APPROXIMATION ALGORITHMS FOR MULTIDIMENSIONAL BIN PACKING

A Thesis
Presented to
The Academic Faculty

by

Arindam Khan

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in
Algorithms, Combinatorics, and Optimization

School of Computer Science
Georgia Institute of Technology
December 2015

APPROXIMATION ALGORITHMS FOR MULTIDIMENSIONAL BIN PACKING

Approved by:

Professor Prasad Tetali, Advisor
School of Mathematics and School of Computer Science
Georgia Institute of Technology

Professor Santosh Vempala
School of Computer Science
Georgia Institute of Technology

Professor Sebastian Pokutta
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

Professor Nikhil Bansal (*Reader*)
Department of Mathematics and Computer Science
Eindhoven University of Technology, Eindhoven, Netherlands

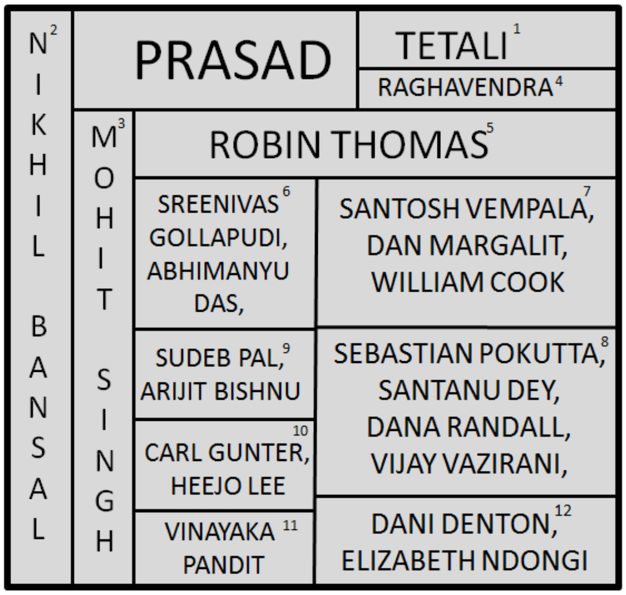
Professor Dana Randall
School of Computer Science
Georgia Institute of Technology

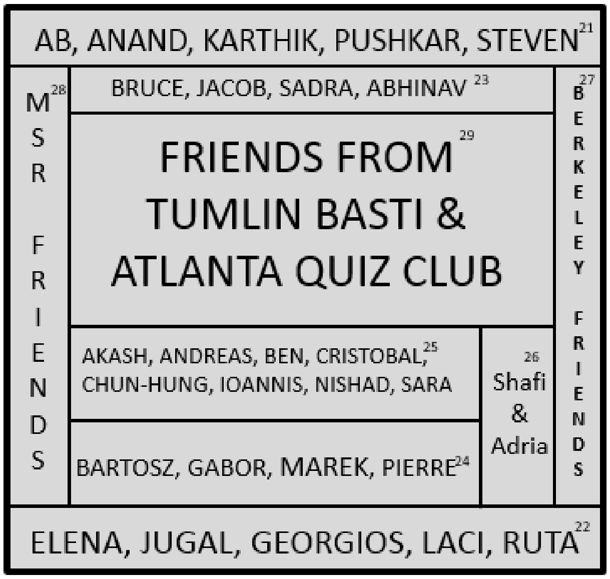
Professor Santanu Dey
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

Date Approved: 10 August 2015

To Maa, Baba and Bhai

ACKNOWLEDGEMENTS


NIKHIL BANSAL[2]
PRASAD
TETALI[1]
RAGHAVENDRA[4]
MOHIT SINGH[3]
ROBIN THOMAS[5]
SREENIVAS[6] GOLLAPUDI, ABHIMANYU DAS,
SANTOSH VEMPALA,[7] DAN MARGALIT, WILLIAM COOK
SUDEB PAL,[9] ARIJIT BISHNU
SEBASTIAN POKUTTA,[8] SANTANU DEY, DANA RANDALL, VIJAY VAZIRANI,
[10] CARL GUNTER, HEEJO LEE
VINAYAKA[11] PANDIT
DANI DENTON,[12] ELIZABETH NDONGI


AB, ANAND, KARTHIK, PUSHKAR, STEVEN[21]
MSR FRIENDS[28]
BRUCE, JACOB, SADRA, ABHINAV[23]
BERKELEY FRIENDS[27]
FRIENDS FROM[29] TUMLIN BASTI & ATLANTA QUIZ CLUB
AKASH, ANDREAS, BEN, CRISTOBAL,[25] CHUN-HUNG, IOANNIS, NISHAD, SARA
[26] Shafi & Adria
BARTOSZ, GABOR, MAREK, PIERRE[24]
ELENA, JUGAL, GEORGIOS, LACI, RUTA[22]

(1) First of all, I would like to thank my advisor Prasad Tetali, for his constant support, encouragement and freedom that he gave throughout my stay at Georgia Tech. His ideas and personality were a great source of inspiration for me.

(2) I am also greatly indebted to Nikhil Bansal. He has been like a second advisor to me. A major part of this thesis has its origin in the discussions at TU Eindhoven during my two visits.

(3) I deeply thank Mohit Singh for being an amazing mentor during my internship at Theory group at Microsoft Research, Redmond. I have learnt a lot from him on academic writing and presentation.

(4) For hosting me during a wonderful time at Simons Institute, Berkeley.

(5) My sincere gratitude to Robin Thomas and ACO program for their support and encouragement.

(6) For introducing me to Bay Area and MSR culture.

(7) For teaching me amazing courses and inspiring.

(8) For many helpful discussions, collaboration and encouragement.

(9) For motivating me to pursue graduate studies.

(10) For giving me research opportunities during my UG studies.

(11) For introducing me to the bin packing problem.

(12) For being amazing real-world problem-solvers.

(21) For *Trashball Champions*!

(22) Deserve kudos for teaching, friendship and collaboration.

(23) For making the theory lab *awesome*.

(24) For collaboration and many interesting discussions.

(25) For being incredible friends.

(26) For Sunday morning discussions and indelible friendship.

(27)(28)(29) For all the amazing and memorable times.

- A special thanks to Saswati for her love and support. I deeply thank *Didima* and *Mejoma* for their blessings. Finally, I would like to thank my parents and brother for their unconditional love, faith and encouragement.

Contents

List of Tables

List of Figures

SUMMARY

The bin packing problem has been the corner stone of approximation algorithms and has been extensively studied starting from the early seventies. In the classical *bin packing problem*, we are given a list of real numbers in the range $(0, 1]$, the goal is to place them in a minimum number of *bins* so that no bin holds numbers summing to more than 1. In this thesis we study approximation algorithms for three generalizations of bin packing: geometric bin packing, vector bin packing and weighted bipartite edge coloring.

In two-dimensional (2-D) *geometric bin packing*, we are given a collection of rectangular items to be packed into a minimum number of unit size square bins. Geometric packing has vast applications in cutting stock, vehicle loading, pallet packing, memory allocation and several other logistics and robotics related problems. We consider the widely studied *orthogonal packing* case, where the items must be placed in the bin such that their sides are parallel to the sides of the bin. Here two variants are usually studied, (i) where the items cannot be rotated, and (ii) they can be rotated by 90 degrees. We give a polynomial time algorithm with an asymptotic approximation ratio of $\ln(1.5) + 1 \approx 1.405$ for the versions with and without rotations. We have also shown the limitations of rounding based algorithms, ubiquitous in bin packing algorithms. We have shown that any algorithm that rounds at least one side of each large item to some number in a constant size collection values chosen independent of the problem instance, cannot achieve an asymptotic approximation ratio better than 3/2.

In d-dimensional *vector bin packing* (VBP), each item is a d-dimensional vector that needs to be packed into unit vector bins. The problem is of great significance in resource constrained scheduling and also appears in recent virtual machine placement in cloud computing. Even in two dimensions, it has novel applications in layout design, logistics, loading and scheduling problems. We obtain a polynomial time algorithm with an asymptotic approximation ratio of $\ln(1.5) + 1 \approx 1.405$ for 2-D VBP. We also obtain a polynomial time algorithm with almost tight (absolute) approximation ratio of $1 + \ln(1.5)$ for 2-D VBP. For d dimensions, we give a polynomial time algorithm with an asymptotic approximation ratio of $\ln(d/2) + 1.5 \approx \ln d + 0.81$. We also consider vector bin packing under resource augmentation. We give a polynomial time algorithm that packs vectors into $(1 + \epsilon)\mathsf{Opt}$ bins when we allow augmentation in $(d - 1)$ dimensions and Opt is the minimum number of bins needed to pack the vectors into $(1, 1)$ bins.

In *weighted bipartite edge coloring* problem, we are given an edge-weighted bipartite graph $G = (V, E)$ with weights $w : E \to [0, 1]$. The task is to find a *proper weighted coloring* of the edges with as few colors as possible. An edge coloring of the weighted graph is called a *proper weighted coloring* if the sum of the weights of the edges incident to a vertex of any color is at most one. This problem is motivated by rearrangeability of 3-stage Clos networks which is very useful in various applications in interconnected networks and routing. We show a polynomial time approximation algorithm that returns a proper weighted coloring with at most $\lceil 2.2223m \rceil$ colors where m is the minimum number of unit sized bins needed to pack the weight of all edges incident at any vertex. We also show that if all edge weights are $> 1/4$ then $\lceil 2.2m \rceil$ colors are sufficient.

Chapter I

INTRODUCTION

The bin packing problem has been the corner stone of approximation algorithms and has been extensively studied starting from the early seventies. In the classical *bin packing problem*, we are given a list $I = \{i_1, i_2, \ldots, i_n\}$ of real numbers in the range $(0, 1]$, the goal is to place them in a minimum number of *bins* so that no bin holds numbers summing to more than 1.

Bin packing is a special case of the one-dimensional *cutting stock* problem [88], loading problem [66] and several scheduling related problems [44]. In theoretical computer science, the bin packing problem was probably first studied by Garey, Graham and Ullman in 1972 [84], from the standpoint of memory allocation problems such as table formatting, prepaging and file allocation. They noticed that finding a general placement algorithm for attaining the minimum number of bins appears to be impractical, and thus provided four heuristics: first fit (FF), best fit (BF), first fit decreasing height (FFDH) and best fit decreasing heights (BFDH). Soon Johnson, Demers, Ullman, Garey and Graham [128] published the first definitive analysis of the worst case guarantees of several bin packing approximation algorithms. The problem is well-known to be NP-hard [86] and the seminal work of Johnson et al. initiated an extremely rich research area in approximation algorithms [107]. In fact the term *approximation algorithm* was coined by David S. Johnson [127] in an influential and prescient paper in 1974 where he studied algorithms for bin packing and other packing and covering related optimization problems.

Bin packing is extremely useful in practice and has a lot of applications in various fields. Skiena [191] has presented *market research* for the field of combinatorial optimization and algorithms, attempting to determine which algorithmic problems are most in demand for applications, by studying WWW traffic. Both bin packing and related knapsack problem were among top five most popular NP-hard problems. The implementations of bin packing and knapsack were the most needed among all NP-hard problems, even more than problems such as set-cover, traveling salesman and graph-coloring.

Garey and Johnson [87], followed by Coffman et al. [43], gave comprehensive surveys on bin packing algorithms. Coffman and Lueker also covered probabilistic analyses of packing algorithms in detail [41]. Galambos and Woeginger [82] gave an overview restricted mainly to online variants of bin packing problems. There had been many surveys on bin packing problems thereafter [93, 40, 52]. The most recent, extensive coverage on 1-D bin packing was given by Coffman et al. [42].

In this thesis, we primarily focus on packing in higher dimensions due to its prominence in many real world applications. These generalizations of bin packing also help us to understand the power and limitations of existing algorithmic techniques. An ambitious goal is to translate insights from these classical problems to general results for other related combinatorial optimization problems.

We primarily consider three generalizations of bin packing: GEOMETRIC BIN PACKING, VECTOR BIN PACKING and WEIGHTED BIPARTITE EDGE COLORING.

In two-dimensional (2-D) *geometric bin packing* (GBP), we are given a collection of rectangular items to be packed into a minimum number of unit-size square bins. This variant and other higher dimensional GBP variants have vast applications in cutting stock, vehicle loading, pallet packing, memory allocation and several other logistics and robotics related problems [88, 175]. In two dimensions, packing objects into containers have many important applications, e.g., in the context of cutting out a given set of patterns from a given large piece of material minimizing waste, typically in sheet metal processing and apparel fabrication. In three dimensions, these problems are frequently encountered in minimizing storage space or container space for transportation. In this thesis we consider the widely studied *orthogonal packing* case, where the items must be placed in the bin such that their sides are parallel to the sides of the bin. In any feasible solution, items are not allowed to overlap. Here two variants are usually studied, (i) where the items cannot be rotated (packing by *translations*), and (ii) they can be rotated by 90 degrees (packing by *restricted rigid* motions). These variants are also recurrent in practice, e.g., in apparel production usually there are patterns of weaving or texture on the material so that the position where a piece should be cut cannot be rotated arbitrarily.

In d-dimensional *vector bin packing* (VBP), each item is a d-dimensional vector that needs to be packed into unit vector bins. The problem is of great significance in resource constrained scheduling and appeared also recently in virtual machine placement in cloud computing [169]. For example, consider each job (item) has multiple resource requirements (dimensions) such as CPU, memory, I/O, disk, network etc. and each server (bin) has a bounded amount of these resources. The goal to assign all jobs to minimum number of servers, without violating the resource constraints, translates to the vector packing problem. Even in two dimensions, vector packing has many novel applications in layout design, logistics, loading and scheduling problems [180, 193].

In the *weighted bipartite edge coloring* problem, we are given an edge-weighted bipartite graph $G = (V, E)$ with weights $w : E \to [0, 1]$. The task is to find a *proper weighted coloring* of the edges with as few colors as possible. An edge coloring of the weighted graph is called a *proper weighted coloring* if the sum of the weights of the edges incident to a vertex of any color is at most one. This problem is motivated by rearrangeability of 3-stage Clos networks which is very useful in various applications in interconnected networks and routing [126, 113]. This problem is a generalization of two classical optimization problems: bin packing and bipartite edge coloring problem.

These generalizations have been well studied since the 1970s. Baker, Coffman, and Rivest first considered orthogonal packings in two dimensions [9]. At the same time Coffman et al. [129] gave performance bounds for level-oriented two-dimensional packing algorithms such as Next Fit Decreasing Height and First Fit Decreasing Height. Lodi, Martello and Monaci first gave a survey on two-dimensional packing problems [152]. Epstein and van Stee gave a survey in [93] on multi-dimensional bin packing. There has been consistent progress in the area since then. We will provide a detailed survey of prior works in the later corresponding chapters.

1.1 Contribution of the Thesis

The dissertation obtains improved approximation algorithms for the three generalizations of bin packing problems, mentioned above. We summarize the contributions below.

Geometric Bin Packing: We give a polynomial time algorithm with an asymptotic approximation ratio of $\ln(1.5) + 1 \approx 1.405$ for 2-D GBP. This holds both for the versions with and without rotations.

The main idea behind this result is to show that the *Round and Approx* (R&A) framework introduced by Bansal, Caprara and Sviridenko [13] (See section 2.6.3) can be applied to a recent $(1.5 + \epsilon)$-approximation result of Jansen and Prädel [116]. Roughly speaking, this framework states that, given a packing problem, if (i) the configuration LP for the problem (with the original item sizes) can be solved up to error $1 + \epsilon$ for any $\epsilon > 0$, and (ii) there is a ρ-approximation for the problem that is *subset-oblivious* (See section 2.6.3 for a formal description); then one can obtain a $(1 + \ln \rho)$-asymptotic approximation for the problem.

In [13], it was shown that the APTAS for 1-D BP due to [55] and the 2-D BP algorithm of [27] are subset-oblivious. However, the notion of subset-obliviousness as defined in [13], is based on various properties of dual-weighting functions, making it somewhat tedious to apply and also limited in scope (e.g. it is not clear to us how to apply this method directly to the algorithm of [116]).

We give a more general argument to apply the R&A framework directly to a wide class of algorithms[1], and without any reference to dual-weighting functions. In particular, we show that *any* algorithm based on rounding the (large) items into $O(1)$ types, is subset-oblivious. The main observation is that any ρ-approximation based on rounding the item sizes, can be related to another configuration LP (on rounded item sizes) whose solution is no worse than ρ times the optimum solution. As the item sizes are rounded, there are only $O(1)$ constraints in this LP and it can be easily shown to be subset-oblivious.

For the particular case of 2-D BP, we present the algorithm of Jansen and Prädel that directly fits in the above framework. As most algorithms for bin-packing problems are based on rounding into $O(1)$ types, this also makes the framework widely applicable. For example, this gives much simpler proofs of all the results in [13].

Finally, we give some results to show the limitations of rounding based algorithms in obtaining better approximation ratios. Rounding of items to $O(1)$ types has been used either implicitly [18] or explicitly [55, 133, 27, 116, 132], in almost all bin packing algorithms. There are typically two types of rounding: either the size of an item in some coordinate (such as width or height) is rounded up in an instance-oblivious way (e.g. Harmonic rounding [147, 27], or Geometric rounding [133]), or it is rounded up in an input sensitive way (e.g. linear grouping [55]). We show that any rounding based algorithm that rounds at least one side of each large item to some number in a constant-size collection values chosen independent of problem instance (let us call such rounding *input-agnostic*), can not have an approximation ratio better than 3/2.

These results are based on joint work with Nikhil Bansal.

[1] This includes all known algorithms that we know of for bin-packing type problems, except the ones based on R&A method.

Vector Bin Packing: Our main result is improved approximation for multidimensional vector packing. We first give a polynomial time algorithm with an asymptotic approximation ratio of $(1 + \ln(1.5) + \epsilon) \approx (1.405 + \epsilon)$ for 2-D vector packing and a $\ln d + 0.807 + o_d(1) + \epsilon$-approximation for d-dimensional vector packing. For 2-D this already gives significant improvement over the current best $(1+\ln(2)+\epsilon) \approx (1.693+\epsilon)$ result [13], but more importantly, it overcomes a natural barrier of $(1 + \ln d)$ of R&A framework due to the fact that one can not obtain better than d-approximation using rounding based algorithms. We circumvent this problem based on two ideas.

First, we show a structural property of vector packing that any optimal packing of m bins can be transformed into nearly $\lceil \frac{3m}{2} \rceil$ bins of two types:

1. Either a bin contains at most two big items, or

2. The bin has slack in one dimension (i.e., the sum of all vectors in the bin is at most $1 - \delta$ for some constant δ). We then search (approximately) over the space of such "well-structured" 1.5-approximate solutions. However, as this structured solution (necessarily) uses unrounded item sizes, it is unclear how to search over the space of such solutions efficiently. So a key idea is to define this structure carefully based on matchings, and use an elegant recent algorithm for the multiobjective-multibudget matching problem by Chekuri, Vondrák, and Zenklusen [35]. As we show, this allows us to both use unrounded sizes and yet enumerate the space of solutions like in rounding-based algorithms.

The second step is to apply the subset oblivious framework to the above algorithm. There are two problems. First, the algorithm is not rounding-based. Second, even proving subset obliviousness for rounding based algorithms for vector packing is more involved than for geometric bin-packing. To get around these issues, we use additional technical observations about the structure of d-dimensional VBP.

Another consequence of the these techniques is the following tight (absolute) approximation guarantee. We show that for any small constant $\epsilon > 0$, there is a polynomial time algorithm with an absolute approximation ratio of $(1.5 + \epsilon)$ for 2-D vector packing, improving upon the guarantee of 2 by Kellerer and Kotov [136].

We extend the approach for $d = 2$ to give a $(d+1)/2$ approximation (for $d = 2$, this is precisely the $3/2$ bound mentioned above) and then show how to incorporate it into R&A. However, applying the R&A framework is more challenging here and instead of the ideal $1+\ln((d+1)/2)$, we get a $(1.5+\ln(d/2)+o_d(1)+\epsilon) \approx (\ln d+0.807+o_d(1)+\epsilon)$-approximation.

Along the way, we also prove several additional results which could be of independent interest. For example, in Section 4.5 we obtain several results related to resource augmented packing which has been studied for other variants of bin packing [122, 19].

These results are based on joint work with Nikhil Bansal and Marek Elias.

Weighted Bipartite Edge Coloring: We show a polynomial time approximation algorithm that returns a proper weighted coloring with at most $\lceil 2.2223m \rceil$ colors where m is the minimum number of unit-sized bins needed to pack the weight of all edges incident at any vertex.

This makes progress towards the resolution of the conjecture [38] that there is always a proper weighted coloring using at most $2m - 1$ colors. In our algorithm and analysis, we exploit that WEIGHTED BIPARTITE EDGE COLORING problem displays features of the classical edge coloring problem as well as the bin packing problem. Our algorithm starts by decomposing the *heavy* weight edges into matchings by applying König's theorem [143] to find an edge coloring of the subgraph induced by these edges. For the light weight edges, we employ the *first-fit decreasing* heuristic where we consider the remaining edges in decreasing order of weight and give them the first available color.

Our work diverges from previous results on this problem in the analysis of this simple combinatorial algorithm. We employ strong mathematical formulations for the bin packing problem; in particular, we use the *configuration linear program (LP)* for the bin packing problem. This linear program has been used to design the best approximation algorithm for the bin packing problem [178, 106]. In our work, we use it as follows. We show that if the algorithm is not able to a color an edge (u, v), then the edges incident at u or v cannot be packed in m bins as promised. To show this, we formulate the configuration linear program for the two bin packing problems, one induced by edges incident at u and the other induced by edges incident at v. We then construct feasible dual solutions to these linear programs showing that the optimal primal value, and therefore the optimal bin packing number, is more than m for at least one of the programs, giving us the desired contradiction. While the weights on the edges incident at u (or v) can be arbitrary reals between 0 and 1, we group the items according to weight classes and how our algorithm colors these edges. This allows us to reduce the number of item types, reducing the complexity of the configuration LP and makes it easier to analyze. While the grouping according to weight classes is natural in bin packing algorithms; the grouping based on the output of our algorithm helps us relate the fact that the edge (u, v) could not be colored by our algorithm to the bin packing bound at u and v. Our analysis can also be extended to show $\lceil 2.2m \rceil$ colors are sufficient when all edge weights are $> 1/4$. We also give an alternate proof of König's Theorem using the skew-supermodular theorem (See Section 5.2), which might be of independent interest. Our techniques might be useful in the analysis of other algorithms related to Clos networks.

These results are based on joint work with Mohit Singh.

1.2 Organization of the Thesis

In Chapter 2 we discuss related definitions and techniques for approximation algorithms and bin packing. In Chapter 3 we discuss geometric bin packing. In Chapter 4 we cover vector bin packing. In Chapter 5 we cover weighted bipartite edge coloring. We conclude in Chapter 6 with a list of open problems.

Chapter II

PRELIMINARIES

In this chapter we introduce relevant notation and definitions required to define, analyze and classify bin packing related problems. Additional definitions will be introduced later on as required.

2.1 Combinatorial Optimization and Complexity

Combinatorial optimization problems are ubiquitous in theory and practice. In a combinatorial optimization problem, the goal is to find a solution that maximizes or minimizes a certain objective value amidst a discrete set of feasible solutions. Edmonds in his seminal paper [65], advocated that an algorithm is *efficient* if the number of atomic operations the algorithm takes to return a solution is polynomial in the size of the problem instance. In many cases, a brute force search might take exponential or superpolynomial time in the problem size, and hence, is not considered efficient. Depending on this time complexity, the problems are naturally classified into *simple* and *hard* problems. Cook [45] formalized this classification defining the complexity classes P (the set of languages that can be recognized by Turing machines in deterministic polynomial time) and NP (the set of languages that can be recognized by Turing machines in nondeterministic polynomial time), and the notion of NP-completeness (A decision problem Π is NP-complete or in NPC, if $\Pi \in \mathsf{NP}$ and every problem in NP is reducible to Π in polynomial time). Karp's seminal work [134] established the pervasive nature of NP-completeness by showing a vast majority of problems in combinatorial optimization are NP-complete. These NP-complete problems do not admit a polynomial time algorithm unless $\mathsf{P} = \mathsf{NP}$. For a detailed introduction to complexity classes, we refer the readers to the book by Arora and Barak [5].

Notion of NP and NP-completeness is defined in terms of decision problems. In this thesis we deal with NP optimization problems.

Definition 2.1.1. An NP optimization problem Π is a four-tuple $(\mathcal{I}, \mathcal{S}, obj, optimize)$ such that:

- $\mathcal{I}$ is the set of input instances of Π and is recognizable in polynomial time.
- For any instance $I \in \mathcal{I}$, $\mathcal{S}(I)$ is the set of feasible solutions for $\mathcal{I}$. Solutions for an instance I are constrained to be polynomially bounded in $|I|$, the size of I; i.e., there exists a polynomial *poly* such that $j \in \mathcal{S}(I)$ implies that $|j| \leq poly(|I|)$. Further, there should exist a polynomial time computable boolean function f such that $f(I, j)$ is true if and only if $j \in \mathcal{S}(I)$.
- For each instance I, the objective function *obj* assigns a positive value to each solution $j \in \mathcal{S}(I)$ and this function is polynomial time computable.
- The parameter *optimize* $\in \{max, min\}$ specifies whether the objective function should be minimized or maximized.

An NP optimization problem $\mathscr{P}$ is NP-hard if there is a polynomial time algorithm $\mathcal{A}$ for a problem $\mathscr{Q} \in \mathsf{NPC}$, given $\mathcal{A}$ has an oracle access to problem $\mathscr{P}$. In general, completeness and hardness are natural notions associated with complexity classes. A problem is *complete* for a class if it is a member of the class and all other problems in the class reduce to it under an appropriate notion of reducibility. A problem is *hard* for a complexity class if all problems in the class reduce to it (it need not be a member of the class). Now let us define some other closely related terms.

Definition 2.1.2. Pseudo-polynomial time algorithm : An algorithm runs in pseudo-polynomial time if its running time is polynomial in the numeric value of the input (i.e., polynomial in the size of input if the numeric data are encoded in unary), but is exponential in the length of the input if the numeric data are encoded in binary.

Definition 2.1.3. Weakly NP-complete : An NP-complete problem $\mathscr{F}$ called *weakly* NP*-complete* or *binary* NP*-complete* if it has a pseudo-polynomial time algorithm.

Definition 2.1.4. Strongly NP-complete : An NP-complete problem $\mathscr{F}$ is called *strongly* NP*-complete* or *unary* NP*-complete* if it is proven that it cannot be solved by a pseudo-polynomial time algorithm unless $\mathsf{P} = \mathsf{NP}$. These problems are NP-complete even when their numeric data are encoded in unary.

The strong/weak kinds of NP-hardness are defined analogously.

Definition 2.1.5. Quasi-polynomial time algorithm : Quasi-polynomial time algorithms are algorithms which run slower than polynomial time, yet not so slow as to be exponential time. The worst case running time of a quasi-polynomial time algorithm is $2^{O((\log n)^c)}$ for some fixed $c > 1$.

The complexity class QP consists of all problems which have quasi-polynomial time algorithms. It can be defined in terms of DTIME as follows. $\mathsf{QP} = \bigcup_{c \in \mathbb{N}} \mathsf{DTIME}(2^{O(\log n)^c})$. *Exponential time hypothesis* implies that NP-complete problems do not have quasi-polynomial time algorithms.

We have already seen in the previous section that several variants of bin packing problem are indispensable in many practical applications. These problems are strongly NP-hard [85] and do not even admit a pseudo-polynomial time algorithm unless $\mathsf{P} = \mathsf{NP}$. Thus it is imperative to develop heuristics to cope with their intractability. A large body of work has emerged to deal with intractability by exploiting additional structures:

- Parameterized Complexity: Here the goal is to design algorithms that are efficient on inputs where the associated parameter is small (For details, see [62]).
- Efficient algorithms for problems over special classes of problem instances such as problems restricted to bipartite, planar or bounded tree-width graphs (for graph problems) or constant number of item types (for packing problems).
- Algorithms that are guaranteed to perform well with very high probability when the input data is coming from a certain distribution.

However in many real-life scenarios, the inputs are generated from complex processes that make discovering additional structure in them a formidable task. In this thesis we restrict ourselves to approximation algorithms that have provable worst-case guarantees even when there is no or little additional information available about the inputs. We will mention few other related practical heuristics in the relevant chapters.

2.2 Approximation Algorithms and Inapproximability

Approximation Algorithm is an attempt to systematically measure, analyze, compare and improve the performance of heuristics for intractable problems. It gives theoretical insight on how to find fast solutions for practical problems, provides mathematical rigor to study and analyze heuristics, and also gives a metric for the difficulty of different discrete optimization problems.

Definition 2.2.1. Approximation ratio : Given an algorithm $\mathcal{A}$ for a *minimization* problem Π, the (multiplicative) *approximation ratio* is:

$$\rho_{\mathcal{A}} = sup_{I \in \mathcal{I}} \left\{ \frac{\mathcal{A}(I)}{\mathsf{Opt}(I)} \right\},$$

where $\mathcal{A}(I)$ is the value of the solution returned by algorithm $\mathcal{A}$ on instance $I \in \mathcal{I}$ and $\mathsf{Opt}(I)$ is the value of the corresponding optimal solution.

In other words, an algorithm $\mathcal{A}$ for a minimization problem Π is called a ρ-approximation algorithm if $\mathcal{A}(I) \leq \rho \cdot \mathsf{Opt}(I)$ holds for every instance I of Π. An algorithm $\mathcal{A}$ for a maximization problem Π is called a ρ-approximation algorithm if $\mathcal{A}(I) \geq \frac{1}{\rho} \cdot \mathsf{Opt}(I)$ holds for every instance I of Π. This asymmetry ensures that $\rho \geq 1$ for all approximation algorithms.

In some cases, quality of the heuristic is measured in terms of additive approximation. In other words, an algorithm $\mathcal{A}$ for a minimization problem Π is called a σ-additive approximation algorithm if $\mathcal{A}(I) \leq \mathsf{Opt}(I) + \sigma$ holds for every instance I of Π. Additive approximation algorithms are relatively rare. Karmarkar-Karp's algorithm [133] for one-dimensional bin packing is one such example.

For detailed introduction to approximation algorithms, we refer the readers to the books on approximation algorithms [201, 204].

Definition 2.2.2. Polynomial time approximation scheme (PTAS) : A problem is said to admit a *polynomial time approximation scheme* (PTAS) if for every constant $\epsilon > 0$, there is a *poly*(n)-time algorithm with approximation ratio $(1 + \epsilon)$ where n is the size of the input. Here running time can be as bad as $O(n^{f(1/\epsilon})$ for any function f that depends only on ϵ.

If the running time of PTAS is $O(f(1/\epsilon) \cdot n^c)$ for some function f and a constant c that is independent of ϵ, we call it to be an *efficient polynomial time approximation scheme* (EPTAS).

On the other hand, if the running time of PTAS is polynomial in both n and $1/\epsilon$, it is said to be a *fully polynomial time approximation scheme* (FPTAS).

Assuming $\mathsf{P} \neq \mathsf{NP}$, a PTAS is the best result we can obtain for a strongly NP-hard problem. Already in the 1D case, a simple reduction from the Partition problem shows that it is NP-hard to determine whether a set of items can be packed into two bins or not, implying that no approximation better than $3/2$ is possible. However, this does not rule out the possibility of an $\mathsf{Opt} + 1$ guarantee where Opt is the number of bins required in the optimal packing. Hence it is insightful to consider the *asymptotic approximation ratio.*

Definition 2.2.3. Asymptotic approximation ratio (AAR) **:** The *asymptotic approximation ratio* of an algorithm $\mathcal{A}$ is ρ if the output of the algorithm has value at most $\rho \cdot \mathsf{Opt}(I) + \delta$ for some constant δ, for each instance I.

In this context the approximation ratio defined as in Definition 2.2.1, is also called to be the (absolute) approximation ratio. If $\delta = 0$, then $\mathcal{A}$ has (absolute) approximation guarantee ρ.

Definition 2.2.4. Asymptotic PTAS (APTAS) **:** A problem is said to admit an *asymptotic polynomial time approximation scheme* (APTAS) if for every $\epsilon > 0$, there is a poly-time algorithm with asymptotic approximation ratio of $(1 + \epsilon)$.

If the running time of APTAS is polynomial in both n and $1/\epsilon$, it is said to be an *asymptotic fully polynomial time approximation scheme* (AFPTAS).

Note that NP optimization problems whose decision versions are all polynomial time reducible to each other (due to NP-completeness), behave very differently in their approximability. For example classical bin packing problem admits an APTAS, whereas no polynomial factor approximation is known for the traveling salesman problem. This anomaly is due to the fact that reductions between NP-complete problems preserve polynomial time computability, but not the quality of the approximate solution.

PTAS is the class of problems that admit polynomial time approximation scheme. On the other hand, APX is the class of problems that have a constant-factor approximation. Clearly $\mathsf{PTAS} \subseteq \mathsf{APX}$. In fact the containment is strict unless $\mathsf{P} = \mathsf{NP}$.

Theorem 2.2.5. [48] If a problem $\mathscr{F}$ is APX-hard then it does not admit PTAS unless $\mathsf{P} = \mathsf{NP}$.

Online Algorithms: Bin packing is also one of the key problems in online algorithms. Let us define the notion of a competitive ratio which will be useful when we discuss some related results in online algorithms in later chapters.

Definition 2.2.6. Competitive Ratio : *Competitive ratio* of an online algorithm $\mathcal{A}$ is the worst case ratio of the solution returned by $\mathcal{A}$ and the best offline algorithm.

Asymptotic competitive ratio is defined analogously.

There are few others metrics to measure the quality of a packing, such as *random-order ratio* [137], *accommodation function* [25], *relative worst-order ratio* [24], *differential approximation measure* [56] etc.

2.3 Relaxation and Rounding

A plethora of approximation algorithms follow a two-step approach. It considers a *relaxation* of the original problem, followed by a *rounding* of the solution of the relaxation. Especially, linear program (LP) and semidefinite program (SDP) relaxations have been instrumental in the design of approximation algorithms and have led to the development of several algorithmic techniques such as deterministic rounding, randomized rounding, primal-dual methods, dual-fitting, iterative methods, entropy-based rounding etc.

2.3.1 Relaxation

The space of feasible solutions of a combinatorial problem is discrete and hence, every combinatorial problem can be reformulated as an optimization problem with integral variables, i.e., an integer program (IP). Therefore, given an instance I of a combinatorial optimization problem Π, we can encode it as maximizing or minimizing a function of a set of variables (say $\{z_1, z_2, \cdots, z_n\}$) that take certain integer values (say $\{0, 1\}$ or $\{+1, -1\}$) and are required to satisfy a set of constraints specific to the problem. As a polynomial time reformulation, the resulting integer program is also NP-hard. By a suitable relaxation, intractable IP is converted to a convex optimization problem that can be solved in polynomial time. Specifically, the relaxation allows the variables to be assigned real numbers or even vectors, instead of just integer values. This is called a *relaxation* as the relaxation permits more solutions than the original program does. Thus if $\mathsf{Opt}(I)$ is the optimal value of the problem with instance I and $\mathsf{Conv}(I)$ is the optimal value of the relaxation, then $\mathsf{Conv}(I) \leq \mathsf{Opt}(I)$. However it is not clear, how good an estimate $\mathsf{Conv}(I)$ is. *Integrality gap* is a coarse measure of this quality.

Definition 2.3.1. Integrality Gap: The integrality gap is the worst case ratio between $\mathsf{Opt}(I)$ and $\mathsf{Conv}(I)$ over all instances I.

The hard instances for a relaxation, where worst-case ratio is obtained, are called *integrality gap instances.*

2.3.2 Relaxation techniques: LP and SDP

Linear Programs: A vast majority of approximation algorithms use a specific type of convex relaxation - *linear programming* (LP). A linear program consists of optimizing a linear function over real-valued variables while satisfying certain linear constraints among them. Simplex method and its variants [53] are extensively used to solve linear programs in practice, whereas interior point methods [164] are provably efficient. In this thesis we will use linear programs and their duals extensively.

First let us define primal and dual linear programs. Consider the following minimization problem. We call this to be the *primal* problem.

$$\begin{array}{lrcll} \textit{minimize} & \displaystyle\sum_{j=1}^{n} c_j x_j & & & \\ \textit{subject} \quad \textit{to}: & \displaystyle\sum_{j=1}^{n} a_{ij} x_j & \geq & b_i, & i \in [m] \\ & x_j & \geq & 0, & j \in [n]. \end{array}$$

We get the corresponding dual problem by introducing variable y_i for the i'th inequality.

$$\begin{array}{llcll} \textit{maximize} & \sum_{i=1}^{m} b_i y_i & & & \\ \textit{subject} \quad \textit{to}: & \sum_{i=1}^{m} a_{ij} y_i & \leq & c_j, & j \in [n] \\ & y_i & \geq & 0, & i \in [m]. \end{array}$$

Theorem 2.3.2. LP-Duality Theorem : The primal program has finite optimum if and only if its dual has finite optimum. Moreover if $\mathbf{x}^* := (x_1^*, x_2^*, \cdots, x_n^*)$ and $\mathbf{y}^* := (y_1^*, y_2^*, \cdots, y_m^*)$ are optimal solutions for the primal and dual programs respectively, then

$$\sum_{j=1}^{n} c_j x_j^* = \sum_{i=1}^{m} b_i y_i^*.$$

Linear programs can be solved in polynomial time even if it has exponentially many constraints provided a polynomial time *separation oracle* can be constructed, i.e., a polynomial time algorithm that given a point in $\mathbb{R}^n$ (where n is the number of variables in the relaxation) either confirms the point is a feasible solution satisfying all constraints or produces a violated constraint. See [96, 95] for more details on the algorithms to find a solution to an LP, given a separation oracle.

Semidefinite Programs: Now we briefly mention semidefinite program (SDP) relaxations where we allow the variables to be vectors with linear constraints on their inner products. SDPs can be optimized within an error ϵ in time polynomial in $\ln \frac{1}{\epsilon}$ and the size of the program. Seminal work of Goemans and Williamson [91] obtained 0.878 approximation for MAX-CUT whereas linear programs can not yield better than factor 1/2 [31]. SDPs have been helpful in obtaining improved approximations for constraint satisfaction problems (CSP), vertex ordering, sparsest cut, graph decomposition etc. We refer readers to [174] for more related references. Several increasingly stronger relaxations such as Lovász-Schrijver, Lasserre and Sherali-Adams hierarchies [146] are also used to strengthen a convex relaxation. However it is not clear if we can use SDPs or hierarchies to get better approximation for variants of bin packing problems.

2.3.3 Rounding of solution of relaxation

For some relaxations, the optimal solution is always integral. Total Dual Integrality and Total unimodularity [182] were developed as general techniques to show the integrality of linear programming formulations. However in general, the optimal solution to the relaxation might not be integral. Thus rounding scheme is devised to *round* the real (for LP) or vector (for SDP) valued solution to an integral solution incurring little loss in the objective value, say at most losing an α-factor in the value. As $\mathsf{Conv}(I) \leq \mathsf{Opt}(I)$, the value of the rounded solution is at most α-times the value of the optimal solution.

2.4 One Dimensional Bin Packing

Before going to multidimensional bin packing, we give a brief description of the results in 1-D bin packing. We also focus primarily on very recent results. For a detailed survey and earlier results we refer the interested reader to [42].

2.4.1 Offline 1-D Bin Packing

The earliest algorithms for one dimensional (1-D) bin packing were simple greedy algorithms such as First Fit (FF), Next Fit (NF), First Fit Decreasing Heights (FFDH), Next Fit Decreasing Heights (NFDH) etc. In their celebrated work, de la Vega and Lueker [55] gave the first APTAS by introducing linear grouping that reduces the number of different item types. Algorithms based on other item grouping or rounding based techniques have been used in many related problems. The result was substantially improved by Karmarkar and Karp [133] who gave a guarantee of

Table 1: Approximation algorithms for one dimensional bin packing

Algorithm	Performance Guarantee	Techniques
Next Fit [128]	$2 \cdot \mathsf{Opt}$	Greedy, Online,
Next Fit Decreasing [128]	$T_\infty \cdot \mathsf{Opt} + O(1)$ [8]	Presorting
First Fit [128]	$\lfloor 1.7\mathsf{Opt} \rfloor$ [61]	Greedy, Online
First Fit Decreasing[128]	$\frac{11}{9}\mathsf{Opt} + \frac{6}{9}$ [60]	Presorting
de la Vega and Lueker [55]	$(1+\epsilon)\mathsf{Opt} + O(\frac{1}{\epsilon^2})$	Linear grouping
Karp and Karmarkar [133]	$\mathsf{Opt} + O(\log^2 \mathsf{Opt})$	Iterative rounding
Rothvoß [178]	$\mathsf{Opt} + O(\log \mathsf{Opt} \cdot \log\log \mathsf{Opt})$	Discrepancy methods
Hoberg and Rothvoß [106]	$\mathsf{Opt} + O(\log \mathsf{Opt})$	Discrepancy methods

$\mathsf{Opt}+O(\log^2 \mathsf{Opt})$ by providing an iterative rounding for a linear programming formulation. It was then improved by Rothvoß [178] to $\mathsf{Opt} + O(\log \mathsf{Opt} \cdot \log\log \mathsf{Opt})$ using ideas from discrepancy theory. Very recently, Hoberg and Rothvoß [106] achieved approximation ratio of $\mathsf{Opt} + O(\log \mathsf{Opt})$ using discrepancy method coupled with a novel 2-stage packing approach. On the other hand, the possibility of an algorithm with an $\mathsf{Opt} + 1$ guarantee is still open. This is one of the top ten open problems in the field of approximation algorithms mentioned in [204].

Table 1 summarizes different algorithms and their performance guarantees. Here $T_\infty \approx 1.69$ is the well-known *harmonic* constant that appears ubiquitously in the context of bin packing.

The *Gilmore-Gomory LP relaxation* [88] is used in [55, 133, 178] to obtain better approximation. This LP is of the following form:

$$min\Big\{\mathbf{1}^T x | Ax = \mathbf{1}, x \geq 0\Big\} \tag{1}$$

Here A is the pattern matrix that consists of all column vectors $\{p \in \mathbb{N}^n | p^T s \leq \mathbf{1}\}$ where $s := (s_1, s_2, \ldots, s_n)$ is the size vector for the items. Each such column p is called a pattern and corresponds to a feasible multiset of items that can be assigned to a single bin. Now if we only consider patterns $p \in \{0,1\}^n$, LP (1) can be interpreted as an LP relaxation of a set cover problem, in which a set I of items has to be covered by *configurations* from the collection $\mathcal{C} \subseteq 2^I$, where each configuration $C \in \mathcal{C}$ corresponds to a set of items that can be packed into a bin:

$$min\left\{ \sum_{C \in \mathcal{C}} x_C : \sum_{C \ni i} x_C \geq 1 \quad (i \in I), x_C \in \{0,1\} \quad (C \in \mathcal{C}) \right\}. \tag{2}$$

This configuration LP is also used in other algorithms for multidimensional bin packing and we will discuss more on configuration LPs in later sections.

Let Opt and Opt_f be the value of the optimal integer solution and fractional solution for LP (1) respectively. Although LP (1) has an exponential number of variables, one can compute a basic solution x with $\mathbf{1}^T x \leq \mathsf{Opt}_f + \delta$ in time polynomial in n and $1/\delta$ using the Grötschel -Lovász-Schrijver variant of the Ellipsoid method [96] or the Plotkin-Shmoys-Tardos framework [172, 6]. In fact the analysis of [106], only shows an upper bound of $O(\log \mathsf{Opt})$ on the additive integrality gap of LP (1). It has been conjectured in [181] that the LP has the *Modified Integer Roundup Property*, i.e., $\mathsf{Opt} \leq \lceil \mathsf{Opt}_f \rceil + 1$. The conjecture has been proved true for the case when the instance contains at most 7 different item sizes [184]. Recently, Eisenbrand et al. [67] found a connection between coloring permutations and bin packing, that shows that *Beck's Three Permutation Conjecture* (any three permutations can be bi-colored with O(1) discrepancy) would imply a $O(1)$ integrality gap for instances with all items sizes bigger than 1/4. However, Newman, Neiman and Nikolov [165] found a counterexample to Beck's conjecture. Using these insights Eisenbrand et al. [67] showed that a broad class of algorithms can not give an $o(\log n)$ gap. Rothvoß [177] further explored this connection with discrepancy theory and gave a rounding using Beck's entropy method achieving $O(\log^2 \mathsf{Opt})$ gap alternatively. The later improvement to $O(\log \mathsf{Opt})$ in [178, 106] arose from the constructive partial coloring lemma [154] and gluing techniques. Recently Goemans and Rothvoß [90] also have shown polynomiality for bin packing when there are $O(1)$ number of item types.

Bin packing problem is also well-studied when the number of bins is some fixed constant k. If the sizes of the items are polynomially bounded integers, then the problem can be solved *exactly* using dynamic programming in $n^{O(k)}$ time, for an input of length n. Along with APTAS for bin packing, this implies a pseudo-polynomial time PTAS for bin packing, significantly better than 3/2, the hardness of (absolute) approximation for the problem. However, Jansen et al. [115] showed *unary bin packing* (where item sizes are given in unary encoding) is $\mathsf{W}[1]$-hard and thus the running time for fixed number of bins k, can not be improved to $f(k) \cdot n^{O(1)}$ for any function of k, under the standard complexity assumptions.

2.4.2 Online 1-D Bin Packing

An online bin packing algorithm uses *k-bounded space* if, for each item, the choice of where to pack it, is restricted to a set of at most k open bins. Lee and Lee [147] gave a *$O(1)$-bounded space* algorithm that achieve asymptotic competitive ratio of $T_\infty \approx 1.69$. They also showed it to be tight. Seiden [185] gave a new algorithm, HARMONIC++, whose asymptotic performance ratio is at most 1.58889, the current best among online algorithms that are not bounded space. Ramanan et al. [176] showed that Harmonic-type algorithms can not achieve better than 1.58333 asymptotic competitive ratio. In general the best known lower bound for asymptotic competitive ratio is 1.54014 [200]. Very recently, Balogh et al. [11] presented an online bin packing algorithm with an *absolute* competitive ratio of 5/3 which is optimal.

Online bin packing has also been studied under probabilistic setting. Shor [190] gave tight-bounds for average-case online bin packing. Other related algorithms for online stochastic bin packing are *Sum of Squares* algorithm by Csirik et al. [49] and primal-dual based algorithms in [98].

2.5 Multidimensional Bin Packing

In this section we discuss the preliminaries related to multidimensional bin packing. We will consider the offline version, when all items are known a priori. We also briefly survey results in the online version, when the items appear one at a time and we need to decide packing an item without knowing the future items.

2.5.1 Geometric Packing

Definition 2.5.1. Two-Dimensional Geometric Bin Packing (2-D GBP) : In two-dimensional *geometric bin packing* (2-D GBP), we are given a collection of n rectangular items $I := \{r_1, r_2, \ldots, r_n\}$ where each rectangle r_k is specified by its width and height (w_k, h_k) such that w_k, h_k are rational numbers in $[0, 1]$. The goal is to pack all rectangles into a minimum number of unit square bins.

We consider the widely studied *orthogonal packing* case, where the items must be placed in the bin such that their sides are parallel to the sides of the bin. In any feasible solution, items are not allowed to overlap. Here two variants are usually studied, (i) where the items cannot be rotated, and (ii) they can be rotated by 90 degrees.

We will also mention some results related to strip packing and geometric knapsack problems, two other geometric generalizations of bin packing, in Section 3.

Definition 2.5.2. Strip Packing (2-D SP) : In two-dimensional *strip packing* (2-D SP), we are given a strip of unit width and infinite height, and a collection of n rectangular items $I := \{r_1, r_2, \ldots, r_n\}$ where each rectangle r_k is specified by its width and height (w_k, h_k) such that w_k, h_k are rational numbers in $[0, 1]$. The goal is to pack all rectangles into the strip minimizing the height.

This is a variant of cutting stock problem, well studied in optimization.

Definition 2.5.3. Geometric Knapsack (2-D GK) : In two-dimensional *geometric knapsack* (2-D GK), we are given a unit square bin and a collection of two dimensional rectangles $I := \{r_1, r_2, \ldots, r_n\}$ where each rectangle r_k is specified by its width and height (w_k, h_k) and profit p_k such that w_k, h_k, p_k are rational numbers in $[0, 1]$. The goal is to find the maximum profit subset that can be feasibly packed into the bin.

Multidimensional variants of above three geometric problems are defined analogously using d-dimensional rectangular parallelepipeds (also known as d-orthotope, the generalization of rectangles in higher dimensions) and d-dimensional cuboids (also known as d-cube, the generalization of squares in higher dimensions). We will discuss some more on 3-dimensional variants in Section 3.

2.5.2 Vector Packing

Now we define vector bin packing, the nongeometric generalization of bin packing.

Definition 2.5.4. Vector Bin Packing (d-D VBP) : In d-dimensional *vector packing* (d-D VBP), we are given a set of n rational vectors $I := \{v_1, v_2, \ldots, v_n\}$ from $[0, 1]^d$. The goal is to partition them into sets (bins) $B_1, B_2, \ldots, B_m$ such that $||\sigma_{B_j}||_\infty \leq 1$ for $1 \leq j \leq m$ where $\sigma_{B_j} = \sum_{v_i \in B_j} v_i$ is the sum of vectors in B_j, and we want to minimize m, the number of bins.

In other words, the goal is to pack all the vectors into minimum number of bins so that for every bin the sum of packed vectors in the bin should not exceed the vector of the bin in each dimension.

We now define related vector scheduling and vector bin covering problems.

Definition 2.5.5. Vector Scheduling (d-D VS) : In d-dimensional *vector scheduling* (d-D VS), we are given a set of n rational vectors $I := \{v_1, v_2, \ldots, v_n\}$ from $[0, 1]^d$ and an integer m. The goal is to partition I into m sets $B_1, B_2, \ldots, B_m$ such that $max_{1 \leq i \leq m} ||\sigma_{B_i}||_\infty$ is minimized, where $\sigma_{B_i} = \sum_{v_i \in B_i} v_i$ is the sum of vectors in B_i.

For $d = 1$, this just reduces the classical multiprocessor scheduling.

Definition 2.5.6. Vector Bin Covering (d-D VBC) : In d-dimensional *vector bin covering* (d-D VBC), we are given a set of n rational vectors $I := \{v_1, v_2, \ldots, v_n\}$ from $[0, 1]^d$. The goal is to partition them into sets (bins) $B_1, B_2, \ldots, B_m$ such that $\sigma_{B_j} \geq 1$ in all dimensions for all $j \in [m]$, where $\sigma_{B_j} = \sum_{v_i \in B_j} v_i$ is the sum of vectors in B_1, and we want to maximize m, the number of bins.

For $d = 1$, classical bin covering problem admits APTAS [119].

2.5.3 Weighted Bipartite Edge Coloring

Now we discuss the weighted bipartite edge coloring problem.

Definition 2.5.7. Weighted Bipartite Edge Coloring (WBEC) : In *weighted bipartite edge coloring* problem, we are given an edge-weighted bipartite graph $G = (V, E)$ with weights $w : E \to [0, 1]$. The task is to find a *proper weighted coloring* of the edges with as few colors as possible.

Here proper weighted coloring is defined as follows.

Definition 2.5.8. Proper Weighted Coloring : An edge coloring of the weighted graph is called a *proper weighted coloring* if the sum of the weights of the edges incident to a vertex of any color is at most one.

Thus when all edge weights are one, the problem reduces to classical edge coloring problem. Whereas, if there are only two vertices, the problem reduces to classical bin packing problem. On the other hand if the graph has n vertices, it becomes a special case of $O(n)$-dimensional vector packing.

2.5.4 Relation between the problems

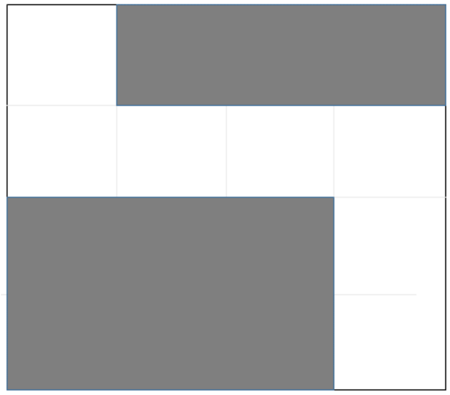


Figure 1: Two rectangles of size $\frac{1}{2} \times \frac{3}{4}$ and $\frac{1}{2} \times \frac{3}{4}$ can be packed into one bin

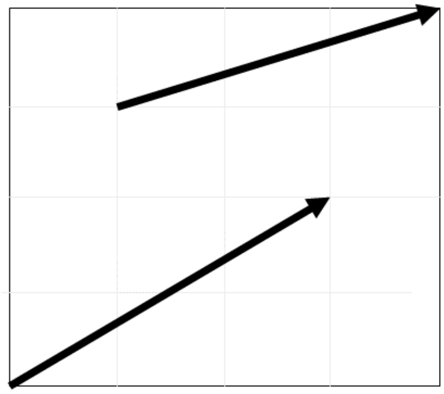


Figure 2: Two vectors $(\frac{1}{2}, \frac{3}{4})$ and $(\frac{1}{4}, \frac{3}{4})$ can not be packed into one vector bin as their sum exceeds one in the second dimension

Figure 1 and 2 show the difference between geometric packing and vector packing. Given a set of vectors, one can easily determine whether they can be packed into one unit bin by just checking whether the sum along each coordinate is at most one or not. However for geometric bin packing, it is NP-hard to determine whether a set of rectangles can be packed into one unit square bin or not, implying that no (absolute) approximation better than 2 is possible even for 2-D GBP.

Note that both geometric knapsack and strip packing are closely related to geometric bin packing. Results and techniques related to strip packing and knapsack have played a major role in improving the approximation for geometric bin packing. If all items have same height then d-dimensional strip packing reduces to $(d-1)$-dimensional geometric bin packing. On the other hand to decide whether a set of rectangles (w_i, h_i) for $i \in [n]$ can be packed into m bins, one can define a 3-D geometric knapsack instance with n items $(w_i, h_i, 1/m)$ and profit $(w_i \cdot h_i \cdot 1/m)$ and decide if there is a feasible packing with profit $\sum_{i \in n}(w_i \cdot h_i \cdot 1/m)$. Figure 3 shows the relation between different generalizations of bin packing.

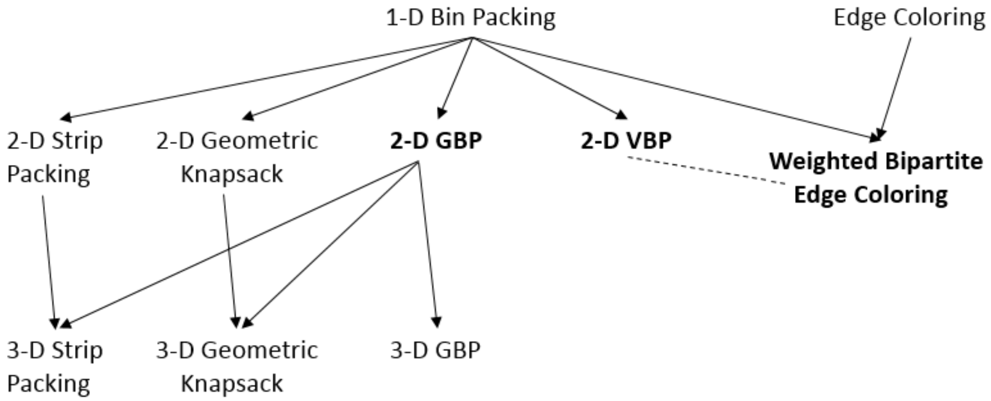


Figure 3: Generalizations of bin packing problems

2.6 Techniques:

Now we describe some techniques, heavily encountered in multidimensional packing.

2.6.1 Next Fit Decreasing Height (NFDH)

Next Fit Decreasing Height (NFDH) procedure was introduced by Coffman et al. [129]. NFDH considers items in a non-increasing order of height and greedily assigns items in this order into *shelves*, where a *shelf* is a row of items having their bases on a line that is either the base of the bin or the line drawn at the top of the highest item packed in the shelf below. More specifically, items are packed left-justified starting from bottom-left corner of the bin, until the next item can not be included. Then the shelf is closed and the next item is used to define a new shelf whose base touches the tallest (left most) item of the previous shelf. If the shelf does not fit into the bin, the bin is closed and a new bin is opened. The procedure continues till all the items are packed. This simple heuristic works quite good for small items. Some key properties of NFDH are following:

Lemma 2.6.1. [158] Let B be a rectangular region with width w and height h. We can pack small rectangles (with both width and height less than ϵ) with total area A using NFDH into B if $w \geq \epsilon$ and $w \cdot h \geq 2A + w^2/8$.

Lemma 2.6.2. [28] Given a set of items of total area of V (each having height at most one), they can be packed in at most $4V + 3$ bins by NFDH.

Lemma 2.6.3. [129] Let B be a rectangular region with width w and height h. If we pack small rectangles (with both width and height less than ϵ) using NFDH into B, total $w \cdot h - (w + h) \cdot \epsilon$ area can be packed, i.e., the total wasted volume in B is at most $(w + h) \cdot \epsilon$.

In fact it can be generalized to d-dimensions.

Lemma 2.6.4. [14] Let C be a set of d-dimensional cubes (where $d \geq 2$) of sides smaller than ϵ. Consider NFDH heuristic applied to C. If NFDH cannot place any other cube in a rectangle R of size $r_1 \times r_2 \times \cdots r_d$ (with $r_i \leq 1$), the total wasted (unfilled) volume in that bin is at most: $\epsilon \sum_{i=1}^{d} r_i$.

2.6.2 Configuration LP

The best known approximations for most bin packing type problems are based on strong LP formulations called configuration LPs. Here, there is a variable for each possible way of feasibly packing a bin (called a *configuration*). This allows the packing problem to be cast as a set covering problem, where each item in the instance I must be covered by some configuration. Let $\mathcal{C}$ denote the set of all valid configurations for the instance I. The configuration LP is defined as:

$$\min \left\{ \sum_{C \in \mathcal{C}} x_C : \qquad \sum_{C \ni i} x_C \geq 1 \quad \forall i \in I, \qquad x_C \geq 0 \quad \forall C \in \mathcal{C} \right\}. \tag{3}$$

As the size of $\mathcal{C}$ can possibly be exponential in the size of I, one typically considers the dual of the LP given by:

$$\max \left\{ \sum_{i \in I} v_i : \qquad \sum_{i \in C} v_i \leq 1 \quad \forall C \in \mathcal{C}, \qquad v_i \geq 0 \quad \forall i \in I \right\}. \tag{4}$$

The separation problem for the dual is the following knapsack problem. Given set of weights v_i, is there a feasible configuration with total weight of items more than 1. From the well-known connection between separation and optimization [94, 172, 96], solving the dual separation problem to within a $(1+\epsilon)$ accuracy suffices to solve the configuration LP within $(1+\epsilon)$ accuracy. We refer the readers to [177] for an explicit proof that, for any set family $\mathcal{S} \subseteq 2^{[n]}$, if the dual separation problem can be approximated to $(1+\epsilon)$-factor in time $T(n,\epsilon)$ then the corresponding column-based LP can be solved within an arbitrarily small additive error δ in time $poly(n, \frac{1}{\delta}) \cdot T(n, \Omega(\frac{\delta}{n}))$. This error term can not be avoided as otherwise we can decide the PARTITION problem in polynomial time. For 1-D BP, dual separation problem admits FPTAS, i.e., can be solved in time $T(n,\epsilon) = poly(n, \frac{1}{\epsilon})$. Thus the configuration LP can be solved within arbitrarily small additive constant error δ in time $poly(n, \frac{1}{\delta}) \cdot poly(n, O(\frac{n}{\delta}))$. For multidimensional bin packing, dual separation problem admits PTAS, i.e., can be solved in time $T(n,\epsilon) = O(n^{f(\frac{1}{\epsilon})})$. Thus the configuration LP can be solved within $(1+\delta)$ multiplicative factor in time $poly(n, \frac{1}{\delta}) \cdot T(n, \Omega(\frac{\delta n}{n}))$, i.e., in time $O(n^{O(f(\frac{1}{\delta}))})$.

Note that the configurations in (3) are defined based on the original item sizes (without any rounding). However, for more complex problems (say 3-D GBP) one cannot hope to solve such an LP to within $1+\epsilon$ accuracy, as the dual separation problem becomes at least as hard as 2-D GBP. In general, given a problem instance I, one can define a configuration LP in multiple ways (say where the configurations are based on rounded sizes of items in I, which might be necessary if the LP with original sizes is intractable). For the special case of 2-D GBP, the separation problem for the dual (4) is the 2-D geometric knapsack problem for which the best known result is only a 2-approximation. However, Bansal et al. [12] showed that the configuration LP (3) with original sizes can still be solved to within $1+\epsilon$ accuracy (this is a non-trivial result and requires various ideas).

Similarly for the case of vector bin packing, the separation problem for the dual (4) is the vector knapsack problem which could be solved to within $1 + \epsilon$ accuracy [80]. However, there is no EPTAS even for 2-D vector knapsack [145].

The fact that solving the configuration LP does not incur any loss for 2-D GBP or VBP plays a key role in the present best approximation algorithms.

2.6.3 Round and Approx (R&A) Framework

Now we describe the R&A Framework as described in [13]. It is the key framework used to obtain present best approximation algorithms for the 2-D geometric bin packing and vector bin packing.

1. Solve the LP relaxation of (3) using the APTAS([12] for 2-D GBP, [80] for VBP). Let x^* be the (near)-optimal solution of the LP relaxation and let $z^* = \sum_{C \in \mathcal{C}} x^*_C$. Let r the number of configurations in the support of x^*.

2. Initialize a $|\mathcal{C}|$-dimensional binary vector x^r to be a all-0 vector. For $\lceil (\ln \rho) z^* \rceil$ iterations repeat the following: select a configuration $C' \in \mathcal{C}$ at random with probability $x^*_{C'}/z^*$ and let $x^r_{C'} := 1$.

3. Let S be the remaining set of items not covered by x^r, i.e., $i \in S$ if and only if $\sum_{C \ni i} x^r_C = 0$. On set S, apply the ρ-approximation algorithm $\mathcal{A}$ that rounds the items to $O(1)$ types and then pack. Let x^a be the solution returned by $\mathcal{A}$ for the residual instance S.

4. Return $x = x^r + x^a$.

Let $\mathsf{Opt}(S)$ and $\mathcal{A}(S)$ denote the value of the optimal solution and the approximation algorithm used to solve the residual instance, respectively. Since the algorithm uses randomized rounding in step 2, the residual instance S is not known in advance. However, the algorithm should perform "well" independent of S. For this purpose, Bansal, Caprara and Sviridenko [13] defined the notion of *subset-obliviousness* where quality of approximation algorithm to solve the residual instance is expressed using a small collection of vectors in $\mathbb{R}^{|I|}$.

Definition 2.6.5. An asymptotic ρ-approximation for the set covering problem defined in (1), is called subset-oblivious if, for any fixed $\epsilon > 0$, there exist constants k, Λ, β (possibly dependent on ϵ), such that for every instance I of (1), there exist vectors $v^1, v^2, \cdots, v^k \in \mathbb{R}^{|I|}$ that satisfy the following properties:

1. $\sum_{i \in C} v_i^j \leq \Lambda$, for each configuration $C \in \mathcal{C}$ and $j = 1, 2, \cdots, k$;

2. $\mathsf{Opt}(I) \geq \sum_{i \in I} v_i^j$ for $j = 1, 2, \cdots, k$;

3. $\mathcal{A}(S) \leq \rho \sum_{i \in S} v_i^j + \epsilon \mathsf{Opt}(I) + \beta$, for each $S \subseteq I$ and $j = 1, 2, \cdots, k$.

Roughly speaking, the vectors are analogues to the sizes of items and are introduced to use the properties of dual of (1). Property 1 says that the vectors divided by constant Λ must be feasible for (2). Property 2 provides lower bound for $OPT(I)$ and Property 3 guarantees that the $\mathcal{A}(S)$ is not significantly larger than ρ times the lower bound in Property 2 associated with S.

The main result about the R&A is the following.

Theorem 2.6.6. (simplified) If a problem has a ρ-asymptotic approximation algorithm that is subset oblivious, and the configuration LP with original item sizes can be solved to within $(1+\epsilon)$ accuracy in polynomial time for any $\epsilon > 0$, then the R&A framework gives a $(1 + \ln \rho)$-asymptotic approximation.

One can derandomize the above procedure and get a deterministic version of R&A method in which Step 2 is replaced by a greedy procedure that defines x^r guided by a suitable potential function. See [13] for the details of derandomization.

2.6.4 Algorithms based on rounding items to constant number of types:

Rounding of items to $O(1)$ types has been used either implicitly [18] or explicitly [55, 133, 27, 116, 132], in almost all bin packing algorithms to reduce the problem complexity and make it tractable. One of the key properties of rounding items is as follows:

Lemma 2.6.7. [133] Let I be a given set of items and s_x be the size of item $x \in I$. Define a partial order on bin packing instances as follows: $I \leq J$ if there exists a bijective function $f : I \to J$ such that $s_x \leq s_{f(x)}$ for each item $s \in I$. J is then called a rounded up instance of I and $I \leq J$ implies $\mathsf{Opt}(I) \leq \mathsf{Opt}(J)$.

There are typically two types of rounding: either the size of an item in some coordinate (such as width or height) is rounded in an instance-oblivious way (e.g. in harmonic rounding [147, 27], or rounding sizes to geometric powers[133]), or it is rounded in an input sensitive way (e.g. in linear grouping [55]).

Linear grouping: Linear grouping was introduced by Fernandez de la Vega and Lueker [55] in the first approximation scheme for 1-D bin packing problem. It is a technique to reduce the number of distinct item sizes. The scheme works as follows, and is based on a parameter k, to be fixed later. Sort the n items by nonincreasing size and partition them into $\lceil 1/k \rceil$ groups such that the first group consists of the largest $\lceil nk \rceil$ pieces, next group consists of the next $\lceil nk \rceil$ largest items and so on, until all items have been placed in a group. Apart from the last group all other groups contain $\lceil nk \rceil$ items and the last group contains $\leq nk$ items. The rounded instance $\tilde{I}$ is created by discarding the first group, and for each other group, the size of an item is rounded up to the size of the largest item in that group. Following lemma shows that the optimal packing of these rounded items is very close to the optimal packing of the original items.

Lemma 2.6.8. [55] Let $\tilde{I}$ be the set of items obtained from an input I by applying linear grouping with group size $\lceil nk \rceil$, then

$$\mathsf{Opt}(\tilde{I}) \leq \mathsf{Opt}(I) \leq \mathsf{Opt}(\tilde{I}) + \lceil nk \rceil$$

and furthermore, any packing of $\tilde{I}$ can be used to generate a packing of I with at most $\lceil nk \rceil$ additional bins.

If all items are $> \epsilon$, then $n\epsilon < \mathsf{Opt}$. So, for $k = \epsilon^2$ we get that any packing of $\tilde{I}$ can be used to generate a packing of I with at most $\lceil n\epsilon^2 \rceil \leq n\epsilon^2 + 1 < \epsilon \cdot \mathsf{Opt} + 1$ additional bins. In Chapter 4, we will use a slightly modified version of linear grouping that does not loose the additive constant of 1.

Geometric grouping: Karmarkar and Karp [133] introduced a refinement of linear grouping called geometric grouping with parameter k. Let $\alpha(I)$ be the smallest item size of an instance I. For $r = 0, 1, \ldots, \lfloor \log_2 \frac{1}{\alpha(I)} \rfloor$, let I_r be the instance consisting of items $i \in I$ such that $s_i \in (2^{-(r+1)}, 2^{-r}]$. Let J_r be the instances obtained by applying linear grouping with parameter $k \cdot 2^r$ to I_r. If $J = \cup_r J_r$ then:

Lemma 2.6.9. [133] $\mathsf{Opt}(J) \leq \mathsf{Opt}(I) \leq \mathsf{Opt}(J) + k \lceil \log_2 \frac{1}{\alpha(I)} \rceil$.

Harmonic rounding: Lee and Lee [147] introduced harmonic algorithm ($harmonic_k$) for online 1-D bin packing, where each item j with $s_j \in (\frac{1}{q+1}, \frac{1}{q}]$ for $q \in \{1, 2, \cdots, (k-1)\}$, is rounded to $1/q$. Then q items of type $1/q$ can be packed together in a bin. So for each type q, only items of the same type are packed into the same bin and new bins are opened when the current bin is full. Let $t_1 = 1$, $t_{q+1} := t_q(t_q + 1)$ for $q \geq 1$. Let $m(k)$ be the integer with $t_{m(k)} < k < t_{m(k)+1}$. It is shown in [147] that the asymptotic approximation ratio of $harmonic_k$ is $T_k = \sum_{q=1}^{m(k)} \frac{1}{t_q} + \frac{1}{t_{m(k)+1}} \cdot \frac{k}{k-1}$. When $k \to \infty$, $T_\infty = 1.691..$, this is the harmonic constant, ubiquitous in bin packing. Caprara [27] introduced *harmonic decreasing height* algorithm for 2-D GBP with asymptotic approximation ratio of T_∞, where widths are rounded according to harmonic rounding and then same width items are packed using NFDH. We refer the readers to [71] for more related applications of the Harmonic algorithm in online and bounded space algorithms.

Chapter III

GEOMETRIC BIN PACKING

In this chapter we discuss our results related to geometric bin packing (GBP) and its variants. First let us briefly mention our results and techniques and then we give a detailed survey of previous work and technical details of our results in the later chapters. Our main result is an improved algorithm for the 2-D bin packing problem stated as follows:

Theorem 3.0.10. For any $\epsilon > 0$, there is a polynomial time algorithm with an asymptotic approximation ratio of $\ln(1.5) + 1 + \epsilon \approx 1.405 + \epsilon$ for 2-D geometric bin packing. This holds both for the versions with and without rotations.

The main idea behind Theorem 3.0.10 is to show that the *Round and Approx* (R&A) framework introduced by Bansal, Caprara and Sviridenko [13] (See section 2.6.3) can be applied to a 1.5-approximation algorithm of Jansen and Prädel [116]. Roughly speaking, this framework states that given a packing problem, if (i) the configuration LP for the problem (with the original item sizes) can be solved up to error $1 + \epsilon$ for any $\epsilon > 0$, and (ii) there is a ρ-approximation for the problem that is *subset-oblivious*; then one can obtain a $(1 + \ln \rho)$-asymptotic approximation for the problem.

In [13], it was shown that the APTAS for 1-D BP due to [55] and the 2-D BP algorithm of [27] are subset-oblivious. However, the notion of subset-obliviousness as defined in [13] is based on various properties of dual-weighting functions, making it somewhat tedious to apply and also limited in scope (e.g. it is unclear to us how to apply this method directly to the algorithm of [116]).

In this chapter we give a more general argument to apply the R&A framework directly to a wide class of algorithms[1], and without any reference to dual-weighting functions. In particular, we show that any algorithm based on rounding the (large) items into $O(1)$ types, is subset-oblivious.

The main observation is that any ρ-approximation based on rounding the item sizes can be related to another configuration LP (on rounded item sizes) whose solution is no worse than ρ times the optimum solution. As the item sizes are rounded, there are only $O(1)$ constraints in this LP and it can easily be shown to be subset-oblivious.

For the particular case of 2-D BP, we present the algorithm of Jansen and Prädel that directly fits in the above framework. As most algorithms for bin-packing problems are based on rounding into $O(1)$ types, this also makes the framework widely applicable. For example, this gives much simpler proofs of all the results in [13].

Finally, we give some results to show the limitations of rounding based algorithms in obtaining better approximation ratios. Rounding of items to $O(1)$ types has been used implicitly [18] or explicitly [55, 133, 27, 116, 132], in almost all bin packing algorithms. There are typically two types of rounding: either the size of an item in some coordinate (such as width or height) is rounded up in an instance-oblivious way (e.g. in Harmonic rounding [147, 27], or rounding sizes to geometric powers [133]), or it is rounded up in an input sensitive way (e.g. in linear grouping [55]). We also show limitations of rounding based algorithms as follows.

Theorem 3.0.11. Any rounding algorithm that rounds at least one side of each large item to some number in a constant-size collection of numbers, independent of the problem instance (let us call such rounding input-agnostic), cannot have an approximation ratio better than 3/2.

[1] This includes all known algorithms that we know of for bin-packing type problems, except the ones based on the R&A method.

Organization of the chapter : In Section 3.1, we survey previous works in geometric bin packing and its variants. In Section 3.2, we describe how the Round and Approx framework can be applied to rounding based algorithms. In Section 3.3, we present the 1.5 approximation algorithm of [116] and show how the Round and Approx framework applies to it. Then in Section 3.4, we show our lower bounds for rounding based algorithms. Finally, in Section 3.5 we make some final remarks.

3.1 Prior Works

In this section we give an extensive survey of the literature related to geometric packing and other related problems.

3.1.1 Geometric Bin Packing

Two-dimensional geometric bin packing (GBP) is substantially different from the 1-D case. Bansal et al. [14] showed that 2-D bin packing in general does not admit an APTAS unless $\mathsf{P} = \mathsf{NP}$.

On the positive side, there has also been a long sequence of works giving improved approximation algorithms. We refer readers to [152] for a review of several greedy heuristics such as Next Fit Decreasing, First Fit Decreasing, Best Fit Decreasing, Finite Best Strip, Floor-Ceiling algorithm, Finite First Fit, Knapsack Packing algorithm, Finite Bottom-Left, Alternate Directions etc. For the case when we do not allow rotation, until the mid 90's the best known bound was a 2.125 approximation [37], which was improved by Kenyon and Rémila [138] to a $2 + \epsilon$ approximation (this comes as a corollary of their APTAS for 2-D strip packing) for any $\epsilon > 0$. Caprara in his break-through paper [27] gave an algorithm for 2-D bin packing attaining an asymptotic approximation ratio of $T_\infty (\approx 1.69103)$.

For the case when rotations are allowed, Miyazawa and Wakabayashi [161] gave an algorithm with an asymptotic performance guarantee of 2.64. Epstein and Stee [73] improved it to 2.25 by giving a packing where, in almost all bins, an area of 4/9 is occupied. Finally an asymptotic approximation guarantee arbitrarily close to 2 followed from the result of [124].

This was later improved by Bansal et al. [13] to $(\ln(T_\infty)+1) \approx 1.52$, for both the cases with and without rotation, introducing a novel randomized rounding based framework: *Round and Approx (R & A)* framework. Jansen and Prädel [116] improved this guarantee further to give a 1.5-approximation algorithm. Their algorithm is based on exploiting several non-trivial structural properties of how items can be packed in a bin. This is the best algorithm known so far, and holds both for the case with and without rotations.

We remark that there is still a huge gap between these upper bounds and known lower bounds. In particular, the best known explicit lower bound on the asymptotic approximation for 2-D BP is currently $1 + 1/3792$ and $1 + 1/2196$, for the versions with and without rotations, respectively [36]. The best asymptotic worst-case ratio that is achievable in polynomial time for d-dimensional GBP for $d > 2$ is T_∞^{d-1} [27], and in fact it can be achieved by an online algorithm using bounded space. There are no known explicit better lower bounds for higher dimensions.

In *non-asymptotic setting*, without rotations there are a 3-approximation algorithms by Zhang [207] and also by Harren and van Stee [105]. Harren and van Stee [103] gave a non-asymptotic 2-approximation with rotations. Independently this approximation guarantee is also achieved for the version without rotations by Harren and van Stee [104] and Jansen et al. [118]. These 2-approximation results match the non-asymptotic lower bound for this problem, unless $\mathsf{P} = \mathsf{NP}$.

3.1.2 Square Packing

Leung et al. [148] have shown that even the special case of packing squares into square is still NP-hard. Kohayakawa et al. [142] gave a $(2-(2/3)^d+\epsilon)$ approximation for packing d-dimensional cubes into unit cubes. Later Bansal et al. [14] have given an

APTAS for the problem of packing d-dimensional cubes into d-dimensional unit cubes.

3.1.3 Online Packing:

Coppersmith and Raghavan [46] first studied online 2-D GBP and gave algorithms with asymptotic performance ratio of 3.25 and 6.35 for $d = 2$ and 3 respectively. Csirik and van Vliet [50] gave an algorithm with performance ratio T_∞^d (This gives 2.859 for 2-D) for arbitrary dimension d. Epstein and van Stee [73] achieved the same ratio of T_∞^d only using bounded space and showed it to be the optimal among all bounded space algorithms. In 2002, Seiden and van Stee [186] proposed an elegant algorithm called $H \otimes C$, comprised of the Harmonic algorithm H and the Improved Harmonic algorithm C, for the 2-D online bin packing problem and proved that the algorithm has an asymptotic competitive ratio of at most 2.66013. Since the best known online algorithm for one-dimensional bin packing is the Super Harmonic algorithm [185], a natural question was whether a better upper bound could be achieved by using the Super Harmonic algorithm instead of the Improved Harmonic algorithm? Han et al. [100] gave a positive answer for this question and a new upper bound of 2.5545 is obtained for the two-dimensional online bin packing. The main idea is to develop new weighting functions for the Super Harmonic algorithm and propose new techniques to bound the total weight in a rectangular bin. The best known lower bound is 1.907 by Blitz, van Vliet and Woeginger [21]. We refer the readers to [199] for a survey of online algorithms for geometric bin packing in multiple dimensions.

When we allow rotation, Epstein [70] gave an algorithm with asymptotic performance ratio of 2.45. Later Epstein and van Stee [74] gave an algorithm with asymptotic performance ratio of 2.25.

For the special case where items are squares, there are also a large number of results. Coppersmith and Raghavan [46] showed their algorithm gives asymptotic performance ratio of 2.6875 in this case. They also gave a lower bound of 4/3. Seiden and van Stee [186] gave an algorithm with asymptotic performance ratio of 2.24437. Epstein and van Stee [72] have shown an upper bound of 2.2697 and a lower bound of 1.6406 for online square packing, and an upper bound of 2.9421 and a lower bound of 1.6680 for online cube packing. The upper bound for squares can be further reduced to 2.24437 using a computer-aided proof. Later Han et al. [101] get an upper bound of 2.1187 for square packing and 2.6161 for cube packing. For bounded space online algorithms, Epstein and van Stee [75] showed lower and upper bounds for optimal online bounded space hypercube packing till dimensions 7. In particular, for 2-D it lies in $(2.3634, 2.3692)$ and for 3-D it lies in $(2.956, 3.0672)$.

Table 2: Present state of the art for geometric bin packing

Problem	Dim.	Subcase	Best algorithm	Best lower bound
Geometric Bin Packing	2	OFF-REC-WR	asymp[2]: 1.5 [116]	$1 + 1/3792$ [36]
			abs[3]: 2 [103]	2 (folklore)
		OFF-REC-NR	asymp: 1.5 [116]	$1 + 1/2196$ [36]
			abs: 2 [104]	2 (folklore)
	d	OFF-REC-NR	asymp: T_∞^{d-1} for $d > 2$ [27]	$1 + 1/2196$ [36]
		OFF-CUB	asymp: PTAS[14]	NP-hard
			abs: 2 [14]	2 [78]
	2	ON-REC-NR	asymp: 2.5545 [100]	1.907 [21]
		ON-REC-WR	asymp: 2.25 [74]	1.6406 [72]
		ON-CUB	asymp: 2.1187[101]	1.6406 [72]
	3	ON-REC-NR	asymp: 4.3198 [100]	1.907 [21]
		ON-CUB	asymp: 2.6161[101]	1.6680 [72]

[2] Here asymp. means asymptotic approximation guarantee

[3] Here abs. means absolute approximation guarantee

Table 2 summarizes present best approximation/inapproximability results for geometric bin packing. Here OFF denotes offline, ON denotes online, REC denotes rectangles, CUB denotes cubes, WR denotes with rotation and NR denotes without rotation.

3.1.4 Heuristics

Lodi et al. have reviewed several exact algorithms based on enumerative approach or branch-and-bound in [152]. They have also studied several integer programming models for 2-D GBP and other related problems. Pisinger and Sigurd [171] gave an algorithm based on Dantzig-Wolfe decomposition. Here the master problem is a set covering problem, solved by delayed column generation. The subproblem deals with the packing of rectangles into a single bin and is solved as a constraint-satisfaction problem (CSP). Martello and Vigo [156] had considered exact solutions for 2-D GBP for instance sizes upto 120.

Jylanki [130] reviewed several greedy heuristics in detail and did empirical study. Primarily he considered five broad classes of algorithms: shelf algorithms, fit-based guillotine algorithms, split-based guillotine algorithms, maximal rectangle algorithms and skyline algorithms. Lodi [151] also has surveyed several one-phase, two-phase and non-shelf algorithms.

In the past two decades, many local search and meta-heuristic algorithms for rectangle packing problems have been proposed. We refer interested readers to Aarts and Lenstra [1] and Glover and Laguna [89] for a detailed survey. Dowsland [63] proposed a meta-heuristic approach to strip packing using simulated annealing. During the search, the objective is to minimize pair-wise overlapping area of rectangles and the neighborhood contains all solutions representing vertical or horizontal items shifting. Jakobs [114] presented a genetic algorithm, based on the representation of packing pattern by an order given by some permutation, and packing positions are determined by a Bottom-Left strategy. There are many similar algorithms based on a *permutation coding scheme*. These algorithms consist of two phases: (1) first, find a good permutation using some meta-heuristic, (2) then the decoding algorithm puts items following the permutation order. Several interesting coding schemes have been proposed such as n-leaf binary tree [123], sequence pair [162], bounded sliceline grid (BCG) [163], O-tree [97], B^*-tree [33], quarter-state sequence [179] etc. Pasha [170] has studied geometric bin packing algorithms for arbitrary shapes and presented several genetic algorithms and simulated annealing based approach. Kroger [144] introduced a sequential and a parallel genetic algorithm based on guillotine cuts.

Lodi et al. [153] introduced a Tabu Search framework exploiting a new constructive heuristic for the evaluation of the neighborhood for 3-D GBP. Kell and Hoeve [135] investigated the application of multivalued decision diagrams (MDDs) to multi-dimensional bin packing problems. Faroe et al. [76] have given a guided local search based algorithm for 3-D GBP. For other heuristics for cutting and packing related problems, see the application-oriented research bibliography by Sweeney and Paternoster [197]. For further details on meta-heuristics for rectangle packing we refer the readers to [112, 110, 23].

For practical problems such as pallet packing in warehouses, several other factors are needed to be considered such as the stability of packing (under gravity), elasticity, interlocking etc. Several heuristics are considered in [183, 57] for the stable pallet packing problem. In VLSI design, simulated annealing algorithms are used in practice to solve 3-D Bin Packing problem. Though these algorithms do not have a good worst-case guarantee, they still sometimes work well in practice. There are multiple ways e.g., sequence triple, transitive closure, 3D-CBL [155] to map each solution of bin packing to a list of 0-1 integers to apply simulated annealing along with different ways to move in the solution space.

3.1.5 Resource Augmentation

Due to pathological worst-case examples, bin packing has been well-studied under *resource augmentation*, i.e., the side length of the bin is augmented to $(1+\epsilon)$ instead of one. This is also known as *bin stretching.* Though 2-D GBP does not admit an APTAS, Bansal et al. [14] gave a polynomial time algorithm to pack rectangles into at most m number of bins of size $(1+\epsilon) \times (1+\epsilon)$ where m is the optimal number of unit bins needed to pack all items. Later Bansal and Sviridenko [19] showed that this is possible even when we relax the size of the bin in only one dimension.

3.1.6 Strip Packing

A closely related problem is the *Strip Packing* problem. It is another generalization of one dimensional bin packing problem and closely tied with the geometric bin packing problem. As we had stated earlier, the best approximation algorithm for 2-D GBP used to be factor $2+\epsilon$ and was a corollary from the APTAS for 2-D strip packing due to Kenyon and Rémila [138]. The 2-D variant, where we are given a strip with width one and unlimited height and the goal is to pack 2-D rectangular items into the strip so as to minimize the height of the packing, is also known as the *Cutting Stock Problem.* In three dimensions, we are given 3-D rectangular items each of whose dimensions is at most one and they need to be packed into a single 3-D box of unit depth, unit width and unlimited height so as to minimize the height of the packing.

First let us discuss offline algorithms for 2-D strip packing. Baker et al. [9] introduced the problem in 1980 and gave an algorithm with absolute approximation ratio of 3. Later Coffman et al. [129] introduced Next-Fit Decreasing Height (NFDH), First-Fit Decreasing Height (FFDH) [129] for 2-D strip packing without rotations, achieving asymptotic approxiamtion ratio as 2 and 1.7 respectively. The upper bound of 2 for NFDH remains valid even for the case with rotations, since the proofs use only area arguments. Epstein and Stee [73] gave a 3/2 approximation algorithm for the problem with rotation. Finally APTAS was given for 2-D strip packing without rotations [138] and with rotations in [124] using a nice interplay of techniques like fractional strip packing, linear grouping and a variant of NFDH. For the absolute approximation, Harren et al. [102] have given a $(5/3 + \epsilon)$-approximation whereas the lower bound is $3/2$ which follows from one dimensional bin packing.

Now we discuss online algorithms for 2-D strip packing. Baker and Schwartz [10] showed that First-Fit Shelf has asymptotic performance ratio 1.7. Csirik and Woeginger [51] improved it to $T_\infty \approx 1.691$ using the *Harmonic* algorithm as a subroutine. They also mention a lower bound of 1.5401. For the absolute performance ratio, Brown et al. [26] have given a lower bound of 2.

3-D strip packing is a common generalization of both the 2-D bin packing problem (when each item has height exactly one) and the 2-D strip packing problem (when each item has width exactly one). Li and Cheng [150] were among the first people who considered the problem. They showed 3-D versions of FFDH and NFDH have unbounded worst-case ratio. They gave a 3.25 approximation algorithm, and later gave an online algorithm with upper bound of $T_\infty^2 \approx 2.89$ [149] using the Harmonic algorithm as a subroutine. Bansal et al. [16] gave a 1.69 approximation for the offline case. Recently Jansen and Prädel [117] further improved it to 1.5. Both these two algorithms extend techniques from 2-D bin packing.

3.1.7 Shelf and Guillotine Packing

For $d = 2$, many special structures of packings have been considered in the literature, because they are both easy to deal with and important in practical applications. Among these, very famous are the two-stage packing structures, leading to two-dimensional shelf bin packing (2SBP) and two-dimensional shelf strip packing (2SSP). Two-stage packing problems were originally introduced by Gilmore and Gomory [20] and, thinking in terms of cutting instead of packing, requires that each item be obtained from the associated bin by at most two stages of cutting.

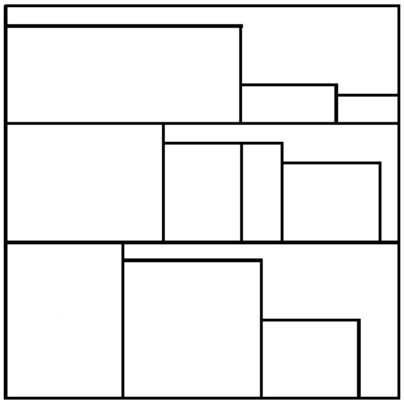


Figure 4: Example of two-stage packing

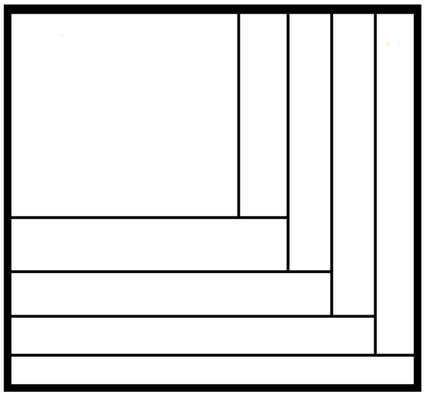


Figure 5: Example of guillotine packing

In two-stage packing, in the first stage, the bins are horizontally cut into shelves. The second stage produces slices, which contain a single item by cutting the shelves vertically. Finally, an additional stage (called trimming) is allowed in order to separate an item from a waste area. See Figure 4 for an example of two-stage packing. Two-stage packing is equivalent to packing the items into the bins in shelves, where a *shelf* is a row of items having their bases on a line that is either the base of the bin or the line drawn at the top of the highest item packed in the shelf below. Formally, a shelf is a set S of items such that total width $\sum_{j \in S} w_j \leq 1$; its height $h(S)$ is

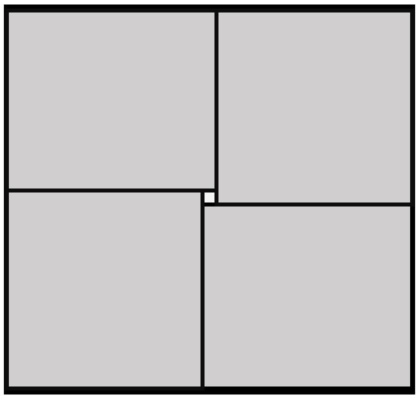


Figure 6: Example of non-guillotine packing

given by $max_{j \in S} h_j$. Many classical heuristics for 2-D strip packing ([129], [10], [51]) and 2-D GBP ([37]), including NFDH and FFDH, construct solutions that are in fact feasible for the two-stage versions. Moreover, Caprara et al. [28] presented an APTAS, each for 2SBP and 2SSP. Given this situation, it is natural to ask for the asymptotic worst-case ratio of general packing versus two-stage packing. Csirik and Woeginger [51] showed ratio of 2SSP versus 2-D strip packing is equal to T_∞. Caprara [27] showed the ratio of 2SBP versus 2-D GBP is also equal to T_∞. Both their algorithms are online and based on *Harmonic Decreasing Height* (HDH) heuristic. Now consider the optimal 2SBP solution in which the shelves are horizontal as well as the optimal 2SBP solution in which they are vertical. (Recall that near-optimal 2SBP solutions can be found in polynomial time [28].) There is no evidence that the asymptotic worst-case ratio between the best of these two solutions and the optimal 2-D GBP can be as bad as T_∞, and in fact Caprara conjectured that this ratio is 3/2. On the other hand, he also mentions that there are examples where we cannot do better than T_∞, if both solutions are formed by the HDH algorithm in [27].

Seiden and Woeginger [187] observed that the APTAS of Kenyon and Rémila [138] can easily be adapted to produce a near-optimal packing in three stages for 2-D strip packing, showing that the asymptotic worst-case ratio of 2-D strip packing versus its k-stage version is 1 for any $k > 2$, and leading to an APTAS for the latter.

Bansal et al. [18] provided an APTAS for the *guillotine case*, i.e., the case in which the items have to be packed in alternate horizontal and vertical stages but there is no limit on the number of stages that can be used. In the guillotine case, there is a sequence of edge-to-edge cuts parallel to one of the edges of the bin. See Figure 5 for an example of guillotine packing and Figure 6 for an example that is not a guillotine packing. Recently Abed et al. [2] studied other related packing problems under guillotine cuts. They also made a conjecture that, for any set of n non-overlapping axis-parallel rectangles, there is a guillotine cutting sequence separating $\Omega(n)$ of them. A proof of this conjecture will imply a $O(1)$-approximation for Maximum Independent Set Rectangles, a related NP-hard problem.

3.1.8 Geometric Knapsack

For 2-D Geometric Knapsack (GK), a result of Steinberg [194] for strip packing translates into a $(3+\epsilon)$ approximation [29]. Present best known approximation algorithm is due to Jansen and Zhang [125] and has an approximation guarantee of $(2+\epsilon)$. On the other hand, no explicit inapproximability results are known. PTAS are known for special cases when resource augmentation is allowed in one dimension [120], all items are square [121] or all items are small [79]. Bansal et al. [12] gave a PTAS for the special case when the range of the profit-to-area ratio of the rectangles is bounded by a constant. Recently Adamaszek and Wiese [3] gave a quasi-PTAS for the problem. This implies that the problem is not APX-hard (thus we can still hope for a PTAS) unless NP $\subseteq$ QP. Very recently, Abed et al. [2] obtained another quasi-PTAS using only sequence of guillotine cuts.

For 3-D, Diedrich et al. [58] have given $7+\epsilon$ and $5+\epsilon$ approximation, for the cases without and with rotations, respectively.

Table 3: Present state of the art for strip packing and geometric knapsack

Problem	Dim.	Subcase	Best algorithm	Best lower bound
Strip Packing	2	OFF-REC-WR	asymp: PTAS[124]	NP-hard
			abs: $\frac{5}{3} + \epsilon$ [102]	3/2
		OFF-REC-NR	asymp: PTAS[138]	NP-hard
	3	OFF-REC-NR	asymp: 1.5 [117]	$1 + 1/2196$ [36]
		OFF-CUB	asymp: PTAS[16]	NP-hard
	2	ON-REC-NR	asymp: T_∞ [51]	1.5401 [51]
	3	ON-REC-NR	asymp: 2.5545 [100] [4]	1.907 [21]
	$d > 3$	ON-REC-NR	asymp: T_∞^d [50]	$\rightarrow 3$ (for $d \rightarrow \infty$) [199]
Geometric Knapsack	2	OFF-REC-NR	$2 + \epsilon$ [125]	NP-hard
		OFF-CUB	PTAS[121]	NP-hard
	3	OFF-REC-NR	$7 + \epsilon$ [58]	NP-hard
		OFF-REC-WR	$5 + \epsilon$ [58]	NP-hard

Table 3 summarizes present best results for strip packing and geometric knapsack. As previously, OFF denotes offline, ON denotes online, REC denotes rectangles, CUB denotes cubes, WR denotes with rotation and NR denotes without rotation.

3.2 R&A Framework for Rounding Based Algorithms

We describe here a general approach to show that a wide class of algorithms for bin-packing type problems, in particular those based on rounding the item sizes to $O(1)$ types, is subset-oblivious. While such algorithms are hard to define formally, we state their general approach below, which subsumes all the known algorithms that we are aware of.

[4] See [199] for the modified algorithm

General form of a rounding based algorithm. Now let us intuitively present the general form of a rounding based algorithm, which appears frequently in the literature [173, 28]. A typical rounding based algorithm for a two-dimensional problem has the following form. Given some accuracy parameter $\epsilon > 0$, one first defines two functions $f(\epsilon)$ and $g(\epsilon)$ (that only depend on ϵ) with $g(\epsilon) \ll f(\epsilon)$.

Now items are divided into three types based on their sizes:

- *Big :* if all its coordinates are at least $f(\epsilon)$,
- *Small:* if all its coordinates are at most $g(\epsilon)$,
- *Medium:* if at least one coordinate lies in the range $(g(\epsilon), f(\epsilon))$.

A standard argument [28, 173] shows that the functions g and f can be chosen such that their ratio is as large as desired, while ensuring that the volume (area in 2-D) of medium items is at most $\frac{\epsilon}{8}$ times $\mathrm{Vol}(I)$, the total volume of items in the input instance I. We refer readers to Lemma 3.21 in [173] for a detailed proof. Now these medium items can be ignored as they can be packed in $(\frac{\epsilon}{2} \cdot \mathsf{Opt} + O(1))$ separate bins using NFDH from Lemma 2.6.2.

Now, all items have each coordinate either small (in $[0, g(\epsilon)]$) or big (in $[f(\epsilon), 1]$). Call an item *skewed* if it is neither big nor small (i.e., some coordinates are less than $g(\epsilon)$ and some more than $f(\epsilon)$). Skewed items can be classified into at most $2^d - 2 = 2$ types based on which subset of coordinates is large, where $d = 2$ is the total number of coordinates.

Now, the algorithm rounds the large dimensions of big and skewed items to $O_\epsilon(1)$ values (a constant depending on only ϵ). This rounding can possibly be in a very complex way, including the guessing of sizes.

Now consider function $h(\epsilon)$ that only depends on ϵ with $g(\epsilon) << h(\epsilon) \cdot f(\epsilon) \leq f(\epsilon)$. In a rounding based algorithm, one argues that in any packing almost all skewed and small items are placed (possibly fractionally) in large regions called *containers*, where each container has large size in each dimension. Specifically one dimension has size $\geq f(\epsilon)$ and the other dimension has size $\geq f(\epsilon) \cdot h(\epsilon)$ and these containers have O(1) types of sizes. Thus these containers can be viewed as big items and the algorithm focuses on packing of constant (say q) types of large items and containers. Now there are only O(1) number of possible configurations as there are O(1) types of items (large items or containers). So there are only polynomial number of possible packings of these large items. One can simply use brute-force method to enumerate all possible packings and select the best. The running time can be improved using integer programming techniques, for fast mixed linear integer programs with few integer variables from [68]. Later almost all skewed items are packed integrally into containers using an *assignment LP*. Afterwards the small items and the remaining skewed items are filled using NFDH in the empty spaces in the packing of big and skewed items. The large separation between g and f ensures that this incurs negligible loss in volume. Then one defines some algorithm $\mathcal{A}$ that finds an almost optimal (say within a factor $(1 + \epsilon_a)$) packing of these rounded big items and containers.

It can be easily verified that the $(1.5 + \epsilon)$-approximation algorithm of Jansen and Prädel [116], as stated in Section 3.3, falls directly in this framework. In their algorithm they choose $f(\epsilon) = \delta, g(\epsilon) = \delta^4, h(\epsilon) = \delta/2$, all big items are rounded to $O(\frac{1}{\delta^2} \cdot \frac{1}{\delta^2})$ types and there are $O(\frac{1}{\delta^6})$ types of containers where $\delta = \epsilon/48$. We will explain the properties of this algorithm in [116] in more detail in the later section. We remark that the rounding of sizes in their algorithm is non-trivial and actually depends on the bin patterns that are used in the optimum solution (which the algorithm will guess).

Relating the algorithm to the configuration LP with rounded item sizes. For concreteness, let us consider the case of 2-D GBP. Suppose $\mathcal{A}$ is a rounding based ρ-approximation algorithm for it. Then, $\mathcal{A}(I) \leq \rho \cdot \mathsf{Opt}(I)$ for any instance I of the problem. The items are classified into big and small. There are two types of skewed items: long (with height $\geq f(\epsilon)$ and width $\leq g(\epsilon)$) and wide (with width $\geq f(\epsilon)$, height $\leq g(\epsilon)$). Upon rounding, the big items are rounded to $O(1)$ types and long (and wide) items are assigned to $O(1)$ groups of different heights (or widths). Let $\tilde{I}$ denote instance obtained from I by rounding the large dimensions according to the rounding performed by $\mathcal{A}$. Clearly, $\mathcal{A}(I) \geq \mathsf{Opt}(\tilde{I})$ and hence

$$\mathsf{Opt}(\tilde{I}) \leq \rho \cdot \mathsf{Opt}(I). \tag{5}$$

Now, let us define the configuration LP on the instance $\tilde{I}$, where the configurations correspond to feasible packing of items in $\tilde{I}$. As there are only a constant number (say t) of such item types, this LP has only a constant number t of non-trivial constraints, one for each item type.

For $i \in \{1, \dots, p_1\}$, let B_i denote the group of big items rounded to type i and $c^r_{B_j}$ be the number of items of type j (i.e., from the set B_j) in the r'th configuration. Long items are rounded to p_2 types of heights and assigned to groups $L_1, \cdots, L_{p_2}$. Similarly wide items are rounded to p_3 types of widths and assigned to groups $W_1, \cdots, W_{p_3}$. Let $w(L_k)$ denote the total width of items in L_k, and $h(W_\ell)$ denote the total height of items in W_ℓ. Similarly, let $c^r_{L_k}$ (resp. $c^r_{W_\ell}$) denote the total width of items in L_k (resp. total height of items in W_ℓ) in the configuration r. Let T be the set of small items and c^r_T be the total volume (area in 2-D) of small items in configuration r. Let $vol_T(\tilde{I})$ be the total volume of small items (i.e., total area of small items in 2-D) in $\tilde{I}$.

Consider the following configuration LP: LP($\tilde{I}$)

$$
\begin{array}{rrcll}
\text{Min} & \sum_r x_r & & & \\
\text{s.t.} & \sum_r c^r_{B_j} x_r & \geq & |B_j| & \forall j \in [p_1], \\
& \sum_r c^r_{L_k} x_r & \geq & w(L_k) & \forall k \in [p_2], \\
& \sum_r c^r_{W_\ell} x_r & \geq & h(W_\ell) & \forall \ell \in [p_3], \\
& \sum_r c^r_{T} x_r & \geq & vol_T(\tilde{I}), & \\
& x_r & \geq & 0 & \forall r \in [m].
\end{array}
$$

Let $t = p_1 + p_2 + p_3 + 1$. As the LP has only t constraints,

$$\text{IP}^*(\tilde{I}) \leq \text{LP}^*(\tilde{I}) + t. \tag{6}$$

where IP$^*(\tilde{I})$ and LP$^*(\tilde{I})$ are the optimal integral and fractional solutions for LP($\tilde{I}$) respectively.

Intuitively, the integral solution of LP gives a fractional packing of items. Big items are packed integrally. For long items it preserves the total width in each class of same height. Similarly for wide items it preserves the total height in each class of same width. However individual long (resp. wide) items may be needed to be sliced vertically (resp. horizontally) and packed fractionally in their small dimension. For small items it preserves the total volume, but the items might be needed to pack fractionally (may be sliced vertically and/or horizontally).

Now any packing of such sliced rectangles can be transformed into a feasible packing using the following lemma, implicit in Section 3.3.2 in [173].

Lemma 3.2.1. [173] For any $\epsilon_b > 0$, if there is a packing of $\tilde{I}$ into m bins, where skewed items are allowed to be sliced along their smaller dimension and small items are allowed to be sliced along horizontally and/or vertically to be packed into containers, then one can get a packing into $(1 + \epsilon_b) \cdot m + O(1)$ bins where all items are packed integrally.

Hence, we get,

$$\begin{aligned}\mathcal{A}(\tilde{I}) &\leq (1+\epsilon_a)\cdot \mathsf{Opt}(\tilde{I}) + O(1) && (7)\\ &\leq (1+\epsilon_a)(1+\epsilon_b)\cdot \mathrm{IP}^*(\tilde{I}) + O(1) && (8)\\ &\leq (1+\epsilon_a)(1+\epsilon_b)\cdot (\mathrm{LP}^*(\tilde{I}) + t) + O(1) && (9)\\ &\leq (1+\epsilon_a)(1+\epsilon_b)\cdot \mathrm{LP}^*(\tilde{I}) + O(1). && (10)\end{aligned}$$

Here inequality (7) follows from the fact that $\mathcal{A}$ finds optimal solution within factor $(1+\epsilon_a)$. Inequality (8) follows from Lemma 3.2.1 and inequality (9) follows from (6).

R&A framework for rounding based algorithms.

We can now show the following.

Theorem 3.2.2. For any $\epsilon > 0$, if there is a ρ approximation algorithm $\mathcal{A}$ that rounds the large coordinates of items to $O(1)$ types before packing (these sizes could depend on the instance I), then the R&A method gives a $(1+\ln\rho+\epsilon)$ asymptotic approximation bin packing for I.

Proof. First, we consider the configuration LP on original items in Step 1 of the R&A framework (see Section 2.6.3) and apply the randomized rounding step to it. The probability that an item $i \in I$ is not covered by any configurations selected in some iteration, is at most $(1 - \sum_{C\ni i} x^*_C/z^*)$. Let S be the set of residual items not covered by any of the bins selected in $\lceil(\ln\rho)z^*\rceil$ iterations. Thus the probability that i is not covered in any of the $\lceil(\ln\rho)z^*\rceil$ independent iterations is at most:

$$\begin{aligned}\mathbb{P}(i\in S) &\leq (1-\sum_{C\ni i} x^*_C/z^*)^{\lceil(\ln\rho)z^*\rceil}\\ &\leq (1-\sum_{C\ni i} x^*_C/z^*)^{(\ln\rho)z^*}\\ &\leq (1-1/z^*)^{(\ln\rho)z^*} && (11)\\ &\leq e^{(-\ln\rho)} && (12)\\ &= \frac{1}{\rho}, && (13)\end{aligned}$$

where inequality (11) follows as $\sum_{C \ni i} x^*_C \geq 1$ for all $i \in I$ and inequality (12) follows from $(1 - x^{-1})^{\alpha x} \leq e^{-\alpha}$ for $x > 0$.

Let $\mathsf{Opt}(I)$ be the number of bins used in the optimal packing of I. Now in Step 2 of R & A framework at most $\lceil (\ln \rho) z^* \rceil \leq 1 + (\ln \rho) \cdot \mathsf{Opt}$ bins were used. Let S denote the set of items that are still unpacked. It remains to bound the number of bins used for packing S using $\mathcal{A}$.

To this end, consider the rounding that $\mathcal{A}$ would apply to the items when given instance I, and consider the instance obtained by applying this rounding to items in S. Let us denote this instance as $\tilde{I} \cap S$. Now consider the following configuration LP for $\tilde{I} \cap S$: LP($\tilde{I} \cap S$)

$$\begin{array}{rrcll}
\text{Min} & \sum_r x_r & & & \\
\text{s.t.} & \sum_r c^r_{B_j} x_r & \geq & |B_j \cap S| & \forall j \in [p_1], \\
& \sum_r c^r_{L_k} x_r & \geq & w(L_k \cap S) & \forall k \in [p_2], \\
& \sum_r c^r_{W_l} x_r & \geq & h(W_\ell \cap S) & \forall \ell \in [p_3], \\
& \sum_r c^r_T x_r & \geq & vol_T(\tilde{I} \cap S), & \\
& x_r & \geq & 0 & \forall r \in [m].
\end{array}$$

Now by (13), we get the following expected values:

$$\mathbb{E}[|B_j \cap S|] \leq |B_j|/\rho,$$

$$\mathbb{E}[w(L_k \cap S)] \leq w(L_k)/\rho,$$

$$\mathbb{E}[h(W_\ell \cap S)] \leq h(W_\ell)/\rho,$$

$$\mathbb{E}[vol_T(\tilde{I} \cap S)] = vol_T(\tilde{I})/\rho.$$

Now we will show that these values are in fact not very far from the expectations using a concentration inequality.

We will need the following concentration inequality [157] in the analysis.

Lemma 3.2.3. (Independent Bounded Difference Inequality) [157]
Let $X = (X_1, \dots, X_n)$ be a family of independent random variables, with $X_j \in A_j$ for $j = 1, \dots, n$, and $\phi : \prod_{j=1}^{n} A_j \to \mathbb{R}$ be a function such that

$$\phi(x) - \phi(x') \leq c_j,$$

whenever the vectors x and x' differ only in the j-th coordinate. Let $\mathbb{E}[\phi(X)]$ be the expected value of the random variable $\phi(X)$. Then for any $t \geq 0$,

$$\mathbb{P}[\phi(X) - \mathbb{E}(\phi(X)) \geq t] \leq e^{-2t^2/\sum_{i=1}^{n} c_j^2}.$$

Consider the function $\phi_i^B(x) = B_i \cap S$, i.e., the number of items of type B_i in S. This is a function of $X = (X_1, \dots, X_{\lceil (\ln \rho) z^* \rceil})$, i.e., $\lceil (\ln \rho) z^* \rceil$ independent random variables (selected configurations in randomized rounding phase of R & A framework) where X_i corresponds to the random variable for the configuration selected in the ith iteration. Now changing value of any of these random variables may lead to the selection of a different configuration C' in place of configuration C in the corresponding iteration. Let vectors x and x' differ only in j'th coordinate, i.e., a different configuration C' is selected in place of configuration C in j th iteration. This might lead to a different set of residual items S'. Then

$$\begin{aligned} \phi_i^B(x) - \phi_i^B(x') &\leq (|B_i \cap S| - |B_i \cap S'|) \\ &\leq max\{|B_i \cap C|, |B_i \cap C'|\} \\ &\leq 1/f(\epsilon). \end{aligned} \tag{14}$$

Here inequality (14) follows from the fact that there can be at most $1/f(\epsilon)$ items of type B_i in a bin, as big items are $\geq f(\epsilon)$ in at least one of the dimensions.

Therefore, from the above-mentioned independent bounded difference inequality, we get,

$$\mathbb{P}[|B_i \cap S| - \mathbb{E}(B_i \cap S) \geq \gamma z^*] \leq e^{-2(\gamma z^*)^2/(\frac{\lceil (\ln \rho) z^* \rceil}{(f(\epsilon))^2})}$$

where $z^* = LP^*(\tilde{I})$.

Similarly consider the function $\phi_k^w(x) = w(L_k \cap S)$. We get,

$$\begin{aligned} \phi_k^w(x) - \phi_k^w(x') &\leq (w(L_k \cap S) - w(L_k \cap S)) \\ &\leq max\{w(L_k \cap C), w(L_k \cap C')\} \\ &\leq 1/f(\epsilon). \end{aligned} \tag{15}$$

Therefore, once again from independent bounded difference inequality, we get,

$$\mathbb{P}[w(L_k \cap S) - \mathbb{E}(w(L_k \cap S)) \geq \gamma z^*] \leq e^{-2(\gamma z^*)^2/(\frac{\lceil (\ln \rho) z^* \rceil}{(f(\epsilon))^2})}.$$

Similarly for wide items we get,

$$\mathbb{P}[h(W_\ell \cap S) - \mathbb{E}(h(W_\ell \cap S))) \geq \gamma z^*] \leq e^{-2(\gamma z^*)^2/(\frac{\lceil (\ln \rho) z^* \rceil}{(f(\epsilon))^2})}.$$

For small items, we take $\phi^T(x) = vol(T \cap S)$ and we get $\phi^T(x) - \phi^T(x') \leq 1$ as the total volume of small items in a bin is at most 1. Thus,

$$\mathbb{P}[vol_T(\tilde{I} \cap S) - \mathbb{E}(vol_T(\tilde{I} \cap S))) \geq \gamma z^*] \leq e^{-2(\gamma z^*)^2/(\frac{\lceil (\ln \rho) z^* \rceil}{(f(\epsilon))^2})}.$$

Thus in the asymptotic case (when z^* is sufficiently large, i.e., $>> \frac{t \cdot \ln \rho}{\gamma^2 (f(\epsilon))^2}$) we can take the union bound over all t cases and with high probability for all large item classes, $\phi_i^B(X) - \mathbb{E}(\phi_i^B(X)) < \gamma z^*$, $w(L_k \cap S) - \mathbb{E}(w(L_k \cap S)) < \gamma z^*$, $h(W_\ell \cap S) - \mathbb{E}(h(W_\ell \cap S)) < \gamma z^*$, and $vol_T(\tilde{I} \cap S) - \mathbb{E}(vol_T(\tilde{I} \cap S)) < \gamma z^*$.

Let ϵ_x be a suitable small constant that we will choose later. Now take $\gamma = \frac{\epsilon_x}{5t\rho}$, then from each of t classes, items of volume at most γz^* can be removed and packed into a total of $\frac{\epsilon_x z^*}{\rho}$ bins. Let us call J to be these removed elements. Then after removing these items with high probability, the right hand side for each constraint in LP$((\tilde{I} \cap S) \setminus J)$ is at most $1/\rho$ times the right hand side of the corresponding constraint in LP$(\tilde{I})$.

Thus,

$$\text{LP}(\tilde{I} \cap S) \leq \text{LP}((\tilde{I} \cap S) \setminus J) + \frac{\epsilon_x z^*}{\rho} \leq \frac{(1 + \epsilon_x)}{\rho} \text{LP}^*(\tilde{I}). \tag{16}$$

This gives us that,

$$
\begin{aligned}
\mathcal{A}(\tilde{I} \cap S) &\leq (1+\epsilon_a)(1+\epsilon_b) \cdot \mathrm{LP}^*(\tilde{I} \cap S) && (17) \\
&\leq (1+\epsilon_a)(1+\epsilon_b) \cdot \frac{(1+\epsilon_x)}{\rho} \mathrm{LP}^*(\tilde{I}) + O(1) && (18) \\
&\leq \frac{(1+\frac{\epsilon}{2})}{\rho} \mathsf{Opt}(\tilde{I}) + O(1) && (19) \\
&\leq (1+\frac{\epsilon}{2}) \mathsf{Opt}(I) + O(1). && (20)
\end{aligned}
$$

Here, inequality (17) follows from (10). Inequality (18) follows from (16), inequality (19) follows by choosing ϵ_x such that $(1+\frac{\epsilon}{2}) = (1+\epsilon_a)(1+\epsilon_b)(1+\epsilon_x)$ and the last inequality follows by (5).

Thus in Step 2 at most $\lceil (\ln \rho) z^* \rceil \leq 1 + (\ln \rho) \cdot \mathsf{Opt}(I)$ bins were used. Medium items are packed into at most $(\frac{\epsilon}{2} \cdot \mathsf{Opt}(I) + O(1))$ additional bins and at most $(1+\frac{\epsilon}{2})\mathsf{Opt}(I) + O(1)$ bins were used to pack the remaining items. This gives the desired $(1 + \ln \rho + \epsilon)$ asymptotic approximation. □

The above algorithm can be derandomized using standard techniques as in [13].

3.3 A rounding based 1.5-approximation algorithm

In this section we present the Jansen-Prädel algorithm [116] that rounds the large dimensions of items into $O(1)$ types of sizes before packing them into bins.

3.3.1 Technique

The algorithm works in two stages. In the first stage, the large dimensions of items in the input instance are rounded to $O(1)$ (specifically $O(\frac{1}{\epsilon^2})$) types. By guessing structures of the rounded items, one guesses the rounded values and how many items are rounded to each such value. In the second stage rounded rectangles are packed into bins. The algorithm uses the following structural theorem.

Theorem 3.3.1. [116] For any $\epsilon_c > 0$, and for any solution that fits into m bins, the widths and the heights of the rectangles can be rounded up so that they fit into

$(3/2+5\epsilon_c)m+O(1)$ bins, while the packing of each of the bins satisfies either *Property 1.1* (The width of each rectangle in bin $\mathcal{B}_i$ of width at least ϵ_c is a multiple of $\frac{\epsilon_c^2}{2}$) or *Property 1.2* (The height of each rectangle in bin $\mathcal{B}_i$ of height at least ϵ_c is a multiple of $\frac{\epsilon_c^2}{2}$).

Using the above structural theorem they show that, given any optimal packing, one can remove all items intersected with a thin strip in the bin and round one side of all remaining items to some multiple of $\epsilon_c^2/2$. Then they pack the cut items separately to get a packing into at most $(3/2+\epsilon)\cdot\mathsf{Opt}+O(1)$ bins that satisfy either Property 1.1 or Property 1.2. After rounding one side of the rectangle, the other side is rounded using techniques similar to those used by [138]. In this version of the algorithm after items are rounded to $O(1)$ types, we can find the optimal packing of these rounded items by brute-force. The algorithm is actually guessing the structure of optimal packing, i.e., rounded values for each item, to use the structural theorem to get a feasible packing in $\leq (\frac{3}{2}+\epsilon)\mathsf{Opt}+O(1)$ bins. The main structure of their algorithm is described below:

Input: A set of items $I := \{r_1, r_2, \cdots, r_n\}$ where $r_j \in (0,1] \times (0,1]$ for all $j \in [n]$ and set of bins of size 1×1.

Output: An orthogonal packing of I.

Algorithm:

1. **Guess Opt:** Guess Opt by trying all values between 1 and n.

2. *For* each guessed values of Opt *do*

(a) **Classification of rectangles:** Compute δ using methods similar to [122] to classify rectangles and pack medium rectangles using Next Fit Decreasing Height (NFDH).

(b) **Guessing structures:** Enumerate suitably over all structures (sizes to which items are rounded to, and for each size the number of items that are rounded to that size) of the set of big, long, wide rectangles and the set of wide and long containers.

(c) *For* each guess *do* **Packing:**

  - Assign big rectangles by solving a flow-network with the algorithm of Dinic [59];
  - Do greedy Assignment of long and wide rectangles into $O(1)$ groups;
  - Pack $O(1)$ number of groups of long and wide rectangles into $O(1)$ types of containers using a linear program;
  - Pack the small rectangles using Next Fit Decreasing Height;
  - Pack $O(1)$ types of containers and $O(1)$ types of big rectangles into bins using brute force or mixed integer linear programs [131];

3. *Return* a feasible packing;

3.3.2 Details of the Jansen-Prädel Algorithm:

In this section we describe the algorithm in [116], to fit in our framework. For more details on the algorithm, we refer the readers to [173].

3.3.2.1 Binary Search for Opt*:*

Using binary search between the number of rectangles (an upper bound) and total area of the rectangles (a natural lower bound), the algorithm finds the minimum m such that there exists a feasible solution with $(3/2 + \epsilon) \cdot m + O(1)$ bins. For each guess of Opt, we first guess the rounding in the following way and then we pack the rounded items into the bins.

3.3.2.2 Classification of Rectangles:

A value δ ($\leq \epsilon' = \epsilon/48$) is selected similar to [122], such that $\frac{1}{\delta}$ is a multiple of 24 and rectangles with at least one of the side lengths between δ and δ^4 have a small area ($\leq \epsilon' \cdot \mathsf{Opt}$). Now we classify the rectangles into five types:

- Big: both width and height is at least δ.
- Wide: width is at least δ, height is smaller than δ^4.
- Long: height is at least δ, width is smaller than δ^4.
- Small: both width and height is less than δ^4.
- Medium: either width or height is in $[\delta^4, \delta)$. As medium rectangles are of total size $\leq \epsilon' \cdot \mathsf{Opt}$, they can be packed using NFDH into at most additional $O(\epsilon' \cdot \mathsf{Opt})$ bins. Choose $\epsilon_c = \delta$.

3.3.2.3 Further Classification:

First let us show that given any optimal packing, we can get a packing in at most $(\frac{3}{2} + \epsilon)\mathsf{Opt} + O(1)$ bins where all large dimensions of items are rounded to $O(1)$ types. Then we will guess the structure of optimal packing to assign items to its rounded type.

Assuming we are given the optimal packing, we can get rounding of one side by using Theorem 3.3.1. Now we will find the rounding of the other side using linear grouping. Let $\mathsf{Opt} = m$ and out of these m bins, m_1 bins $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_{m_1}$ are of type 1 (that satisfy Property 1.1, i.e., the width and the x-coordinate of each rectangle in $\mathcal{B}_i$ of width at least ϵ_c is a multiple of $\frac{\epsilon_c^2}{2}$) and the remaining m_2 $(= m - m_1)$ bins $\mathcal{B}_{m_1+1}, \mathcal{B}_{m_1+2}, \ldots, \mathcal{B}_m$ are of type 2 (that satisfy Property 1.2, i.e., the height and the y-coordinate of each rectangle in $\mathcal{B}_i$ of height at least ϵ_c is a multiple of $\frac{\epsilon_c^2}{2}$). Thus the widths of big and wide rectangles in bins of type 1 and the heights of big and long rectangles in bins of type 2 are rounded to some multiples of $\delta^2/2$. Let B_i^w and W_i^w be the set of big and wide rectangles, respectively, that are packed in type 1 bin with widths rounded to $i \cdot \delta^2/2$ for $i \in \{2/\delta, 2/\delta+1, \ldots, 2/\delta^2\}$. Similarly, let B_i^h and L_i^h be the set of big and long rectangles, respectively, that are packed in type 2 bin with heights rounded to $i \cdot \delta^2/2$. Let L^w and W^h be the set of long rectangles in type 1 bin and the set of wide rectangles in type 2 bins, respectively. Set of small and medium rectangles are denoted by M and S respectively.

3.3.2.4 Rounding of big, long and wide rectangles:

The rounded widths of rectangles in B_i^w and W_i^w and rounded heights of rectangles in B_i^h and L_i^h are known. In this step we find the rounding of heights of rectangles in B_i^w and L^w and rounding of widths of rectangles in B_i^h and W^h using linear grouping techniques similar to Kenyon-Rémila [138] and introduced by Fernandez de la Vega and Lueker [55] .

For any set B_i^w, we sort the items $r_i^1, r_i^2 \cdots r_i^{|B_i^w|}$ according to non-increasing height, i.e., $h(r_i^e) \geq h(r_i^f)$ for $e \leq f$. Now define at most $\frac{1}{\delta^2}$ subsets $B_{i,j}^w$, each of which contains $\lceil \delta^2 \cdot |B_i^w| \rceil$ rectangles except possibly for the last subset. For any two rectangles $r_A \in B_{i,j_1}^w$ and $r_B \in B_{i,j_2}^w$ and $j_2 \geq j_1$, $h(r_A) \geq h(r_B)$. We round the heights of all rectangles in each set $B_{i,j}^w$ to the height of the tallest rectangle in the subset (we call it to be the *round rectangle* of the subset). Set apart the set $B_{i,1}^w$ and for other rectangles in $B_{i,j}^w$ place them in the position of rectangles of $B_{i,j-1}^w$. This is possible as all subsets (except possibly the last) have the same cardinality and all rectangles have same width.

Similarly, sort long rectangles in L^w according to non-increasing height. We divide the set L^w into at most $\frac{1}{\delta^2}$ subsets $L_1^w, \ldots, L_{1/\delta^2}^w$ such that every subset has total width $\delta^2 \cdot w(L^w)$. If needed, items are sliced vertically to create subsets. For any two rectangles $r_A \in L_{j_1}^w$ and $r_B \in L_{j_2}^w$ and $j_2 \geq j_1$, $h(r_A) \geq h(r_B)$. We round the height of each rectangle to the height of the tallest rectangle in it. Apart from L_1^w, rectangles of L_j^w are packed in the position of rectangles of L_{j-1}^w.

The rectangles in $L_1^w, B_{2/\delta,1}^w, \cdots, B_{2/\delta^2,1}^w$ are packed separately into additional bins using NFDH. Note that the width of all rectangles in $L^w, B_{2/\delta}^w, \cdots, B_{2/\delta^2}^w$ is at most $\frac{1}{\delta} \cdot m_1$ as each rectangle has height at least δ. So, $w(L_1^w) + w(B_{2/\delta,1}^w), \ldots, w(B_{2/\delta^2,1}^w) \leq \delta^2 . w(L^w) + w(B_{2/\delta}^w), \cdots, w(B_{2/\delta^2}^w) \leq \delta^2 \cdot \frac{1}{\delta} \cdot m_1 = \delta \cdot m_1$. Thus the total area of the rectangles in $L_1^w \cup B_{2/\delta,1}^w \cup \cdots \cup B_{2/\delta^2,1}^w$ is $O(\delta \cdot m_1)$ and thus can be packed into additional $O(\delta \cdot m_1)$ bins using NFDH.

Widths of rectangles in B_i^h, W^h are rounded in a similar manner.

3.3.2.5 Rounding of containers:

We have not so far rounded the width of long rectangles and heights of wide rectangles. Now we construct rectangular containers for the wide and long rectangles for that purpose. We only show the rounding of containers for type 1 bins. Rounding containers for type 2 bins can be done analogously. Let C_L^w be the set of containers for long rectangles and C_W^w be the set of containers for wide rectangles in type 1 bins. Define $2/\delta^2$ vertical slots of width $\delta^2/2$ in each type 1 bin $\mathcal{B}_i$. A long container is part of a slot that contains at least one long rectangle, and the container is bounded at the top and bottom by a wide or big rectangle, or the boundary (ceiling or floor) of the bin. There can be at most $(1/\delta - 1)$ long containers in a slot. Thus there are at most $O(\delta^3)$ long containers per bin.

Next, construct wide containers by extending upper and lower edges of big rectangles and long containers in both directions till they hit another big rectangle, a long container or the boundary (left or right side of bin). Wide and small rectangles are horizontally cut by these lines. As there are $O(\delta^3)$ big rectangles and long containers, there are $O(\delta^3)$ wide containers in $\mathcal{B}_i$. This way any packing in optimal bin is transformed into a packing of big rectangles and long and wide containers. Note that skewed and small rectangles might be packed fractionally in the long and wide containers.

Now we do the rounding of containers. Heights of all containers in C_W^w are rounded down to the nearest multiple of δ^4, cutting the uppermost wide and short rectangles. There are $O(1/\delta^3) \cdot m_1$ wide containers and note that the small rectangles have height less than δ^4. Thus the cut wide and short rectangles are packed using NFDH in additional $O(\delta \cdot m_1)$ bins. For long containers we remove the short rectangles and push all long rectangles vertically down till they touch the top of another rectangle or the boundary. Then we round down the heights to either the nearest multiple of δ^4 or a combination of rounded heights of the long rectangles. Note that these heights are rounded down, although large, but still to $O(1)$ number of types. Total area loss for each container is $O(\delta^4 \cdot \delta^2/2)$ and the number of long containers is $O(1/\delta^3)$. So in the reduced container we pack the small items till we can and the remaining small rectangles are packed into additional $O(\delta^3 \cdot m_1)$ bins using NFDH.

Similarly long containers can be constructed for the additional bins that are used to pack items of L_1^w. These bins will have at most $(2\delta \cdot m_1 + 1) \cdot 2/\delta^2$ long containers of width $\delta^2/2$ and height 1. Note that there are $O(1/\delta^2)^{1/\delta}$ possible heights of containers, which can be reduced to $O(1/\delta^2)$ heights using linear grouping and losing only a small constant.

Thus at the end of the rounding of containers, the containers have the following properties:

2.1. There are at most $O(1/\delta^3) \cdot m_1$ wide containers in C_W^w with width of a multiple of δ^2 and height of a multiple of δ^4.

2.2. There are at most $O(1/\delta^3) \cdot m_2$ long containers in C_L^h with height of a multiple of δ^2 and width of a multiple of δ^4.

2.3. There are at most $O(1/\delta^3) \cdot m_2$ wide containers in C_W^h with $O(1/\delta^2)$ different widths (either a multiple of δ^4 or a combination of rounded width of wide rectangles in W^h) and height $\delta^2/2$.

2.4. There are at most $O(1/\delta^3) \cdot m_2$ long containers in C^w_L with $O(1/\delta^2)$ different heights (either a multiple of δ^4 or a combination of rounded height of long rectangles in L^w) and width $\delta^2/2$.

At the end of the rounding step we get the following theorem.

Theorem 3.3.2. [116] Given an optimal packing I into m bins, it is possible to round the widths and heights of the long, wide and big rectangle to $O(1)$ types such that it fits in at most $(3/2 + O(f_1(\delta)))m + O(f_2(\delta))$ bins for some functions f_1 and f_2 and these bins satisfy either Property 1.1 or Property 1.2. Furthermore the heights of long and big rectangles in L^w and B^w, widths of wide and big rectangles in W^h and B^h are rounded up to $O(1/\delta^2)$ values. The wide and long rectangles are sliced horizontally and vertically, respectively, and packed into containers satisfying Properties 2.1-2.4 and small rectangles are packed fractionally into the wide and long containers. Medium rectangles are packed separately into $O(\delta)$ bins.

3.3.2.6 Transformation of rectangles:

Now we guess the structure of the optimal packing for the assignment of rectangles to the rounded rectangles.

First we have to determine whether width or height of a big rectangle is rounded to a multiple of $\delta^2/2$. We guess the cardinality of sets B^w_i and B^h_i for $i \in \{2/\delta, 2/\delta + 1, \cdots, 2/\delta^2\}$. This can be done by choosing less than $2 \cdot (2/\delta^2)$ values out of n; note that this is polynomial in n. For each such guess we also guess $2 \cdot (2/\delta^2) \cdot (1/\delta^2)$ round rectangles out of n rectangles. These values give us the structure of subsets as discussed in the rounding of big rectangles. Now to find the assignment of big rectangles to these subsets, we create a directed flow network $G = (V, E)$. First we create source(s) and target node(t). For each rectangle $r \in I$, we create a node and add an edge from s to r with capacity one. Next we create nodes for all subsets $B^w_{i,j}$

and $B^h_{i,j}$ and add an edge from r to $B^y_{i,j}$ of capacity one, if r might belong to $B^y_{i,j}$ where $y \in \{w, h\}$. Next add edges between nodes corresponding to subsets and the target node of infinite capacity. Now apply Dinic's algorithm [59] or any other flow algorithm to find if there is an $s - t$ flow of value the same as the number of big rectangles. If there exists such a flow, we get a valid assignment of big rectangles into subsets. On the other hand, if there is no such flow then continue with the other guesses.

Next we need to transform the wide and long rectangles. First we need to decide whether a wide rectangle belongs to type 1 or type 2 bins. The case for long rectangle is analogous. Note that in the linear grouping of wide rectangles in W^h, $1/\delta^2$ subsets were created. The total height of all rectangles in W^h is bounded by $n \cdot \delta^4$. So we approximately guess the total height of W^h, by choosing some $t \in \{1, 2, \cdots, n\}$, so that $t \cdot \delta^4 \leq h(W^h) \leq (t + 1) \cdot \delta^4$. As all subsets $W^h_1, W^h_2, \cdots, W^h_{1/\delta^2}$ have the same height, each subset will have height $h(W^h) \cdot \delta^2$. We also guess the height of rectangles to which all rectangles in each subset are rounded. This can be done by choosing $1/\delta^2$ rectangles out of n rectangles. Thus we can approximately guess the structure of the rectangles in W^h. Remaining wide rectangles are in W^w i.e., their width are rounded to the next multiple of $\delta^2/2$. We guess approximately the total height of the rectangles in $W^w_{2/\delta}, \cdots, W^w_{2/\delta^2}$ by choosing $(2/\delta^2 - 2/\delta + 1)$ integral values t_k such that $t_k \cdot \delta^4 \leq h(W^w_j) < (t_k + 1) \cdot \delta^4$. Thus we can guess the structure of all subsets of wide rectangles and we need to assign the wide rectangles into these subsets. For the assignment we sort the wide rectangles in non-increasing order. We assign wide rectangles greedily into all sets $W^w_k \in W^w$, starting from W^w_{2/δ^2} and continue until the total height exceeds $(t_k + 1) \cdot \delta^4$ for each set W^w_k. The remaining rectangles are similarly greedily assigned into sets $W^h_1, \cdots, W^h_{1/\delta^2}$. It is easy to show that for the right guesses of t_k values, all rectangles will have a valid assignment. Afterwards we remove the shortest rectangles in each subset to reduce the height to at most $t_k \cdot \delta^4$.

It can be shown that the total height of these removed wide rectangles is $O(\delta^2)$ and thus can be packed into $O(1)$ additional bins.

3.3.2.7 Construction of containers:

Here we describe the construction of long and wide containers that are placed in type-1 bins. The construction of long and wide containers that are placed in type-2 bins, is analogous.

Each wide container in C_W^w has height of a multiple of δ^4 and width of multiple of $\delta^2/2$. Hence we can guess $n_{i,j}^w$, number of wide containers that has width $i\delta^2/2$ and height $j \cdot \delta^4$ by choosing $1/\delta^4 \cdot 2/\delta^2$ values out of n.

Similarly long containers in C_L^w have the same width and $O(1/\delta^2)$ types of heights (either a combination of rounded heights of long rectangles or a multiple of δ^4) which we can guess.

3.3.2.8 Packing long and wide rectangles into containers

There are four cases: packing of long rectangles into long containers in type 1 bins, packing of long rectangles into long containers in type 2 bins, packing of wide rectangles into wide containers in type 1 bin and packing of wide rectangles into wide containers in type 2 bins. Here let us only consider the wide containers in type 1 bins, other cases can be handled similarly. There are $O(\delta^3)$ types of wide containers and $O(\delta^2)$ types of wide rectangles. Wide rectangles are packed into containers using a linear programs as in Kenyon-Rémilla [138]. Then we add back the small rectangles using NFDH in the empty regions of the $O(\delta^3)$ types of containers to the extent we can.

3.3.2.9 Complete packing

Now we have big rectangles and containers of $O(1)$ type, thus there are $O(1)$ number of possible configurations of packing of big rectangles and containers into bins. We try out all configurations by brute force to find the optimal packing of big rectangles and containers. Then we add back the remaining rectangles using NFDH in the empty regions of the bin till we can and then we pack the remaining rectangles into additional bins.

3.3.3 Analysis

In the rounding step, separate packing of rectangles in $L_1^w, B_{2/\delta,1}^w, \cdots, B_{2/\delta^2,1}^w, W_1^h$, $B_{2/\delta,1}^h, \cdots, B_{2/\delta^2,1}^h$ need at most $O(\delta \cdot \mathsf{Opt})$ additional bins. In rounding containers the cut wide, long and small rectangles are packed into additional $O(\delta \cdot \mathsf{Opt})$ bins. Packing of medium and remaining small rectangles take $O(\delta \cdot \mathsf{Opt})$ bins. Removed wide rectangles in the step of transformation of wide rectangles require $O(1)$ extra bins. So using the structural theorem, total $(3/2+O(\delta))\mathsf{Opt}+O(\delta)$ bins are sufficient.

The running time of the steps are given as follows. The binary search requires $O(\log n)$ time. Computing δ in a method similar to [122] takes $O(n/\epsilon)$ time. For the structure of the set of big rectangles, we guess $O(1/\delta^2)$ values out of n to guess the cardinality of the sets and for such a guess, $O(1/\delta^4)$ round rectangles are guessed. Similarly, we get the structure of wide and long rectangles, we guess $O(1/\delta^3)$ values out of n. Structure of long and wide containers require guessing $O(1/\delta^6)$ values out of n and guessing $O(1/\delta^2)$ values out of $O(1/\delta^4 + (1/\delta^2)^{1/\delta})$ respectively. Solving the flow network takes $O(n^3)$ time. Assignment of wide and long rectangles into groups will take $O(n \log n)$ time. The running time for packing containers and big rectangles using the brute force method, is a large constant, triple exponential in δ. It can be reduced using integer programs of Kannan et al. [131]. Packing medium and small rectangles using NFDH require $O(n \log n/\delta^3)$ time. In total the running time is bounded by $O(n^{h_1(1/\epsilon)} \cdot h_2(1/\epsilon))$, where h_1, h_2 are polynomial functions. Thus the total running time is polynomial for a fixed $\epsilon > 0$.

3.3.4 Bin packing with rotations

Bin packing with rotation is almost similar to the packing without rotation. When rotation is allowed we only have bins with a packing that satisfy Property 1.1. Remaining rounding steps are analogous to the versions without rotations. The step of transformation of rectangles, however, is slightly different when we allow rotations. For big rectangles, in the flow network we connect a big rectangle with all subsets that can contain the rectangle before and after rotating by 90°. On the other hand, for transformation of wide and long rectangles, we approximately guess $w(L^w)$ and the heights of the sets $W^w_{2/\delta}, \cdots, W^w_{2/\delta^2}$ and the height of the round rectangles in L^w. Now we can rotate all long rectangles to have only wide rectangles and greedily assign them to the wide rectangles in $W^w_{2/\delta}, \cdots, W^w_{2/\delta^2}$. The remaining wide rectangles are rotated back and assigned to the sets $L^w_1 \cdots, L^w_{1/\delta^2}$. The analysis is also similar, however, gives slightly better additive constants in the approximation.

3.4 Lower bound for rounding based algorithms

In this section, we describe some limitations of rounding based algorithms.

Theorem 3.0.11. (restated) Any rounding algorithm that rounds at least one side of each large item to some fixed constant independent of the problem instance (let us call such rounding input-agnostic), cannot have an approximation ratio better than 3/2.

Proof. Consider an input-agnostic algorithm $\mathcal{A}$ that rounds at least one side of each large item to one of the values $c_1, c_2 \dots, c_z$, that are chosen independent of the input instance. Let i and j be such that $c_i < 0.5 \leq c_{i+1}$ and $c_{j-1} \leq 0.5 < c_j$. Let $f = \min\{0.5 - c_i, c_j - 0.5\}$. Here we assume the algorithm rounds identical items with the same height and width to the same types.

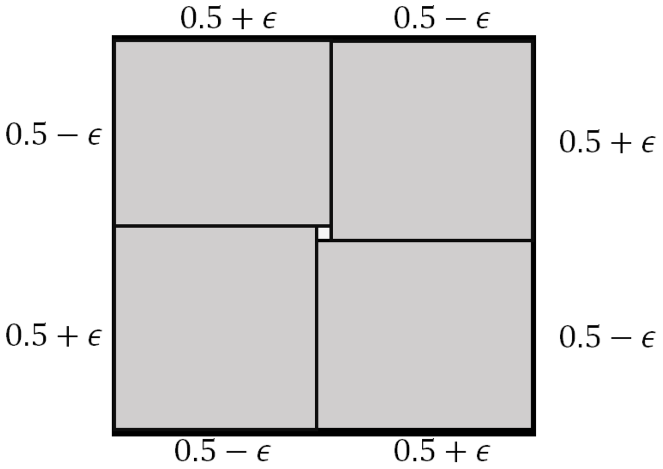


Figure 7: Lower bound example for rounding based algorithms

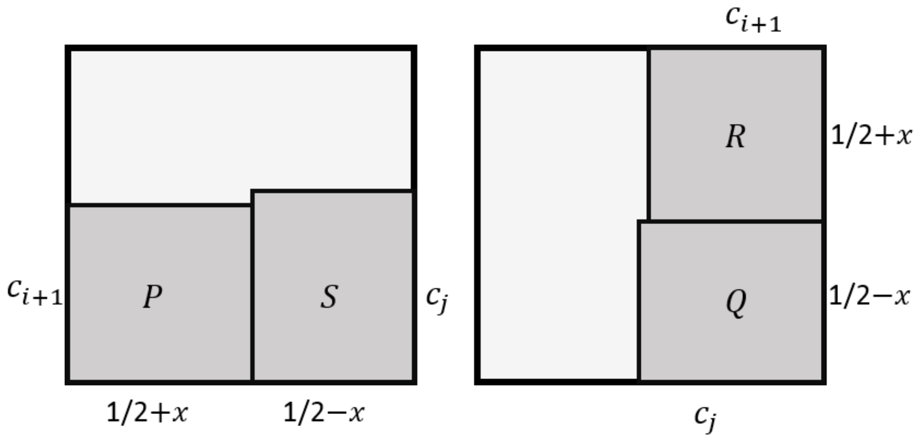


Figure 8: The case when $C_{i+1} > 1/2$

Now consider an optimum packing using $m = 2k$ bins where each bin is packed as in Figure 7, for some fixed $x \in (0, f)$. Under the rounding, an item $(1/2+x)\times(1/2-x)$ is rounded to either $(1/2 + x) \times (c_{i+1})$ (let us call such items of type P) or to $(c_j) \times (1/2 - x)$ (let us call such items of type Q). Similarly, each item $(1/2 - x) \times (1/2 + x)$ is rounded to either $(c_{i+1}) \times (1/2 + x)$ (call these of type R) or to $(1/2 - x) \times (c_j)$ (call these of type S).

Let us first consider the easy case when $c_{i+1} > 1/2$. It is easily checked that in this case, any bin can contain at most 2 rounded items: (i) either a P-item and a

S-item or (ii) a Q-item and a R-item. See, for example Figure 8. This implies that a 2-approximation is the best one can hope for, if 1/2 is not included among the $c_1, c_2 \ldots, c_z$.

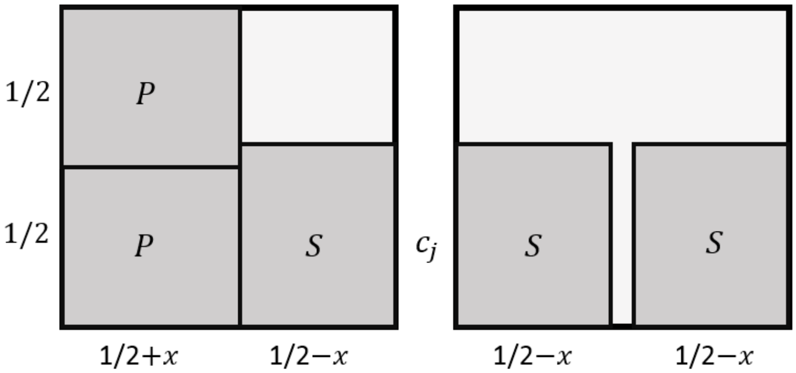


Figure 9: Configurations $\{P, P, S\}$ and $\{S, S\}$

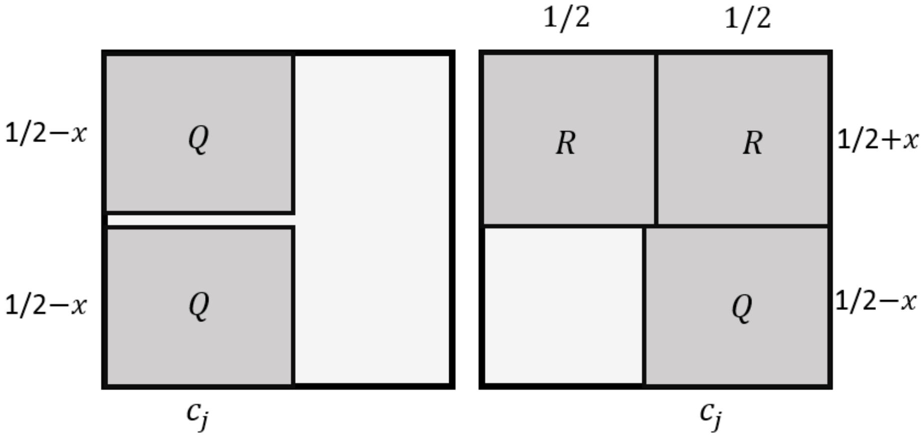


Figure 10: Configurations $\{R, R, Q\}$ and $\{Q, Q\}$

We now consider the case when $c_{i+1} = 1/2$. We claim that the possible bin configurations are:

a) [{ P,P,S } and { S,S }], which happens when the items are rounded to types P and S (see Figure 9). Or,

b) [{ R,R,Q } and { Q,Q }], which happens when items are rounded to types R and Q (see Figure 10).

Furthermore, the remaining two cases can be ignored. That is, when items are rounded to type P and R or when items are rounded to type Q and S, as in these cases at most two items can be packed into a bin.

So, let us consider case (a). The proof for case (b) is analogous. Let x_1 and x_2 denote the number of configurations of type { P,P,S } and { S,S } respectively. Then we get the following configuration LP:

$$\begin{aligned} \text{Min} \quad & x_1 + x_2 \\ \text{s.t.} \quad 2x_1 & \geq 4k \\ x_1 + 2x_2 & \geq 4k \\ x_1, x_2 & \geq 0 \end{aligned}$$

The dual is:

$$\begin{aligned} \text{Max} \quad & 4k(v_1 + v_2) \\ \text{s.t.} \quad 2v_1 + v_2 & \leq 1 \\ 2v_2 & \leq 1 \\ v_1, v_2 & \geq 0 \end{aligned}$$

A feasible dual solution is $v_1 = 0.25, v_2 = 0.5$. This gives the dual optimal as $\geq 3k$. Thus the number of bins needed is $\geq 3k = 3m/2$. □

This in particular implies that to beat 3/2 one would need a rounding that is not input-agnostic, or which rounds identical items with the same height and width to different types — sometimes rounded by width and sometimes by height.

We also note that 4/3 is the lower bound for any rounding algorithm that rounds items to $O(1)$ types. This seems to be a folklore observation, but we state it here for completeness.

Theorem 3.4.1. Any algorithm that rounds items to $O(1)$ types cannot achieve better than 4/3 approximation.

Proof. Consider the packing in Figure 7. Assume there is an optimal packing of $m = 3k$ bins where each bin is having similar packing as in Figure 7 for m different values of $x \in (0.001, 0.01]$. Note that the sum of the height and width is exactly 1 for each rectangle. If we use rounding to $O(1)$ items, then for all but $O(1)$ items i, $w(i) + h(i)$ exceed 1. Without loss of generality, we assume each item touches the boundary. Otherwise for these items, we can extend sides vertically and horizontally so that they touch the boundary or another item. As the total sum of the side lengths of a bin is 4 and each item has an intersection with the boundary of length > 1 , we can only pack 3 rounded items to a bin. Thus $4m = 12k$ items can be packed into at least $4k - O(1) = 4/3m - O(1)$ bins. This example packing is particularly interesting as it also achieves the best known lower bound of 4/3 for guillotine packing. □

3.5 Conclusion

The approach for the R&A framework described here applies directly to a wide variety of algorithms and gives much simpler proofs for previously considered problems (e.g., vector bin packing, one dimensional bin packing) [13]. As rounding large coordinates to $O(1)$ number of types is by far the most widely used technique in bin-packing type problems, we expect wider applicability of this method. In fact in the next chapter we will extend this method to further improve the approximation for vector packing.

Moreover, improving our guarantee for 2-D BP will require an algorithm that is *not* input-agnostic. In particular, this implies that it should have the property that it can round two identical items (i.e., with identical height and width) differently. One such candidate is the guillotine packing approach [18]. It has been conjectured that this approach can give an approximation ratio of 4/3. One way to show this would be to prove a structural result bounding the gap between guillotine and non-guillotine packings. At present the best known upper bound on this gap is $T_\infty \approx 1.69$ [28].

Chapter IV

VECTOR BIN PACKING

In this chapter, we consider the d-dimensional (d-D) vector bin packing (VBP) problem. First, let us briefly mention our results and then give a detailed survey of previous work, followed by technical details of our results in the later sections.

We give several improved results for d-dimensional VBP. The first of our main results is as follows:

Theorem 4.0.1. For any small constant $\epsilon > 0$, there is a polynomial time algorithm with an asymptotic approximation ratio of $(1 + \ln(1.5) + \epsilon) \approx (1.405 + \epsilon)$ for 2-D vector packing.

Our techniques for R&A framework from Chapter 3 can be extended to get a simpler proof of previous best $(1 + \ln d)$ approximation [13] for vector packing. However we will show that this $(1+\ln d)$ is a natural barrier, as rounding based algorithms can not beat d. Thus Theorem 4.6.8 gives a substantial improvement upon the current $(1 + \ln 2 + \epsilon) \approx 1.693 + \epsilon$ bound [13] for 2-D vector packing, but, more importantly, it overcomes the barrier of $(1 + \ln d)$ of the R&A framework.

Theorem 4.6.8 is based on two (perhaps surprising) ideas. First we give a $(1.5+\epsilon)$ asymptotic approximation for any $\epsilon > 0$, without the R&A framework. To do this, we show that there exists a "well-structured" 1.5-approximate solution, and then search (approximately) over the space of such solutions. However, as this structured solution (necessarily) uses unrounded item sizes, it is unclear how to search over the space of such solutions efficiently. So the key idea is to define this structure carefully based on matchings, and use a recent elegant algorithm for the multiobjective-multibudget matching problem by Chekuri, Vondrák, and Zenklusen [35]. As we show, this allows us to both use unrounded sizes and yet enumerate the space of solutions like in rounding-based algorithms. A more detailed overview can be found in Section 4.3.

The second step is to apply the subset oblivious framework to the above algorithm. There are two problems. First, the algorithm is not rounding-based. Second, even proving subset obliviousness for *any* rounding based algorithm for d-dimensional vector packing is more involved than for geometric bin-packing. Roughly, in the geometric version, items with even a single small coordinate have small area, which makes it easier to handle, while in the d-dimensional VBP such skewed items can cause problems. To get around these issues, we use additional technical observations about the structure of the d-dimensional VBP.

Another consequence of these techniques is the following tight (absolute) approximation guarantee[1], improving upon the guarantee of 2 by Kellerer and Kotov [136]. This also shows that 2-D VBP is strictly easier in absolute approximability than 2-D GBP.

Theorem 4.0.2. For any small constant $\epsilon > 0$, there is a polynomial time algorithm with an absolute approximation ratio of $(1.5 + \epsilon)$ for 2-D vector packing.

We extend the approach for $d = 2$ to give a $(d + 1)/2$ approximation (for $d = 2$, this is precisely the $3/2$ bound mentioned above) and then show how to incorporate it into R&A. However, applying the R&A framework is more challenging here and instead of the ideal $1 + \ln((d + 1)/2) \approx 0.307 + \ln d + o_d(1)$, we get the following $(\ln d + 0.807)$-approximation:

Theorem 4.0.3. For any small constant $\epsilon > 0$, there is a polynomial time algorithm with an asymptotic approximation ratio of $(1.5 + \ln(d/2) + o_d(1) + \epsilon) \approx \ln d + 0.807 + o_d(1) + \epsilon$ for d-dimensional VBP.

[1] Recall that 3/2 is tight even for one dimensional vector packing via the Partition problem, and hence for the 2-D VBP. So even though 1-D VBP and 2-D VBP have very different asymptotic approximability, they have very similar absolute approximability.

Along the way, we also prove several additional results which could be of independent interest. For example, in Section 4.5 we obtain several results related to resource augmented packing which has been studied for other variants of bin packing [122, 19]. We specifically show the following.

Theorem 4.0.4. There is a polynomial time algorithm for packing vectors into at most $(1 + 2\epsilon)\mathsf{Opt} + 1$ bins with ϵ resource augmentation in $(d - 1)$ dimensions (i.e., bins have length $(1 + \epsilon)$ in $(d - 1)$ dimensions and 1 in the other dimension), where Opt denotes the minimum number of unit vector bins to pack these vectors.

Organization. The organization of the chapter is as follows. In Section 4.1, we discuss related previous works. Then in Section 4.2, we describe some preliminaries for this chapter. Thereafter in Section 4.3, we give an overview of our algorithm before going to the technical details. In Section 4.4, we consider packing when resource augmentation is allowed in $(d-1)$ dimensions. In Section 4.5, we present the $(d+1)/2$ approximation algorithm for d-D vector packing and some related structural properties of vector packing. Also we present the algorithm with absolute approximation guarantee of 3/2. In Section 4.6 and Section 4.7, we present the improved algorithms using R & A framework for 2-D and d-dimensions respectively.

4.1 Prior Works

In this section we survey the previous work on vector packing and its variants.

4.1.1 Offline Vector Packing:

The first paper to obtain an APTAS for 1-D bin packing by Fernandez de la Vega and Lueker [55], implies a $(d + \epsilon)$ approximation for vector packing problem. Woeginger [205] showed that there exists no APTAS even for $d = 2$ unless P = NP. However some restricted class of vectors may still admit an APTAS. For example, consider the usual partial order on d dimensional vectors, where $(x_1, x_2, \ldots, x_d) \prec (y_1, y_2, \ldots, y_d)$

if and only if $x_i \leq y_i$ for all $i \in [d]$. In Woeginger's gadget for the lower bound, the items are pairwise incompatible. The opposite extreme case, when there is a total order on all items, is easy to approximate. In fact, a slight modification of de la Vega and Lueker [55] algorithm yields an APTAS for subproblems of d-dimensional VBP with constant Dilworth number. After nearly twenty years, offline results for the general case was improved by Chekuri and Khanna [34]. They gave an algorithm with asymptotic approximation ratio of $(1+\epsilon d+H_{1/\epsilon})$ where $H_k = 1+1/2+\cdots+1/k$, is the k'th Harmonic number. Considering $\epsilon = 1/d$, they show that for fixed d, vector bin packing can be approximated to within $O(\ln d)$ in polynomial time. Bansal, Caprara and Sviridenko [13] then introduced the *Round and Approx* framework and the notion of subset oblivious algorithm and improved it further to $(1 + \ln d)$. Both these algorithms run in time that is exponential in d (or worse). Yao [206] showed that no algorithm running in time $o(n \log n)$ can give better than a d-approximation.

For arbitrary d, Chekuri-Khanna [34] showed vector bin packing is hard to approximate to within a $d^{1/2-\epsilon}$ factor for all fixed $\epsilon > 0$ using a reduction from graph coloring problem. This can be improved to $d^{1-\epsilon}$ by using the following simple reduction. Let G be a graph on n vertices. In the d-dimensional VBP instance, there will be $d = n$ dimensions and n items, one for each vertex. For each vertex i, we create an item i that has size 1 in coordinate i and size $1/n$ in coordinate j for each neighbor j of i, and size 0 in every other coordinate. It is easily verified that a set of items S can be packed into a bin if and only if S is an independent set in G. Thus we mainly focus on the case when d is a fixed constant and not part of the input.

The two dimensional case has received special attention. Kellerer and Kotov [136] designed an algorithm for 2-D vector packing with worst case absolute approximation ratio as 2. On the other hand there is a hardness of 3/2 for absolute approximation ratio that comes from the hardness of 1-D bin packing.

4.1.2 Online Vector Packing

A generalization of the First Fit algorithm by Garey et al. [83] gives $d+\frac{7}{10}$ competitive ratio for the online version. Galamobos et al. [81] showed a lower bound on the performance ratio of online algorithms that tends to 2 as d grows. The gap persisted for a long time, and in fact it was conjectured in [69] that the lower bound is super constant, but sublinear.

Recently Azar et al. [7] settled the status by giving $\Omega(d^{1-\epsilon})$ information theoretic lower bound using stochastic packing integer programs and online graph coloring. In fact their result holds for arbitrary bin size $B \in \mathbb{Z}^+$ if the bin is allowed to grow. In particular, they show that for any integer $B \geq 1$, any deterministic online algorithm for VBP has a competitive ratio of $\Omega(d^{\frac{1}{B}-\epsilon})$. For $\{0,1\}$-VBP the lower bound is $\Omega(d^{\frac{1}{B+1}-\epsilon})$. They also provided an improved upper bound for $B \geq 2$ with a polynomial time algorithm for the online VBP with competitive ratio: $O(d^{1/(B-1)} \log d^{B/(B+1)})$, for $[0,1]^d$ vectors and $O(d^{1/B} \log d^{(B+1)/B})$, for $\{0,1\}^d$ vectors.

4.1.3 Vector Scheduling

For d-dimensional vector scheduling, the first major result was obtained by Chekuri and Khanna [34]. They obtained a PTAS when d is a fixed constant, generalizing the classical result of Hochbaum and Shmoys [108] for multiprocessor scheduling. For arbitrary d, they obtained $O(\ln^2 d)$-approximation using approximation algorithms for packing integer programs (PIPs) as a subroutine. They also showed that, when m is the number of bins in the optimal solution, a simple random assignment gives $O(\ln dm/\ln\ln dm)$-approximation algorithm which works well when m is small. Furthermore, they showed that it is hard to approximate within any constant factor when d is arbitrary. This $\omega(1)$ lower bound is still the present best lower bound for the offline case.

In the online setting, Meyerson et al. [160] gave deterministic online algorithms with $O(\log d)$ competitive ratio. Im et al. [111] recently gave an algorithm with $O(\log d/\log\log d)$-competitive ratio. They also show tight information theoretic lower bound of $\Omega(\log d/\log\log d)$. Surprisingly this is also the present best offline algorithm!

4.1.4 Vector Bin Covering

For d-dimensional vector bin covering problem Alon et al. [4] gave an online algorithm with competitive ratio $\frac{1}{2d}$, for $d \geq 2$, and they showed an information theoretic lower bound of $\frac{2}{2d+1}$. For the offline version they give an algorithm with an approximation guarantee of $\Theta(\frac{1}{\log d})$.

Table 4: Present state of the art for vector packing and related variants

Problem	Subcase	Best algorithm	Best lower bound
Vector Bin Packing	Offline (constant d)	$1 + \ln d$ (asymp.[2]) [13]	APX-hard [205]
	$d = 2$	$1 + \ln 2 \approx 1.69$ (asymp.) [13]	APX-hard [205]
		2 (abs.[3]) [136]	3/2 [4]
	Offline (arbitrary d)	$1 + \epsilon d + O(\ln \frac{1}{\epsilon})$ [34]	$d^{1-\epsilon}$ [5]
	Online	$d + \frac{7}{10}$ [83]	$\Omega(d^{1-\epsilon})$ [7]
Vector Scheduling	Offline (constant d)	PTAS[34]	NP-hard
	Offline (arbitrary d)	$O(\log d/\log\log d)$ [111]	$\omega(1)$ [34]
	Online	$O(\log d/\log\log d)$ [111]	$\Omega(\log d/\log\log d)$ [111]

[2] Here asymp. means asymptotic approximation guarantee

[3] Here abs. means absolute approximation guarantee

[4] Follows from the fact that even 1-D bin packing can not be approximated better than 3/2

[5] See the reduction in Section 4.1.1

4.1.5 Heuristics

Heuristics for 2-D VBP were studied in detail by Spieksma [193], who mentions applications in loading, scheduling, and layout design, considers lower bounding and heuristic procedures using a branch-and-bound scheme. Here, upper bounds are derived from a heuristic, adapted from the first fit decreasing (FFD)-rule for bin-packing. To find better lower bounds, properties of pairs of items are investigated. Han et al. [99] present heuristic and exact algorithms for a variant of 2-D VBP, where the bins are not identical. Caprara and Toth [30] also studied 2-D VBP. They analyze several lower bounds for the 2-D VBP. In particular, they determine an upper bound on the worst-case performance of a class of lower bounding procedures derived from the classical 1-D BP. They also prove that the lower bound associated with the huge LP relaxation dominates all the other lower bounds. They then introduce heuristic and exact algorithms, and report extensive computational results on several instance classes, showing that in some cases the combinatorial approach allows for a fast solution of the problem, while in other cases one has to resort to a large formulation for finding optimal solutions. Chang et al. [32] had proposed a greedy heuristic named *hedging*. Otoo et al. [167] studied the 2-D VBP, where each item has 2 distinct weights and each bin has 2 corresponding capacities, and have given linear-time greedy heuristics. An interesting application of the 2-D VBP problem is studied by Vercruyssen and Muller [202]. The application arises in a factory where coils of steel plates (items), each having a certain physical weight and height, have to be distributed over identical furnaces (bins) with a limited capacity for height and weight. Another application of the problem is described by Sarin and Wilhelm [180], in the context of layout design. Here, a number of machines (items) have to be assigned to a number of robots (bins), with each robot having a limited capacity for space, as well as a limited capacity for serving a machine. Many of these heuristics are tailor-made for 2-D.

For the general case, Stillwell et al. [196] studied variants of FFD concluding that the algorithm *FFDAvgSum* is best in practice. They also show that genetic algorithms do not perform well. Panigrahy et al. [169] systematically studied variants of the First Fit Decreasing (FFD) algorithm. Inspired by bad instances for FFD-type algorithms, they propose new geometric heuristics that run nearly as fast as FFD for reasonable values of n and d.

4.2 Preliminaries

Let $I := \{v_1, v_2, \dots, v_n\}$ be a set of d-dimensional vectors where $v_i = (v_i^1, v_i^2, \dots, v_i^d)$ and $v_i^j \in [0,1]$ for $j \in [d]$. Here $[d]$ denotes the set $\{1, 2, \dots, d\}$, for $d \in \mathbb{N}$. We will use the terms dimension and coordinate interchangeably. If α is a vector, we call a bin B_i to be α-packed, if $\sum_{v_j \in B_i} v_j^\ell \le \alpha^\ell$ for all $\ell \in [d]$. For any bin B_i, it has *slack* δ in dimension k if $\sum_{v \in B_i} v^k \le (1-\delta)$. For a set S of vectors, let σ_S be the vector denoting the coordinate-wise sum of all vectors in S, i.e., $\sigma_S = \sum_{v_j \in S} v_j$. We denote $\sigma_S \le v$ if $\sigma_S^l \le v^l$ for all dimensions $l \in [d]$. Thus S is a feasible configuration, if $\sigma_S \le \mathbf{1}$, where $\mathbf{1} = (1, \dots, 1)$ is the unit vector.

We now classify the items into the following two classes based on their sizes. Here β is a parameter depending on ϵ and d and will be fixed later.

- Large or big items ($\mathscr{L}$) : at least one coordinate has size $\ge \beta$, i.e., $v \in \mathscr{L}$ iff $||v||_\infty \ge \beta$.

- Small items ($\mathscr{S}$) : all coordinates have size $< \beta$, i.e., $v \in \mathscr{S}$ iff $||v||_\infty < \beta$.

Let $\mathscr{L}_B$ be the set of big items and $\mathscr{S}_B$ be the set of small items in a bin B. We call a *configuration of big items* in B to be the vector $\sum_{v_i \in \mathscr{L}_B} v_i$. We call a *configuration of small items* to be the remaining space in the bin, i.e., vector $\mathbf{1} - \sum_{v_i \in \mathscr{L}_B} v_i$.

Let $(c_1, c_2, \dots, c_d)$ be a d-tuple of integers such that $c_i \in [0, \lceil \frac{1}{\epsilon} \rceil]$. Each such distinct tuple is called to be a *capacity configuration* and approximately describes how a bin is filled. A packing of a set of vectors V is said to be *viable* for a capacity configuration $(c_1, \ldots, c_d)$ if $\sigma_V \leq \epsilon \cdot (c_1, \ldots, c_d)$. There are $r_c := (1 + \lceil \frac{1}{\epsilon} \rceil)^d$ possible types of capacity configurations.

Now we define a *bin configuration* to be an r_c-tuple of integers $(m_1, m_2, \ldots, m_{r_c})$ where $m_i \in [m]$ and $\sum_i m_i = m$. Bin configuration approximately describes the structure of all packed bins, i.e., there are m_i bins with capacity configuration c_i for $i \in [r_c]$. Total number of bin configurations is $O(m^{r_c})$. A packing of a set of vectors V is said to be *viable* to a bin configuration $M = (m_1, m_2, \ldots, m_{r_c})$, if there is a packing of items in V into m bins such that the bins can be partitioned into sets $\mathscr{B}_1, \mathscr{B}_2, \ldots, \mathscr{B}_{r_c}$ and there are m_i bins in $\mathscr{B}_i$ that are viable to capacity configuration c_i.

4.2.1 Rounding specification and realizability

Consider a partition $R_1 \cup \cdots \cup R_k$ of I into k classes, and a function $R\colon I \to [0,1]^d$ which maps all items $v \in R_i$ to some item $\tilde{v}_i$. We call the instance $\tilde{I} := \{R(v) \mid v \in I\}$ a *rounding* of I to k item types. Sometimes, in our algorithms we will not know which items will be rounded in what way. We will have classes $W_1, \ldots, W_k \subseteq I$, not necessarily disjoint, and for each class W_i, there will be a specified item $\tilde{v}_i$ and a number w_i, meaning that exactly w_i items from W_i are supposed to be rounded to item $\tilde{v}_i$. We call this a *rounding specification.* We say, that a rounding specification is realizable for instance I, if there is a rounding $\tilde{I}$ and a function $R\colon I \to \tilde{I}$ that satisfies the requirements of the rounding specification. This can be checked, for example, by solving a flow problem. Being able to guess the right rounding specification and test its realizability will be crucial in Section 4.5.

4.2.2 Limitations of Round and Approx Framework

Theorem 4.2.1. Any algorithm that rounds large dimensions (with value more than ϵ, where $\epsilon > 0$ is a given accuracy parameter) of items to $O(1)$ types can not achieve better than d approximation for d-D vector packing.

Proof. Let $\mathcal{A}$ be an algorithm that rounds large dimensions of items to r (a constant) types. Let $t \in \mathbb{N}$ be another large constant. Consider following $n = dtrk$ items that can be packed into $m = n/d = trk$ bins $\mathcal{B}_1, \ldots, \mathcal{B}_m$ in the optimal packing. There are tr types of bin in the optimal packing and each type contains $m/tr = k$ bins. Each i'th type bin contains d items $v_{i_1}, v_{i_2}, \ldots, v_{i_d}$ such that $v_{i_h}^h = 1-\epsilon_i$ and $v_{i_h}^l = \epsilon_i/(d-1)$ for $h = 1, 2, \ldots, d$ and $l \in [d] \setminus \{h\}$ where $\epsilon_i = \frac{\xi}{d^{i-1}}$ and ξ is a small constant, i.e., in each bin of type i, for each coordinate h there is exactly one item v_{i_h} with value $(1-\epsilon_i)$ in that coordinate (v_{i_h} has length $\frac{\epsilon_i}{(d-1)}$ in all other coordinates), and $(d-1)$ other items with value $\frac{\epsilon_i}{(d-1)}$ in coordinate h.

Now let the rounding algorithm round the large coordinates of items to constant types $1-\delta_1, 1-\delta_2, \ldots, 1-\delta_r$ such that $1-\epsilon_{\ell_h} \leq 1-\delta_h < 1-\epsilon_{\ell_{h+1}}$ for $h \in [r-1]$ and $1-\epsilon_{tr} \leq 1-\delta_r$. Then apart from possibly the items in bins of type ℓ_h for $h \in [r-1]$ and of type tr, for all other items large coordinates $(1-\epsilon')$ are rounded to some other value $(1-\tilde{\epsilon})$ such that $\epsilon' \geq d\tilde{\epsilon}$. Let us call the bins that are not of type ℓ_h for $h \in [r-1]$ and of type tr to be complex bins. We claim the following.

Claim 4.2.2. No two rounded items from two complex bins, can be packed into one bin.

Proof. Let v_{i_x} and v_{j_y} be two items from complex bins of type i and j where $i \leq j$. From definition of ϵ_i,

$$\epsilon_i = d^{j-i}\epsilon_j \tag{21}$$

Let $\overline{v}_{i_x}$ and $\overline{v}_{j_y}$ be the rounded vector, then

$$\begin{aligned}\overline{v}^y_{j_y} + \overline{v}^y_{i_x} &\geq (1-\overline{\epsilon}_j) + \frac{\epsilon_i}{d-1} \\ &\geq (1-\frac{\epsilon_j}{d}) + \frac{\epsilon_i}{d-1} \qquad \Big[\text{Since, } \epsilon_j \geq d\overline{\epsilon}_j\Big] \\ &\geq (1-\frac{\epsilon_j}{d}) + \frac{\epsilon_j}{d-1} > 1 \qquad \Big[\text{Since, } i \leq j \quad \text{and} \quad d > 1\Big].\end{aligned}$$

Thus two rounded items can not be packed together in a bin. □

Hence, we need at least $(md - rkd)$ bins to pack all items. This implies a lower bound of approximation for these class of algorithms as $\frac{md-rkd}{m} = d(1-\frac{rk}{trk}) = d(1-\frac{1}{t})$. Thus it gives asymptotic approximation hardness for rounding-based algorithms as d. □

Therefore, improving $1 + \ln d$ for d-dimensions is not possible by just using R & A framework with a $O(1)$ rounding based algorithm to pack the residual items.

4.2.3 Multi-objective/multi-budget Matching

In *Multi-objective/multi-budget matching problem (MOMB)*, we are given a graph $G := (V, E)$, k linear functions (called demands) $f_1, f_2, \ldots, f_k : 2^E \to \mathbb{R}_+$, ℓ linear functions (called budgets) $g_1, g_2, \ldots, g_\ell : 2^E \to \mathbb{R}_+$, and the goal is to find a matching $\mathcal{M}$ satisfying $f_i(\mathcal{M}) \geq D_i$ for all $i \in [k]$ and $g_i(\mathcal{M}) \leq B_i$ for all $i \in \ell$.

Chekuri et al. [35] gave an algorithm that solves the problem nearly optimally.

Theorem 4.2.3. [35] For any constant $\gamma > 0$ and any constant number of demands and budgets $k+\ell$, there is a polynomial time algorithm which for any feasible instance of multi-objective matching finds a feasible solution S such that

- Each linear budget constraint is satisfied: $g_i(S) \leq B_i$.

- Each linear demand is nearly satisfied: $f_i(S) \geq (1-\gamma)D_i$.

If such a solution is not found, the algorithm returns a certificate that the instance is not feasible with demands D_i and budget B_i.

4.3 Overview and Roadmap

Before proving our results, we first give some intuition behind the main ideas and techniques. For the sake of simplicity, we mostly focus on the case of $d = 2$.

The starting point is the following simple observation. Suppose there is an optimal packing $\mathcal{P}$ of I where each bin in $B \in \mathcal{P}$ has some fixed slack δ in each dimension. Then one can get optimal packing easily by the following resource augmentation result [34].

Theorem 4.3.1. [34] If a d-dimensional VBP instance can be packed in m bins, then for any $\delta > 0$, a packing in m bins of size $(1 + \delta, \ldots, 1 + \delta)$ can be found in time poly$(n, f(\delta))$ for some function f.

However this is too good to hope for, and there can be a large gap between packings with and without slack. For example, if all items are $(0.5, 0.5)$, then $\mathsf{Opt}(I) = m/2$, while any packing with slack needs m bins. However, note that this instance, or any instance where each bin has at most 2 items, can be easily solved by matching. Similarly an example can be constructed, showing that in d-dimensional case $\mathsf{Opt}_{(1-\delta)}$ can be d-times larger than Opt. Consider the set of md vectors that can be packed into m bins in an optimal packing such that in each bin B_i there are d vectors v_j for $j \in [d]$ having coordinate j equal to $1 - \delta$ and the rest equal to $\frac{\delta}{d-1}$. Then $\mathsf{Opt} = m$, while $\mathsf{Opt}_{(1-\delta)} = md$.

However we can show a related structural result.

Lemma 4.3.2. [Structural lemma for 2-D VBP] Fix any $\delta < 1/5$. For any packing $\mathcal{P}$ using m bins, there exists a packing $\mathcal{P}'$ into at most $\lceil 3m/2 \rceil$ bins, such that for each bin B in $\mathcal{P}'$: (i) either B contains at most 2 items, or (ii) at least one of the dimensions in B has slack at least δ.

We call such a packing $\mathcal{P}'$ a *structured* packing, and a generalized version of this lemma is proved in Lemma 4.5.9 in Section 4.5.

Finding structured-packings: Our goal then is to find such a packing $\mathcal{P}'$ efficiently. To handle bins of type (ii), we first show a variant of Theorem 4.3.1 that only needs resource augmentation in $d-1$ dimensions (instead of d). This is shown in Theorem 4.0.4 in Section 4.4. Now, if we knew which items were packed in the *matching* bins (of type (i) above, i.e., bins that contain at most two items), then an APTAS for $\mathcal{P}'$ would follow by applying Theorem 4.0.4 on the remaining items. However, it is unclear how to guess the items in the matching bins efficiently, as their sizes are not rounded.

To get around this, we flip the idea on its head. We observe that Theorem 4.0.4 for packing of bins with slack is based on rounding item sizes, and hence only requires knowledge of how many items of each size type (according to its internal rounding) are present in the instance. So we guess the thresholds used by the algorithm to define the size classes, and guess how many items of each type are packed in the slack bins. Now, the remaining items must be the ones packed in matching bins (we do not know which are these items, but we know how many items of each type there are). This leads precisely to the multi-objective budgeted matching problem [35].

In particular, we consider a graph with a vertex for each item and an edge between two items if they can be packed together. We then classify the items into $O(1)$ classes, based on the guessed size classification, and then apply Theorem 4.2.3 to find a matching with the guessed quota of items from each class. This gives the 3/2-asymptotic approximation for $d = 2$, and an analogous $(d+1)/2$ approximation for general d. This is described in Section 4.5.

Applying the *R&A* framework: We apply the R&A framework in different ways depending on whether $d = 2$ or $d > 2$. For $d = 2$, we first find a packing in matching bins, and then apply R&A on remaining items. Roughly speaking, this works because the remaining items are packed using the APTAS which is rounding based. The proof has some additional technical difficulties compared to a similar previous result on

2-dimensional geometric bin-packing [17], due to skewed items that are large in one dimension and small in another. This is described in Section 4.6.

For $d > 2$, it is unclear how to make the above idea work as there are no analogous results to Theorem 4.2.3 for multi-objective budgeted d-dimensional matching. So we adopt a different approach. We apply random sampling directly to $\mathcal{P}'$ first and then use multi-objective budgeted matching for the remaining items. Roughly, the reason is that after sampling many bins are left with one or two remaining items, and we can apply Theorem 4.2.3. But the details are more technical and we refer the reader to Section 4.7.

4.4 Vector Packing with Resource Augmentation

In this section we consider packing when resource augmentation is allowed in $(d-1)$-dimensions. We call these dimensions to be *augmentable* dimensions. The only other dimension where we are not allowed to augment, is called *non-augmentable* dimension. Without loss of generality, we assume the last dimension to be the non-augmentable dimension. Now let us prove Theorem 4.0.4, the main result in this section.

Theorem 4.4.1. (Restatement of Theorem 4.0.4) Let $\epsilon > 0$ be a constant and I be an instance of d-dimensional vector packing for which m unit vector bins are required in the optimal packing, then there is a polynomial time algorithm for packing vectors in I into at most $(1 + \epsilon + \frac{\epsilon^2}{2d})m + 1$ bins with ϵ resource augmentation in $(d-1)$ dimensions (i.e., bins have length $(1+\epsilon)$ in $(d-1)$ dimensions and length 1 in the other dimension).

Given ϵ and a guess for the optimal value m, we describe a procedure that either returns a feasible packing into $(1+\epsilon+\frac{\epsilon^2}{2d})m+1$ bins with ϵ resource augmentation in $(d-1)$ dimensions, or proves that the guess is incorrect. First we classify the items into *big* and *small* based on $\beta = \epsilon/2d$. Then we round the big items into a constant number of classes. We set aside few vectors and pack them separately. Thereafter, for each bin configuration C, the algorithm first decides if remaining rounded vectors in $\mathscr{L}$ can be packed viable to C. This is done using dynamic programming and contributes a major slice of the overall time complexity. Afterwards we replace the rounded items by the original ones. In the final step we pack the small items using linear programming into the residual space of the bins as well as in some additional bins as needed.

4.4.1 Rounding of big items

Now let us describe the procedure to round the big items into constant number of item types. We apply different roundings on augmentable and non-augmentable coordinates. Coordinates $1, \ldots, d-1$ (those with the resource augmentation allowed) are rounded to the multiples of α, where $\alpha = \frac{\epsilon^2}{2d^2}$, and the d-th coordinate is rounded using linear grouping. Note that the small items are not rounded and will be considered separately later.

4.4.1.1 Rounding of augmentable coordinates by preprocessing:

We create an instance $\widehat{Q}$ rounded in the first $d-1$ coordinates by replacing each big item p_i of I with an item $\widehat{q_i}$ as follows:

$$\widehat{q_i^\ell} := \begin{cases} \lceil p_i^\ell/\alpha \rceil \alpha & \text{if } \ell \in \{1, \ldots, (d-1)\}, \\ p_i^\ell & \text{if } \ell = d. \end{cases}$$

We split $\widehat{Q}$ into classes $\{W^u | u \in \{1, \ldots, \lceil \frac{1}{\alpha} \rceil\}^{d-1}\}$ where $W^u := \{\widehat{q} \mid \widehat{q}^\ell = u^\ell \cdot \alpha\}$ for each $\ell \in [d-1]$, creating $r_A := (\lceil \frac{1}{\alpha} \rceil)^{d-1}$ classes altogether.

4.4.1.2 Rounding of the non-augmentable coordinate:

We apply rounding of the last coordinate using linear grouping on each W^u separately. Let λ be a very small constant dependent just on ϵ and d which will be fixed in the proof of Lemma 4.4.2. We sort the items in W^u according to nonincreasing size in the last coordinate. Let k_u be the number of vectors in W^u, denoted by $\widehat{q}_{u,1}, \ldots, \widehat{q}_{u,k_u}$ in sorted order according to nonincreasing size in the last coordinate. We define at most $\lceil 1/\lambda \rceil$ subsets $W^{u,j}$ each of which consists of $\lceil \lambda |W^u| \rceil$ vectors except the last subset that might contain less than $\lceil \lambda |W^u| \rceil$ vectors. This is done in the following way. We select the first vector $\widehat{q}_{u,1}$ (call it the first *round vector*), then assign that vector and next $\lceil \lambda |W^u| \rceil - 1$ vectors into one subset $W^{u,1}$. Then again we select the next vector (second round vector) and assign it and next $\lceil \lambda |W^u| \rceil - 1$ vectors into the set $W^{u,2}$ and so on. Thus, the jth subset $W^{u,j}$ contains $\lceil \lambda |W^u| \rceil$ vectors $\{\widehat{q}_{u,((j-1)\cdot\lceil \lambda |W^u| \rceil+1)}, \ldots, \widehat{q}_{u,(j\cdot\lceil \lambda |W^u| \rceil)}\}$ and the jth round vector is $\widehat{q}_{u,((j-1)\cdot\lceil \lambda |W^u| \rceil+1)}$. The last subset can possibly contain less than $\lceil \lambda |W^u| \rceil$ vectors.

To get the final rounded instance $\widetilde{Q}$ we replace each vector $\widehat{q}_i \in W^{u,j}$ by $\widetilde{q}_i$, where

$$\widetilde{q}_i^{\ell} := \widehat{q}_i^{\ell} \quad \text{for } \ell \in [d-1],$$

$$\widetilde{q}_i^{d} := \max\{\widehat{q}^{d} \mid \widehat{q} \in W^{u,j}\}.$$

Thus we round-up the dth dimension to the dth coordinate of the *round vector* of the group. Note that the number of groups in each W^u is $\le \lceil (\frac{|W^u|}{\lceil \lambda |W^u| \rceil}) \rceil \le \lceil (\frac{|W^u|}{\lambda |W^u|}) \rceil \le \lceil \frac{1}{\lambda} \rceil$. Thus the rounded vectors in $\widetilde{Q}$ can be classified into one of $r_L := \lceil 1/\lambda \rceil \cdot r_A = \lceil \frac{1}{\lambda} \rceil \cdot (\lceil \frac{1}{\alpha} \rceil)^{(d-1)}$ distinct classes. Any *configuration* of vectors into one bin can be described as tuple $(k_1, k_2, \ldots, k_{r_L})$ where k_i indicates the number of vectors of the i'th class. As at most d/β vectors from $\widetilde{Q}$ can be there in any bin, there are at most $(d/\beta)^{r_L}$ *configurations.*

Now we prove that the described rounding procedure fulfills the requirements of the following lemma:

Lemma 4.4.2. Let I be an instance of d-dimensional vector packing and $I_1, \ldots, I_m$ be a packing of I into m unit bins. Then there exists a packing of $\widetilde{Q}$, rounding of the big vectors in I using the described rounding procedure, such that:

1. Except for $\frac{\epsilon m}{2}$ items, all other big items in $\widetilde{Q}$ can be packed into m bins $\widetilde{Q}_1, \ldots, \widetilde{Q}_m$ of type $(1+\epsilon, 1+\epsilon, \ldots, 1+\epsilon, 1)$.

2. For $i \in [m]$, the configuration of small items in $\widetilde{Q}_i$ is the same or larger than the configuration of small items in I_i in all coordinates.

Proof. Given a packing of I into m bins, we can construct a packing of $\widetilde{Q}$ in the following way. We set $\lambda := \frac{\epsilon\beta}{2d}$. We keep the *round vectors* in their same positions. For all $u \in [\lceil\frac{1}{\alpha}\rceil]^{d-1}$ and $i \in \left[\lceil\frac{1}{\lambda}\rceil - 1\right]$, we remove the other vectors (i.e., except the ith round vector) from the packing of I corresponding to items in $W^{u,i}$ and put rounded items (except the $(i+1)$th *round vector*) from $W^{u,i+1}$ in their places to construct packing $\widetilde{Q}$. This will give a feasible packing as items in $W^{u,i+1}$ are the same or smaller than the items in $W^{u,i}$ for each $u \in \{1, \ldots, \lceil\frac{1}{\alpha}\rceil\}^{d-1}$ and $i \in \left[\lceil\frac{1}{\lambda}\rceil - 1\right]$ and all subsets of W^u have the same cardinality of $\lceil\lambda|W^u|\rceil$ (except probably the last). Let $\widetilde{Q}'' := \cup_{u\in\{1,\ldots,\lceil\frac{1}{\alpha}\rceil\}^{d-1}}(W^{u,1} \setminus \{\widehat{q}_{u,1}\})$ and $\widetilde{Q}' = \widetilde{Q} \setminus \widetilde{Q}''$. By now, all items in $\widetilde{Q}'$ are packed in a feasible way and we are left with items in $\widetilde{Q}''$ (i.e., the set of all items in $W^{u,1}$ for each $u \in \{1, \ldots, \lceil\frac{1}{\alpha}\rceil\}^{d-1}$ except the round vectors) which we will pack separately. It is clear that in any bin, we did not use more space in the last dimension to pack all items in $\widetilde{Q}'$, than what the big items used in the original packing in the last dimension. Let $\widetilde{Q}_i$ be a bin in the packing of $\widetilde{Q}'$, that we obtained from a packing of I_i using the above procedure. Let us consider the sum of sizes in any coordinate $\ell \in [d-1]$:

$$\begin{aligned}
\sum_{\widetilde{q}\in\widetilde{Q}_i} \widetilde{q}^{\ell} &\leq \sum_{p\in I_i}(\lceil p^{\ell}/\alpha\rceil \cdot \alpha) \\
&\leq \sum_{p\in I_i}(p^{\ell} + \alpha) \\
&= \sum_{p\in I_i} p^{\ell} + \frac{d}{\beta}\alpha \qquad \left[\text{Since, } |I_i| \leq \frac{d}{\beta}\right] \\
&= \sum_{p\in I_i} p^{\ell} + \epsilon \qquad \left[\text{Since, } \alpha = \frac{\epsilon^2}{2d^2} = \frac{\beta\epsilon}{d}\right].
\end{aligned}$$

Hence,

$$(1+\epsilon) - \sum_{\widetilde{q}\in\widetilde{Q}_i} \widetilde{q}^{\ell} \leq 1 - \sum_{p\in I_i} p^{\ell}.$$

Therefore, the augmentation of the bin in coordinate $\ell \in [d-1]$ by ϵ is sufficient and for $i \in [m]$, the configuration of small items (remaining space in the bin) in $\widetilde{Q}_i$ is the same or larger than the configuration of small items in I_i in all coordinates.

Now to conclude the proof we need to show, $|\widetilde{Q}''| \leq \frac{\epsilon m}{2}$. Each big item has size at least β in at least one of the coordinates. In each bin there can be at most $1/\beta$ items for each coordinate which are big when restricted to this coordinate, having at most $\frac{d}{\beta}$ big items in a bin in total. Therefore, we have $\frac{d}{\beta} m \geq \sum |W^u|$. Now,

$$\begin{aligned}
|\widetilde{Q}''| = \sum_u (|W^{u,1}| - 1) &\leq \sum_u (\lceil \lambda |W^u| \rceil - 1) \\
&\leq \sum_u \lambda |W^u| \\
&\leq \lambda \frac{d}{\beta} m \\
&\leq \frac{\epsilon\beta}{2d} \cdot \frac{d}{\beta} m \\
&\leq \frac{\epsilon m}{2}.
\end{aligned}$$

So, we are left with at most $\lfloor \frac{\epsilon m}{2} \rfloor$ many items. Note that we can always pack these extra items greedily into $\lfloor \frac{\epsilon m}{2} \rfloor$ additional bins. □

4.4.2 Packing of big items

From Lemma 4.4.2, it follows that if I is packable into m bins then there is a structured rounding $\widetilde{Q}$ of big items in I. There is a constant number of item types r_L in $\widetilde{Q}$. Thus there are only $r \leq (\frac{d}{\beta})^{r_L}$, number of configurations. Since $m \leq n$, this gives us $O(m^r)$, a polynomial number of configurations of big items for m bins and allows us to guess it, more precisely, to try each configuration until we find the right one.

Lemma 4.4.3. Let $m := Opt(I)$, then the rounded instance $\widetilde{Q}$ can be packed in polynomial time into $m' := m + \lfloor \frac{\epsilon m}{2} \rfloor$ unit bins.

Proof. We guess the configuration of the m' bins which would correspond to the configuration of big items in some packing of I into m bins in the sense of Lemma 4.4.2. We guess the round vectors and pack them separately into $\lfloor \frac{\epsilon m}{2} \rfloor$ bins. If there is a feasible packing of $\widetilde{Q}'$ corresponding to the guessed configuration, we can find it using dynamic programming in time $O\big((\frac{dn}{\beta})^{r_L} \cdot m\big)$ according to the following lemma in [34, Lemma 2.3].

Lemma 4.4.4. [34]. Let $M = (m_1, \ldots m_{r_c})$ be a bin configuration. There exists an algorithm with running time $O\big((\frac{dn}{\beta})^{r_L} \cdot m\big)$ to decide if there is a packing of the items in $\widetilde{Q}'$ that is viable to M.

If we can not find one, we have guessed a wrong configuration and we should try again. If we tried all the configurations and none of them worked, this is a contradiction to the Lemma 4.4.2 and we know that I is not packable into m bins. □

4.4.3 Packing of small items

Lemma 4.4.5. Let $(\widetilde{Q}_1, \ldots, \widetilde{Q}_m)$ be a packing of large vectors in I into m bins of type $(1+\epsilon, \ldots, 1+\epsilon, 1)$ such that there is an optimal packing $(I_1, \ldots, I_m)$ into m unit vector bins, so that for each $k \in [m]$, configuration of small items in $\widetilde{Q}_i$ is the same or larger than the configuration of small items in I_i in all coordinates. Then we can pack all the small items of I into residual space of bins $\widetilde{Q}_1, \ldots, \widetilde{Q}_m$ and at most $\left\lceil \frac{m}{2} \cdot (\epsilon + \frac{\epsilon^2}{d}) \right\rceil$ additional bins.

Proof. To pack the small vectors, we use the idea of Chekuri and Khanna [34]. However in [34], as resource augmentation was allowed in all dimensions, small vectors could have been packed without using any additional bins. In our case we need extra bins. We formulate the following assignment LP:

$$\sum_{j=1}^{m} x_{ij} = 1 \quad \text{for each small item } v_i \in \mathscr{S},$$

$$\sum_{i=1}^{|\mathscr{S}|} x_{ij} v_i^{\ell} \leq (1+\epsilon) - b_j^{\ell} \quad \text{for each bin } j \in [m] \text{ and dimension } \ell \in [d-1],$$

$$\sum_{i=1}^{|\mathscr{S}|} x_{ij} v_i^{\ell} \leq 1 - b_j^{\ell} \quad \text{for each bin } j \in [m] \text{ and dimension } \ell = d,$$

$$x_{ij} \geq 0 \quad \text{for each item } i \in \mathscr{S} \text{ and each bin } j \in [m],$$

where $b_j := \sum_{\widetilde{q} \in \widetilde{Q}_j} \widetilde{q}$, i.e., sum of rounded large vectors in bin j.

Since there is an optimum packing of I that for each bin j uses a smaller or equal space for small items, the assignment LP is feasible. Let x be a basic feasible solution to this LP. The integrally assigned items can be packed directly. We denote F to be the set of items assigned fractionally in x. The number of variables in the LP is $m \cdot |\mathscr{S}|$ and thus from standard polyhedral theory any basic solution to the LP has $m \cdot |\mathscr{S}|$ tight constraints. However there are $dm + |\mathscr{S}|$ non-trivial constraints. Thus at most $dm + |\mathscr{S}|$ variables can be strictly positive. As each vector is assigned to at least one bin, the number of vectors that are fractionally assigned to more than one bin is at most dm. Hence, $|F| \leq dm$. We pack the items in F in separate bins. Now, for any constant $\psi < 1/2$, we have the following inequality:

$$\begin{aligned} \frac{1}{\psi} - (\lfloor \frac{1}{\psi} \rfloor) \cdot (1+2\psi) &\leq \frac{1}{\psi} - (\frac{1}{\psi} - 1) \cdot (1+2\psi) \\ &\leq 2\psi - 1 < 0. \end{aligned}$$

Hence,

$$\frac{1}{(\lfloor \frac{1}{\psi} \rfloor)} < \frac{(1+2\psi)}{(\frac{1}{\psi})}. \tag{22}$$

Since the size of each item is less than β in both dimensions, the number of additional bins are at most

$$\begin{aligned}
\left\lceil \frac{dm}{\lfloor \frac{1}{\beta} \rfloor} \right\rceil &\leq \left\lceil \frac{dm}{\lfloor \frac{2d}{\epsilon} \rfloor} \right\rceil \\
&\leq \left\lceil dm \cdot \frac{(1+\epsilon/d)}{(\frac{2d}{\epsilon})} \right\rceil \qquad \left[\text{From (22), using } \psi = \epsilon/2d\right] \\
&\leq \left\lceil \frac{m}{2} \cdot (\epsilon + \frac{\epsilon^2}{d}) \right\rceil.
\end{aligned}$$

□

4.4.4 Algorithm and proof of Theorem 4.0.4

In Algorithm 1, a summary of an algorithm giving a proof of the main theorem of this section can be found.

```
A. Guessing phase1:
  A1. Guess $m := Opt(I)$,                                       // $O(\log n)$ guesses
  A2. Create $\widetilde{Q}$, the rounded instance of large vectors of $I$,      // $O(n \log n)$
  A3. Guess large configuration $M$ of $m + \lceil \frac{\epsilon m}{2} \rceil$ bins of size
  $(1+\epsilon, \ldots, 1+\epsilon, 1)$,                                      // $O(m^t)$
B. Packing of large items.
  B1. Pack items in $\widetilde{Q}$ into configuration $M$, or            // $O\big((\frac{dn}{\beta})^{r_L} \cdot m\big)$
      return to Guessing phase,
  B2. Replace rounded items of $\widetilde{Q}$ by original items,               // $O(n)$
C. Packing of small items.
  Pack small items using assignment LP, or
      return to Guessing phase.
```

Algorithm 1: d-DIM. VBP WITH RESOURCE AUGMENTATION IN $(d-1)$ DIM.

Proof of Theorem 4.0.4. If there is a packing of I into m unit bins, from Lemma 4.4.2 we know that there is a packing of rounded instance $\widetilde{Q}$ of big vectors of I into $m' := m + \lfloor \frac{\epsilon m}{2} \rfloor$ unit bins. Moreover, Lemma 4.4.2 assures, that there is still enough space to pack all the small vectors except at most dm small items. Thus if we can guess the bin configuration of the optimal packing, we can pack the small items of I using Lemma 4.4.5, getting a packing of I into $m'' := m + \lfloor \frac{\epsilon m}{2} \rfloor + \lceil \frac{m}{2} \cdot (\epsilon + \epsilon^2/d) \rceil \leq m + m \cdot (\epsilon + \frac{\epsilon^2}{2d}) + 1$ bins.

Since items in Q are rounded to a constant number of classes, there is a polynomial number of configurations of m bins for Q and we can, in the worst case, enumerate them all. If we fail in the following steps in packing I into $m'' := m + m \cdot (\epsilon + \frac{\epsilon^2}{2d}) + 1$ bins for every choice of bin configurations, due to Lemma 4.4.2, and 4.4.5, we can claim that $Opt(I) > m$. Otherwise $Opt(I) \leq m''$.

This enables us to guess the value of m using a binary search between $|| \sum_{v \in I} v ||_\infty$ and n. We find the smallest m such that we are able to pack I into m'' bins using Algorithm 1. Then from Lemma 4.4.2 and 4.4.5 we know that $m \leq Opt(I) \leq m'' = m + m \cdot (\epsilon + \frac{\epsilon^2}{2d}) + 1$ and the statement of the theorem follows. □

4.5 Finding a well-structured approximation solution for vector packing

Our goal in this section is to prove the following theorem:

Theorem 4.5.1. Given any constant ϵ, such that $0 < \epsilon < \frac{1}{56d^2}$, and a d-dimensional VBP instance I, there is a polynomial time algorithm to pack I into at most $\lceil (\frac{d+1}{2})\mathsf{Opt}(I) \rceil \cdot (1 + 2\epsilon) + 1$ unit bins.

This already gives $(1.5 + \epsilon)$ and $(2 + \epsilon)$ asymptotic approximations for 2-D and 3-D VBP improving upon the current best guarantees of $(1.693 + \epsilon)$ and $(2.099 + \epsilon)$ respectively.

To prove Theorem 4.5.1, we first show the existence of a structured packing using

$\lceil (\frac{d+1}{2})m \rceil$ bins (Lemma 4.5.9), and then use the APTAS in Section 4.4 to find such a structured packing.

Let δ be a suitably small constant whose value will be specified later.

Definition 4.5.2. A packing $\mathcal{P}$ is *structured* if each bin $B \in \mathcal{P}$ satisfies one of the following:

- B consists of precisely one single large item v. We call this collection of bins $\mathcal{B}_S$.
- B consists of precisely two large items v_i, v_j. We call this collection of bins $\mathcal{B}_T$.
- B has δ-slack in at least $d-1$ dimensions. The collection of bins with δ-slack in dimensions $[d] \setminus \{\ell\}$ is denoted by $\mathcal{B}_\ell$. If B has δ-slack in all dimensions we assign it (arbitrarily) to $\mathcal{B}_1$.

Let us fix $\delta := \frac{\epsilon}{1+\epsilon}$. We claim the following:

Lemma 4.5.3. Let $\epsilon > 0$ be a constant and I be an instance of d-dimensional vector packing having a structured packing into m' bins, then there is a polynomial time algorithm that pack items in I into at most $(1 + \epsilon + \frac{\epsilon^2}{2d})m' + 1$ bins.

An algorithm proving Lemma 4.5.3 is given at the end of this section. Together with Lemma 4.5.9 (proving existence of relatively small structured packings) it already implies Theorem 4.5.1.

4.5.1 Existence of small structured packing

In the following part of this section we prove that there indeed exist small structured packings. First we consider $d = 2$, which is rather simple. Then we will consider the proof of general d, which is more involved.

Lemma 4.5.4. Let $\delta < 1/5$, and let B be a bin of 2-D VBP that is not structured. Then at least one of the following holds:

1. There is a large item $p \in B$ such that $p \leq (1/2, 1/2)$ and p is $> \delta$ in at least one coordinate.
2. There is a subset S of items in B such that $\sum_{p \in S} p \leq (2\delta, 2\delta)$ and either:

   (i) $B \setminus S$ has δ-slack in some dimension, or

   (ii) $|B \setminus S| \leq 2$ with $(\sum_{v \in B \setminus S} v) \geq (1-\delta, 1-\delta)$.

Proof. Let T denote the set of items $v \in B$ such that $v \leq (\delta, \delta)$. As there can be at most two items in a bin with some coordinate strictly $> 1/2$ (at most one item for each coordinate), if there is no p satisfying the first requirement, we know that $|B \setminus T| \leq 2$.

If $(\sum_{v \in T} v) \leq (\delta, \delta)$, we set $S := T$. Then $|B \setminus S| \leq 2$ with $(\sum_{v \in B \setminus S} v) \geq (1-\delta, 1-\delta)$, as desired. Otherwise, $(\sum_{v \in T} v)$ is greater than δ in at least one dimension. So we greedily add items of T to S until $B \setminus S$ contains δ slack in at least one dimension. Since T only contains items $< (\delta, \delta)$, the items of S can not sum to more than $\delta + \delta \leq 2\delta$ in any coordinate. □

We now show that Lemma 4.5.4 implies the following lemma. This immediately implies Lemma 4.5.9 (for $d = 2$), by applying it repeatedly to the unstructured bins in I

Lemma 4.5.5. For any two unstructured bins B_1 and B_2 in I, their items can be repacked into three structured bins.

Proof. As $\delta < 1/5$, we have $4\delta < (1-\delta)$ and $1/2 + \delta < (1-\delta)$. From each B_1 and B_2, let us remove either the large item p or set S (as in Lemma 4.5.4), and pack them into a new bin B'.

We first note that if either p or S is removed from some bin B, then B becomes structured (it either has δ-slack in some dimension, or at most 2 items). Second, if the repacked items are both of type p, then B' is structured as it has exactly 2 large items. Otherwise, if at least one item is of type S, then B' has δ-slack in both dimensions as both $1/2 + 2\delta \leq 1 - \delta$ and $2\delta + 2\delta \leq 1 - \delta$. □

Thus we get the existence for 2-D.

Lemma 4.5.6. Let I be an instance of 2-dimensional vector packing. Any optimal packing of I into m bins can be transformed into a packing into $\lceil \frac{3}{2}m \rceil$ bins of types $\mathcal{B}_S, \mathcal{B}_T, \mathcal{B}_1, \mathcal{B}_2$.

Before proving the d-dimensional structural lemma, we make the following observation.

Lemma 4.5.7. Let $\mathcal{B}$ be a set of d-dimensional vectors that can be feasibly packed into one unit vector bin. Then there is a subset V of big items and another subset S of items of $\mathcal{B}$ such that the bin packed with all the items in $B := \mathcal{B} \setminus (V \cup S)$ contains slack of $\geq \delta$ in at least $d-1$ dimensions, and one of the following holds.

1. $|V| \leq (d-1)$, $S = \emptyset$,
2. $|V| \leq (d-2)$, $S \neq \emptyset$, such that $\sum_{p \in S} p \leq \kappa\delta$ in all coordinates for a small constant κ.

Furthermore, removal of each item in V from $\mathcal{B}$, creates a slack of $\geq \delta$ in at least one distinct coordinate.

Proof. We start with $\mathscr{C} = [d], V = \emptyset$ and then we update $\mathscr{C}$ after removal of items in the following steps by considering the dimensions one by one. In each iteration we select a coordinate $k \in \mathscr{C}$ that we have not considered already. Let v be the vector with highest coordinate along k, i.e., $v^k \geq p^k$ for all $p \in \mathcal{B}$. If removal of v from $\mathcal{B}$ creates δ slack along dimension k, then remove v from $\mathcal{B}$ and update $\mathcal{B} = \mathcal{B} \setminus \{v\}$, $\mathscr{C} = \mathscr{C} \setminus \{k\}, V = V \cup \{v\}$. We continue this while $|\mathscr{C}| > 1$ and while such a removable item v exists. Thus by definition of V, removal of each item in V from $\mathcal{B}$, creates slack in at least one distinct coordinate in $\mathcal{B}$ and $|V| \leq d - 1$.

As element in V creates slack in at least one distinct coordinate, if $|V| = d-1$, $\mathcal{B} \setminus V$ already has slack in $(d-1)$ dimensions and we are done by taking $S = \emptyset$.

Otherwise let $|\mathscr{C}| \geq 2$, where $\mathscr{C}$ denotes the set of coordinates where $\mathcal{B}$ does not contain δ slack. So, $|V| \leq (d-2)$. Hence after removing the items in V, all items in $(\mathcal{B} \setminus V)$ are $< \delta$ in all coordinates of $\mathscr{C}$. Note that they can still be relatively big in coordinates of $\overline{\mathscr{C}} := [d] \setminus \mathscr{C}$. Now we make the following general claim.

Claim 4.5.8. Let $\mathcal{B}'$ be the set of d-dimensional vectors that can be feasibly packed into a unit vector bin and $\mathscr{C}$ be the set of coordinates where the sum of vectors in the bin is $\geq (\theta - \delta)$ for some constant $\theta \leq 1$. If all items in a set $\mathcal{B}'$ are smaller than δ in all coordinates of $\mathscr{C}$ and $\delta < \frac{\theta}{6}$, then there exists a subset S such that $|S| \geq 2$ and $\sum_{p \in S} p \leq \kappa\delta$ in all coordinates for some constant κ and $\sum_{p \in S} p \geq \delta$ in all coordinates in $\mathscr{C}$.

Proof of Claim: Let $\mathcal{B}''$ be set of all items of $\mathcal{B}'$ which are $< \frac{3\delta d}{\theta}$ in all coordinates of $\overline{\mathscr{C}}$. We formulate the following LP:

$$\begin{aligned} \min \sum_{p_i \in \mathcal{B}''} x_i & \\ \delta \leq \sum_{p_i \in \mathcal{B}''} x_i p_i^{\ell} \leq \frac{2\delta}{\theta} & \quad \forall\, \ell \in \mathscr{C}, \\ \sum_{p_i \in \mathcal{B}''} x_i p_i^{\ell} \leq \frac{2\delta}{\theta} & \quad \forall\, \ell \in \overline{\mathscr{C}}, \\ 0 \leq x_i^{\ell} \leq 1 & \quad \forall\, i \in \mathcal{B}'', \ell \in [d]. \end{aligned} \tag{23}$$

To show the feasibility of this LP, let us consider $\boldsymbol{x'} = (\frac{2\delta}{\theta}, \dots, \frac{2\delta}{\theta})$. Let $\mathring{\mathcal{B}} = \mathcal{B}' \setminus \mathcal{B}''$. There could be at most $\frac{d\theta}{3\delta d}$ items in $\mathcal{B}'$ bigger than $\frac{3\delta d}{\theta}$ in some coordinate in $\mathscr{C}$. As these items are $\leq \delta$ in all coordinates in $\mathscr{C}$, $\sum_{p_i \in \mathring{\mathcal{B}}} p_i \leq \delta \cdot \frac{d\theta}{3\delta d} \leq \theta/3$ in all coordinates in $\mathscr{C}$. Now $\sum_{p_i \in B'' \cup \mathring{\mathcal{B}}} p_i \geq (\theta - \delta)$ in all coordinates in $\mathscr{C}$. Thus the sum $\sum_{p_i \in B''} p_i$ is at least $(\theta - \delta - \theta/3) \geq \theta/2$ in all coordinates of $\mathscr{C}$ by taking $\delta < \theta/6$. Therefore, we have $\sum_{p_i \in B''} x_i p_i > \frac{2\delta}{\theta} \cdot \frac{\theta}{2} \geq \delta$ in coordinates of $\mathscr{C}$. Since the items of $\mathcal{B}''$ were packed into a single bin, we know that $\sum_{p_i \in \mathcal{B}''} p_i \leq 1$ and therefore $\sum_{p_i \in \mathcal{B}''} x_i p_i \leq \frac{2\delta}{\theta}$ in all coordinates and thereby $\boldsymbol{x'}$ is a feasible solution of (23).

Let us fix a basic optimal solution $\boldsymbol{x}$ to (23) and set $S = \{p_i | x_i > 0\}$. We claim that the sum $\sum_{p_i \in S} p_i$ is at most $\kappa\delta$ in each dimension for some constant κ dependent just on δ and d. Let F denote the set of fractional variables x_i. We know that each basic solution to the LP (23) fulfils at least $|B''|$ constraints with equality. Apart from constraints $0 \leq x_i^\ell \leq 1$, which would make the variable integral if fulfilled with equality, there are at most $(2|\mathscr{C}| + |\overline{\mathscr{C}}|) \leq 2d$ nontrivial constraints and they allow us $|F| \leq 2d$. Since each item is at most $3\delta d/\theta$ in all dimensions we get $\sum_{p_i \in S} p_i \leq 2\delta + 2d \cdot \frac{3\delta d}{\theta} \leq 7\delta d^2/\theta$ as $d \geq 2$. Hence, the claim follows by taking $\kappa = 7d^2/\theta$.

This completes the proof. □

If we consider $\epsilon < \frac{\theta}{28d^2}$, then $\delta < \frac{\theta}{28d^2}$. Thus $2 \cdot \kappa\delta < (1-\delta)$ and $1/2 + \kappa\delta < (1-\delta)$. Thus from each bin we remove S and for any two bins, the removed elements can be packed into bins that have slack $\geq \delta$ in at least $d-1$ coordinates or contains at most two items.

Lemma 4.5.9. Let I be an instance of d-dimensional vector packing where $d > 2$. Any optimal packing of I into m bins can be transformed to a packing into $\lceil \frac{d+1}{2} m \rceil$ bins of types $\mathcal{B}_S, \mathcal{B}_T, \mathcal{B}_1, \ldots, \mathcal{B}_d$.

Proof. For each bin B_i in the optimal packing, let V_i and S_i denote the sets guaranteed by Lemma 4.5.7. So, the bin with items in $B_i \setminus (V_i \cup S_i)$ is already of type $\mathcal{B}_\ell$ for some $\ell \in [d]$. We have to pack items in $V_i \cup S_i$ into bins of types $\mathcal{B}_S, \mathcal{B}_T, \mathcal{B}_1, \ldots, \mathcal{B}_d$. Now there are two cases.

Case A. *If d is odd*, i.e., $d-1$ is even. If $|V_i| \leq d-3$ we can pack all items in $|V_i|$ in pairs into at most $\frac{d-3}{2}$ bins. Otherwise if $|V_i| \geq d-2$, there is an item $v \in V_i$ such that v is $< 1/2$ in all coordinates except one (say d_y). Pack v with S_i (if $S_i \neq \emptyset$) or an arbitrary another element from V_i (if $S = \emptyset$) separately in another bin B'. Note that B' either has at most two items (if $S_i = \emptyset$) or has slack in all dimensions except d_y, satisfying condition of $\mathcal{B}_\ell$ bins. We can pack all remaining items in $|V_i|$ in pairs into at most $\frac{d-3}{2}$ bins. Total bins needed $\leq (\frac{(d-3)}{2} + 1 + 1)m \leq \lceil \frac{d+1}{2} m \rceil$.

Case B. *If d is even*, i.e., $d-1$ is odd.
If $|V_i| \leq (d-2)$ and $S_i \neq \emptyset$, then for any two bins B_i and B_j we can pack corresponding sets S_i and S_j together and we get a bin with at least δ slack in all dimensions. We pack items in V_i into pairs so that the total number of bins is at most $\leq \frac{d-2}{2} \cdot m + m + \lceil \frac{m}{2} \rceil \leq \lceil \frac{d+1}{2} m \rceil$ bins.

Only remaining case is when $|V_i| = (d-1)$ and $S_i = \emptyset$. Now if there is a set $R_i \subseteq B_i$ such that either $R_i = \{p_i\}$ with $p_i \leq 1/2$ in all coordinates or R_i is a set S'_i with $\sigma_{S'_i} \leq \frac{1}{2} - \delta$ in all dimensions except one, then we will show that all sets of type R_i from different bins can be paired together into bins of type $\mathcal{B}_T$ or $\mathcal{B}_\ell$. So, all R_i items from m bins can be packed into at most $\lceil \frac{m}{2} \rceil$ bins of type $\mathcal{B}_S, \mathcal{B}_T, \mathcal{B}_\ell$. For each bin we need to show existence of such R_i and the remaining items are packable in $\frac{d}{2}$ bins. Note that as each element in V_i creates slack in at least one distinct coordinate, $B_i \setminus V_i$ has slack in all coordinates except at most one, thus already in one of the structured types. Now there are two cases.

Case B1. There is an element $p_i \in V_i$ with $p_i \leq 1/2$. We can set $R_i = \{p_i\}$, pack $(d-2)$ other elements in $V_i \setminus \{p_i\}$ in pairs, $B_i \setminus V_i$ into a bin of type $\mathcal{B}_\ell$.

Case B2. There is no element $p_i \in V_i$ with $p_i \leq 1/2$. Thus all elements in V_i are $> 1/2$ in at least one distinct coordinate. Then there exists at least one vector v_x

that is $> 1/2$ in only one coordinate and $\leq 1/2$ in all other coordinates as $d > 2$. Without loss of generality, assume $V_i = \{v_{V,1}, \ldots, v_{V,d-1}\}$ and $v_{V,i} > 1/2$ in dimension i. Now consider the vector $v_{V,d}$ in $B_i \setminus V_i$ with highest value on d-dimension. There are three cases:

Case B1a. $v^d_{V,d} > 1/2$.

In this case the sum of all vectors in $B_i \setminus (V_i \cup \{v_{V,d}\})$ is $< 1/2$ in all dimensions. Let $\mathcal{Q}_i := \{q_1, \cdots, q_{|\mathcal{Q}_i|}\}$ be the set of dimensions where $\sigma_{B_i \setminus (V_i \cup \{v_{V,d}\})}$ is $> \frac{1}{2} - \delta$. If $\mathcal{Q}_i \leq 1$, take $S'_i = B_i \setminus (V_i \cup \{v_{V,d})\}$ and pack remaining d vectors in pairs in $d/2$ bins. Otherwise if $\mathcal{Q}_i \geq 2$, pack $\mathcal{P}_i := \{v_{V,q_1}, \cdots, v_{V,q_{|\mathcal{Q}_i|}}, v_{V,l_1}\}$ in one bin where $l_1 \notin \mathcal{Q}_i$. For dimensions in $\mathcal{Q}_i$, $\mathcal{P}_i$ has slack $\geq \frac{1}{2} - \delta$ and in all other dimensions except possibly l_1, $\mathcal{P}_i$ has slack $> 1/2$. Thus this set has slack in all dimensions except possibly in dimension l_1. Pack elements in $B_i \setminus (V_i \cup \{v_{V,d})\}$ with another remaining vector v_{V,l_2}. Clearly this set has slack in all dimensions except dimension l_2. Remaining $(d - 1 - |\mathcal{Q}| - 1)$ vectors are packed in pairs. Total number of bins needed $\leq \lceil \frac{(d-1-|\mathcal{Q}|-1)}{2} \rceil + 1 + 1 \leq \lceil \frac{d}{2} \rceil$ as $|\mathcal{Q}_i| \geq 2$ and $d \geq 4$ (Since, $d > 2$ and d is even).

Case B1b. $1/2 \geq v^d_{V,d} \geq \delta$.

Simply take $p_i = \{v_{V,d}\}$. Pack v_x with $B_i \setminus (V_i \cup \{p_i\})$ in a bin that will have slack in all dimensions except possibly in dimension x. Remaining $(d-2)$ vectors in V_i can be packed in pairs.

Case B1c. $v^d_{V,d} < \delta$.

If $B_i \setminus V_i \leq \frac{1}{2} - \delta$ in dimension d, pack v_x with $B_i \setminus V_i$ in a bin that will have slack in all dimensions except possibly in dimension x. Pack remaining vectors in pairs. Last case is when $B_i \setminus V_i > \frac{1}{2} - \delta$ in dimension d. Let $\mathcal{H}_i$ be the set of coordinates where $B_i \setminus V_i > \frac{1}{2} - \delta$. Apply the Claim in the proof of Lemma 4.5.7 restricted to coordinates in $\mathcal{H}_i$ and take $\mathscr{C} = \{d\}$, $\theta = \frac{1}{2}$ to get a set S'_i that is $> \delta$ in $\mathscr{C}$ and $< \kappa\delta$ in all coordinates in $\mathcal{H}_i$. Thus, v_x can be packed with $B_i \setminus (V_i \cup S'_i)$ that have slack in all dimensions except possibly dimension x. Remaining vectors can be packed in pairs. □

4.5.2 Finding the best structured packing

As a corollary of the result on resource augmentation we get,

Theorem 4.5.10. Let I_ℓ be an instance having a packing into m_ℓ bins of type $\mathcal{B}_\ell$. There is a poly-time algorithm which finds packing of I_ℓ into $(1+2\epsilon)m_\ell$ unit bins.

We note the following implication for future reference.

Observation 4.5.11. *To pack large items of I_ℓ, we only need to know the rounding specification, i.e., the sizes defining the groups $W_\ell^{u,j}$ and the number of items $w_\ell^{u,j}$ in these groups, and not the precise items themselves.*

4.5.2.1 The Overall Algorithm:

Now we describe the overall algorithm. From Lemma 4.5.9, we know that any optimal packing of I can be transformed into a packing of $m' := \lceil \frac{d+1}{2} m \rceil$ bins of types $\mathcal{B}_S, \mathcal{B}_T, \mathcal{B}_1, \dots, \mathcal{B}_d$. Let m_S, m_T, m_ℓ be the number of bins of types $\mathcal{B}_S, \mathcal{B}_T, \mathcal{B}_\ell$ respectively for $\ell \in [d]$. So, $\sum_\ell m_\ell + m_S + m_T \le \lceil \frac{d+1}{2} \cdot m \rceil$. Assume we can partition instance I into $d+2$ sets $I_S, I_T, I_1, \dots, I_d$ such that they correspond to elements of bins of type $\mathcal{B}_S, \mathcal{B}_T, \mathcal{B}_1, \dots, \mathcal{B}_d$ respectively in some structured packing of at most $\lceil \frac{d+1}{2} m \rceil$ bins. Now let us consider a graph G with vertices as items in $\mathcal{B}_T$, and there is an edge between two items v_i, v_j if they can be packed together in a bin, i.e., $\sigma_{\{v_i, v_j\}} \le \mathbf{1}$. Thus if we find the *maximum matching* in G, we get an optimal packing of items in $\mathcal{B}_T$. Then we can pack each item in I_S separately in m_S bins, pack items in I_T into m_T bins using the matching and pack items in I_ℓ into $(1+\epsilon+\frac{\epsilon^2}{2d})m_\ell + 1$ unit vector bins using Algorithm 1 with resouce augmentation in all coordinates except ℓ, since we have δ slack in all dimensions which could be overpacked by factor $(1+\epsilon)$ and $(1-\delta)(1+\epsilon) = 1$.

However in general there are exponentially many such partitions of I, thus we will try to get a good approximate partition. If we simply guess the number of items in bins of type $\mathcal{B}_S$ and $\mathcal{B}_T$ bins and rounding size classes used in Algorithm 1 for each $\ell \in [d]$, then the above approach might not work. For example, if we separate I_S and sub-instances I_ℓ first, since Algorithm 1 recognizes the items just up to the rounding, it can happen that the $2m_T$ items which are leftover for the matching type bins are not matchable together. On the other hand, if we try to pack the matching bins first, the standard matching can possibly select too small items which would result in moving a smaller item from the bins packed using Algorithm 1 and putting a larger one back. This way we loose guarantee for the performance of Algorithm 1, since we are in fact packing different items. Therefore, to get a better control over the items which will be matched, we guess for each rounding class how many items with similar or slightly smaller sizes will be placed in bins of type $\mathcal{B}_T$ or $\mathcal{B}_S$, and we find a matching respecting these requirements. Though matching will be found on the items with original sizes, this will ensure that even though we might not pack the same items into the matching bins as the structured packing did, we packed items from the same rounding class and therefore we do not affect the performance of Algorithm 1. We will also show polynomially many guesses are sufficient in this approach.

To describe the algorithm, we will use similar notations and results as in Section 4.4. Remember, r_A is the number of rounding classes of augmentable coordinates and there are $\lceil \frac{1}{\lambda} \rceil$ number of linear grouping classes inside each of them.

4.5.2.2 Preprocessing phase:

First, we separate the small items S of I, this can be done in linear time. We just put the small items aside for now, they will be packed later. Let us denote $I' := I \setminus S$.

4.5.2.3 Guessing phase:

Our algorithm is trying to find m', the minimum number of bins in a structured packing for I. It is clear that m' is at most n and at least $\lceil ||\sigma_I||_\infty \rceil$. So we can guess m' by binary search between n and $\lceil ||\sigma_I||_\infty \rceil$ in the following way. If the *Packing* phase of the algorithm finds a structured packing into m' bins, we decrease its value. Otherwise we increase it. After guessing of m', we have to guess the partition of m' bins to bins of types $\mathcal{B}_S, \mathcal{B}_T$ and $\mathcal{B}_1, \ldots, \mathcal{B}_d$. It is clear that there is a polynomial number of choices of m' and $m_S, m_T, m_1, \ldots, m_d$ and in the worst case we can enumerate them all in $O((m')^{(d+2)})$. As $m' = O(md)$ from Lemma 4.5.9, the next lemma follows.

Lemma 4.5.12. Let I be an instance having a structured packing into m' bins. The guessing phase will guess $m_S, m_T, m_1, \ldots, m_d$ in $O((md)^{(d+2)})$ time.

For each $\ell \in [d]$, we define the rounding classes W_ℓ^u for $u \in [\lceil \frac{1}{\alpha} \rceil]^{[d]\setminus\{\ell\}}$ according to the first part of the rounding procedure in Section 4.4. For each $\ell \in [d]$ we have r_A (constant) number of classes, none of them clearly containing more than n items, and inside of each class the items are to be rounded to size of one of $\lceil \frac{1}{\lambda} \rceil$ round vectors (again a constant), so there are at most $r_A \cdot n^{(\lceil \frac{1}{\lambda} \rceil)}$ possible choices of round vectors. This way we can classify $d \cdot r_L$ rounding classes $W_\ell^{u,j}$.

For each W_ℓ^u, we guess numbers $w_\ell^{u,1} = \cdots = w_\ell^{u,\lceil \frac{1}{\lambda} \rceil - 1}$ — the sizes of the regular linear grouping classes and $w_\ell^{u,\lceil \frac{1}{\lambda} \rceil} \leq w_\ell^{u,1}$ the size of the last possibly smaller group. Note, that by now we have prepared rounding instances for the resource augmented packing as in Algorithm 1. We also guess numbers $s_\ell^{u,j}$ and $t_\ell^{u,j}$ — the numbers of items of size corresponding to the class $W_\ell^{u,j}$ but contained in the bins of type $\mathcal{B}_S$ and $\mathcal{B}_T$, respectively. Note that any item v in I_S or I_T are actually not rounded. They are just assigned to a rounded class with size $\geq v$ and they are packed using matching with their original sizes.

As we are guessing 4 numbers for each of dr_L number of $W_\ell^{u,j}$ classes and there are n possibilities for each of these numbers, there are n^{4dr_L}, a polynomial number of possibilities to enumerate in total. Thus we get the next lemma.

Lemma 4.5.13. We can guess rounding $W_\ell^{u,i}$ (sizes of rounding classes) and the number of items $w_\ell^{u,j}, t_\ell^{u,j}, s_\ell^{u,j}$ for $u \in [\lceil \frac{1}{\alpha} \rceil]^{[d] \setminus \{\ell\}}, \ell \in [d], j \in \lceil \frac{1}{\lambda} \rceil]$ in polynomial time.

4.5.2.4 Assignment of items into rounded size:

We have generated some rounded instances by specifying rounding and number of items belonging to each class. However, it is not clear for an item v, which dimension should be the nonaugmentable dimension in rounding. To find out whether there exists an assignment of items in I into the rounding classes such that it will satisfy the requirements on the number of items we use a technique from [173]. We specify the following flow network $G := (V, E)$:

- we have vertices s and t for source and sink,
- create a vertex for each item and for each class $W_\ell^{u,i}$,
- add edge of capacity 1 from s to each item,
- for each item add an edge of capacity 1 from the item to all d possible rounding classes to which the item could belong,
- we add an edge of capacity $t_\ell^{u,j} + s_\ell^{u,j} + w_\ell^{u,j}$ from each class $W_\ell^{u,j}$ to t.

Using the algorithm of Dinic [59] we find a maximum integral flow from s to t in time $O(|E| \cdot |V|^2) \leq O(n + nd + dr_L).(n + dr_L)^2 = O_{d,\epsilon}(n^3)$. If it does not saturate some edge outgoing from s or incoming to t, this means that we either cannot assign some item to any class or some class cannot have the required number of items assigned. In this case we know that no valid assignment exists and the rounding is not feasible. We need to go back to the guessing phase and make another guess. Otherwise, we get a legal assignment of items into the rounding classes. Then we first separate and pack I_T. We pack I_S and I_ℓ afterwards.

4.5.2.5 Packing of I_T:

To find a packing of I_T into bins of type $\mathcal{B}_T$, we construct a graph (I', E), where $(p_i, p_j) \in E$ if p_i and p_j fit together into a bin i.e., $\sigma_{\{p_i,p_j\}} \leq \mathbf{1}$. Here we use vectors with original sizes and they are just assigned to some class W_ℓ^{ij} with size greater or equal to the size of the vector. We want to find a matching such that from each class W_ℓ^{ij}, t_ℓ^{ij} vectors are saturated by the matching. We formulate the following LP:

$$\begin{aligned} \sum x_{uv} &\leq 1 \quad \text{for all vertices } v \in V, \\ \sum_{u \in W_\ell^{ij}} x_{uv} &= t_\ell^{ij} \quad \text{for all classes } W_\ell^{ij}, \\ 0 \leq x_e &\leq 1 \quad \text{for all edge } e \in E. \end{aligned}$$

Note that this is a matching LP with a constant number of additional linear constraints, since the number of classes depends only on the value of ϵ. To solve this LP we use an algorithm of Chekuri, Vondrák, and Zenklusen [35] which either returns a matching which covers between t_ℓ^{ij} and $(1-\nu)t_\ell^{ij}$ items from each class W_ℓ^{ij} or it returns a certificate that this LP is infeasible (See Theorem 4.2.3). If the LP is infeasible, then either the rounding classes or the numbers t_ℓ^{ij} are incorrect and we try another guess.

The residual νt_ℓ^{ij} vectors from each class can be selected arbitrarily and we pack these $\leq 2\nu m_T$ items into $2\nu m_T$ number of additional bins using $(1-\nu)m_T + 2\nu m_T \leq (1+\nu)m_T$ bins in total. We choose $\nu = \epsilon$.

4.5.2.6 Packing of I_S:

From each class $W_\ell^{u,j}$ we select arbitrary $s_\ell^{u,j}$ items not belonging to I_T and pack them into separate m_S bins.

4.5.2.7 Packing of $I_1, \dots, I_d$:

For each $\ell \in [d]$ we take the leftover instances of $W_\ell^{u,j}$ which do not belong to I_T nor I_S. We denote I_ℓ the union of all classes $W_\ell^{u,j}$. Thus we get the following lemma.

Lemma 4.5.14. For a correct guess, we can pack I_ℓ into $(1+\epsilon)m$ bins in polynomial time. Moreover, the free space configuration will satisfy requirements of Lemma 4.4.5.

Proof. For a correct guess we can pack rounded items of type I_ℓ into $m_\ell + \lfloor \frac{\epsilon m_\ell}{2} \rfloor$ bins using Lemma 4.4.3 with ℓ to be the nonaugmentable dimension. Hence, total number of bins needed to pack all items in $\cup_\ell I_\ell$ is $\leq \sum_\ell (m_\ell + \lfloor \frac{\epsilon m_\ell}{2} \rfloor)$. □

4.5.2.8 Packing of small vectors:

Finally we pack the small vectors. From Lemma 4.5.14, $\sum_\ell m_\ell$ bins obtained in the previous step have at least the same small configuration as was the small configuration in corresponding bins of structured packing. So there is a feasible fractional packing of all small vectors into the space in $\sum_\ell m_\ell$ bins. We use another assignment LP similar to Lemma 4.4.5 to get the packing in at most another additional $\left\lceil \frac{\sum_\ell m_\ell}{2} \cdot (\epsilon + \frac{\epsilon^2}{d}) \right\rceil$ bins.

Proof of Theorem 4.5.3. From Lemma 4.5.12 and 4.5.13, the guessing stage takes polynomial time. For a correct guess, I_T items are packed into $(1+\epsilon)m_T$ bins, I_S items are packed into m_S bins. Remaining I_ℓ type items are packed in $\sum_\ell (m_\ell + \lfloor \frac{\epsilon m_\ell}{2} \rfloor)$ bins and small items are packed into additional $\sum_\ell \frac{m_\ell}{2} \cdot (\epsilon + \frac{\epsilon^2}{d}) + 1$ bins. Hence, total number of bins

$$\begin{aligned}
&\leq (1+\epsilon)m_T + m_S + \sum_\ell (m_\ell + \lfloor \frac{\epsilon m_\ell}{2} \rfloor) + + \sum_\ell \frac{m_\ell}{2} \cdot (\epsilon + \frac{\epsilon^2}{d}) + 1 \\
&\leq (1 + \epsilon + \frac{\epsilon^2}{2d})(m_S + m_T + \sum_\ell m_\ell) + 1 \\
&\leq (1 + \epsilon + \frac{\epsilon^2}{2d})(m') + 1.
\end{aligned}$$

□

```
    1. Separate set $S$ of small items
**Guessing stage:**
    2. Guess $m', m_S, m_T, m_1, \ldots, m_d$.
    3. For each $\ell \in [d]$: create classes $W_\ell^{u,j}$ and guess numbers $w_\ell^{u,j}, t_\ell^{u,j}, s_\ell^{u,j}$.
**Packing stage:**
    4. Find assignment of items into rounding classes using flow-net,
    5. Find multiobjective matching and pack $I_T$ into bins $\mathcal{B}_T$,
    6. Select $I_S$ arbitrarily according to numbers $s_\ell^{u,j}$ and pack it into bins $\mathcal{B}_S$,
    7. For each $\ell \in [d]$: Pack $I_\ell$ into $\mathcal{B}_\ell$ using Algorithm 1,
    8. Pack $S$ using assignment LP as in Lemma 4.4.5,
    9. If any step fails, then go to the next guess in Guessing stage.
    10. **return** packing $\mathcal{B}_S, \mathcal{B}_T, \mathcal{B}_1, \ldots, \mathcal{B}_d$.
```

Algorithm 2: d-DIM. VBP WITH $(\frac{(d+1)}{2} + \epsilon)$-ASYMPTOTIC APPROXIMATION

Proof of Theorem 4.5.1. From Lemma 4.5.9, there is a packing into $\lceil \frac{d+1}{2} m \rceil$ bins of type $\mathcal{B}_S, \mathcal{B}_T, \mathcal{B}_1, \ldots \mathcal{B}_d$. Then using Lemma 4.5.3, we get the number of required bins is at most $\leq \lceil \frac{d+1}{2} m \rceil \cdot (1 + \epsilon + \frac{\epsilon^2}{2d}) + 1$. □

4.5.3 Tight absolute approximation of $(3/2 + \gamma)$ for 2-D VBP

Now we will give an algorithm with absolute approximation ratio of $(1.5 + \gamma)$ for 2-D vector packing. Note that even for 1-D bin packing the lower bound of absolute approximation is 3/2.

Theorem 4.5.15. (Restatement of Theorem 4.0.2)
For any constant γ with $0 < \gamma < 1/4$, there is a polynomial time algorithm with an absolute approximation ratio of $(1.5 + \gamma)$ for 2-D vector packing.

Proof. Take $\gamma' = \gamma/2$. We give the algorithm in Algorithm 3. From Theorem 4.5.1, we can get a packing into $3m/2 + \gamma' m + c_2$ bins in polynomial time if m is the number of bins in the optimal packing. There are two cases.

Case A. $\gamma'\mathsf{Opt} \leq c_2$: As there are at most $c_1 = c_2/\gamma'$ (a constant) number of bins in the optimal solution, there are at most $c_1 d/\beta$ big items. Thus there are only a constant number of big configurations. So in polynomial time, we can find the big configurations in optimal packing and small vectors can be added using assignment LP except dc_1 number of small items. We can pack these small items into one additional bin by taking small β such that $\beta dc_1 \leq 1$.

Case B. $\gamma'\mathsf{Opt} > c_2$: From Theorem 4.5.1, we can get a packing into $3m/2 + \gamma' m + c_2 \leq 3m/2 + (2\gamma')m \leq (1.5 + \gamma)m$. □

Algorithm:

```
A. Find if there is a feasible packing into at most $m + 1$ bins for all $m \in [c_2/\gamma']$,
  A1. Find the large configurations in optimal packing,
  A2. Small vectors (except possibly at most $dc_1$ number of small items) are
  then added using assignment LP in the remaining space,
  A3. Pack remaining small items into one additional bin,
    Otherwise try the next value of $m$;
B. Run Algorithm 2 to find a feasible packing into $3m/2 + \gamma' m + c_2$ bins,
return any feasible packing found by the algorithm for minimim $m$.
```

Algorithm 3: 2-D VBP WITH $(\frac{3}{2} + \gamma)$- ABSOLUTE APPROXIMATION

This is interesting as it shows vector packing is much easier to approximate than geometric packing in terms of absolute approximability. On the other hand, though 1-D BP and 2-D VBP are quite distinct in asymptotic approximability, they are arbitrarily close for the absolute approximability setting.

4.6 Improved Approximation using R&A Framework

In this section, we combine the ideas from the previous section with the R&A framework. We describe the 2-dimensional case first and then consider the d-dimensional case.

We first show the following key result. We will use this property in the analysis of algorithms for 2-D and d-D vector packing. For a d-dimensional VBP instance I, suppose there is a ρ approximate *rounded-packing* $\mathcal{P}$, where big items are rounded to a constant number of types. Then the R&A framework gives a $(1 + \ln \rho)$ asymptotic approximation. We will again need the following concentration inequality [157] in the analysis.

Lemma 4.6.1. (Independent Bounded Difference Inequality) [157] Let $X = (X_1, \ldots, X_n)$ be a family of independent random variables, with $X_j \in A_j$ for $j = 1, \ldots, n$, and $f : \prod_{j=1}^{n} A_j \to \mathbb{R}$ be a function such that

$$f(x) - f(x') \leq c_j,$$

whenever the vectors x and x' differ only in the j-th coordinate. Let $\mathbb{E}[f(X)]$ be the expected value of the random variable $f(X)$. Then for any $t \geq 0$,

$$\mathbb{P}[f(X) - \mathbb{E}(f(X)) \geq t] \leq e^{-2t^2/\sum_{i=1}^{n} c_j^2}.$$

In particular, we show the following:

Theorem 4.6.2. Let I be a set of items and $\tilde{I}$ is a rounding of items in I such that all big items are rounded to t_1 (constant) types. Assume that there exists an optimal packing $\mathcal{P}$ of $\tilde{I}$ into m bins. If for $\lceil (\ln \rho) z^* \rceil$ iterations we select a configuration $C' \in \mathcal{C}$ at random with probability $x^*_{C'}/z^*$ as in R & A framework applied to I and J is the residual set of elements at the end of these iterations, then with high probability, for any small constant $\epsilon' > 0$, there is a *rounded packing* $\mathcal{P}_J$ of J into $\frac{(1+\epsilon')m}{\rho} + c_m$ bins where all big items are rounded to t_1 types (thus there are also only constant types of small configurations) and c_m is a constant.

Proof. As there are t_1 (constant) types of big items and there can be at most d/β big items in a configuration, there are $(d/\beta)^{t_1}$ types of big configurations and thus $(d/\beta)^{t_1}$ types of small configurations. Let t_2 $(\leq (d/\beta)^{t_1})$ be the number of types of small configurations in $\mathcal{P}$. Let $\mathcal{R}_{j_1}$ be the set of big items of j_1'th type and $\mathcal{S}_{j_2}$ be the set of small items belonging to j_2'th type of small configurations in $\mathcal{P}$ for $j_1 \in [t_1]$ and $j_2 \in [t_2]$. Let $|\mathcal{R}_{j_1}| = r_{j_1}$ for $j_1 \in [t_1]$, i.e., r_{j_1} is the number of items in the class of big items $\mathcal{R}_{j_1}$. Also let s_{j_2} be the number of bins with small configuration of type j_2. Let $h^C_{j_1}$ be the number of items of type $\mathcal{R}_{j_1}$ in configuration C and $g^C_{j_2} = 1$ if configuration C has small configuration of type j_2. Consider the following $LP(\tilde{I})$ where $\mathcal{C}$ is the set of configurations of $\tilde{I}$:

$$\begin{aligned} \min \sum_{C \in \mathcal{C}} x_C & \\ \sum_{C \in \mathcal{C}} h^C_{j_1} x_C \geq r_{j_1} & \quad \forall\, j_1 \in [t_1], \\ \sum_{C \in \mathcal{C}} g^C_{j_2} x_C \geq s_{j_2} & \quad \forall\, j_2 \in [t_2], \\ x_C \geq 0 & \quad \forall\, C \in \mathcal{C}. \end{aligned} \tag{24}$$

Let $t = t_1 + t_2$. As the LP has t (which is a constant) number of constraints and the optimal integral solution of $LP(\tilde{I})$ has value m,

$$m \leq LP(\tilde{I}) + t. \tag{25}$$

Intuitively in $LP(\tilde{I})$ we consider small configurations of each bin as a single item. Let $\epsilon_1 > 0$ be a small constant that we will choose later. Now define another instance J' such that it contains $\frac{r_{j_1}(1+\epsilon_1)}{\rho}$ big items of type $j_1 \in [t_1]$ and $\frac{s_{j_2}(1+\epsilon_1)}{\rho}$ items of type $j_2 \in [t_2]$ such that the size of each item in type j_2 is equal to the size of j_2'th type of small configuration. Intuitively J' is a shrunk down version of $\tilde{I}$ where each small configuration is replaced by a single item of that same size.

Now consider the following $LP(J')$:

$$\begin{aligned}
&\min \sum_{C\in\mathcal{C}} x_C \\
&\sum_{C\in\mathcal{C}} h_j^C x_C \geq \frac{r_j(1+\epsilon_1)}{\rho} \quad \forall\, j \in [t_1], \\
&\sum_{C\in\mathcal{C}} g_j^C x_C \geq \frac{s_j(1+\epsilon_1)}{\rho} \quad \forall\, j \in [t_2], \\
&\qquad x_C \geq 0 \quad \forall\, C \in \mathcal{C}.
\end{aligned} \tag{26}$$

As the right hand side of constraints in $LP(\tilde{I})$ and $LP(J')$ differ by a factor of $\frac{(1+\epsilon_1)}{\rho}$, we get:

$$\frac{(1+\epsilon_1)}{\rho} LP(\tilde{I}) = LP(J'). \tag{27}$$

Let $\mathsf{Opt}(J')$ be the optimal integral solution for the above LP.

Now consider applying the same rounding as in $\tilde{I}$ to the items in J. Let us denote this instance to be $(\tilde{I} \cap J)$. To conclude the proof we will show that the items in $(\tilde{I} \cap J)$ can be packed in $(1+\epsilon_2)\mathsf{Opt}(J')$ bins, for a small positive constant ϵ_2 that we will choose later..

Lemma 4.6.3. $\mathsf{Opt}(\tilde{I} \cap J) \leq (1+\epsilon_2)\mathsf{Opt}(J')$.

Proof. As J is the set of residual items that are not covered by any bins selected in $\lceil (\ln \rho) z^* \rceil$ iterations in *packing using randomized rounding.* thus for any item $i \in I$,

$$\begin{aligned}
\mathbb{P}(i \in J) &= (1 - \sum_{C \ni i} x_C^*/z^*)^{\lceil (\ln \rho) z^* \rceil} \\
&\leq (1 - \sum_{C \ni i} x_C^*/z^*)^{(\ln \rho) z^*} \\
&\leq (1 - 1/z^*)^{(\ln \rho) z^*} && (28) \\
&\leq e^{(-\ln \rho)} && (29) \\
&= \frac{1}{\rho}. && (30)
\end{aligned}$$

Here inequality (28) follows from the fact that $\sum_{C \ni i} x^*_C \geq 1$ for all $i \in I$ and inequality (29) follows from the fact that $(1 - \frac{1}{x})^{\alpha x} \leq e^{-\alpha}$ for $x > 0$.

Hence from (30), $\mathbb{E}[|\mathcal{R}_i \cap J|] \leq |\mathcal{R}_i|/\rho$ for all $i \in [t_1]$. Now we will the concentration around the mean using independent bounded difference inequality.

Consider the function $f^B_i(x) = \mathcal{R}_i \cap J$, i.e., the number of items of type $\mathcal{R}_i$ in J. This is a function of $X = (X_1, \ldots, X_{\lceil (\ln \rho) z^* \rceil})$, i.e., $\lceil (\ln \rho) z^* \rceil$ independent random variables (selected configurations in randomized rounding phase of R & A framework) where X_i corresponds to the random variable for the configuration selected in ith iteration. Now changing value of any of these random variables may lead to selection of a different configuration C' in place of configuration C in the corresponding iteration. Let vectors x and x' differ only in j'th coordinate, i.e., a different configuration C' is selected in place of configuration C. This might lead to a different set of residual items J'. Then

$$\begin{aligned} f^B_i(x) - f^B_i(x') &\leq (|\mathcal{R}_i \cap J| - |\mathcal{R}_i \cap J'|) \\ &\leq max\{|\mathcal{R}_i \cap C|, |\mathcal{R}_i \cap C'|\} \\ &\leq 1/\beta. \end{aligned} \tag{31}$$

Here inequality (31) follows from the fact that there can be at most β items of type $\mathcal{R}_i$ in a bin as big items are $\geq \beta$ in at least one of the dimensions.

Therefore, from independent bounded difference inequality, we get,

$$\mathbb{P}[f^B_i(X) - \mathbb{E}(f^B_i(X)) \geq \gamma z^*] \leq e^{-2(\gamma z^*)^2 / (\frac{\lceil (\ln \rho) z^* \rceil}{\beta^2})}.$$

Thus in the asymptotic case (when z^* is sufficiently large, i.e., $>> \frac{\ln \rho \cdot t_1 t_2}{\gamma^2 \beta^2}$) we can take union bound over all t_1 cases and with high probability for all large item classes, $f^B_i(X) - \mathbb{E}(f^B_i(X)) < \gamma z^*$. We can take $\gamma = \frac{\epsilon_3^2}{t_1 t_2 \rho}$, for some small positive constant ϵ_3 that we will choose later. As in the packing of J', we have $\frac{(1+\epsilon_1)}{\rho}|\mathcal{R}_i|$ big items of type $\mathcal{R}_i$, we can pack $\mathbb{E}(f^B_i(X)) \leq |\mathcal{R}_i|/\rho$ items in $\tilde{I} \cap J$ in the slots of same type items in J'. Then we can pack all remaining big items in at most $t_1\gamma z^* \leq \frac{\epsilon_3^2 z^*}{t_2\rho} \leq \frac{\epsilon_3^2 LP(\tilde{I})}{t_2\rho} \leq \frac{\epsilon_3^2 LP(J')}{t_2} \leq \frac{\epsilon_3^2 \mathsf{Opt}(J')}{t_2}$ extra bins.

Now let us consider the small items. Consider a small configuration $\mathscr{S}_j = (h_1, \ldots, h_d)$. Let $\mathcal{S}_j$ be the set of items in all small configurations of type $\mathscr{S}_j$ and s_j be the number of small configurations of type $\mathscr{S}_j$. Let function $f^k_{\mathcal{S}_j}$ be $\sum_{v\in\mathcal{S}_j\cap J} v^k.\frac{1}{h_k}$, i.e., the length of items in $\mathcal{S}_j \cap J$ in k'th dimension scaled by a factor $\frac{1}{h_k}$. Intuitively the scaling makes each dimension of $\mathscr{S}_j$ to be one. This is again function of $\lceil(\ln\rho)z^*\rceil$ independent random variables (selected configurations). Thus as above, whenever the vectors x and x' differ only in one coordinate, we get:

$$\begin{aligned} f^k_{\mathcal{S}_j}(x) - f^k_{\mathcal{S}_j}(x') &\leq \sum_{v\in\mathcal{S}_j\cap J} v^k.\frac{1}{h_k} - \sum_{v\in\mathcal{S}_j\cap J'} v^k.\frac{1}{h_k} \\ &\leq max\{\sum_{v\in\mathcal{S}_j\cap C} v^k.\frac{1}{h_k}, \sum_{v\in\mathcal{S}_j\cap C'} v^k.\frac{1}{h_k}\} \\ &\leq \frac{1}{h_k}\cdot h_k \leq 1. \end{aligned} \tag{32}$$

Thus, from independent bounded difference inequality, we get,

$$\mathbb{P}[f^k_{\mathcal{S}_j}(X) - \mathbb{E}(f^k_{\mathcal{S}_j}(X)) \geq \gamma z^*] \leq e^{-2(\gamma z^*)^2/(\lceil(\ln\rho)z^*\rceil)}.$$

Now if $s_j < \frac{\epsilon_3 z^*}{t_1 t_2}$, one can pack these small items in all t_2 classes of small configurations into at most $\epsilon_3 z^*/t_1$ additional bins. Otherwise if $s_j \geq \frac{\epsilon_3 z^*}{t_1 t_2}$, then with high probability, $f^k_{\mathcal{S}_j}(X) \leq (\frac{s_j}{\rho} + \gamma z^*) \leq (1+\epsilon_3)\cdot\frac{s_j}{\rho}$. Thus, all small items in $\mathcal{S}_j \cap J$ can be packed into the corresponding small configurations of $\frac{(1+\epsilon_3)}{\rho}s_j$ bins in J' fractionally by assigning each item to each bin with $\frac{\rho}{(1+\epsilon_3)s_j}$ fraction.

So using assignment LP we can get an integral packing of all items into corresponding small configurations of $\frac{(1+\epsilon_3)}{\rho}s_j$ bins except $\lceil d\cdot\frac{(1+\epsilon_3)}{\rho}s_j\rceil$ items. However the original sizes of these $\lceil d\cdot\frac{(1+\epsilon_3)}{\rho}s_j\rceil$ items are $\leq \beta$ in all dimensions. So, we can pack them in extra $\leq \lceil\frac{\lceil d\frac{(1+\epsilon_3)}{\rho}s_j\rceil}{\lfloor 1/\beta\rfloor}\rceil \leq \lceil(d\beta + \beta^2 d)\frac{(1+\epsilon_3)}{\rho}s_j\rceil$ number of bins.

Therefore in the asymptotic case, $\mathsf{Opt}(\tilde{I}\cap J)$, the total number of bins in the optimal packing of $(\tilde{I} \cap J)$ is:

$$(1+\epsilon_3)\mathsf{Opt}(J')+\frac{\epsilon_3\mathsf{Opt}(J')}{t_1}+\lceil(d\beta+\beta^2 d)\frac{(1+\epsilon_3)}{\rho}\mathsf{Opt}(J')\rceil+\frac{\epsilon_3^2\mathsf{Opt}(J')}{t_2} \leq (1+\epsilon_2)\mathsf{Opt}(J'),$$

by choosing $\epsilon_2 > \epsilon_3 + \epsilon_3/t_1 + {\epsilon_3}^2/t_2 + (d\beta + \beta^2 d)(1+\epsilon_3)/\rho$. □

Therefore, in the asymptotic case when z^* is sufficiently large,

$$\begin{aligned} \mathsf{Opt}(J) \leq \mathsf{Opt}(\tilde{I} \cap J) &\leq (1+\epsilon_2)\mathsf{Opt}(J') && (33)\\ &\leq (1+\epsilon_2)(LP(J')+t) && (34)\\ &\leq \frac{(1+\epsilon_1)(1+\epsilon_2)}{\rho}LP(\tilde{I})+O(1) && (35)\\ &\leq \frac{(1+\epsilon')}{\rho}LP(\tilde{I})+O(1) && (36)\\ &\leq \frac{(1+\epsilon')}{\rho}\mathsf{Opt}(\tilde{I})+c_m. && (37) \end{aligned}$$

Inequality (33) follows from Lemma 4.6.3. Inequality (34) follows from the fact that there are t number of constraints in LP (26). Inequality (35) follows from inequality (27). Inequality (36) is obtained by chossing $\epsilon_1, \epsilon_2, \epsilon_3$ such that $\epsilon' = (1+\epsilon_1)(1+\epsilon_2)-1$ and c_m is the additive constant.

This completes the proof. □

Note that one can find such a (near)-optimal rounded packing in polynomial time by guessing the round vectors as we discussed in the previous section. So we need to show existence of structural packing with better guarantees.

4.6.1 Approximation Algorithm for 2-D vector packing:

Now we show a $(1+\ln(3/2))$-asymptotic approximation algorithm for any set of items I. First let us briefly mention the main idea for the algorithm. Note that Theorem 4.6.2 applies only to packings where all items are rounded. However, as we already know from Section 4.3, if we round all the items of I to the constant number of types, the optimum of the rounded instance can become as large as $2\,\mathsf{Opt}(I)$. To overcome this difficulty, we identify problematic items which we do not want to round, pack them into separate bins, and apply R&A to the rest of the instance. This leads us to the following definitions:

Definition 4.6.4. A bin B is *compact*, if it has a subset of items K with $|K| \leq 2$ and $(\sum_{v \in K} v) \geq (1-\delta, 1-\delta)$.

Definition 4.6.5. A bin B is *non-compact*, if it has no subset of items K with $|K| \leq 2$ and $(\sum_{v \in K} v) \geq (1-\delta, 1-\delta)$.

We claim, that it is just the compact bins what cause troubles, and R&A can be applied to non-compact bins successfully. Let m_C and m_N denote the number of compact and non-compact bins respectively in the optimum packing of I.

Separating compact bins: We separate pairs of large items belonging to the compact bins using an idea similar to the preceding section. First, we guess rounding classes $W_\ell^{u,j}$ for rounding of the non-compact bins together with the numbers $c_\ell^{u,j}, w_\ell^{u,j}$ of items from each class which are to be packed into compact bins and noncompact bins, respectively. The graph $G = (I, E)$ is a little bit different: we add edge between v and v', if they form a compact bin together. Then we find a MOMB matching in G satisfying guessed requirements $c_\ell^{u,j}$, and pack the matched items into roughly m_C separate bins.

Single large items bigger than $(1-\delta)$ in both coordinates can be separated easily in linear time. Since the volume of the small items in compact bins is negligible, they cause no harm to the R&A part (they can be packed into at most $2\delta m_C \leq 2\delta m$ additional bins). Therefore, we do not need to separate them.

Packing non-compact bins using R&A: We apply R&A to the rest of the instance: we solve the configuration LP and perform the randomized rounding (B2 in Algorithm 4). Note, that we use at most $\lceil m \cdot \ln \rho \rceil$ bins in this step. Let us denote

S the set of residual items. Now, we use an important property of non-compact bins: By the arguments in Lemma 4.5.4 and Lemma 4.5.5, it can be shown that items from m_N non-compact bins can be rounded to constant number of types and repacked into $\frac{3}{2}m_N$ bins[6]. Therefore, we can apply Theorem 4.6.2 which roughly says, that $\mathsf{Opt}(S) \leq \frac{2}{3}m_N$. Then we pack S using $\frac{3}{2}$-approximation algorithm into roughly m_N bins. Altogether, we used roughly $m_C + \lceil \ln \rho \rceil m + m_N$ bins.

First, we prove the following two lemmas regarding packing of compact and non-compact bins.

Lemma 4.6.6. All items in m_N non-compact bins can be packed into $\lceil 3m_N/2 \rceil(1+2\epsilon)$ rounded bins, i.e., bins where large items are rounded to constant types.

Proof. From structural properties in Lemma 4.5.4, for non-compact bins either there is a large item $p \in B$ such that $p \leq (1/2, 1/2)$ and p is $> \delta$ in at least one coordinate, or there is a subset S of items in B such that $\sum_{p \in S} p \leq (2\delta, 2\delta)$ and $B \setminus S$ has δ-slack in some dimension. Thus the removal of item(s) p or S from such bin B creates δ slack in B in at least one dimension. So from the proof of Lemma 4.5.5, any two such bins can be repacked into 3 bins such that either it has slack δ in one dimension or it contains two p type items. Now if bins have δ-slack in at least one dimension, using Algorithm 1 we can get a rounded packing loosing only a factor $(1+\epsilon+\epsilon^2/d) \leq (1+2\epsilon)$ for small ϵ. On the other hand, for bins containing two p type items, one can just round p type items into (1/2,1/2). So for m_N non-compact bins we can produce a rounded-packing in $\lceil 3m_N/2 \rceil(1+2\epsilon)$ rounded bins. □

Lemma 4.6.7. All items in m_C compact bins can be packed into $(1+2\delta)m_C + 1$ bins where m_C bins contain single or two items and remaining $2\delta m_C + 1$ bins contain only small items.

[6] Precisely, can be repacked into bins which either have δ-slack in some dimension, or can be viewed as having two items of size at most $(1/2, 1/2)$. In both cases, the items can be rounded to a constant number of types.

Proof. Given m_C compact bins, we can remove the items $\leq (\delta, \delta)$ in the compact bins and greedily pack them into additional $\lceil \frac{m_C}{\lfloor 1/\delta \rfloor} \rceil \leq (\delta + 2\delta^2)m_C + 1 \leq 2\delta m_C + 1$ extra bins for small values of δ. After removing these items remaining compact bins are of type $\mathcal{B}_S$ or $\mathcal{B}_T$. $\square$

Thus there is a packing into $(1+2\delta)m_C + \frac{3}{2}(1+2\epsilon)m_N + 2$ bins such that m_C bins are of type $\mathcal{B}_S$ or $\mathcal{B}_T$, where as large items in $\frac{3}{2}(1+2\epsilon)m_N + 1$ bins are rounded, and other $2\delta m_C + 1$ bins contain small items. We use this observation in our algorithm given below.

```
A. Guessing stage:
  A1. Guess $\mathsf{Opt}(I), m_C$, and $m_N$, and take $\epsilon = \frac{\epsilon'}{6}$,
  A2. For each $\ell \in [d]$:
  Create classes $W_\ell^{u,j}$ and guess corresponding round vectors and $w_\ell^{u,j}, c_\ell^{u,j}$, the
  number of items in compact and noncompact bins in each size classes,
B. Packing stage:
  B1. Use MOMB matching on the original item sizes to pack $c_\ell^{u,j}$ items from
  class $W_\ell^{u,j}$ into $(1+\delta)m_C$ bins,
  B2. Solve configuration LP restricted to remaining items and apply
  randomized rounding with parameter $\rho = 3/2$ for $\lceil z^* \cdot \ln(\rho) \rceil$ iterations,
  B3. Let $S$ be set of remained items, pack it using algorithm 2 into
  $m_N + 4\epsilon m + c_m + 3$ bins, where $c_m$ is the additive constant from Theorem
  4.6.2.
  B4. If packing in Step B1 or B3 fails, go to next guess.
```

Algorithm 4: $(1.405 + \epsilon')$-approximation for 2-D VBP

4.6.1.1 Analysis of the Algorithm

Theorem 4.6.8. (Restatement of Theorem) For any small constant $\epsilon' > 0$, there is a polynomial time algorithm (Algorithm 4) with an asymptotic approximation ratio of $(1 + \ln(1.5) + \epsilon') \approx (1.405 + \epsilon')$ for 2-D vector packing.

Proof. The binary search to find m $(:= \mathsf{Opt}(I))$ takes $O(\log n)$ time. For each guess there are $O(n^2)$ guesses for m_C, m_N. Also as there are total t_L (constant) number of round vectors, they can be guessed in $O(n^{t_L})$ time. Thus in polynomial time we can guess the suitable values for the *guessing stage* in the algorithm.

Now there are three steps in the *packing stage* of the algorithm. In the first step we pack $(1+\delta)m_C$ number of bins using multi-objective multi-budget matching on the original item sizes. In the second step we pack using *randomized rounding of configurations* and we use at most $\lceil (\ln \rho) z^* \rceil \leq \ln \rho \cdot m + 1$ bins. We choose $\rho = 3/2$. In the last step we pack the remaining items in S into $m_N + 4\epsilon m + c_m + 3$ number of bins. So, for any feasible solution the total number of bins needed $=$ $(1+\delta)m_C + (\ln \rho)m + m_N + 4\epsilon m + O(1) \leq (1+\epsilon)m_C + (\ln \rho + 4\epsilon)(m_C + m_N) + m_N + O(1) \leq$ $(1+\epsilon' + \ln \rho)(m_C + m_N) + O(1)$ bins, where $\epsilon' = 6\epsilon$. This gives $(1 + \ln(\frac{3}{2}) + \epsilon')$ asymptotic approximation if the algorithm returns a feasible solution.

To complete the proof we need to show that the algorithm always returns a feasible packing for a correct guess.

In the *packing using matching* step, we create an edge between two nodes v_i and v_j only if they form a compact bin together, i.e., $v_i + v_j \geq (1-\delta, 1-\delta)$. Note that we can always separate very large items $v_i \geq (1-\delta, 1-\delta)$ and pack them separately. In this step the items we pack in m_C bins might not be the same items in the compact bins in the optimal solution. However the packed items in these m_C bins are very similar to those items in the compact bins. We pack c_ℓ^{ij} items from W_ℓ^{ij} using MOMB matching into $(1+\delta)m_C$ bins as in Section 4.5 with $\nu = \delta$. So, for a correct guess at the end of *packing using matching*, each class W_ℓ^{ij} has only $\approx (c_\ell^{ij} + w_\ell^{ij}) - c_\ell^{ij} = w_\ell^{ij}$

items left.

So after packing using matching, from Lemma 4.6.6 and 4.6.7 remaining items have a rounded packing in $\frac{3m_N}{2}(1+2\epsilon)+2\delta m_C+2$ bins. Then we use randomized rounding. Now S is the set of residual items that are not covered by any bins selected in $\lceil(\ln\rho)z^*\rceil$ iterations in *packing using randomized rounding.* Thus from Theorem 4.6.2, the large items in S can be rounded to $O(1)$ types and packed into at most $(\frac{3m_N}{2}(1+2\epsilon)+2\delta m_C+2)(1+\epsilon)/\rho+c_m \leq m_N+4\epsilon(m_N+m_C)+c_m+3$ bins, where c_m is the additive constant from Theorem 4.6.2.

This concludes the proof. □

4.7 Improved Approximation Algorithm for d-Dimensional Vector Packing

For d-dimensional VBP, the approach for 2-D can not be directly applied as there are no analogous results for multiobjective d-dimensional matching. So we adopt a somewhat different approach. We first note the following two structural properties for d-dimensional VBP.

Theorem 4.7.1. For any $\epsilon'' > 0$, if there exists a feasible packing of a d-dimensional VBP instance I into m bins, there is a packing into at most $(2+\epsilon'')m+1$ bins such that m bins contain at most $(d-1)$ items in each of them and all other bins have $O(1)$ types of large items and small configurations.

Proof. Choose δ such that $\epsilon'' = \kappa\delta\cdot(1+2\kappa\delta)$. Let V_i, S_i be the sets from Lemma 4.5.7, for bin B_i for $i \in [m]$. Then after removal of $(V_i \cup S_i)$, remaining items in B_i bin have slack in $d-1$ dimensions and can be rounded using Algorithm 1. If $|S_i| \leq 1$, pack S_i and V_i together in one bin B'. Note that $|B'| \leq (d-1)$. Otherwise, pack V_i in one bin. As elements in S_i are $\leq \kappa\delta$ in all dimensions, we pack them separately in additional $\lceil m/(\lfloor\frac{1}{\kappa\delta}\rfloor)\rceil \leq \kappa\delta(1+2\kappa\delta)\cdot m+1$ bins. □

Lemma 4.7.2. If there exists a feasible packing of μn_1 items into a packing $\mathcal{P}_1$ of n_1 bins, one can pack them into a packing $\mathcal{P}_2$ of $\frac{(\mu+1)}{2} \cdot n_1$ bins in polynomial time.

Proof. We will show there exists $\frac{(\mu+1)}{2} \cdot n_1$ bins such that each bin contains one or two items. Then we can find a packing of μn_1 items into $\frac{(\mu+1)}{2} \cdot n_1$ bins using maximum matching. Assume x be number of bins in $\mathcal{P}_1$ that contain odd number of items. Pack one item from each bin with odd number of items, separately into x bins. Remaining bins contain even number of items. So, they can be packed in $(\mu n_1 - x)/2$ bins using matching. Hence, total $x + \frac{(\mu n_1 - x)}{2} \leq \frac{\mu n_1 + x}{2} \leq \frac{\mu n_1 + n_1}{2}$ bins are sufficient. □

Consider the packing in $(2+\epsilon'')m+1$ bins as mentioned in Theorem 4.7.1 where $m := \mathsf{Opt}(I)$. Let I_D be the set of items in those m bins containing at most $(d-1)$ items. Let $I_N = I \setminus I_D$, are the items in the remaining rounded bins. Note that if either I_D or I_N is small, we can pack them separately into an additional additive constant number of bins. Therefore, without loss of generality in the asymptotic case, let us assume I_D and I_N to be sufficiently large.

We use randomized rounding affront with $\rho = (d+1)/2$ and let J be the residual set of items. We show that we are left with nearly $2m$ items from I_D, which we pack using Lemma 4.7.2 and MOMB matching into $\frac{(1+3\epsilon)(d-1)m}{2\rho} + \frac{(1+\epsilon)m}{2} + c_m(1+\epsilon)$ bins, where c_m is the additive constant from Theorem 4.6.2. Using Theorem 4.6.2, we show that remaining items have a rounded packing in to $\frac{(1+10\epsilon)m}{\rho} + c_m(1+2\epsilon)+1$ bins. As before, let W_ℓ^{ij}'s correspond to the classes of the rounded vectors. We can arbitrarily assign vectors in I_D to any class with size dominating the size of the vector.

4.7.0.2 Analysis

Theorem 4.7.3. (Restatement of Theorem 4.0.3)
For any constant ϵ_c, such that $0 < \epsilon_c < \frac{1}{3d^2}$, there is a poly-time algorithm (Algorithm 5) with an asymptotic approximation ratio of $(1.5 + \ln(d/2) + o_d(1) + \epsilon_c) \approx \ln d + 0.807 + o_d(1) + \epsilon_c$ for d-dimensional VBP.

```
A. Guessing stage:
    A1. Guess $m := \mathsf{Opt}(I)$ and create rounding classes $W_\ell^{u,j}$, and take $\epsilon = \frac{\epsilon_c}{20}$,
B. Packing stage:
    B1. Choose $\rho = \frac{(d+1)}{2}$ and solve configuration LP and apply randomized rounding for $\lceil LP(I) \cdot \ln(\rho) \rceil$ iterations,
    B2. Let $J$ be the set of items left at the end of randomized rounding. Guess numbers $\mu_\ell^{ij} = |(I_D \cap J) \cap W_\ell^{ij}|, \nu_\ell^{ij} = |(I_N \cap J) \cap W_\ell^{ij}|$,
    B3. Use MOMB matching to pack $\mu_\ell^{u,j}$ items from classes $W_\ell^{u,j}$ into $\frac{(1+3\epsilon)(d-1)m}{2\rho} + \frac{(1+\epsilon)m}{2} + c_m(1+\epsilon)$ bins,
    B4. Pack remaining items using Algorithm 2 in $\frac{(1+10\epsilon)m}{\rho} + c_m(1+2\epsilon) + 1$ bins,
    B5. If packing in Step B3 or B4 fails, go to next guess.
```

Algorithm 5: $(\ln(d/2) + 1.5 + o_d(1) + \epsilon_c)$-APPROXIMATION FOR d-DIM. VBP.

Proof. In packing using randomized rounding we use at most $\ln(\frac{d+1}{2}) \cdot m + 1$ bins where $m = \mathsf{Opt}(I)$. If there is a feasible solution, packing with matching takes $\frac{(1+3\epsilon)(d-1)m}{2\rho} + \frac{(1+\epsilon)m}{2} + c_m(1+\epsilon)$ and packing using $O(1)$ rounding based algorithm takes $\frac{(1+10\epsilon)m}{\rho} + c_m(1+2\epsilon) + 1$ bins. In total, the number of bins is at most $(\ln(d/2) + 1.5 + o(1) + \epsilon_c)m + O(1)$ as $\epsilon_c = 20\epsilon$.

To complete the proof, we show the algorithm always returns a feasible packing. From (30), for any item $i \in I$, $\mathbb{P}(i \in S) = 1/\rho$. So, $\mathbb{E}[|I_D \cap S|] = (d-1)m/\rho$. One can use *independent bounded difference inequality* as in Theorem 4.6.2, to show concentration around the expectation. Thus with high probability, $|I_D \cap S| \leq \frac{(1+\epsilon)(d-1)m}{\rho} + c_m$, where c_m is the additive constant from Theorem 4.6.2. Note that it is not necessary that each of the m bins in the optimal solution contains exactly two items from $(I_D \cap S)$. Some of those bins can contain 1, 3 or other number of items from $(I_D \cap S)$. Thus though these items in $I_D \cap S$ are packable in m bins, we may not pack them into m bins in polynomial time using matching. That is where we use Lemma

4.7.2. Using Lemma 4.7.2 we can pack these $\frac{(1+\epsilon)(d-1)m}{\rho} + c_m$ items from $I_D \cap S$ into $\Big((\frac{(1+\epsilon)(d-1)m}{\rho} + m)/2\Big)(1+\epsilon) + c_m(1+\epsilon) \leq \frac{(1+3\epsilon)(d-1)m}{2\rho} + \frac{(1+\epsilon)m}{2} + c_m(1+\epsilon)$ bins using multi-objective multi-budget matching as in Section 4.5. Here the inequality followed from the fact $(1+\epsilon)^2 \leq (1+3\epsilon)$. Note that in this case the graph is created on items left after randomized rounding and an edge is created between two items if and only if they are packable together in a bin.

There is a rounded-packing of items in I_N into $m(1+\epsilon'')$ bins. Thus from Theorem 4.6.2, large items in J can be rounded to $O(1)$ types and be packed in $\frac{m(1+\epsilon'')(1+\epsilon)}{\rho} + c_m$ bins. So using Algorithm 1 we can get a packing into $(1+2\epsilon)\frac{m(1+\epsilon)(1+\epsilon)}{\rho} + c_m(1+2\epsilon) + 1 \leq \frac{(1+10\epsilon)m}{\rho} + c_m(1+2\epsilon) + 1$ bins by taking $\epsilon'' < \epsilon$ and from the fact that $(1+\epsilon)^2(1+2\epsilon) \leq (1+10\epsilon)$.

Thus in the asymptotic case, the number of bins needed is at most

$$\leq \quad m \cdot \ln(\rho) + \frac{(1+\epsilon)m}{2} + \frac{m(d-1)(1+3\epsilon)}{2\rho} + \frac{m(1+10\epsilon)}{\rho} + (2 + c_m(2+3\epsilon))$$

$$\leq \quad m \cdot \ln(\rho) + \frac{m}{2} + \frac{m(d+1)}{2\rho} + \epsilon m(\frac{1}{2} + \frac{3(d-1)}{2\rho} + \frac{10}{\rho}) + (2 + c_m(2+3\epsilon)) \quad (38)$$

$$\leq \quad m \cdot \ln(\frac{d+1}{2}) + \frac{m}{2} + m + \epsilon_c m + (2 + c_m(2+\epsilon_c)) \quad (39)$$

$$\leq \quad m(\frac{3}{2} + \ln(\frac{(d+1)}{2}) + \epsilon_c) + O(1). \quad (40)$$

Inequality (39) follows by taking $\rho = \frac{d+1}{2}$ and $\epsilon_c = 20\epsilon$. □

One can derandomize the algorithm using potential function based standard arguments in [13].

Note that the above algorithm obtains the present best approximation for all $d > 4$. For $d = 2$ and 3, our previous algorithms obtain the present best asymptotic approximation of $(1.405 + \epsilon)$ and $(2 + \epsilon)$ respectively.

4.8 Conclusion

Our bounds for small values of d can probably be improved by proving and using multi-objective multi-budget variant of d-dimensional matching for $d > 2$. However, this approach will not give an $(1-\delta)\ln d$ approximation for large d as there is already an $\Omega(d/\ln d)$ lower bound for d-dimensional matching.

As there is still no known explicit hardness for d-D VBP as function of d, it will be interesting to show a $f(d)$ hardness using some reduction from d-dimensional matching or other related problems.

Chapter V

WEIGHTED BIPARTITE EDGE COLORING

In this chapter we discuss our results on the weighted bipartite edge coloring which were motivated by the study of Clos networks [39]; this coloring problem also generalizes both bin packing and the classical edge coloring problem.

Clos Networks. Clos networks were introduced by Clos [39] in the context of designing interconnection networks with a small number of links to route multiple simultaneous connection requests such as telephone calls. Since then it has found various applications in data communications and parallel computing systems [126, 113]. The symmetric 3-stage Clos network is generally considered to be the most basic multistage interconnection network. Let $C(m, \mu, r)$ denote a symmetric 3-stage Clos network, where the input (first) stage consists of r crossbars of size $m \times \mu$, the center (second) stage consists of μ crossbars of size $r \times r$ and the output (third) stage consists of r crossbars of size $\mu \times m$. Moreover, there exists one link between every center switch and each of the r input or output switches. No link exists between other pair of switches. An example $C(2, 3, 4)$ network is shown in Figure 11.

A *request frame* is a collection of connection requests between inlets and outlets in the network such that each inlet or outlet is associated with at most one request. A request frame is *routable* if all requests are routed through a middle switch such that no two requests share the same link. An interconnection network is said to be *rearrangeably nonblocking* if all request frames are routable. In the classic switching environment all connection requests fully use a link and all have the same bandwidth. However in present networks, different requests might have different bandwidths (due

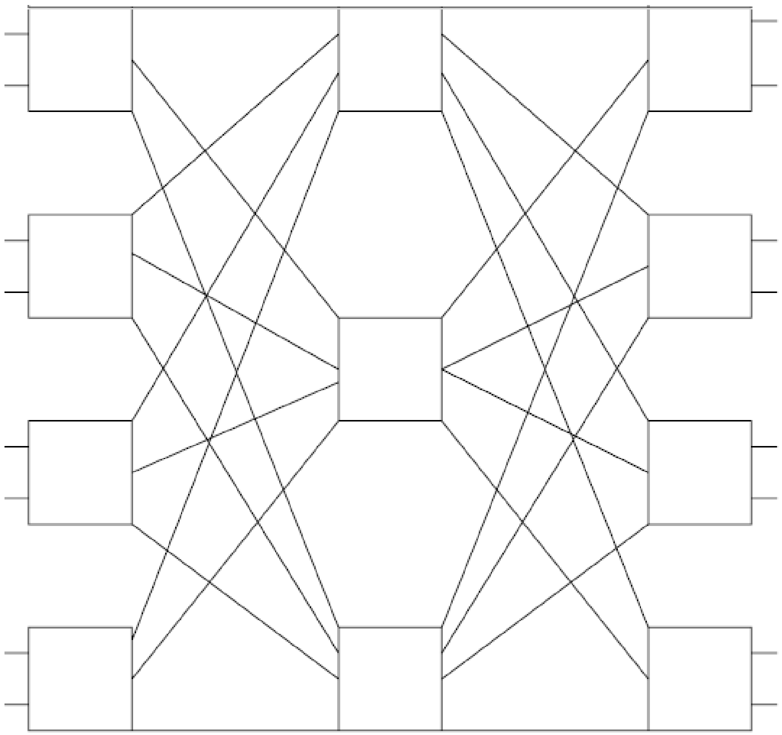


Figure 11: A Clos network with $m = 2, \mu = 3, r = 4$ [64]

to wide range of traffic such as voice, video, facsimile etc.) and may be combined in a given link if the combined request does not exceed the link capacity. In this *multirate* setting, a connection request is a triple (i, j, w), where i, j, w are inlet, outlet and demand of the connection, respectively, and all links have capacity one. Here a request frame is a collection of connection requests between inlets and outlets in the network such that the total weight of all requests in the frame for any particular inlet or outlet is at most one. The central question in 3-stage Clos networks is finding the minimum number of switches (crossbars) μ $(= \mu(m, r))$ in the middle stage such that all request frames are routable. It is particularly interesting to obtain bounds independent of r.

Weighted Bipartite Edge Coloring. Nonblocking rearrangeable properties of a 3-stage Clos network $C(m, \mu, r)$ can be translated to the following graph theoretic problem. Formally, in a WEIGHTED BIPARTITE EDGE COLORING problem, we are given an edge-weighted bipartite (multi)-graph $G := (V, E)$ with bipartitions A, B ($|A| = |B| = r$) and edge weights $w : E \to [0, 1]$. Let w_e denote the weight of edge $e \in E$. The goal is to obtain a *proper weighted coloring* of all the edges with a minimum number of colors. An edge coloring of the weighted bipartite graph is called a *proper weighted coloring* if the sum of (the weights of) the edges of the same color incident to a vertex is at most one for any color and any vertex. Here the sets A and B correspond to the input and output switches, edge (u, v) corresponds to a request between input switch u and output switch v. A routable request frame translates into the condition that weights of all incident edges to any vertex can be assigned a proper weighted coloring using m colors (or packed into m unit-sized bins) and the switches in the middle stage correspond to the colors (or bins). We refer the reader to Correa and Goemans [47] for a detailed discussion of this reduction.

Now let us introduce some notation. Let $\chi'_w(G)$ denote the minimum number of colors needed to obtain a proper weighted coloring of G. Let $m, r \in \mathbb{Z}^+$, and $\mu(m, r) = max_G \chi'_w(G)$ where the maximum is taken over all bipartite graphs, $G = (A \cup B, E)$ with $|A| = |B| = r$, and where m is the maximum over all the vertices of the number of unit-sized bins needed to pack the weights of incident edges. Chung and Ross [38] made the following conjecture:

Conjecture 5.0.1. *Given an instance of the* WEIGHTED BIPARTITE EDGE COLORING *problem, there is a proper weighted coloring using at most $2m - 1$ colors where m denotes the maximum over all the vertices of the number of unit-sized bins needed to pack the weights of edges incident at the vertex. In other words, $\mu(m, r) \leq 2m - 1$.*

There has been a series of results achieving weaker bounds on $\mu(m, r)$ (see related works for details), and the current best bound (due to Feige and Singh [77]) shows that $\mu(m, r) \leq 2.25m$.

Our results. Our main result is to make progress towards a resolution of Conjecture 5.0.1 by showing $\mu(m, r) \leq \frac{20m}{9} + o(m)$.

Theorem 5.0.2. There is a polynomial time algorithm for the WEIGHTED BIPARTITE EDGE COLORING problem which returns a proper weighted coloring using at most $\lceil 2.2223m \rceil$ colors where m denotes the maximum over all the vertices of the number of unit-sized bins needed to pack the weights of incident edges.

In our algorithm and analysis, we exploit the fact that the WEIGHTED BIPARTITE EDGE COLORING problem displays features of the classical edge coloring problem as well as the bin packing problem. We will use classical König's Theorem [143] as a subroutine in our algorithm. We will give an alternate proof of König's Theorem from skew-supermodularity in Section 5.2. Though these skew-supermodularity results are not used in our algorithm, they might be useful in other related problems and are of independent interest.

Our algorithm starts by decomposing the *heavy* weight edges into matchings by applying König's theorem to find an edge coloring of the subgraph induced by these edges. For the light weight edges, we employ the *first-fit decreasing* heuristic where we consider the remaining edges in decreasing order of weight and give them the first available color. The detailed algorithm is given in Figure 12 and builds on the algorithm by Feige and Singh [77].

Our work diverges from the previous results on this problem in the analysis of this simple combinatorial algorithm. We employ strong mathematical formulations for the bin packing problem; in particular, we use the *configuration linear program (LP)* for the bin packing problem. This linear program has been used to design the best approximation algorithm for the bin packing problem [133, 178, 106]. In our work, we use it as follows. We show that if the algorithm is not able to color an edge (u, v), then the edges incident at u or v cannot be packed into m bins as promised. To show this, we formulate the configuration linear program for the two bin packing problems

– one induced by edges incident at u and the other induced by edges incident at v. We then construct feasible dual solutions to these linear programs showing that the optimal primal value, and therefore the optimal bin packing number, is more than m for at least one of the programs, giving us the desired contradiction. While the weights on the edges incident at u (or v) can be arbitrary reals between 0 and 1, we group the items based on their weight classes and on how our algorithm colors these edges. This allows us to reduce the number of item types, reducing the complexity of the configuration LP, making it easier to analyze. While the grouping according to weight classes is natural in bin packing algorithms, the grouping based on the output of our algorithm helps us relate the fact that the edge (u, v) could not be colored by our algorithm to the bin packing bound at u and v. Our analysis can also be extended to show that $\lceil 2.2m \rceil$ colors are sufficient when all the edge weights are $> 1/4$.

Overview of the chapter. In Section 5.1, we survey related works. In Section 5.2, we present an alternate proof of König's Theorem using skew-supermodularity. Finally in Section 5.3, we present our edge-coloring algorithm and the related analysis.

5.1 Related Works

Edge-coloring problem has been one of the central problems in graph theory and discrete mathematics since its appearance in 1880 [198] in relation with the *four-color problem.* Three classical results on edge coloring are König's theorem [143] for coloring a bipartite graph with Δ colors, Vizing's theorem [203] for coloring any simple graph with $\Delta+1$ colors and Shannon's theorem [189] for coloring any multigraph with at most $3\Delta/2$ colors where Δ is the maximum degree of the graph . The *chromatic index* of a graph is the number of colors required to color the edges of the graph such that no two adjacent edges have the same color. Though one can find optimal edge coloring for a bipartite graph in polynomial time using König's theorem, Holyer [109] showed that it is even NP-hard to decide whether the chromatic index of a cubic graph is 3 or 4. We refer the readers to [195] for a survey on edge coloring.

Now let us review the literature related to weighted bipartite edge coloring. First let us introduce some more notation. When the weight function $w : E \rightarrow I$ is restricted to a subinterval $I \subseteq [0, 1]$, then we denote the minimum number of colors by $\mu_I(m, r)$. Slepian [192] showed that $\mu_{[1,1]}(m, r) = m$ using König's theorem. Melen and Turner [159] showed that $\mu_{[0,B]}(m, r) \leq \frac{m}{1-B}$ for $B \leq 1$. In particular $\mu_{[0,1/2]}(m, r) \leq 2m - 1$. There has been a series of works improving the bounds for $\mu(m, r)$ [38, 64, 166, 47]. The best known lower bound for $\mu(m, r)$ is 5/4 due to Ngo and Vu [166]. Correa and Goemans introduced a novel graph decomposition result and perfect packing of an associated continuous one-dimensional bin packing instance to show $\mu(m, r) \leq 2.5480m + o(m)$. The present best algorithm is due to Feige and Singh [77] who showed $\mu(m, r) \leq 9/4$. Their result holds even if m is the maximum over all the vertices of the total weight of edges incident at the vertex. For related results in general graphs we refer the readers to [77].

5.2 König's Edge-coloring Theorem

First we state the König's Theorem since we use it as a subroutine in our algorithm to ensure a decomposition of the set of edges into matchings.

Theorem 5.2.1. [143] Given a bipartite graph $G = (V, E)$, there exists a coloring of edges with $\Delta = max_{v \in V} deg_E(v)$ colors such that all edges incident at a common vertex receive distinct colors. Moreover, such a coloring can be found in polynomial time.

Now we give an alternate proof of König's Theorem. First let us define some preliminaries.

Definition 5.2.2. Intersecting family: A family $\mathcal{C}$ is called an *intersecting family* if for all $T, U \in \mathcal{C}$ with $T \cap U \neq \emptyset$, both $T \cap U$ and $T \cup U$ are in $\mathcal{C}$.

Definition 5.2.3. Intersecting supermodular: Given an intersecting family $\mathcal{C}$, a function $f : \mathcal{C} \to \mathbb{R}$ is called *intersecting supermodular*, if for all $T, U \in \mathcal{C}$ with $T \cap U \neq \emptyset$, we have:

$$f(T) + f(U) \leq f(T \cup U) + f(T \cap U)$$

Definition 5.2.4. Skew-supermodular functions: Given a finite ground set V, a set function $g : 2^V \to \mathbb{Z} \cup \{\infty\}$ is called *skew-supermodular*, if at least one of the following two inequalities holds for every $X, Y \subseteq V$:

$$g(X) + g(Y) \leq g(X \cap Y) + g(X \cup Y) \tag{41}$$

$$g(X) + g(Y) \leq g(X \setminus Y) + g(Y \setminus X) \tag{42}$$

5.2.1 Supermodular coloring and an extension

Schrijver [182] showed the following theorem on supermodular coloring:

Theorem 5.2.5. [182] Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be intersecting families of subsets of a set S, let $g_1 : \mathcal{C} \to \mathbb{Z}$ and $g_2 : \mathcal{C} \to \mathbb{Z}$ be intersecting supermodular, and let $k \in \mathbb{Z}_+$. Then S can be partitioned into classes $L_1, L_2, \ldots, L_k$ such that

$$g_i(U) \leq |\{j \in [k] : L_j \cap U \neq \emptyset\}| \tag{43}$$

for each $i = 1, 2$ and each $U \in \mathcal{C}_i$ if and only if

$$g_i(U) \leq min\{k, |U|\}\,. \tag{44}$$

König's edge-coloring theorem for bipartite graphs is a special case of above theorem. Let V_1 and V_2 be the color classes for bipartite graph G. Let $\mathcal{C}_i := \{\delta(v) | v \in V_i\}$ for $i = 1, 2$. If we define $g_i(\delta(v)) := |\delta(v)|$ for all $v \in V_i$ for $i = 1, 2$, then we get that any bipartite graph can be colored by $\Delta(V) := max_{v \in V} \delta(V)$ colors.

Bernáth and Király showed the following *skew-supermodular coloring theorem* which is an extension of Schrijver's supermodular coloring theorem.

Theorem 5.2.6. Skew-supermodular coloring theorem. [20] Let $p_1, p_2 : 2^V \rightarrow \mathbb{Z} \cup \{\infty\}$ be two skew-supermodular functions and $k \in \mathbb{Z}_+$. Then V can be partitioned into classes $L_1, L_2, \ldots, L_k$ such that

$$p_i(U) \leq |\{j \in [k] : L_j \cap U \neq \emptyset\}| \tag{45}$$

for each $i = 1, 2$ and each $U \subseteq V$ if and only if

$$p_i(U) \leq min\{k, |U|\} \tag{46}$$

However this theorem is not a direct generalization of supermodular coloring as intersecting supermodular functions are not necessarily skew-supermodular. For example, the above defined function g_i is not skew-supermodular.

5.2.2 König's theorem from skew-supermodular coloring

In this section we prove König's edge-coloring theorem from Theorem 5.2.6. Let $G := (V, E)$ be a bipartite graph such that V_1, V_2 are the two partitions. Let $X \subseteq E$ and $\mathcal{I}_i(X) := \{v \in V_i \mid \delta(v) \cap X \neq \emptyset\}$, i.e., $\mathcal{I}_i(X)$ is the set of vertices in V_i that are incident on X. Let $d(v)$ be the degree of vertex v in G. Let $S \subseteq V_i$ and $\eth(S) := v \in S \mid d(v) \leq d(u)\ \forall u \in S$, i.e., the vertex in S with the minimum degree. If there are multiple vertices with the same minimum degree then we arbitrarily choose one vertex. First let us show that the function $f_i(X) = |X| - \sum_{v \in \{\mathcal{I}_i(X) \setminus \{\eth(\mathcal{I}_i(X))\}\}} d(v)$ is skew-supermodular.

Lemma 5.2.7. $f_i(X) = |X| - \sum_{v \in \{\mathcal{I}_i(X) \setminus \{\eth(\mathcal{I}_i(X))\}\}} d(v)$ is skew-supermodular for $i = 1, 2$.

Proof. Let us show that f_1 is skew-supermodular. Proof for skew-supermodularity of f_2 follows similarly. Let $X, Y \subseteq E$, if $X \cap Y = \emptyset$, Condition (42) of skew-supermodular functions hold trivially. So, assume $X \cap Y \neq \emptyset$ and we will show that in this case condition (41) of skew-supermodular functions is true. Let $g(X) = |X|$ and $h(X) = \sum_{v \in \{\mathcal{I}_1(X) \setminus \{\eth(\mathcal{I}_1(X))\}\}} d(v)$. Then, $f_1(X) = g(X) - h(X)$. Clearly $g(X)$ is modular, i.e., $g(X) + g(Y) = g(X \cup Y) + g(X \cap Y)$. So we need to show $h(X) + h(Y) \geq h(X \cup Y) + h(X \cap Y)$ if $X \cap Y \neq \emptyset$. W.l.o.g. assume $d(\eth(\mathcal{I}_1(X))) \geq d(\eth(\mathcal{I}_1(Y)))$.

Now,

$$\begin{align*}
& h(X \cup Y) + h(X \cap Y) \\
\leq\ & \sum_{v \in \mathcal{I}_1(X \cup Y)} d(v) + \sum_{v \in \mathcal{I}_1(X \cap Y)} d(v) - d(\eth(\mathcal{I}_1(X \cup Y))) - d(\eth(\mathcal{I}_1(X \cap Y))) \\
\leq\ & \sum_{v \in \mathcal{I}_1(X)} d(v) + \sum_{v \in \mathcal{I}_1(Y)} d(v) - d(\eth(\mathcal{I}_1(X \cup Y))) - d(\eth(\mathcal{I}_1(X \cap Y))) \tag{47} \\
\leq\ & \sum_{v \in \mathcal{I}_1(X)} d(v) + \sum_{v \in \mathcal{I}_1(Y)} d(v) - d(\eth(\mathcal{I}_1(X))) - d(\eth(\mathcal{I}_1(Y))) \tag{48} \\
\leq\ & h(X) + h(Y) \tag{49}
\end{align*}$$

This is what we needed to prove. Here inequality (47) follows from the submodularity property of cut function. Inequality (48) follows from the fact that $d(\eth(\mathcal{I}_1(X \cap Y))) \geq d(\eth(\mathcal{I}_1(X)))$ and $d(\eth(\mathcal{I}_1(X \cup Y))) = min\{d(\eth(\mathcal{I}_1(X))), d(\eth(\mathcal{I}_1(Y)))\} = d(\eth(\mathcal{I}_1(Y)))$.

□

Now we are ready to prove the edge-coloring theorem.

Proof of Theorem 5.2.1. Let us we define $p_i(X) := |X| - \sum_{v \in \{\mathcal{I}_i(X) \setminus \{\eth(\mathcal{I}_i(X))\}\}} d(v)$ for $i = 1, 2$. From Lemma 5.2.7, p_i is skew-submodular. Also from the definition it is clear that $p_i(U) = |U| - h(U) \leq |U|$ as $h(U) \geq 0$. Now let $X_v = \{e \mid e \in \delta(v) \cap X\}$, for any vertex $v \in \mathcal{I}(X)$. Note that

$$|X_v| \leq d(v) \leq \Delta\,. \tag{50}$$

Then

$$\begin{aligned}
p_i(X) &= |X| - \sum_{v \in \{\mathcal{I}_i(X) \setminus \{\eth(\mathcal{I}_i(X))\}\}} d(v) \\
&= |X_{\eth(\mathcal{I}_i(X))}| - \sum_{v \in \{\mathcal{I}_i(X) \setminus \{\eth(\mathcal{I}_i(X))\}\}} (d(v) - |X_v|) \\
&\leq |X_{\eth(\mathcal{I}_i(X))}| \\
&\leq \Delta \quad \big[\text{From (50)}\big] .
\end{aligned}$$

Thus $p_i(U) \leq min\{\Delta, |U|\}$ for any $U \subseteq E$. Therefore, using Theorem 5.2.6, we get a proper edge coloring with Δ colors. □

It will be interesting if one can extend Theorem 5.2.6 to get a better bound for weighted bipartite edge coloring.

5.3 Edge-coloring Weighted Bipartite Graphs

In this section we present our main result and prove Theorem 5.0.2.

Theorem 5.3.1. [Restatement of Theorem 5.0.2] There is a polynomial time algorithm for the WEIGHTED BIPARTITE EDGE COLORING problem which returns a proper weighted coloring using at most $\lceil 2.2223m \rceil$ colors, where m denotes the maximum over all the vertices of the number of unit-sized bins needed to pack the weights of incident edges.

The Algorithm:

Our complete algorithm for edge-coloring weighted bipartite graphs is given in Figure 12. In the algorithm, we set a threshold $\gamma = \frac{1}{10}$ and consider the subgraph induced by edges with weights more than γ and apply a combination of König's Theorem and a greedy algorithm with $\lceil tm \rceil$ colors where $t = 2.2223 > 20/9$. The remaining edges of weights at most γ are then added greedily.

1. $F \leftarrow \emptyset$, $t \leftarrow 2.2223$.
2. Include edges with weight $> \gamma = \frac{1}{10}$ in F in nonincreasing order of weight maintaining the property that $deg_F(v) \leq \lceil tm \rceil$ for all $v \in V$.
3. Decompose F into $r = \lceil tm \rceil$ matchings $M_1, \ldots, M_r$ and color them using colors $1, \ldots, r$. Let $F_i \leftarrow M_i$ for each $1 \leq i \leq r$.
4. Add remaining edges with weight $> \gamma$ in nonincreasing order of weight to any of the F_i's maintaining that weighted degree of each color at each vertex is at most one, i.e., $\sum_{e \in \delta(v) \cap F_i} w_e \leq 1$ for each $v \in V$ and $1 \leq i \leq r$.
5. Add remaining edges with weight $\leq \gamma$ in nonincreasing order of weight to any of the F_i's maintaining that weighted degree of each color at each vertex is at most one, i.e., $\sum_{e \in \delta(v) \cap F_i} w_e \leq 1$ for each $v \in V$ and $1 \leq i \leq r$.

Figure 12: Algorithm for Edge Coloring Weighted Bipartite Graphs

Analysis:

Now we prove Theorem 5.0.2. Though the algorithm is purely combinatorial, the analysis uses configuration LP and other techniques from bin packing to prove the correctness of the algorithm.

The following lemma from Correa and Goemans [47] (which was also implicit in [64]) ensures that if the algorithm succeeds in coloring all edges of weight at least γ, the greedy coloring will be able to color the remaining edges of weight at most γ.

Lemma 5.3.2. [47, 64] Consider a bipartite weighted graph $G = (V, E)$ with a coloring of all edges of weight $> \gamma$ using at least $\frac{2m}{1-\gamma}$ colors for some $\gamma > 0$. Then the greedy coloring will succeed in coloring the edges with weight at most γ without any additional colors.

In our setting, we have $\gamma = \frac{1}{10}$ and the number of colors is at least $\frac{20}{9}m = \frac{2m}{1-\frac{1}{10}}$ and thus Lemma 5.3.2 applies. Hence, it suffices to show that the algorithm is able to color all edges with weights $> \frac{1}{10}$ using $\lceil tm \rceil$ colors, as the remaining smaller edges can be colored greedily. Thus, without loss of generality, we assume that the graph has no edges of weight $\leq \frac{1}{10}$ and prove the following lemma.

Lemma 5.3.3. If all edges have weight more than $\frac{1}{10}$ and $t = 2.2223$ ($> 20/9$) then the algorithm in Figure 12 returns a coloring of all edges using $\lceil tm \rceil$ colors such that the weighted degree of each color at each vertex is at most one, i.e., $\sum_{e \in \delta(v) \cap F_i} w_e \leq 1$.

Proof. Suppose for the sake of contradiction, the algorithm is not able to color all edges. Let $e := (u, v)$ be the first edge that cannot be colored by any color in Step (3) or Step (4) of the algorithm. Let the weight of edge e, w_e, be α. Moreover, when e is considered in Step (2), degree of either u or v is already $\lceil tm \rceil$, else we would have included e in F. Without loss of generality let that vertex be u, i.e., $deg_F(u) = \lceil tm \rceil$.

For each color $1 \leq i \leq \lceil tm \rceil$, we must have that $\sum_{f \in \delta(v) \cap F_i} w_f > 1 - \alpha$ or $\sum_{f \in \delta(u) \cap F_i} w_f > 1 - \alpha$, else we can color e in Step (4). Let $H_v = \{i | \sum_{f \in \delta(v) \cap F_i} w_f > 1 - \alpha\}$, $\beta m = |H_v|$. Now for each color $i \notin H_v$, we have $\sum_{f \in \delta(u) \cap F_i} w_f > 1 - \alpha$. Moreover, $deg_F(u) = \lceil tm \rceil$ and each of these edges weighs at least $w_e = \alpha$. Hence, for each color $1 \leq i \leq \lceil tm \rceil$, there is an edge incident at u colored with color i with weight at least α. Let us call a color i *tight* at u if $\sum_{f \in \delta(u) \cap F_i} w_f > (1 - \alpha)$ and a color i *open* at u if $\sum_{f \in \delta(u) \cap F_i} w_f \in [\alpha, 1 - \alpha]$. Let τ be the number of tight colors at u and θ be the number of open colors at u. Thus we have,

$$\tau \geq (t - \beta)m \tag{51}$$

$$\theta = (tm - \tau) \leq \beta m\,. \tag{52}$$

Now consider the following lemma.

Lemma 5.3.4. [77] Each edge of weight at least 1/t is in F.

It gives the following inequality.

$$\alpha < 1/t\,. \tag{53}$$

Now consider all edges incident on v. We get,

$$m > \beta m(1 - \alpha)$$

$$\Rightarrow \qquad 1 > \beta(1-\alpha)\,. \tag{54}$$

Similarly considering all edges incident on u. We get,

$$m > (tm - \beta)(1-\alpha) + (\beta m)\alpha$$

$$\Rightarrow \qquad 1 > t(1-\alpha) + \beta(2\alpha - 1)\,. \tag{55}$$

In fact we can strengthen inequality (53) to the following:

$$\alpha \leq 1/3\,. \tag{56}$$

This follows from the fact that a unit-sized bin can contain at most two items with weight $> 1/3$. As all edges incident to a vertex can be packed into m unit-sized bins, there can be at most $2m$ edges incident to a vertex with weight $> 1/3$. Since $t > 2$, we get that all edges with weight more than $\frac{1}{3}$ must have been included in F in Step (2). Thus $\alpha \leq 1/3$. Moreover we also get from (54):

$$\beta \quad < \quad 1/(1-\alpha) \quad \leq \quad 3/2\,. \tag{57}$$

Now there are two cases:

Case A: $\alpha \leq 1/4$. Consider the RHS of (55): $t(1-\alpha) + \beta(2\alpha - 1)$. If we show that the expression is always greater than 1 for $\alpha \leq 1/4$, then we arrive at a contradiction. Now,

$$\begin{aligned} t(1-\alpha) + \beta(2\alpha-1) - 1 \quad &> \quad t(1-\alpha) - \frac{(1-2\alpha)}{(1-\alpha)} - 1, \qquad \text{[From (54)]} \\ &\geq \quad \frac{20(1-\alpha)^2 - 9(1-2\alpha) - 9(1-\alpha)}{9(1-\alpha)} \qquad \text{[Since, } t > 20/9\text{]} \\ &\geq \quad \frac{(20\alpha^2 - 13\alpha + 2)}{9(1-\alpha)} \\ &\geq \quad \frac{(4\alpha-1)(5\alpha-2)}{9(1-\alpha)} \\ &\geq \quad 0, \text{as } \alpha \leq 1/4\,. \end{aligned} \tag{58}$$

Thus $t(1-\alpha)+\beta(2\alpha-1)>1$, which contradicts (55).

Case B: $1/4<\alpha\leq 1/3$. In this case, we will show in Lemma 5.3.5 that if $\beta\leq 13/9$, then all edges incident at u can not be packed into m bins. On the other hand, in Lemma 5.3.12 we show that if $\beta>13/9$, then all edges incident at v can not be packed into m bins. These two facts together give us the desired contradiction.

Lemma 5.3.5. If $\beta\leq 13/9$, then edges incident at u can not be packed into m bins.

Proof. To give a lower bound on the number of bins required, we will consider a relaxation to the bin packing problem for edges incident at vertex u and show that the optimal value of the relaxation, and thus the optimal number of bins required, is greater than m. The lower bound will be exhibited by constructing a feasible dual solution to the relaxation to the bin packing problem.

Since $deg_F(u)=\lceil tm\rceil$ when edge e was considered in Step (2) of the algorithm and not included in F, we have that all edges incident at u in F have weight at least the weight of e. Moreover, edges are considered in the decreasing order of weight in Step (4), the weight of all edges incident at u when e is considered in Step (4) is $\geq w_e$. We restrict our attention to these edges incident at u with weight $\geq\alpha$ and show that they cannot be packed in m unit-sized bins. Let us divide these edges incident at u into three size classes.

- Large $L:=\{f\in\delta(u):w_f\in(1/2,1]\}$.
- Medium $M:=\{f\in\delta(u):w_f\in(1/3,1/2]\}$.
- Small $S:=\{f\in\delta(u):w_f\in[\alpha,1/3]\}$.

First we have the following observation.

Observation 5.3.6. *In any bin packing solution, in any bin there can be at most one item from L, two items from $L\cup M$ and three items from $L\cup M\cup S$.*

Now let us prove the following two claims.

Claim 5.3.7. Edges in $L \cup M$ are included in Step (2) of the algorithm and thus are a subset of F.

Proof. If we are unable to add an edge f in Step (2), it means one of its endpoints has $\lceil tm \rceil > 2m$ edges with weight $\geq w_f$. Since all edges incident at any vertex $v \in V$ can be packed into m bins, there are at most $2m$ edges incident at it with weight more than $\frac{1}{3}$. Thus all edges of weight more than $\frac{1}{3}$, i.e., all edges in $L \cup M$ must be included in F in Step (2) of the algorithm. □

As a corollary we get,

Claim 5.3.8. For any color i, there is at most one edge in $L \cup M$ with color i.

Proof. All edges in $L \cup M$ must be included in F in Step (2) of the algorithm. In Step (3) of the algorithm, we include at most one edge of F incident at any vertex in each F_i. Thus each color class obtains at most one edge incident at each vertex from F and therefore, from $L \cup M$. □

Using the observation and the above claim, we itemize the configuration of each of the tight colors depending on the size of edges with that color. Note that tight colors must have weight $> 1 - \alpha \geq 1 - 1/3 = 2/3$.

1. Case 1. *The tight color has a single edge f.* Then we have that $w_f > \frac{2}{3}$ and only possibility is $i)(L)$; Here by (L), we denote that the bin contains only one item and that item is an item from the set L.

2. Case 2. *The tight color contains exactly two edges.* Here (S, S) is not tight as the total weight of edges in such a bin is $\leq 2/3$. So, the bin can contain at most one item from S. On the other hand, the bin can contain at most one item from $L \cup M$ from Claim 5.3.8. Thus the possible size types of these edges are

$ii)(L,S); iii)(M,S)$; As above, by (L,S) we denote that the bin contains only two items: exactly one item from set L and exactly one item from S.

3. Case 3. *The tight color contains three edges.* The bin can contain at most one item from $L \cup M$ from Claim 5.3.8. However if it contains one L item, sum of weights of three items exceeds one. Thus the only possible size types of these edges are $iv)(M,S,S); v)(S,S,S)$.

Now consider the following LP: $LP_{bin}(u)$:

$$\begin{aligned} & min \sum_{i=1}^{5} y_i \\ x_1 + x_2 + x_3 + x_4 + x_5 &\geq \tau && (59) \\ y_1 + y_2 &\geq x_1 + x_2 + z_1 && (60) \\ y_1 + 2y_3 + y_4 &\geq x_3 + x_4 + z_2 && (61) \\ y_2 + y_3 + 2y_4 + 3y_5 &\geq x_2 + x_3 + 2x_4 + 3x_5 + z_3 && (62) \\ z_1 + z_2 + z_3 &\geq \theta && (63) \\ x_j, y_k, z_l &\geq 0 \quad \forall j \in [5], k \in [5], l \in [3]. && (64) \end{aligned}$$

Lemma 5.3.9. The optimal number of unit-sized bins needed to pack all edges incident at u is at least the optimum value of $LP_{bin}(u)$.

Proof. Given a feasible packing of edges incident at u in at most m unit-sized bins, we construct a feasible solution $(\bar{x}, \bar{y}, \bar{z})$ to the linear programming relaxation whose objective is at most the number of unit-sized bins needed in the packing. In the feasible solution $(\bar{x}, \bar{y}, \bar{z})$, the variables $\bar{x}$ and $\bar{z}$ are constructed using the coloring given by the algorithm. The variables $\bar{y}$ are constructed using the optimal bin packing.

We first define the variables $\bar{x}$. Let $\bar{x}_1, \bar{x}_2, \bar{x}_3, \bar{x}_4, \bar{x}_5$ be the number of tight colors at u of type $(L), (L,S), (M,S)$, $(M,S,S), (S,S,S)$, respectively. Since the coloring of the edges incident at u is one of the five types described above and there are at least τ

tight colors, we have that $\sum_{i=1}^{5} \bar{x}_i \geq \tau$ and thus the solution satisfies constraint (59). Now we define the variables $\bar{z}_1, \bar{z}_2, \bar{z}_3$ to be the number of items in open colors from L, M and S respectively. There are θ open colors. Each open color contains at least one item $L \cup M \cup S$. Thus, $\bar{z}_1 + \bar{z}_2 + \bar{z}_3 \geq \theta$ and thus the solution satisfies constraint (63).

To construct the solution $\bar{y}$, we will group the bins in the optimal bin packing solutions depending on the subset of items present in them into five classes and the number of bins in each class will define the variables $\bar{y}$. The constraints (60)-(62) will correspond to making sure that the optimal bin packing solution has appropriate number of items of each size type.

We now have the following claim where we characterize the possible bin configurations.

Claim 5.3.10. Consider any feasible bin-packing of edges incident at u restricted to edges in $L \cup M \cup S$. Then each bin must contain items which correspond to a subset of one of the following 5 configurations or subsets of these configurations.

$C_1 : (L, M)$ $\quad$ $C_2 : (L, S)$ $\quad$ $C_3 : (M, M, S)$

$C_4 : (M, S, S)$ $\quad$ $C_5 : (S, S, S)$

Proof. Observe that in any bin there can be at most one item from L, two items from $L \cup M$ and three items from $L \cup M \cup S$. Now let us consider two cases.

Case 1. The bin contains an item from L. In this case, the bin can not contain three items as the sum of their weights exceeds one. So, it can contain at most one item from L and one item from $M \cup S$. Thus C_1 and C_2 cover such two cases.

Case 2. The bin does not contain any item from L. In this case, the bin can contain three items from $M \cup S$ and at most two of these items can be from M. Thus C_3, C_4 and C_5 cover such possibilities. □

We map each configuration in the optimal bin packing solution to one of types C_i

where the configuration is either C_i or its subset. Let $\bar{y}_i$ denote the number of bins mapped to type C_i for each $1 \leq i \leq 5$. We now count the number of items of each type to show feasibility of the constraints of the linear program.

> Constraint (60). Items of type L equal $\bar{x}_1 + \bar{x}_2 + \bar{z}_1$ and can only be contained in configuration C_1 and C_2. Thus we have $\bar{y}_1 + \bar{y}_2 \geq \bar{x}_1 + \bar{x}_2 + \bar{z}_1$ satisfying constraint (60).
>
> Constraint (61). Items of type M equal $\bar{x}_3 + \bar{x}_4 + \bar{z}_2$ and are contained once in configurations C_1, C_4 and twice in configuration C_3. Thus we have $\bar{y}_1 + 2\bar{y}_3 + \bar{y}_4 \geq \bar{x}_3 + \bar{x}_4 + \bar{z}_2$ satisfying constraint (61).
>
> Constraint (62). Items of type S equal $\bar{x}_2 + \bar{x}_3 + 2\bar{x}_4 + 3\bar{x}_5 + \bar{z}_3$ and occur once in configurations C_2, C_3, twice in configuration C_4 and thrice in C_5. Thus, we have $\bar{y}_2 + \bar{y}_3 + 2\bar{y}_4 + 3\bar{y}_5 \geq \bar{x}_2 + \bar{x}_3 + 2\bar{x}_4 + 3\bar{x}_5 + \bar{z}_3$ showing feasibility of constraint (62).

This implies that $(\bar{x}, \bar{y}, \bar{z})$ is a feasible solution to LP_{bin} and its objective equals the number of bins needed to pack the edges incident at u in unit-sized bins. Thus we have the lemma. □

We now show a contradiction by showing the optimal value of the $LP_{bin}(u)$ is more than m.

Lemma 5.3.11. The optimal solution to $LP_{bin}(u)$ is strictly more than m.

Proof. We prove this by considering the dual linear program of the $LP_{bin}(u)$. Since every feasible solution to the dual LP gives a lower bound on the objective of the primal $LP_{bin}(u)$, it is enough to exhibit a feasible dual solution of objective strictly more than m to prove the lemma. Now the dual of the LP_{bin} is following:

A feasible dual solution is: $v_1 = \frac{2}{3}, v_2 = \frac{2}{3}, v_3 = \frac{1}{3}, v_4 = \frac{1}{3}, v_5 = \frac{1}{3}$.

$$
\begin{array}{ll}
max \quad \tau \cdot v_1 + \theta \cdot v_5 & \\
\text{Subject to:} & \\
v_1 - v_2 \leq 0, & v_1 - v_2 - v_4 \leq 0, \\
v_1 - v_3 - v_4 \leq 0, & v_1 - v_3 - 2v_4 \leq 0, \\
v_1 - 3v_4 \leq 0, & v_2 + v_3 \leq 1, \\
v_2 + v_4 \leq 1, & 2v_3 + v_4 \leq 1, \\
v_3 + 2v_4 \leq 1, & 3v_4 \leq 1, \\
v_5 - v_2 \leq 0, & v_5 - v_3 \leq 0, \\
v_5 - v_4 \leq 0, & v_i \geq 0 \quad \forall i \in [4].
\end{array}
$$

Thus dual optima $\geq \frac{2\tau}{3} + \frac{\theta}{3}$ and we need at least these many colors to color items in τ tight colors and θ open colors. Using the fact that $\theta = tm - \tau, \tau \geq (t - \beta)m$ and $t > \frac{20}{9}m, \beta \leq \frac{13}{9}$, we obtain that the number of bins required to pack all items incident on u is:

$$
\begin{aligned}
&\geq \tau \cdot v_1 + \theta \cdot v_4 \\
&\geq \frac{2}{3}\tau + \frac{1}{3}(tm - \tau) \\
&= \frac{1}{3}\tau + \frac{1}{3}(tm) \\
&\geq \frac{1}{3}(t - \beta)m + \frac{1}{3}(tm) \\
&\geq \frac{2t}{3}m - \frac{\beta}{3}m \\
&> m(\frac{2}{3} \cdot \frac{20}{9} - \frac{1}{3} \cdot \frac{13}{9}) \\
&= m\,.
\end{aligned}
$$

Thus the number of bins required to pack all items incident on u is strictly greater than m. This is a contradiction. □

This concludes the proof of Lemma 5.3.5. □

Lemma 5.3.12. If $\beta > 13/9$, then edges incident at v can not be packed into m bins.

Proof. Similar to the previous lemma, to give a lower bound on the number of bins required, we will consider a relaxation to the bin packing problem for edges incident at vertex v and show that the optimal value of the relaxation, and thus the optimal number of bins required, is greater than m. Again, the lower bound will be exhibited by constructing a feasible dual solution to the relaxation to the bin packing problem.

As $\beta(1-\alpha) < 1$, we get,

$$\alpha > 1 - 1/\beta \geq 4/13 > 0.3. \tag{65}$$

Let us call a color i *tight* at v, if $\sum_{f \in \delta(v) \cap F_i} w_f > (1-\alpha)$. Now consider any tight color $\mathcal{B}$ at v. At most one edge f in $\mathcal{B}$ was colored in Step (3) of the algorithm and remaining edges (if any) in $\mathcal{B}$ were colored in Step (4) of the algorithm. Now, w_f can be smaller than w_e as it might be the case that when e was considered in Step (2) then already degree of other endpoint u was $\lceil tm \rceil$. However, edges are considered in the nonincreasing order of weight in Step (4), thus the weight of all edges incident at v when e is considered in Step (4) is also $\geq w_e$. Thus, all the remaining edges (if any) in $\mathcal{B}$ that were colored in Step (4) of the algorithm have weights $> \alpha$.

We restrict our attention to the edges at tight colors at v and show that if $\beta > \frac{13}{9}$ they cannot be packed in m unit-sized bins. Let us divide these edges incident at u into three size classes.

- Large $L := \{f \in \delta(v) : w_f \in (1/2, 1]\}$.
- Medium $M := \{f \in \delta(v) : w_f \in (1/3, 1/2]\}$.
- Small $S := \{f \in \delta(v) : w_f \in [\alpha, 1/3]\}$.
- Tiny $T := \{f \in \delta(v) : w_f \in (1/10, \alpha)\}$.

First we have the following observation.

Observation 5.3.13. *In any bin packing solution, in any bin there can be at most one item from L, two items from $L \cup M$, three items from $L \cup M \cup S$ and nine items from $L \cup M \cup S \cup T$.*

Now let us prove the following claim.

Claim 5.3.14. For any tight color i at v, all edges added in Step (4) of the algorithm are in S. As a corollary, there is at most one edge incident on v with color i that is in $L \cup M \cup T$ and it can only be added in Step (2) of the algorithm.

Proof. From Claim 5.3.7, it follows that all edges in $L \cup M$ must be included in F in Step (2) of the algorithm. On the other hand, as all edges colored in Step (4) have weights $\geq \alpha$, they can not be in T. Hence, only edges in S are colored in Step (4). Edges in $L \cup M \cup T$ are colored in Step (3). In Step (3) of the algorithm, we include at most one edge of F incident at any vertex in each F_i. Thus each color class obtains at most one edge incident at each vertex from F and therefore, from $L \cup M \cup T$. □

Using the observation and the claim, we itemize the configuration of each of the tight colors depending on the size of edges with that color. Note that in this case tight colors have weights $> 1 - \alpha \geq 1 - 1/3 = 2/3$.

1. If the tight color has a single edge f. Then we have that $w_f > 2/3$ and only possibility is $i)(L)$; Here, (L) denotes that the bin contains only one item and that item is an item from the set L.

2. If the tight color contains exactly two edges. From Claim 5.3.14, the bin can contain at most one item from $L \cup M \cup T$. On the other hand, (S, S) or (T, S) has weight $\leq 2/3$. So the bin can contain at most one item from S and one item from $L \cup M$. Thus the possible size types of these edges are $ii)(L, S); iii)(M, S)$; As above, by (L, S) we denote that the bin contains only two items: exactly one item from set L and exactly one item from S.

3. If the tight color contains three edges. From Claim 5.3.14, the bin can contain at most one item from $L \cup M \cup T$. However if the bin contains an item from L,

the sum of weights of an item from L and two items from S exceeds one. Thus the possible size types of these edges are $iv)(M,S,S); v)(S,S,S); vi)(T,S,S)$.

Now consider the following configuration LP based on the items at v: $LP_{bin}(v)$:

$$min \sum_{i=1}^{19} y_i$$

$$\begin{aligned}
x_1 + x_2 + x_3 + x_4 + x_5 + x_6 &\geq \beta m && (66)\\
y_{14} &\geq x_1 && (67)\\
y_8 + y_9 + y_{10} + y_{15} &\geq x_2 && (68)\\
y_3 + y_7 + y_9 + y_{11} + 2y_{12} + y_{16} &\geq x_3 && (69)\\
y_2 + 2y_4 + y_6 + y_{10} + y_{11} + 2y_{13} + y_{17} &\geq x_4 && (70)\\
3y_1 + 2y_2 + 2y_3 + y_4 + 2y_5 + y_6 + y_7 + y_8 + y_{18} &\geq x_2 + x_3 + 2x_4 + 3x_5 + 2x_6 && (71)\\
(3y_5 + 3y_6 + 3y_7 + y_8 + y_9 + y_{10} + 3y_{11} + 3y_{12} & && \\
+3y_{13} + 3y_{14} + 4y_{15} + 6y_{16} + 6y_{17} + 6y_{18} + 9y_{19}) &\geq x_6 && (72)\\
y_j, x_k &\geq 0 \quad \forall j \in [19], k \in [6]. && (73)
\end{aligned}$$

Lemma 5.3.15. The optimal number of unit-sized bins needed to pack all edges incident at u is at least the optimum value of $LP_{bin}(v)$.

Proof. Given a feasible packing of edges incident at v in at most m unit-sized bins, we construct a feasible solution $(\bar{x}, \bar{y})$ to the linear programming relaxation whose objective is at most the number of unit-sized bins needed in the packing. In the feasible solution $(\bar{x}, \bar{y})$, the variables $\bar{x}$ are constructed using the coloring given by the algorithm. The variables $\bar{y}$ are constructed using the optimal bin packing.

We first define the variables $\bar{x}$. Let $\bar{x}_1, \bar{x}_2, \bar{x}_3, \bar{x}_4, \bar{x}_5, \bar{x}_6$ be the number of tight colors at v of type (L), (L,S), (M,S), (M,S,S), (S,S,S), (T,S,S), respectively. Since the coloring of the edges incident at v is one of the six types described above and there are at least βm tight colors at v, we have that $\sum_{i=1}^{6} \bar{x}_i \geq \beta m$ and thus the solution satisfies constraint (78).

To construct the solution $\bar{y}$, we will group the bins in the optimal bin packing solutions depending on the subset of items present in them into nineteen classes and the number of bins in each class will define the variables $\bar{y}$. The constraints (79)-(84) will correspond to making sure that the optimal bin packing solution has appropriate number of items of each size type.

To define the 19 different classes of bin types in the optimal solution, we need to further classify items according to size. Let $L_1, L_2 \subseteq L$ be the set of large edges that appear in the configurations of the type (L) and (L, S), respectively, in the tight colors.

As for any item $l_1 \in L_1$, $w_{l_1} + \alpha > 1$. We get,

$$w_{l_1} > 1 - \alpha \geq 1 - 1/3 = 2/3. \tag{74}$$

Let M_1, M_2 be the set of medium edges that appear in the tight colors of type (M, S) and (M, S, S), respectively.

We now have the following claim where we characterize the possible bin configurations. We show that each bin contains items which correspond to one of 19 possible configurations or their subsets.

Claim 5.3.16. Consider any feasible bin-packing of edges incident at v restricted to edges in $L \cup M \cup S \cup T$. Then each bin must contain items which correspond to a subset of one of the following 19 configurations.

$C_1 : (S, S, S)$
$C_2 : (M_2, S, S)$
$C_3 : (M_1, S, S)$
$C_4 : (M_2, M_2, S)$
$C_5 : (S, S, T, T, T)$
$C_6 : (M_2, S, T, T, T)$
$C_7 : (M_1, S, T, T, T)$
$C_8 : (L_2, S, T)$
$C_9 : (L_2, M_1, T)$
$C_{10} : (L_2, M_2, T)$
$C_{11} : (M_1, M_2, T, T, T)$
$C_{12} : (M_1, M_1, T, T, T)$
$C_{13} : (M_2, M_2, T, T, T)$
$C_{14} : (L_1, T, T, T)$
$C_{15} : (L_2, T, T, T, T)$
$C_{16} : (M_1, T, T, T, T, T, T)$
$C_{17} : (M_2, T, T, T, T, T, T)$
$C_{18} : (S, T, T, T, T, T, T)$
$C_{19} : (T, T, T, T, T, T, T, T, T)$

Proof. Observe that since items in L have weight more than $\frac{1}{2}$, items in M have weight more than $\frac{1}{3}$, items in S have weight more than $\frac{1}{4}$ and items in T have weight more than $\frac{1}{10}$, there can be at most one item from L, two items from $L \cup M$, at most three items in total from $L \cup M \cup S$ and at most nine items from $L \cup M \cup S \cup T$ in any feasible packing.

1. Bins with three items from $D := L \cup M \cup S$. As the sum of weights of three elements from D is more than $3\alpha > 0.9$, elements from T can not appear in these bins as $3\alpha + w_f > 1$ for any $f \in T$. Moreover, configurations which contain at least one item of L cannot have three items from D without the weight exceeding one. Thus, the packing can contain only items from M and S. When the bin contains only S items, it corresponds to configuration C_1. Packings which contain one item from $M_1 \cup M_2$ and two items from S are exactly the configurations C_2, C_3.

   Now let us consider the case when we have two items from $M_1 \cup M_2$. First observe that for each $h \in M_1$, there exists a $s \in S$ such that (h, s) are the only edges colored with a tight color. Thus we have that $w_h + w_s > 1 - \alpha$. But then for any other $h' \in M_1 \cup M_2$ and $s' \in S$, we have that

   $$w_h + w_{h'} + w_{s'} \geq w_h + w_s + \alpha > 1\,, \tag{75}$$

   where the inequality follows since $w_{h'} \geq w_s$ and $w_{s'} \geq \alpha$. This implies that configurations of type (M_1, M_1, S) or (M_1, M_2, S) are not feasible. Hence, the only possible remaining configuration is C_4.

2. Bins with two items from D. Here we consider maximal configurations which are not subsets of configurations which contain three items from D. When the configuration contains two items $s_1, s_2 \in S$, we have that $w_{s_1} + w_{s_2} > 0.6$ and thus the only maximal configuration is C_5. Configurations C_6, C_7 cover the case when the configuration contains one item from S and one item from M. Let $l \in L_1$. Since l appears alone in a tight color, we have that $w_l + \alpha > 1$. Since every item in $M \cup S$ has weight at least α, there is no valid configuration with

two items from D such that one of them is in L_1. If the configuration contains $l_2 \in L_2$ and $g \in M_1 \cup M_2 \cup S$, it can at most contain one element $t \in T$ as $w_{l_2} + w_g + w_t > 0.5 + 0.3 + 0.1 > 0.9$. Thus configuration C_8 covers the case when there is one item from S and one item from L. Now we are left with cases when there are no S items in the bin. If there is one L item and one M item, C_9, C_{10} cover such possibilities.

Similarly if the configuration contains two items from M, it can contain at most 3 elements from T. Configurations C_{11}, C_{12}, C_{13} cover all such the possibilities.

3. Bins with one item from D. Here we consider configurations which are not subsets of configurations which contain at least two items from D. Note that as for any item $l_1 \in L_1$, from inequality (74), $w_{l_1} > 2/3$. Thus (L_1, T, T, T) is the maximal configuration containing one L_1 item. C_{14} is the corresponding configuration. The other four possible configurations are $C_{15}, C_{16}, C_{17}, C_{18}$ where the bins contain one item from L_2, M_1, M_2, S respectively. In these cases the number of T items are upper bounded by $4, 6, 6, 6$ respectively from the lower bound of size of items in the corresponding classes in D.

4. Bins with no item from D. Only possible maximal configuration is C_{19}.

□

We map each configuration in the optimal bin packing solution to one of types C_i where the configuration is either C_i or its subset. Let $\bar{y}_i$ denote the number of bins mapped to type C_i for each $1 \leq i \leq 19$. We now count the number of items of each type to show feasibility of the constraints of the linear program.

Constraint (79). Items of type L_1 equal $\bar{x}_1$ and can only be contained in configuration C_{14}. Thus we have $\bar{y}_{14} \geq \bar{x}_1$.

Constraint (80). Similarly, items of type L_2 equal $\bar{x}_2$. They are contained in configurations C_8, C_9, C_{10} and C_{15}. Thus we have $\bar{y}_8 + \bar{y}_9 + \bar{y}_{10} + \bar{y}_{15} \geq \bar{x}_2$.

Constraint (81). Items of type M_1 equal $\bar{x}_3$ and are contained once in configurations $C_3, C_7, C_9, C_{11}, C_{16}$ and twice in configuration C_{12}. Thus we have $\bar{y}_3 + \bar{y}_7 + \bar{y}_9 + \bar{y}_{11} + 2\bar{y}_{12} + \bar{y}_{16} \geq \bar{x}_3$.

Constraint (82). Items of type M_2 equal $\bar{x}_4$ and are contained once in configurations $C_2, C_6, C_{10}, C_{11}, C_{17}$ and twice in configurations C_4, C_{13}. Thus we have $\bar{y}_2 + 2\bar{y}_4 + \bar{y}_6 + \bar{y}_{10} + \bar{y}_{11} + 2\bar{y}_{13} + \bar{y}_{17} \geq \bar{x}_4$ satisfying constraint (82).

Constraint (83). Items of type S equal $\bar{x}_2 + \bar{x}_3 + 2\bar{x}_4 + 3\bar{x}_5 + 2\bar{x}_6$ and occur once in configurations $C_4, C_6, C_7, C_8, C_{18}$, twice in configurations C_2, C_3, C_5 and thrice in C_1. Thus, we have $3\bar{y}_1 + 2\bar{y}_2 + 2\bar{y}_3 + \bar{y}_4 + 2\bar{y}_5 + \bar{y}_6 + \bar{y}_7 + \bar{y}_8 + \bar{y}_{18} \geq \bar{x}_2 + \bar{x}_3 + 2\bar{x}_4 + 3\bar{x}_5 + 2\bar{x}_6$.

Constraint (84). Items of type T equal $\bar{x}_6$ and occur once in configurations C_8, C_9, C_{10}, thrice in configurations $C_5, C_6, C_7, C_{11}, C_{12}, C_{13}, C_{14}$, four times in configuration C_{15}, six times in configurations C_{16}, C_{17}, C_{18} and nine times in configuration C_{19}. Thus, we have $(3\bar{y}_5 + 3\bar{y}_6 + 3\bar{y}_7 + \bar{y}_8 + \bar{y}_9 + \bar{y}_{10} + 3\bar{y}_{11} + 3\bar{y}_{12} + 3\bar{y}_{13} + 3\bar{y}_{14} + 4\bar{y}_{15} + 6\bar{y}_{16} + 6\bar{y}_{17} + 6\bar{y}_{18} + 9\bar{y}_{19}) \geq \bar{x}_6$.

This implies that $(\bar{x}, \bar{y})$ is a feasible solution to $LP_{bin}(v)$ and its objective equals the number of bins needed to pack the edges incident at v in unit-sized bins. Thus we have the lemma. □

We now show a contradiction by showing that the optimal value of the $LP_{bin}(v)$ is more than m.

Lemma 5.3.17. The optimal solution to the $LP_{bin}(v)$ is strictly more than m.

Proof. We prove this by considering the dual linear program of the $LP_{bin}(v)$. Since every feasible solution to the dual LP gives a lower bound on the objective of the primal $LP_{bin}(v)$, it is enough to exhibit a feasible dual solution of objective strictly more than m to prove the lemma. Now the dual of the $LP_{bin}(v)$ is given below:

$$\begin{array}{ll}
max \quad \beta m \cdot v_1 & \\
\text{Subject to:} & \\
v_1 - v_2 \leq 0, & v_1 - v_3 - v_6 \leq 0, \\
v_1 - v_4 - v_6 \leq 0, & v_1 - v_5 - 2v_6 \leq 0, \\
v_1 - 3v_6 \leq 0, & v_1 - 2v_6 - v_7 \leq 0, \\
3v_6 \leq 1, & v_5 + 2v_6 \leq 1, \\
v_4 + 2v_6 \leq 1, & 2v_5 + v_6 \leq 1, \\
2v_6 + 3v_7 \leq 1, & v_5 + v_6 + 3v_7 \leq 1, \\
v_4 + v_6 + 3v_7 \leq 1, & v_3 + v_6 + v_7 \leq 1, \\
v_3 + v_4 + v_7 \leq 1, & v_3 + v_5 + v_7 \leq 1, \\
v_4 + v_5 + 3v_7 \leq 1, & 2v_4 + 3v_7 \leq 1, \\
2v_5 + 3v_7 \leq 1, & v_2 + 3v_7 \leq 1, \\
v_3 + 4v_7 \leq 1, & v_4 + 6v_7 \leq 1, \\
v_5 + 6v_7 \leq 1, & v_6 + 6v_7 \leq 1, \\
9v_7 \leq 1, & v_i \geq 0 \quad \forall i \in [7].
\end{array}$$

A feasible dual solution is

$$v_1 = \frac{9}{13}, v_2 = \frac{9}{13}, v_3 = \frac{7}{13}, v_4 = \frac{5}{13}, v_5 = \frac{1}{13}, v_6 = \frac{4}{13}, v_7 = \frac{1}{13}. \tag{76}$$

Thus dual optima is at least

$$\beta m \cdot \frac{9}{13} > m\,.$$

Thus, we need more than m bins to pack all items incident at v, a contradiction. □

This completes the proof of Lemma 5.3.12. □

Therefore, the proof of Theorem 5.0.2 is complete. □

5.3.1 Better approximation when all edge weights are $> 1/4$

If we assume all items have weights $> 1/4$, then similar analysis shows that $2.2m$ colors are sufficient.

Theorem 5.3.18. If all edge-weights are $> 1/4$, then there is a polynomial time algorithm for the WEIGHTED BIPARTITE EDGE COLORING problem which returns a proper weighted coloring using at most $\lceil 2.2m \rceil$ colors, where m denotes the maximum over all the vertices of the number of unit-sized bins needed to pack the weights of incident edges.

Proof. In this section we will show that Algorithm 12 with $t > \frac{11}{5}$ is sufficient to get a proper weighted coloring when all edges are $> 1/4$. Similar to Section 5.3, we will show in Lemma 5.3.19 that if $\beta \leq 7/5$, then all edges incident at u can not be packed into m bins. On the other hand, in Lemma 5.3.20 we show that if $\beta > 7/5$, then all edges incident at v can not be packed into m bins. These two facts together give us the desired contradiction.

Lemma 5.3.19. If $\beta \leq 7/5$, then edges incident at u can not be packed into m bins.

Proof. From the analysis of Lemma 5.3.11, we get that the number of bins to pack all items incident on u is :

$$\begin{aligned} &\geq \frac{2t}{3}m - \frac{\beta}{3}m \\ &> \frac{2}{3}\cdot\frac{11}{5}m - \frac{1}{3}\cdot\frac{7}{5}m \quad \left[\text{Since, } t > \frac{11}{5}\right] \\ &= m\,. \end{aligned} \tag{77}$$

This is a contradiction. □

Lemma 5.3.20. If $\beta > 7/5$, then edges incident at v can not be packed into m bins.

Proof. As in Lemma 5.3.12, we restrict our attention to the edges at tight colors at v and show that if $\beta > \frac{7}{5}$ they cannot be packed in m unit-sized bins. The only difference is that we now have edges that are $> 1/4$. Let us divide these edges incident at u into three size classes.

- Large $L := \{f \in \delta(v) : w_f \in (1/2, 1]\}$.

- Medium $M := \{f \in \delta(v) : w_f \in (1/3, 1/2]\}$.

- Small $S := \{f \in \delta(v) : w_f \in [\alpha, 1/3]\}$.
- Tiny $T' := \{f \in \delta(v) : w_f \in (1/4, \alpha)\}$.

Claim 5.3.21. Consider any feasible bin-packing of edges incident at v restricted to edges in $L \cup M \cup S \cup T$. Then each bin must contain items which correspond to a subset of one of the following 19 configurations.

$C_1 : (S, S, S)$
$C_2 : (M_2, S, S)$
$C_3 : (M_1, S, S)$
$C_4 : (M_2, M_2, S)$
$C_5 : (S, S, T')$
$C_6 : (M_2, S, T')$
$C_7 : (M_1, S, T')$
$C_8 : (L_2, S)$
$C_9 : (L_2, M_1)$
$C_{10} : (L_2, M_2)$
$C_{11} : (M_1, M_2, T')$
$C_{12} : (M_1, M_1, T')$
$C_{13} : (M_2, M_2, T')$
$C_{14} : (L_1, T')$
$C_{15} : (L_2, T')$
$C_{16} : (M_1, T', T')$
$C_{17} : (M_2, T', T')$
$C_{18} : (S, T', T')$
$C_{19} : (T', T', T')$

Proof. In Claim 5.3.16, we considered maximal configurations in tight colors. Let us consider any configuration in Claim 5.3.16 and let us assume that in a tight color belonging to that configuration there are k items from class T and we can not add any more item. Then $(\sum_{i \in L \cup M \cup S} w_i + (k+1) \cdot \frac{1}{10} + \epsilon) > 0$ for any $\epsilon > 0$. Hence, $(\sum_{i \in L \cup M \cup S} w_i + (k+1) \cdot \frac{2}{5} \cdot \frac{1}{4} + \epsilon) > 0$ for any $\epsilon > 0$. Hence if there were k items of class T in configurations in Claim 5.3.16, then we can replace items in T with at most $\lceil (k+1) \cdot \frac{2}{5} \rceil - 1$ items from class T'. Thus we replace $1, 2, 3, 4, 6, 9$ items in T in the configurations in Claim 5.3.16 with items $0, 1, 1, 2, 3$ items in T' respectively. This gives us the configurations in this claim from the configurations in Claim 5.3.16. □

Thus we get the following LP similar to the configuration LP $LP_{bin}(v)$ in the proof of Lemma 5.3.12.

$$min \quad \sum_{i=1}^{19} y_i$$

$$\begin{align} x_1+x_2+x_3+x_4+x_5+x_6 &\geq \beta m \tag{78}\\ y_{14} &\geq x_1 \tag{79}\\ y_8+y_9+y_{10}+y_{15} &\geq x_2 \tag{80}\\ y_3+y_7+y_9+y_{11}+2y_{12}+y_{16} &\geq x_3 \tag{81}\\ y_2+2y_4+y_6+y_{10}+y_{11}+2y_{13}+y_{17} &\geq x_4 \tag{82}\\ 3y_1+2y_2+2y_3+y_4+2y_5+y_6+y_7+y_8+y_{18} &\geq x_2+x_3+2x_4+3x_5+2x_6 \tag{83}\\ (y_5+y_6+y_7+y_{11}+y_{12}+y_{13}+y_{14} & \nonumber\\ +y_{15}+2y_{16}+2y_{17}+2y_{18}+3y_{19}) &\geq x_6 \tag{84}\\ y_j, x_k &\geq 0 \quad \forall j \in [19], k \in [6]. \tag{85} \end{align}$$

Note that all constraints are same as in the proof of Lemma 5.3.12, except for the last constraint as now we count the number of items in T' instead of T. We also get the following dual which differs only in the constraints corresponding to configurations containing type T items.

$$\begin{array}{ll} max \quad \beta m \cdot v_1 & \\ \text{Subject to:} & \\ v_1 - v_2 \leq 0, & v_1 - v_3 - v_6 \leq 0, \\ v_1 - v_4 - v_6 \leq 0, & v_1 - v_5 - 2v_6 \leq 0, \\ v_1 - 3v_6 \leq 0, & v_1 - 2v_6 - v_7 \leq 0, \\ 3v_6 \leq 1, & v_5 + 2v_6 \leq 1, \\ v_4 + 2v_6 \leq 1, & 2v_5 + v_6 \leq 1, \\ 2v_6 + v_7 \leq 1, & v_5 + v_6 + v_7 \leq 1, \\ v_4 + v_6 + v_7 \leq 1, & v_3 + v_6 \leq 1, \\ v_3 + v_4 \leq 1, & v_3 + v_5 \leq 1, \\ v_4 + v_5 + v_7 \leq 1, & 2v_4 + v_7 \leq 1, \\ 2v_5 + v_7 \leq 1, & v_2 + v_7 \leq 1, \\ v_3 + v_7 \leq 1, & v_4 + 2v_7 \leq 1, \\ v_5 + 2v_7 \leq 1, & v_6 + 2v_7 \leq 1, \\ 3v_7 \leq 1, & v_i \geq 0 \quad \forall i \in [7]. \end{array}$$

Here a feasible dual solution is

$$v_1 = \frac{5}{7}, v_2 = \frac{5}{7}, v_3 = \frac{4}{7}, v_4 = \frac{3}{7}, v_5 = \frac{1}{7}, v_6 = \frac{2}{7}, v_7 = \frac{1}{7}. \tag{86}$$

Thus dual optima is at least

$$\beta m \cdot \frac{7}{5} > m\,.$$

This completes the proof of Lemma 5.3.20. □

Therefore, the proof of Theorem 5.3.18 is complete. □

5.4 Conclusion

In this chapter, we have shown that $\lceil 2.2223m \rceil$ colors are sufficient to get a polynomial time proper weighted coloring for bipartite graphs. Our techniques based on configuration LP and other structural properties of bin packing, might be useful in the analysis of other algorithms related to Clos networks. We remark that considering the case $1/4 \geq \alpha > 1/5$ separately, might improve the bound further by more case analysis and the fact that there are at most $4m$ items. However we can at most attain $35m/16 \approx 2.19m$ by that analysis.

There is no better known existential result even with exponential running time. Finding a better approximation algorithm (independent of m) or inapproximability, and extending our techniques to general graphs are other interesting directions.

Chapter VI

CONCLUSION

In this thesis we obtained improved approximation for three classical generalizations of bin packing: geometric bin packing, vector bin packing and weighted bipartite edge coloring. We have made related concluding remarks in the corresponding chapters. Now in this chapter we conclude by listing some of the open problems related to multidimensional bin packing for future work.

Problem 1. Tight approximability of bin packing.

The present best algorithm for 1-D BP by Hoberg and Rothvoß [106], uses $\mathsf{Opt} + O(\log \mathsf{Opt})$ bins. Proving one could compute a packing with only a constant number of extra bins will be a remarkable progress and is mentioned as one of the ten most important problems in approximation algorithms [204]. Consider the seemingly simple 3-Partition case in which all n items have sizes $s_i \in (1/4, 1/2)$. Recent progress by [165] suggests that either $O(\log n)$ bound is the best possible for 3-Partition or some fundamentally new ideas are needed to make progress.

Problem 2. Integrality gap of Gilmore-Gomory LP.

It has been conjectured in [181] that the Gilmore-Gomory LP for 1-D BP has *Modified Integer Roundup Property*, i.e., $\mathsf{Opt} \leq \lceil \mathsf{Opt}_f \rceil + 1$. The conjecture has been proved true for the case when the instance contains at most 7 different item sizes [184]. Settling the status for the general case is an important open problem in optimization.

Problem 3. Tight asymptotic competitive ratio for 1-D online BP.

The present best algorithm for online bin packing is by Seiden [185] and has asymptotic performance ratio at most 1.58889. Ramanan et al. [176] showed that these *Harmonic-type* algorithms can not achieve better than 1.58333 asymptotic competitive ratio. In general the best known lower bound for asymptotic competitive ratio is 1.54014 [200]. Giving a stronger lower bound using some other construction is an important question in online algorithms.

Problem 4. Improved inapproximability for multidimensional bin packing. There is a huge gap between the best approximation guarantee and hardness of multidimensional bin packing. There are no explicit inapproximability bounds known for multidimensional bin packing as function of d, apart from the APX-hardness in 2-D. Thus there is a huge gap between the best algorithm (1.69^{d-1}, i.e., exponential in d for geometric packing and $O(\ln d)$ for vector packing for $d > 2$) and the hardness. Improved inapproximability, as a function of d, will be an interesting hardness result.

Problem 5. Tight ratio between optimal Guillotine packing and optimal bin packing. Improving the present guarantee for 2-D GBP will require an algorithm that is not *input-agnostic*. In particular, this implies that it should have the property that it can round two identical items (i.e., with identical height and width) differently. One such candidate is the guillotine packing approach [18]. It has been conjectured that this approach can give an approximation ratio of 4/3 for 2-D GBP. At present the best known upper bound on this gap is $T_\infty \approx 1.69$ [28]. Guillotine cutting also has connections with other geometric packing problems such as GEOMETRIC KNAPSACK and MAXIMUM INDEPENDENT SET RECTANGLES [2].

Problem 6. Tight ratio between optimal two-stage packing and optimal bin packing. Caprara conjectured [27] that there is a two-stage packing that gives

3/2 approximation for 2-D bin packing. As there are PTAS for 2-stage packing [28], this will give another 3/2 approximation for 2-D BP and coupled with our R&A method this will give another $(1.405 + \epsilon)$ approximation. Presently the upper bound between best two-stage packing and optimal bin packing is $T_\infty \approx 1.69$. As 2-stage packings are very well-studied, this question is of independent interest and it might give us more insight on the power of Guillotine packing.

Problem 7. Extending R&A framework to d-D GBP and 3-D SP.

One key bottleneck to extend R&A framework to d-D GBP or other related problems, is to find a good approximation algorithm to find the solution of the configuration LP. A $poly(d)$ asymptotic approximation for the LP will give us a $poly(d)$ asymptotic approximation for d-D GBP, a significant improvement over the current best ratio of $2^{O(d)}$ for $d > 2$.

Problem 8. Resolving Chung-Ross Conjecture [38].

The first conjecture says that given an instance of the WEIGHTED BIPARTITE EDGE COLORING problem, there is a proper weighted coloring using at most $2m - 1$ colors where m denotes the maximum over all the vertices of the number of unit-sized bins needed to pack the weights of edges incident at the vertex.

A stronger version of the conjecture is that, given an instance of the WEIGHTED BIPARTITE EDGE COLORING problem, there is a proper weighted coloring using at most $2n - 1$ colors where n is the smallest integer greater than the maximum over all the vertices of the total weight of edges incident at the vertex.

Finally, finding faster algorithms that work well in practice, is also a very important problem.

REFERENCES

[1] Aarts, E. H. and Lenstra, J. K., *Local search in combinatorial optimization.* Princeton University Press, 2003.

[2] Abed, F., Chalermsook, P., Correa, J., Karrenbauer, A., Pérez-Lantero, P., Soto, J. A., and Wiese, A., "On guillotine cutting sequences," in *APPROX (to appear)*, 2015.

[3] Adamaszek, A. and Wiese, A., "A quasi-ptas for the two-dimensional geometric knapsack problem," in *SODA*, pp. 1491–1505, 2015.

[4] Alon, N., Azar, Y., Csirik, J., Epstein, L., Sevastianov, S. V., Vestjens, A. P. A., and Woeginger, G. J., "On-line and off-line approximation algorithms for vector covering problems," *Algorithmica*, vol. 21, no. 1, pp. 104–118, 1998.

[5] Arora, S. and Barak, B., *Computational complexity: a modern approach.* Cambridge University Press, 2009.

[6] Arora, S., Hazan, E., and Kale, S., "The multiplicative weights update method: a meta-algorithm and applications," *Theory of Computing*, vol. 8, no. 1, pp. 121–164, 2012.

[7] Azar, Y., Cohen, I. R., Kamara, S., and Shepherd, B., "Tight bounds for online vector bin packing," in *STOC*, pp. 961–970, 2013.

[8] Baker, B. S. and Coffman, Jr, E. G., "A tight asymptotic bound for next-fit-decreasing bin-packing," *SIAM Journal on Algebraic Discrete Methods*, vol. 2, no. 2, pp. 147–152, 1981.

[9] Baker, B. S., Jr., E. G. C., and Rivest, R. L., "Orthogonal packings in two dimensions," *SIAM J. Comput.*, vol. 9, no. 4, pp. 846–855, 1980.

[10] Baker, B. S. and Schwarz, J. S., "Shelf algorithms for two-dimensional packing problems," *SIAM J. Comput.*, vol. 12, no. 3, pp. 508–525, 1983.

[11] Balogh, J., Békési, J., Dósa, G., Sgall, J., and van Stee, R., "The optimal absolute ratio for online bin packing," in *SODA*, pp. 1425–1438, 2015.

[12] Bansal, N., Caprara, A., Jansen, K., Prädel, L., and Sviridenko, M., "A structural lemma in 2-dimensional packing, and its implications on approximability," in *ISAAC*, pp. 77–86, 2009.

[13] Bansal, N., Caprara, A., and Sviridenko, M., “A new approximation method for set covering problems, with applications to multidimensional bin packing,” *SIAM J. Comput.*, vol. 39, no. 4, pp. 1256–1278, 2009.

[14] Bansal, N., Correa, J. R., Kenyon, C., and Sviridenko, M., “Bin packing in multiple dimensions: Inapproximability results and approximation schemes,” *Math. Oper. Res.*, vol. 31, no. 1, pp. 31–49, 2006.

[15] Bansal, N., Elias, M., and Khan, A., “Improved approximation for vector packing,” *Under Submission*, 2015.

[16] Bansal, N., Han, X., Iwama, K., Sviridenko, M., and Zhang, G., “Harmonic algorithm for 3-dimensional strip packing problem,” in *SODA*, pp. 1197–1206, 2007.

[17] Bansal, N. and Khan, A., “Improved approximation algorithm for two-dimensional bin packing,” in *SODA*, pp. 13–25, 2014.

[18] Bansal, N., Lodi, A., and Sviridenko, M., “A tale of two dimensional bin packing,” in *FOCS*, pp. 657–666, 2005.

[19] Bansal, N. and Sviridenko, M., “Two-dimensional bin packing with one-dimensional resource augmentation,” *Discrete Optimization*, vol. 4, no. 2, pp. 143–153, 2007.

[20] Bernáth, A. and Király, T., “Covering skew-supermodular functions by hypergraphs of minimum total size,” *Oper. Res. Lett.*, vol. 37, no. 5, pp. 345–350, 2009.

[21] Blitz, D., van Vliet, A., and Woeginger, G., “Lower bounds on the asymptotic worst-case ratio of online bin packing alorithms,” *Manuscript*, 1996.

[22] Bobba, R., Fatemieh, O., Khan, F., Khan, A., Gunter, C. A., Khurana, H., and Prabhakaran, M., “Attribute-based messaging: Access control and confidentiality,” *ACM Transactions on Information and System Security (TISSEC)*, vol. 13, no. 4, p. 31, 2010.

[23] Bortfeldt, A., “A genetic algorithm for the two-dimensional strip packing problem with rectangular pieces,” *European Journal of Operational Research*, vol. 172, no. 3, pp. 814–837, 2006.

[24] Boyar, J. and Favrholdt, L. M., “The relative worst order ratio for online algorithms,” *ACM Transactions on Algorithms (TALG)*, vol. 3, no. 2, p. 22, 2007.

[25] Boyar, J., Larsen, K. S., and Nielsen, M. N., “The accommodating function: A generalization of the competitive ratio,” *SIAM Journal on Computing*, vol. 31, no. 1, pp. 233–258, 2001.

[26] Brown, D. J., Baker, B. S., and Katseff, H. P., "Lower bounds for on-line two-dimensional packing algorithms," *Acta Informatica*, vol. 18, no. 2, pp. 207–225, 1982.

[27] Caprara, A., "Packing 2-dimensional bins in harmony," in *FOCS*, pp. 490–499, 2002.

[28] Caprara, A., Lodi, A., and Monaci, M., "Fast approximation schemes for two-stage, two-dimensional bin packing," *Math. Oper. Res.*, vol. 30, no. 1, pp. 150–172, 2005.

[29] Caprara, A. and Monaci, M., "On the two-dimensional knapsack problem," *Oper. Res. Lett.*, vol. 32, no. 1, pp. 5–14, 2004.

[30] Caprara, A. and Toth, P., "Lower bounds and algorithms for the 2-dimensional vector packing problem," *Discrete Applied Mathematics*, vol. 111, no. 3, pp. 231–262, 2001.

[31] Chan, S. O., Lee, J. R., Raghavendra, P., and Steurer, D., "Approximate constraint satisfaction requires large lp relaxations," in *FOCS*, pp. 350–359, IEEE, 2013.

[32] Chang, S. Y., Hwang, H.-C., and Park, S., "A two-dimensional vector packing model for the efficient use of coil cassettes," *Computers & operations research*, vol. 32, no. 8, pp. 2051–2058, 2005.

[33] Chang, Y.-C., Chang, Y.-W., Wu, G.-M., and Wu, S.-W., "B*-trees: a new representation for non-slicing floorplans," in *Proceedings of the 37th Annual Design Automation Conference*, pp. 458–463, ACM, 2000.

[34] Chekuri, C. and Khanna, S., "On multidimensional packing problems," *SIAM J. Comput.*, vol. 33, no. 4, pp. 837–851, 2004.

[35] Chekuri, C., Vondrák, J., and Zenklusen, R., "Multi-budgeted matchings and matroid intersection via dependent rounding," in *SODA*, pp. 1080–1097, 2011.

[36] Chlebík, M. and Chlebíková, J., "Inapproximability results for orthogonal rectangle packing problems with rotations," in *CIAC*, pp. 199–210, 2006.

[37] Chung, F., Garey, M. R., and Johnson, D. S., "On packing two-dimensional bins," *SIAM J. Algebraic Discrete Methods*, vol. 3, pp. 66–76, 1982.

[38] Chung, S.-P. and Ross, K. W., "On nonblocking multirate interconnection networks," *SIAM Journal on Computing*, vol. 20, no. 4, pp. 726–736, 1991.

[39] Clos, C., "A study of nonblocking switching networks," *Bell System Technical Journal*, vol. 32, no. 2, pp. 406–424, 1953.

[40] Coffman, E. G., Galambos, G., Martello, S., and Vigo, D., "Bin packing approximation algorithms: Combinatorial analysis," in *Handbook of combinatorial optimization*, pp. 151–207, Springer, 1999.

[41] Coffman, E. G. and Lueker, G. S., *Probabilistic analysis of packing and partitioning algorithms*. Wiley New York, 1991.

[42] Coffman Jr, E. G., Csirik, J., Galambos, G., Martello, S., and Vigo, D., "Bin packing approximation algorithms: Survey and classification," in *Handbook of Combinatorial Optimization*, pp. 455–531, Springer, 2013.

[43] Coffman Jr, E. G., Garey, M. R., and Johnson, D. S., "Approximation algorithms for bin-packin : an updated survey," in *Algorithm design for computer system design*, pp. 49–106, Springer, 1984.

[44] Conway, R. W., Maxwell, W. L., and Miller, L. W., "Theory of scheduling, 1967," *Addison-Wesley*, 1971.

[45] Cook, S. A., "The complexity of theorem-proving procedures," in *STOC*, pp. 151–158, 1971.

[46] Coppersmith, D. and Raghavan, P., "Multidimensional on-line bin packing: algorithms and worst-case analysis," *Operations Research Letters*, vol. 8, no. 1, pp. 17–20, 1989.

[47] Correa, J. and Goemans, M., "Improved bounds on nonblocking 3-stage clos networks," *SIAM Journal of Computing*, vol. 37, pp. 870–894, 2007.

[48] Crescenzi, P. and Panconesi, A., "Completeness in approximation classes," *Information and Computation*, vol. 93, no. 2, pp. 241–262, 1991.

[49] Csirik, J., Johnson, D. S., Kenyon, C., Orlin, J. B., Shor, P. W., and Weber, R. R., "On the sum-of-squares algorithm for bin packing," *Journal of the ACM (JACM)*, vol. 53, no. 1, pp. 1–65, 2006.

[50] Csirik, J. and Van Vliet, A., "An on-line algorithm for multidimensional bin packing," *Operations Research Letters*, vol. 13, no. 3, pp. 149–158, 1993.

[51] Csirik, J. and Woeginger, G. J., "Shelf algorithms for on-line strip packing," *Inf. Process. Lett.*, vol. 63, no. 4, pp. 171–175, 1997.

[52] Csirik, J. and Woeginger, G. J., *On-line packing and covering problems*. Springer, 1998.

[53] Dantzig, G. B., *Linear programming and extensions*. Princeton university press, 1998.

[54] Das, A., Gollapudi, S., Khan, A., and Leme, R. P., "Role of conformity in opinion dynamics in social networks," in *Proceedings of ACM conference on Online social networks (COSN)*, pp. 25–36, 2014.

[55] de la Vega, W. F. and Lueker, G. S., "Bin packing can be solved within 1+epsilon in linear time," *Combinatorica*, vol. 1, no. 4, pp. 349–355, 1981.

[56] Demange, M., Grisoni, P., and Paschos, V. T., "Differential approximation algorithms for some combinatorial optimization problems," *Theoretical Computer Science*, vol. 209, no. 1, pp. 107–122, 1998.

[57] Demisse, G., Mihalyi, R., Okal, B., Poudel, D., Schauer, J., and Nüchter, A., "Mixed palletizing and task completion for virtual warehouses," in *Virtual Manufacturing and Automation Competition (VMAC) Workshop at the Int. Conference of Robotics and Automation (ICRA)*, 2012.

[58] Diedrich, F., Harren, R., Jansen, K., Thöle, R., and Thomas, H., "Approximation algorithms for 3d orthogonal knapsack," *J. Comput. Sci. Technol.*, vol. 23, no. 5, pp. 749–762, 2008.

[59] Dinic, E., "An algorithm for solution of a problem of maximum flow in a network with power estimation," *Soviet Mathematics Doklady*, 1970.

[60] Dósa, G., Li, R., Han, X., and Tuza, Z., "Tight absolute bound for first fit decreasing bin-packing: Ffd(l) $\leq$ 11/9 OPT(L) + 6/9," *Theoretical Computer Science*, vol. 510, pp. 13–61, 2013.

[61] Dósa, G. and Sgall, J., "First fit bin packing: A tight analysis," in *STACS 2013*, pp. 538–549, 2013.

[62] Downey, R. G. and Fellows, M. R., *Parameterized complexity*, vol. 3. Springer Heidelberg, 1999.

[63] Dowsland, K. A., "Some experiments with simulated annealing techniques for packing problems," *European Journal of Operational Research*, vol. 68, no. 3, pp. 389–399, 1993.

[64] Du, D. Z., Gao, B., Hwang, F. K., and Kim, J. H., "On multirate rearrangeable clos networks," *SIAM Journal of Computing*, vol. 28, no. 2, pp. 463–470, 1999.

[65] Edmonds, J., "Paths, trees, and flowers," *Canadian Journal of mathematics*, vol. 17, no. 3, pp. 449–467, 1965.

[66] Eilon, S. and Christofides, N., "The loading problem," *Management Science*, vol. 17, no. 5, pp. 259–268, 1971.

[67] Eisenbrand, F., Pálvölgyi, D., and Rothvoss, T., "Bin packing via discrepancy of permutations," *ACM Transactions on Algorithms*, vol. 9, no. 3, p. 24, 2013.

[68] Eisenbrand, F. and Shmonin, G., "Carathéodory bounds for integer cones," *Operations Research Letters*, vol. 34, no. 5, pp. 564–568, 2006.

[69] Epstein, L., “On variable sized vector packing,” *Acta Cybern.*, vol. 16, no. 1, pp. 47–56, 2003.

[70] Epstein, L., “Two dimensional packing: the power of rotation,” in *Mathematical Foundations of Computer Science 2003*, pp. 398–407, Springer, 2003.

[71] Epstein, L., “Harmonic algorithm for bin packing,” in *Encyclopedia of Algorithms*, 2015.

[72] Epstein, L. and van Stee, R., “Online square and cube packing,” *Acta Inf.*, vol. 41, no. 9, pp. 595–606, 2005.

[73] Epstein, L. and van Stee, R., “Optimal online algorithms for multidimensional packing problems,” *SIAM Journal on Computing*, vol. 35, no. 2, pp. 431–448, 2005.

[74] Epstein, L. and Van Stee, R., “This side up!,” in *Approximation and Online Algorithms*, pp. 48–60, Springer, 2005.

[75] Epstein, L. and Van Stee, R., “Bounds for online bounded space hypercube packing,” *Discrete optimization*, vol. 4, no. 2, pp. 185–197, 2007.

[76] Faroe, O., Pisinger, D., and Zachariasen, M., “Guided local search for the three-dimensional bin-packing problem,” *INFORMS Journal on Computing*, vol. 15, no. 3, pp. 267–283, 2003.

[77] Feige, U. and Singh, M., “Edge coloring and decompositions of weighted graphs,” in *ESA*, pp. 405–416, 2008.

[78] Fereira, C. E., Miyazawa, F. K., and Wakabayashi, Y., “Packing of squares into squares,” in *Pesquisa Operacional*, Citeseer, 1998.

[79] Fishkin, A. V., Gerber, O., and Jansen, K., “On efficient weighted rectangle packing with large resources,” in *ISAAC*, pp. 1039–1050, 2005.

[80] Frieze, A. and Clarke, M., “Approximation algorithms for the m-dimensional 0–1 knapsack problem: Worst-case and probabilistic analyses,” *European Journal of Operational Research*, vol. 15, no. 1, pp. 100–109, 1984.

[81] Galambos, G., Kellerer, H., and Woeginger, G. J., “A lower bound for on-line vector-packing algorithms,” *Acta Cybern.*, vol. 11, no. 1-2, pp. 23–34, 1993.

[82] Galambos, G. and Woeginger, G. J., “On-line bin packing - A restricted survey,” *Math. Meth. of OR*, vol. 42, no. 1, pp. 25–45, 1995.

[83] Garey, M. R., Graham, R. L., and Johnson, D. S., “Resource constrained scheduling as generalized bin packing,” *J. Comb. Theory, Ser. A*, vol. 21, no. 3, pp. 257–298, 1976.

[84] Garey, M. R., Graham, R. L., and Ullman, J. D., "Worst-case analysis of memory allocation algorithms," in *STOC*, pp. 143–150, 1972.

[85] Garey, M. R. and Johnson, D. S., "Complexity results for multiprocessor scheduling under resource constraints," *SIAM Journal on Computing*, vol. 4, no. 4, pp. 397–411, 1975.

[86] Garey, M. R. and Johnson, D. S., "Computers and interactibility: A guide to the theory of np-completeness," *W.H. Freeman & Co., SF, CA*, 1979.

[87] Garey, M. R. and Johnson, D. S., "Approximation algorithms for bin packing problems: A survey," in *Analysis and design of algorithms in combinatorial optimization*, pp. 147–172, Springer, 1981.

[88] Gilmore, P. C. and Gomory, R. E., "A linear programming approach to the cutting-stock problem," *Operations research*, vol. 9, no. 6, pp. 849–859, 1961.

[89] Glover, F. and Laguna, M., *Tabu search*. Springer, 1999.

[90] Goemans, M. X. and Rothvoss, T., "Polynomiality for bin packing with a constant number of item types," in *SODA 2014*, pp. 830–839, 2014.

[91] Goemans, M. X. and Williamson, D. P., "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming," *Journal of the ACM (JACM)*, vol. 42, no. 6, pp. 1115–1145, 1995.

[92] Gollapudi, S., Khan, A., Kulkarni, J., Talwar, K., and Yazdanbod, S., "A geometric approach to diverse group formation," *Under Submission*, 2015.

[93] Gonzalez, T. F., *Handbook of approximation algorithms and metaheuristics*. CRC Press, 2007.

[94] Grigoriadis, M. D., Khachiyan, L. G., Porkolab, L., and Villavicencio, J., "Approximate max-min resource sharing for structured concave optimization," *SIAM Journal on Optimization*, vol. 11, no. 4, pp. 1081–1091, 2001.

[95] Grötschel, M., Lovász, L., and Schrijver, A., "The ellipsoid method and its consequences in combinatorial optimization," *Combinatorica*, vol. 1, no. 2, pp. 169–197, 1981.

[96] Grötschel, M., Lovász, L., and Schrijver, A., "Geometric algorithm and combinatorial optimization," *Algorithms and Combinatorics: Study and Research Texts, 2. Springer-Verlag, Berlin*, 1988.

[97] Guo, P.-N., Takahashi, T., Cheng, C.-K., and Yoshimura, T., "Floorplanning using a tree representation," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 20, no. 2, pp. 281–289, 2001.

[98] Gupta, V. and Radovanovic, A., “Online stochastic bin packing,” *arXiv preprint arXiv:1211.2687*, 2012.

[99] Han, B. T., Diehr, G., and Cook, J. S., “Multiple-type, two-dimensional bin packing problems: Applications and algorithms,” *Annals of Operations Research*, vol. 50, no. 1, pp. 239–261, 1994.

[100] Han, X., Chin, F. Y. L., Ting, H.-F., Zhang, G., and Zhang, Y., “A new upper bound 2.5545 on 2d online bin packing,” *ACM Transactions on Algorithms*, vol. 7, no. 4, p. 50, 2011.

[101] Han, X., Ye, D., and Zhou, Y., “A note on online hypercube packing,” *Central European Journal of Operations Research*, vol. 18, no. 2, pp. 221–239, 2010.

[102] Harren, R., Jansen, K., Prädel, L., and van Stee, R., “A $(5/3 + \epsilon)$-approximation for strip packing,” *Comput. Geom.*, vol. 47, no. 2, pp. 248–267, 2014.

[103] Harren, R. and van Stee, R., “Packing rectangles into 2opt bins using rotations,” in *SWAT*, pp. 306–318, 2008.

[104] Harren, R. and van Stee, R., “Improved absolute approximation ratios for two-dimensional packing problems,” in *APPROX-RANDOM*, pp. 177–189, 2009.

[105] Harren, R. and van Stee, R., “Absolute approximation ratios for packing rectangles into bins,” *J. Scheduling*, vol. 15, no. 1, pp. 63–75, 2012.

[106] Hoberg, R. and Rothvoss, T., “A logarithmic additive integrality gap for bin packing,” *CoRR*, vol. abs/1503.08796, 2015.

[107] Hochbaum, D. S., *Approximation algorithms for NP-hard problems.* PWS Publishing Co., 1996.

[108] Hochbaum, D. S. and Shmoys, D. B., “Using dual approximation algorithms for scheduling problems theoretical and practical results,” *Journal of the ACM (JACM)*, vol. 34, no. 1, pp. 144–162, 1987.

[109] Holyer, I., “The np-completeness of edge-coloring,” *SIAM Journal on Computing*, vol. 10, no. 4, pp. 718–720, 1981.

[110] Hopper, E. and Turton, B., “An empirical investigation of meta-heuristic and heuristic algorithms for a 2d packing problem,” *European Journal of Operational Research*, vol. 128, no. 1, pp. 34–57, 2001.

[111] Im, S., Kell, N., Kulkarni, J., and Panigrahi, D., “Tight bounds for online vector scheduling,” *CoRR*, vol. abs/1411.3887, 2014.

[112] Imahori, S., Yagiura, M., and Nagamochi, H., "Practical algorithms for two-dimensional packing," *Handbook of Approximation Algorithms and Metaheuristics. Chapman & Hall/CRC Computer & Information Science Series*, vol. 13, 2007.

[113] Itoh, A., Takahashi, W., Nagano, H., Kurisaka, M., and Iwasaki, S., "Practical implementation and packaging technologies for a large-scale atm switching system," *Journal of Selected Areas in Communications*, vol. 9, pp. 1280–1288, 1991.

[114] Jakobs, S., "On genetic algorithms for the packing of polygons," *European Journal of Operational Research*, vol. 88, no. 1, pp. 165–181, 1996.

[115] Jansen, K., Kratsch, S., Marx, D., and Schlotter, I., "Bin packing with fixed number of bins revisited," *Journal of Computer and System Sciences*, vol. 79, no. 1, pp. 39–49, 2013.

[116] Jansen, K. and Prädel, L., "New approximability results for two-dimensional bin packing," in *SODA*, 2013.

[117] Jansen, K. and Prädel, L., "A new asymptotic approximation algorithm for 3-dimensional strip packing," in *SOFSEM 2014: Theory and Practice of Computer Science*, pp. 327–338, Springer, 2014.

[118] Jansen, K., Prädel, L., and Schwarz, U. M., "Two for one: Tight approximation of 2d bin packing," in *WADS*, pp. 399–410, 2009.

[119] Jansen, K. and Solis-Oba, R., "An asymptotic fully polynomial time approximation scheme for bin covering," *Theoretical Computer Science*, vol. 306, no. 1, pp. 543–551, 2003.

[120] Jansen, K. and Solis-Oba, R., "New approximability results for 2-dimensional packing problems," in *MFCS*, pp. 103–114, 2007.

[121] Jansen, K. and Solis-Oba, R., "A polynomial time approximation scheme for the square packing problem," in *IPCO*, pp. 184–198, 2008.

[122] Jansen, K. and Solis-Oba, R., "Rectangle packing with one-dimensional resource augmentation," *Discrete Optimization*, vol. 6, no. 3, pp. 310–323, 2009.

[123] Jansen, K. and van Stee, R., "Placement algorithms for arbitrarily shaped blocks," in *DAC*, p. 474, 1979.

[124] Jansen, K. and van Stee, R., "On strip packing with rotations," in *STOC*, pp. 755–761, 2005.

[125] Jansen, K. and Zhang, G., "Maximizing the total profit of rectangles packed into a rectangle," *Algorithmica*, vol. 47, no. 3, pp. 323–342, 2007.

[126] John Beetem, M. D. and Weingarten, D., "The gf11 supercomputer," in *International Symposium on Computer architecture*, pp. 108–115, 1985.

[127] Johnson, D. S., "Approximation algorithms for combinatorial problems," in *STOC*, pp. 38–49, 1973.

[128] Johnson, D. S., Demers, A., Ullman, J. D., Garey, M. R., and Graham, R. L., "Worst-case performance bounds for simple one-dimensional packing algorithms," *SIAM Journal on Computing*, vol. 3, no. 4, pp. 299–325, 1974.

[129] Jr., E. G. C., Garey, M. R., Johnson, D. S., and Tarjan, R. E., "Performance bounds for level-oriented two-dimensional packing algorithms," *SIAM J. Comput.*, vol. 9, no. 4, pp. 808–826, 1980.

[130] Jylänki, J., "A thousand ways to pack the bin-a practical approach to two-dimensional rectangle bin packing," *retrived from http://clb. demon. fi/files/RectangleBinPack. pdf*, 2010.

[131] Kannan, R., "Minkowskis convex body theorem and integer programming," *Mathematics of Operations Research*, vol. 12, no. 3, p. 415440, 1987.

[132] Karger, D. R. and Onak, K., "Polynomial approximation schemes for smoothed and random instances of multidimensional packing problems," in *SODA*, pp. 1207–1216, 2007.

[133] Karmarkar, N. and Karp, R. M., "An efficient approximation scheme for the one-dimensional bin-packing problem," in *FOCS*, pp. 312–320, 1982.

[134] Karp, R. M., "Reducibility among combinatorial problems," in *Symposium on the Complexity of Computer Computations*, pp. 85–103, 1972.

[135] Kell, B. and van Hoeve, W.-J., "An mdd approach to multidimensional bin packing," in *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, pp. 128–143, Springer, 2013.

[136] Kellerer, H. and Kotov, V., "An approximation algorithm with absolute worst-case performance ratio 2 for two-dimensional vector packing," *Oper. Res. Lett.*, vol. 31, no. 1, pp. 35–41, 2003.

[137] Kenyon, C., "Best-fit bin-packing with random order," in *SODA*, vol. 96, pp. 359–364, 1996.

[138] Kenyon, C. and Rémila, E., "A near-optimal solution to a two-dimensional cutting stock problem," *Math. Oper. Res.*, vol. 25, no. 4, pp. 645–656, 2000.

[139] Khan, A., Pal, S. P., Aanjaneya, M., Bishnu, A., and Nandy, S. C., "Diffuse reflection diameter and radius for convex-quadrilateralizable polygons," *Discrete Applied Mathematics*, vol. 161, no. 10-11, pp. 1496–1505, 2013.

[140] Khan, A. and Raghavendra, P., "On mimicking networks representing minimum terminal cuts," *Information Processing Letters*, vol. 114, no. 7, pp. 365–371, 2014.

[141] Khan, A. and Singh, M., "On weighted bipartite edge coloring," *Under Submission*, 2015.

[142] Kohayakawa, Y., Miyazawa, F. K., Raghavan, P., and Wakabayashi, Y., "Multidimensional cube packing," *Electronic Notes in Discrete Mathematics*, vol. 7, pp. 110–113, 2001.

[143] König, D., "Graphok s alkalmazsuk a determinnsok s a halmazok elmletre," *Mathematikai s Termszettudomnyi rtesit*, vol. 34, pp. 104–119, 1916.

[144] Kröger, B., "Guillotineable bin packing: A genetic approach," *European Journal of Operational Research*, vol. 84, no. 3, pp. 645–661, 1995.

[145] Kulik, A. and Shachnai, H., "There is no eptas for two-dimensional knapsack," *Information Processing Letters*, vol. 110, no. 16, pp. 707–710, 2010.

[146] Laurent, M., "A comparison of the sherali-adams, lovász-schrijver, and lasserre relaxations for 0-1 programming," *Mathematics of Operations Research*, vol. 28, no. 3, pp. 470–496, 2003.

[147] Lee, C. C. and Lee, D. T., "A simple on-line bin-packing algorithm," *J. ACM*, vol. 32, no. 3, pp. 562–572, 1985.

[148] Leung, J. Y., Tam, T. W., Wong, C., Young, G. H., and Chin, F. Y., "Packing squares into a square," *Journal of Parallel and Distributed Computing*, vol. 10, no. 3, pp. 271–275, 1990.

[149] Li, K. and Cheng, K. H., "Heuristic algorithms for on-line packing in three dimensions," *Journal of Algorithms*, vol. 13, no. 4, pp. 589–605, 1992.

[150] Li, K. and Cheng, K.-H., "On three-dimensional packing," *SIAM Journal on Computing*, vol. 19, no. 5, pp. 847–867, 1990.

[151] Lodi, A., "Algorithms for two-dimensional bin packing and assignment problems," *Doktorarbeit, DEIS, Universita di Bologna*, 1999.

[152] Lodi, A., Martello, S., and Monaci, M., "Two-dimensional packing problems: A survey," *European Journal of Operational Research*, vol. 141, no. 2, pp. 241–252, 2002.

[153] Lodi, A., Martello, S., and Vigo, D., "Heuristic algorithms for the three-dimensional bin packing problem," *European Journal of Operational Research*, vol. 141, no. 2, pp. 410–420, 2002.

[154] Lovett, S. and Meka, R., "Constructive discrepancy minimization by walking on the edges," in *FOCS*, pp. 61–67, 2012.

[155] Ma, Y., Hong, X., Dong, S., and Cheng, C., “3d cbl: An efficient algorithm for general 3d packing problems,” in *Circuits and Systems, 2005. 48th Midwest Symposium on*, pp. 1079–1082, IEEE, 2005.

[156] Martello, S. and Vigo, D., “Exact solution of the two-dimensional finite bin packing problem,” *Management science*, vol. 44, no. 3, pp. 388–399, 1998.

[157] McDiarmid, C., “Concentration,” in *Probabilistic methods for algorithmic discrete mathematics*, pp. 195–248, Springer, 1998.

[158] Meir, A. and Moser, L., “On packing of squares and cubes,” *Journal of combinatorial theory*, vol. 5, no. 2, pp. 126–134, 1968.

[159] Melen, R. and Turner, J. S., “Nonblocking multirate distribution networks,” *IEEE Transactions on Communications*, vol. 41, no. 2, pp. 362–369, 1993.

[160] Meyerson, A., Roytman, A., and Tagiku, B., “Online multidimensional load balancing,” in *APPROX*, pp. 287–302, 2013.

[161] Miyazawa, F. K. and Wakabayashi, Y., “Packing problems with orthogonal rotations,” in *LATIN*, pp. 359–368, Springer, 2004.

[162] Murata, H., Fujiyoshi, K., Nakatake, S., and Kajitani, Y., “Vlsi module placement based on rectangle-packing by the sequence-pair,” *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 15, no. 12, pp. 1518–1524, 1996.

[163] Nakatake, S., Fujiyoshi, K., Murata, H., and Kajitani, Y., “Module packing based on the bsg-structure and ic layout applications,” *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 17, no. 6, pp. 519–530, 1998.

[164] Nesterov, Y., Nemirovskii, A., and Ye, Y., *Interior-point polynomial algorithms in convex programming*, vol. 13. SIAM, 1994.

[165] Newman, A., Neiman, O., and Nikolov, A., “Beck’s three permutations conjecture: A counterexample and some consequences,” in *FOCS*, pp. 253–262, 2012.

[166] Ngo, H. Q. and Vu, V. H., “Multirate rearrangeable clos networks and a generalized edge-coloring problem on bipartite graphs,” *SIAM J. Comput.*, vol. 32, no. 4, pp. 1040–1049, 2003.

[167] Otoo, E., Pinar, A., and Rotem, D., “A linear approximation algorithm for 2-dimensional vector packing,” *arXiv preprint arXiv:1103.0260*, 2011.

[168] Pandit, V., Kenkre, S., and Khan, A., “Discovering bucket orders from data,” in *SDM*, pp. 872–883, 2011.

[169] Panigrahy, R., Talwar, K., Uyeda, L., and Wieder, U., "Heuristics for vector bin packing," *research. microsoft. com*, 2011.

[170] Pasha, A., *Geometric bin packing algorithm for arbitrary shapes.* PhD thesis, University of Florida, 2003.

[171] Pisinger, D. and Sigurd, M., "Using decomposition techniques and constraint programming for solving the two-dimensional bin-packing problem," *INFORMS Journal on Computing*, vol. 19, no. 1, pp. 36–51, 2007.

[172] Plotkin, S., Shmoys, D., and Tardoso, E., "Fast approximate algorithm for fractional packing and covering problems," *Mathematics of Operations Research*, vol. 20, pp. 257–301, 1995.

[173] Prädel, L. D., *Approximation Algorithms for Geometric Packing Problems.* PhD thesis, Christian-Albrechts-Universität, Kiel, 2013.

[174] Raghavendra, P., *Approximating np-hard problems efficient algorithms and their limits.* PhD thesis, University of Washington, Seattle, 2009.

[175] Ram, B., "The pallet loading problem: A survey," *International Journal of Production Economics*, vol. 28, no. 2, pp. 217–225, 1992.

[176] Ramanan, P., Brown, D. J., Lee, C.-C., and Lee, D.-T., "On-line bin packing in linear time," *Journal of Algorithms*, vol. 10, no. 3, pp. 305–326, 1989.

[177] Rothvoss, T., "The entropy rounding method in approximation algorithms," in *SODA*, pp. 356–372, 2012.

[178] Rothvoss, T., "Approximating bin packing within o(log opt * log log opt) bins," in *FOCS*, pp. 20–29, 2013.

[179] Sakanushi, K., Kajitani, Y., and Mehta, D. P., "The quarter-state-sequence floorplan representation," *Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on*, vol. 50, no. 3, pp. 376–386, 2003.

[180] Sarin, S. and Wilhelm, W., "Prototype models for two-dimensional layout design of robot systems," *IIE transactions*, vol. 16, no. 3, pp. 206–215, 1984.

[181] Scheithauer, G. and Terno, J., "Theoretical investigations on the modified integer round-up property for the one-dimensional cutting stock problem," *Operations Research Letters*, vol. 20, no. 2, pp. 93–100, 1997.

[182] Schrijver, A., *Combinatorial optimization: polyhedra and efficiency*, vol. 24. Springer, 2003.

[183] Schuster, M., Bormann, R., Steidl, D., Reynolds-Haertle, S., and Stilman, M., "Stable stacking for the distributor's pallet packing problem," in *IROS*, pp. 3646–3651, IEEE, 2010.

[184] Sebo, A. and Shmonin, G., "Proof of the modified integer round-up conjecture for bin packing in dimension 7," *Manuscript*, 2009.

[185] Seiden, S. S., "On the online bin packing problem," *J. ACM*, vol. 49, no. 5, pp. 640–671, 2002.

[186] Seiden, S. S. and van Stee, R., "New bounds for multi-dimensional packing," in *SODA*, pp. 486–495, 2002.

[187] Seiden, S. S. and Woeginger, G. J., "The two-dimensional cutting stock problem revisited," *Math. Program.*, vol. 102, no. 3, pp. 519–530, 2005.

[188] Seo, D., Khan, A., and Lee, H., "A study on detecting malcodes distribution sites," in *Korean Information Processing Society (KIPS) Fall Conference*, pp. 1425–1428, 2008.

[189] Shannon, C. E., "A theorem on coloring the lines of a network," *Journal of Mathematics and Physics*, vol. 28, no. 2, pp. 148–151, 1949.

[190] Shor, P. W., "How to pack better than best fit: tight bounds for average-case online bin packing," in *FOCS*, pp. 752–759, 1991.

[191] Skiena, S., "Who is interested in algorithms and why?: lessons from the stony brook algorithms repository," *ACM SIGACT News*, vol. 30, no. 3, pp. 65–74, 1999.

[192] Slepian, T., "Two theorems on a particular crossbar switching," *unpublished manuscript*, 1958.

[193] Spieksma, F. C., "A branch-and-bound algorithm for the two-dimensional vector packing problem," *Computers & operations research*, vol. 21, no. 1, pp. 19–25, 1994.

[194] Steinberg, A., "A strip-packing algorithm with absolute performance bound 2," *SIAM Journal on Computing*, vol. 26, no. 2, pp. 401–409, 1997.

[195] Stiebitz, M., Scheide, D., Toft, B., and Favrholdt, L. M., *Graph Edge Coloring: Vizing's Theorem and Goldberg's Conjecture*, vol. 75. John Wiley & Sons, 2012.

[196] Stillwell, M., Schanzenbach, D., Vivien, F., and Casanova, H., "Resource allocation algorithms for virtualized service hosting platforms," *Journal of Parallel and Distributed Computing*, vol. 70, no. 9, pp. 962–974, 2010.

[197] Sweeney, P. E. and Paternoster, E. R., "Cutting and packing problems: a categorized, application-orientated research bibliography," *Journal of the Operational Research Society*, pp. 691–706, 1992.

[198] Tait, P. G., "Remarks on the colouring of maps," in *Proc. Roy. Soc. Edinburgh*, vol. 10, pp. 501–503, 1880.

[199] van Stee, R., “Sigact news online algorithms column 26: Bin packing in multiple dimensions,” *ACM SIGACT News*, vol. 46, no. 2, pp. 105–112, 2015.

[200] van Vliet, A., “An improved lower bound for on-line bin packing algorithms,” *Inf. Process. Lett.*, vol. 43, no. 5, pp. 277–284, 1992.

[201] Vazirani, V. V., *Approximation algorithms.* Springer Science & Business Media, 2001.

[202] Vercruyssen, D. and Muller, H., “Simulation in production,” *A report of the University of Gent, Belgium*, 1987.

[203] Vizing, V. G., “On an estimate of the chromatic class of a p-graph (in russian),” *Diskret. Analiz*, vol. 3, pp. 23–30, 1964.

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

[205] Woeginger, G. J., “There is no asymptotic ptas for two-dimensional vector packing,” *Inf. Process. Lett.*, vol. 64, no. 6, pp. 293–297, 1997.

[206] Yao, A. C.-C., “New algorithms for bin packing,” *Journal of the ACM (JACM)*, vol. 27, no. 2, pp. 207–227, 1980.

[207] Zhang, G., “A 3-approximation algorithm for two-dimensional bin packing,” *Oper. Res. Lett.*, vol. 33, no. 2, pp. 121–126, 2005.

VITA

Arindam Khan obtained his bachelor's degree in Computer Science and Engineering at Indian Institute of Technology, Kharagpur, India and is currently a PhD candidate in the *Algorithms, Combinatorics and Optimization* program at Georgia Institute of Technology, Atlanta. His research [17, 15, 140, 54, 92, 139, 22, 168, 188, 141] spans several areas of theoretical computer science including approximation algorithms and inapproximability results for NP-hard optimization problems, graph algorithms, algorithmic game theory as well as applied algorithms for social networks and data mining. Previously he has been a research intern at Theory group, Microsoft Research Redmond and Microsoft Research Silicon Valley and a blue scholar at Analytics and Optimization group, IBM Research, India. He welcomes your comments to akhan67@gatech.edu.