the same x are on a vertical line and sets with the same y are on the same horizontal line (see Figure 4.1). The constraints of Matrix Tiling are the same as before: the pairs selected from $S[x, y]$ and $S[x + 1, y]$ agree in the first coordinate, while the pairs selected from $S[x, y]$ and $S[x, y + 1]$ agree in the second coordinate. Matrix Tiling with $\leq$ is defined similarly. With this notation, we can give a very clean and transparent L-reduction to Maximum Independent Set of Unit Disk Graphs.

4.2 Reduction and relation between solutions of both the instances.

Construction of R

It will be convenient to work with open disks of radius $\frac{1}{2}$ (diameter 1). Two disks are nonintersecting if and only if the distance between their centers is at least 1.
Let $I = (n, k, S)$ be an instance of $MT \leq$. We construct a set $\mathcal{D}$ of unit disks. Let $\epsilon = 1/n^2$. For every $1 \leq x, y \leq k$ and every $(a, b) \in S[x, y] \subset [n] \times [n]$, we introduce into $\mathcal{D}$ an open disk of radius $\frac{1}{2}$ centered at $(x + \epsilon a, y + \epsilon b)$; Let $D[x, y]$ be the set of these $|S[x, y]|$ disks introduced for a fixed x and y (see Fig.). Note that the disks in $D[x, y]$ all intersect each other. Therefore, if $D' \subseteq \mathcal{D}$ is a set of pairwise nonintersecting disks, then $|D'| \leq k^2$ and $|D'| = k^2$ is possible only if D' contains exactly one disk from each $D[x, y]$. We need the following observation first. Consider two disks centered at $(x + \epsilon a, y + \epsilon b)$ and $(x + 1 + \epsilon a', y + \epsilon b')$ for some $(a, b), (a', b') \in [n] \times [n]$. We claim that they are nonintersecting if and only if $a \leq a'$. Indeed, if $a > a'$, then the square of the distance of the two centers is

$$\begin{aligned}(1 + \epsilon(a' - a))^2 + \epsilon^2(b' - b)^2 &\leq (1 + \epsilon(a' - a))^2 + \epsilon^2 n^2 \\ &\leq (1 - \epsilon)^2 + \epsilon = 1 - \epsilon + \epsilon^2 < 1\end{aligned}$$

(in the first inequality, we have used $b', b \leq n$; in the second inequality, we have used $a \geq a' + 1$ and $\epsilon = 1/n^2$). On the other hand, if $a \leq a'$, then the square of the distance is at least $(1 + \epsilon(a' - a))^2 \geq 1$, hence the two disks do not intersect (recall that the disks are open). This proves our claim. A similar claim shows that disks centered at $(x + \epsilon a, y + \epsilon b)$ and $(x + \epsilon a', y + 1 + \epsilon b')$ are nonintersecting if and only if $b \leq b'$. Moreover, it is easy to see that the disks centered at $(x + \epsilon a, y + \epsilon b)$ and $(x' + \epsilon a', y' + \epsilon b')$ for some $1 \leq a, a', b, b' \leq n$ cannot intersect if $|x - x'| + |y - y'| \geq 2$: the square of the distance between the two centers is at least $2(1 - \epsilon n)^2 > 1$.

$S[1,3]$: (1,1) (2,5) (3,3)	$S[2,3]$: (3,2) (2,3)	$S[3,3]$: (5,4) (3,4)
$S[1,2]$: (5,1) (1,4) (5,3)	$S[2,2]$: (3,1) (2,2)	$S[3,2]$: (1,1) (2,3)
$S[1,1]$: (1,1) (3,1) (2,4)	$S[2,1]$: (2,2) (1,4)	$S[3,1]$: (1,3) (2,3) (3,3)

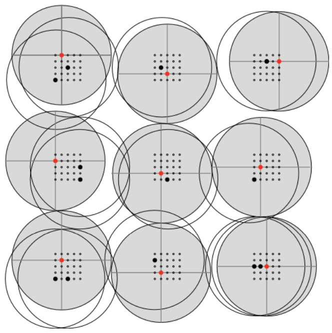


Figure 4.1: An instance of Matrix Tiling with $\leq$ with $k = 3$ and $n = 5$ and the corresponding instance of Maximum Independent Set on Unit Disk Graph constructed in the reduction. The small dots show the potential positions for the centers, the large dots are the actual centers in the constructed instance. The shaded disks with red centers correspond to the solution of Matrix Tiling with $\leq$ shown on the left [9, Figure 14.7]

Lemma 11. *If we have some any solution $k^2 - e$ for I we can have at least $k^2 - e$ for our constructed instance of $\mathcal{D}$.*

Proof. If we have $k^2 - e$ as a solution for I, it means we have the number of $\star$'s in the solution is e, for all the cells with the $\star$ in the solution don't pick any disk from that corresponding $D[x, y]$, and for the $k^2 - e$ cells which are non-$\star$ pairs. Let these non-$\star$ pairs be $s[x, y] = (a[x, y], b[x, y])$ for the $S[x, y]$'th cell. For every non-$\star$ cell, we select the disk $d[x, y]$ centered at $(x+\epsilon a[x, y], y+\epsilon b[x, y]) \in D[x, y]$.

Now we prove that all the disks selected this way do not intersect each other:

If the neighboring (right) cell in the solution of I was $\star$ we are not selecting the disk from that particular set ($D[x + 1, 1]$, where cell $S[x + 1, y]$ is $\star$). then we do not need to worry about the intersecting condition as the next closest disk (on the right side) will be in the set $D[x + 2, y]$ which will imply that $|x - x'| + |y - y'| \geq 2$ and as proved earlier $d[x, y]$ and $d[x+2, y]$ cannot intersect.

We can prove in the similar fashion all the remaining three cases:

- where the left cell is $\star$,
- where the above cell is $\star$,
- and where the below cell is $\star$.

Now we will prove that the disks selected in the suggested way will not intersect even in the case where all the four immediate neighboring cells are non-$\star$ in the solution of I. We can prove this is the following way: As have seen, if $|x - x'| + |y - y'| \geq 2$, then $d[x, y]$ and $d[x', y']$ cannot intersect. As the $s[x, y]$'s form a solution of the instance I, we have that $a[x, y] \leq a[x + 1, y]$. Therefore, by our claim above, the disks $d[x, y]$ and $d[x + 1, y]$ do not intersect. Similarly, we have $b[x, y] \leq b[x, y + 1]$, implying that $d[x, y]$ and $d[x, y + 1]$ do not intersect either. Hence there is indeed a set of at least $k^2 - e$ pairwise nonintersecting disks in $\mathcal{D}$.

$\square$

Now using the Lemma 11, we can prove the following relation for $OPT(x) = k^2 - a$ and $OPT(R(x)) = k^2 - b$

$$\begin{aligned} OPT(R(x)) &\geq OPT(x) \\ \implies k^2 - b &\geq k^2 - a \\ \implies -b &\geq -a \\ \implies b &\leq a \end{aligned} \tag{4.1}$$

Construction of S

Let $D' \subseteq \mathcal{D}$ be a set of $k^2 - n$ pairwise independent disks. As mentioned earlier, the disks in $D[x, y]$ all intersect each other, which would imply that there at most one disk for each $D[x, y]$. Look at $D[x, y]$ and if there is a disk $d[x, y]$ centered at $(x + \epsilon a[x, y], y + \epsilon b[x, y])$ for some $(a[x, y], b[x, y]) \in [n] \times [n]$, first $d[x, y] \in D[x, y]$ implies that $s[x, y] = (a[x, y], b[x, y]) \in S[x, y]$, select this pair to form the solution for the instance I.
If there is no $d[x, y]$ selected from $D[x, y]$ in D' select $\star$ for the $S[x, y]$ cell to form the solution for the instance I.
We claim that the solution formed this way satisfies the conditions of Matrix Tiling with $\leq$. For any a disk $d[x, y]$ centered at $(x + \epsilon a[x, y], y + \epsilon b[x, y])$, we have two cases:

1. The neighbor of $s[x, y] = (a[x, y], b[x, y]) \in S[x, y]$ (i.e. $S[x, y + 1]$) is star: In this case, as mentioned earlier we pick $\star$ for the $S[x, y + 1]$ for the solution of I instance, so we do not have to check the Matrix Tiling with $\leq$ condition of the pair selected for $s[x, y] = (a[x, y], b[x, y])$.

2. The neighbors $S[x + 1, y]$ and $S[x, y + 1]$ of $s[x, y] = (a[x, y], b[x, y]) \in S[x, y]$ are non-$\star$:
We know $S[x + 1, y], S[x, y]$ and $S[x, y + 1]$ are non-$\star$, because there are $d[x, y], d[x + 1, y]$ and $d[x, y + 1]$ is selected from $D[x, y], D[x + 1, y]$ and $D[x, y+1]$ in D'. As we have seen above, the fact that $d[x, y]$ and $d[x+1, y]$ do not intersect implies that $a[x, y] \leq a[x + 1, y]$. Similarly, the fact that $d[x, y]$ and $d[x, y + 1]$ do not intersect each other implies that $b[x, y] \leq b[x, y + 1]$.

Note: In the above proof for neighboring cases we only proved for the below neighbor ($D[x, y + 1]$), but all the remaining cases (right neighbor $D[x + 1, y]$, top neighbor $D[x, y - 1]$, and left neighbor $D[x - 1, y]$) can be proved in the similar fashion.
Thus the $s[x, y]$'s selected this way indeed form a solution for the instance I.
Here we can also conclude that if we construct the solution $c_A(S(y)) = k^2 - n$ of the instance I, from $c_B(y) = k^2 - m$ a solution of $\mathcal{D}$ this way, we have the following equality:

$$\begin{aligned} k^2 - n &= k^2 - m \\ \implies n &= m \end{aligned} \tag{4.2}$$

4.2.1 Relation between the optimal solutions and any approximate solutions of I and $\mathcal{D}$

We can notice that the optimum for I is always at least $\frac{k^2}{4}$: if i and j are both odd, then let $s_{i,j}$ be an arbitrary element of $S_{i,j}$; otherwise, let $s_{i,j} = \star$. And we have the upper bound on the optimum: k^2, which gives us the following inequality:

$$k^2/4 \leq OPT(x) \leq k^2 \tag{4.3}$$

For $\mathcal{D}$ the lower bound can be 1, because it cannot be intersected by anything. The upper bound is k^2 because as mentioned earlier, the disks in $D[x, y]$ all intersect each other, which would imply that there at most one disk for each $D[x, y]$, and if we have one disk for each $D[x, y]$ we will get k^2 disks therefore:

$$1 \leq OPT(R(x)) \leq k^2 \tag{4.4}$$

Now from the equation 4.3 and equation 4.4 equations, we can find the value of α for the 3^{rd} condition condition of the L-reduction:

$$\begin{aligned} &OPT(R(x)) \leq k^2 = 4k^2/4 = 4OPT(x) \\ &\implies OPT(R(x)) \leq 4OPT(x) \end{aligned} \tag{4.5}$$

Thus for $\alpha = 4$, we have $OPT(R(x)) \leq \alpha OPT(x)$.

4.2.2 Relation between the optimal solutions and any approximate solutions of I and $\mathcal{D}$.

Let

$$OPT(x) = k^2 - a,$$
$$OPT(R(x)) = k^2 - b,$$
$$c_B(y) = k^2 - m,$$
$$c_A(S(y)) = k^2 - n.$$

from equation 4.2 and equation 4.1 in follows:

$$\begin{aligned} n &= m \\ \implies n - a &= m - a \\ \implies n - a &= m - a \leq m - b \\ \implies n - a &\leq (1)(m - b). \end{aligned} \tag{4.6}$$

we now look at the 4^{th} condition condition of L-reduction:

$$\begin{aligned} &OPT(x) - c_A(S(y)) \leq (\beta)(OPT(R(x)) - c_B(y)) \\ &\implies (k^2 - a) - (k^2 - n) \leq (\beta)((k^2 - b) - (k^2 - m)) \\ &\implies (n - a) \leq (\beta)(m - b) \end{aligned} \tag{4.7}$$

from equation 4.6 and equation 4.7 we can get $\beta = 1$.
Thus for $\beta = 1$, we can satisfy: $|OPT(x) - c_A(S(y))| \leq \beta|OPT(R(x)) - c_B(y)|$.

Note: Because both the problems I and $\mathcal{D}$ are maximization optimization problems, we have $OPT(x) \geq c_A(S(y))$, and $OPT(R(x)) \geq c_B(y)$. So we can ignore the modulus used in the fourth condition in the L-reduction definition.

This completes L-reduction from Matrix Tiling with $\leq$ to Maximum Independent Set on Unit Disk Graph, where the values of α and β are 4 and 1 respectively. Providing the PTAS lower bound such that there are no $d, \delta > 0$ such that MAXIMUM INDEPENDENT SET ON UNIT DISK GRAPHS has PTAS with the running time $2^{O(1/\epsilon)^d} \cdot n^{O(1/\epsilon)^{1-\delta}}$ unless ETH fails.

CHAPTER 5

Future Work

For future work, one can look at existing exact reductions from Grid Tiling with $\leq$ and can try to get L-reduction, which would be from Matrix Tiling with $\leq$, and from Theorem 3 and Lemma 10 can derive the PTAS lower bound for the optimization version of these problems.

As a concluding note, my current plan is to apply the developed framework to geometric settings, in particular to the Covering Points with Squares problem. This approach is expected to yield a tight PTAS lower bound, thereby connecting the reductions studied here with a natural geometric optimization problem.

Bibliography

[1] S. Banerjee, R. Chitnis, and A. Lahiri. How does fairness affect the complexity of gerrymandering? In *22nd International Conference on Autonomous Agents and Multiagent Systems*, pages 2869–2871. Association for Computing Machinery (ACM), 2023.

[2] K. Bringmann, S. Kisfaludi-Bak, M. Pilipczuk, and E. J. van Leeuwen. On geometric set cover for orthants. In *27th Annual European Symposium on Algorithms, ESA 2019*, page 26. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2019.

[3] R. Chitnis. A tight lower bound for edge-disjoint. In *Algorithms and Complexity: 12th International Conference, CIAC 2021, Virtual Event, May 10–12, 2021, Proceedings*, volume 12701, page 187. Springer Nature, 2021.

[4] R. Chitnis and A. E. Feldmann. A tight lower bound for steiner orientation. In *International Computer Science Symposium in Russia*, pages 65–77. Springer, 2018.

[5] R. Chitnis, A. E. Feldmann, and P. Manurangsi. Parameterized approximation algorithms for bidirected steiner network problems. *ACM Transactions on Algorithms (TALG)*, 17(2):1–68, 2021.

[6] R. Chitnis and N. Saurabh. Tight lower bounds for approximate & exact k-center in $\mathbb{R}^d$. *arXiv preprint arXiv:2203.08328*, 2022.

[7] V. Cohen-Addad, A. De Mesmay, E. Rotenberg, and A. Roytman. The bane of low-dimensionality clustering. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 441–456. SIAM, 2018.

[8] V. Cohen-Addad, É. C. De Verdière, D. Marx, and A. De Mesmay. Almost tight lower bounds for hard cutting problems in embedded graphs. *Journal of the ACM (JACM)*, 68(4):1–26, 2021.

[9] M. Cygan, F. V. Fomin, Ł. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized algorithms*, volume 5. Springer, 2015.

[10] M. de Berg, H. L. Bodlaender, S. Kisfaludi-Bak, and S. Kolay. An eth-tight exact algorithm for euclidean tsp. *SIAM Journal on Computing*, 52(3):740–760, 2023.

[11] M. De Berg, H. L. Bodlaender, S. Kisfaludi-Bak, D. Marx, and T. C. Van Der Zanden. A framework for exponential-time-hypothesis–tight algorithms and lower bounds in geometric intersection graphs. *SIAM Journal on Computing*, 49(6):1291–1331, 2020.

[12] D. Marx. On the optimality of planar and geometric approximation schemes. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pages 338–348, 2007.

[13] D. Marx and M. Pilipczuk. Subexponential parameterized algorithms for graphs of polynomial growth. *arXiv preprint arXiv:1610.07778*, 2016.

[14] D. Marx and A. Sidiropoulos. The limited blessing of low dimensionality: when 1-1/d is the best possible exponent for d-dimensional geometric problems. In *Proceedings of the Thirtieth Annual Symposium on Computational Geometry*, pages 67–76, 2014.

[15] D. P. Williamson and D. B. Shmoys. *The design of approximation algorithms*. Cambridge University Press, 2011.