Approximation Algorithms for Round-UFP and Round-SAP

Debajyoti Kar
Department of Computer Science and Automation, Indian Institute of Science, Bengaluru, India

Arindam Khan
Department of Computer Science and Automation, Indian Institute of Science, Bengaluru, India

Andreas Wiese
Department of Mathematics, Technische Unviersität Müchen, Germany

Abstract

We study Round-UFP and Round-SAP, two generalizations of the classical Bin Packing problem that correspond to the unsplittable flow problem on a path (UFP) and the storage allocation problem (SAP), respectively. We are given a path with capacities on its edges and a set of jobs where for each job we are given a demand and a subpath. In Round-UFP, the goal is to find a packing of all jobs into a minimum number of copies (rounds) of the given path such that for each copy, the total demand of jobs on any edge does not exceed the capacity of the respective edge. In Round-SAP, the jobs are considered to be rectangles and the goal is to find a non-overlapping packing of these rectangles into a minimum number of rounds such that all rectangles lie completely below the capacity profile of the edges.

We show that in contrast to Bin Packing, both problems do not admit an asymptotic polynomial-time approximation scheme (APTAS), even when all edge capacities are equal. However, for this setting, we obtain asymptotic $(2+\varepsilon)$-approximations for both problems. For the general case, we obtain an $O(\log\log n)$-approximation algorithm and an $O(\log\log \frac{1}{\delta})$-approximation under $(1+\delta)$-resource augmentation for both problems. For the intermediate setting of the *no bottleneck assumption* (i.e., the maximum job demand is at most the minimum edge capacity), we obtain an absolute 12- and an asymptotic $(16+\varepsilon)$-approximation algorithm for Round-UFP and Round-SAP, respectively.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases Approximation Algorithms, Scheduling, Rectangle Packing

Digital Object Identifier 10.4230/LIPIcs.ESA.2022.71

Related Version *Full Version*: https://arxiv.org/abs/2202.03492

Funding *Arindam Khan*: Arindam Khan was supported in part by Pratiksha Trust Young Investigator Award, Google CSExplore Award, and Google India Research Award.

Acknowledgements We thank Waldo Gálvez, Afrouz Jabal Ameli, Siba Smarak Panigrahi, and Arka Ray for helpful initial discussions.

1 Introduction

The unsplittable flow on a path problem (UFP) and the storage allocation problem (SAP) are two well-studied problems in combinatorial optimization. In this paper, we study Round-UFP and Round-SAP, which are two related natural problems that also generalize the classical Bin Packing problem.

In both Round-UFP and Round-SAP, we are given as input a path $G=(V,E)$ and a set of n jobs J. We assume that $\{v_0, v_1, \ldots, v_m\}$ are the vertices in V from left to right and then for each $i \in \{1, ..., m\}$ there is an edge $e_i := \{v_{i-1}, v_i\}$. Each job $\mathrm{j} \in J$ has integral demand $d_\mathrm{j} \in \mathbb{N}$, a source $v_{s_\mathrm{j}} \in V$, and a sink $v_{t_\mathrm{j}} \in V$. We say that each job j spans the path P_j which we define to be the path between v_{s_j} and v_{t_j}. For every edge $e \in E$, we are given

© Debajyoti Kar, Arindam Khan, and Andreas Wiese;
licensed under Creative Commons License CC-BY 4.0
30th Annual European Symposium on Algorithms (ESA 2022).
Editors: Shiri Chechik, Gonzalo Navarro, Eva Rotenberg, and Grzegorz Herman; Article No. 71; pp. 71:1–71:19
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

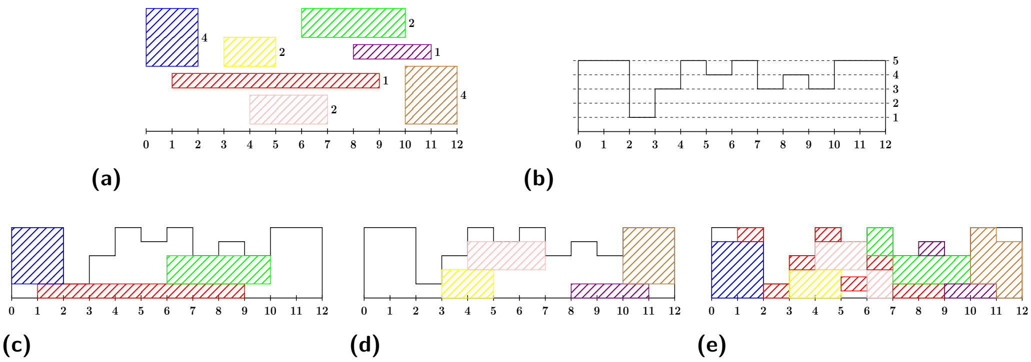

Figure 1 **(a)** A set of 7 jobs with demands written beside; **(b)** The capacity profile; **(c)**, **(d)** Any valid ROUND-SAP packing requires at least 2 rounds; **(e)** A valid ROUND-UFP packing using only 1 round.

an integral capacity c_e. A useful geometric interpretation of the input path and the edge capacities is the following (see Figure 1): consider the interval $[0, m)$ on the x-axis and a function $c\colon [0, m) \to \mathbb{N}$. Each edge e_k corresponds to the interval $[k-1, k)$ and each vertex v_i corresponds to the point i. For edge e_k, we define $c(x) = c_{e_k}$ for each $x \in [k-1, k)$.

In the ROUND-UFP problem, the objective is to partition the jobs J into a minimum number of sets $J_1, ..., J_k$ (that we will denote by *rounds*) such that the jobs in each set J_i form a *valid packing,* i.e., they obey the edge capacities, meaning that $\sum_{j \in J_i : e \in P_j} d_j \le c_e$ for each $e \in E$. In the ROUND-SAP problem, we require to compute additionally for each set J_i a non-overlapping set of rectangles underneath the capacity profile, corresponding to the jobs in J_i (see Figure 1). Formally, we require for each job $j \in J_i$ to determine a height h_j with $h_j + d_j \le c_e$ for each edge $e \in P_j$, yielding a rectangle $R_j = (s_j, h_j) \times (t_j, h_j + d_j)$, such that for any two jobs $j, j' \in J_i$ we have that $R_j \cap R_{j'} = \emptyset$. Again, the objective is to minimize the number of rounds.

Note that unlike ROUND-SAP, in ROUND-UFP we do not need to pack the jobs as contiguous rectangles. Hence, intuitively in ROUND-UFP we can slice the rectangles vertically and place different slices at different heights. See Figure 1.

ROUND-UFP and ROUND-SAP arise naturally in the setting of resource allocation with connections to many fundamental optimization problems. One possible interpretation of the input path is that each vertex is a computer and that these computers are connected via some wired connection, e.g., optical fiber, ethernet, etc. Each direct connection between two computers is modeled by an edge in the input and it has a certain bandwidth, modeled by the capacity of the edge. Also, certain pairs of computers seek to send data from one to the other, and these pairs are represented by the input jobs. The size of each job corresponds to the amount of data that needs to be transported. Now it can happen that the bandwidths do not support all the jobs. In this case, a remedy is to partition the jobs into groups such that each group is supported by the bandwidths. It is desirable to have as few groups as possible, in order to send the data as quickly as possible. Now each of the mentioned groups corresponds to a round in ROUND-UFP, and therefore we seek to minimize the number of rounds. In ROUND-SAP, we require additionally that each job gets a contiguous portion of the communication bandwidths on each edge and also that this portion is the same on each edge. This is a requirement that arises in practice in wavelength division multiplexing and optical fiber minimization, see [4, 51].

An alternative interpretation of the path is that it represents a time axis. For example, the first edge e_1 would correspond to the time interval $[0, 1)$, the second edge e_2 to the interval $[1, 2)$, etc. The capacities of the edges represent the available amount of some resource of a machine, e.g., memory or transmission frequencies, etc., which might change over time. Each input job then represents a job that needs to be executed during some time interval, i.e., the time interval that corresponds to the edges of the path of the job. Similar to the above, it might be the case that not all jobs can be executed on the same machine because not enough resources are available on a machine. Then, we want to distribute the jobs on several machines, such that all jobs assigned to the same machine can run on that machine, i.e., enough resources are available. Here we assume that each machine has its own resources, e.g., its own CPUs or memory. The jobs on the same machine then form a round according to our definition of Round-UFP. Naturally, we seek to minimize the number of rounds, which corresponds to minimizing the number of machines. Similarly as above, in Round-SAP, we additionally require that each job gets a contiguous portion of the resource, e.g., a contiguous portion of the frequency spectrum or the memory [11]. Another application is ad-placement, where each job is an advertisement that requires a contiguous portion of the banner [45].

Round-UFP and Round-SAP are APX-hard as they contain the classical Bin Packing problem as a special case when G has only one single edge. However, while for Bin Packing there exists an asymptotic polynomial time approximation scheme (APTAS), [1] it is open whether such an algorithm exists for Round-UFP or Round-SAP. The best known approximation algorithm for Round-UFP is a $O(\min\{\log n, \log m, \log\log c_{\max}\})$-approximation [35]. For the special case of Round-UFP of uniform edge capacities, Pal [49] gave a 3-approximation. Elbassioni et al. [21] gave a 24-approximation algorithm for the problem under the no-bottleneck assumption (NBA) which states that the maximum job demand is upper-bounded by the minimum edge capacity. A result in (the full version of) [45] states that any solution to an instance of UFP can be partitioned into at most 80 sets of jobs such that each of them is a solution to the corresponding SAP instance. This immediately yields approximation algorithms for Round-SAP: a 240-approximation for the case of uniform capacities, a 1920-approximation under the NBA, and a $O(\min\{\log n, \log m, \log\log c_{\max}\})$-approximation for the general case. These are the best known results for Round-SAP.

1.1 Our Contributions

First, we show that both Round-SAP and Round-UFP, unlike the classical Bin Packing problem, do not admit an APTAS, even in the uniform capacity case. We achieve this via a gap preserving reduction from the 3D matching problem. We create a numeric version of the problem and define a set of hard instances for both Round-SAP and Round-UFP. Together with a result of Chlebik and Chlebikova [15], we derive an explicit lower bound on the asymptotic approximation ratio for both problems. Our hardness result holds even for the case in which in the optimal packing no round contains more than $O(1)$ jobs, i.e., a case in which we can even enumerate all possible packings in polynomial time.

For the case of uniform edge capacities, we give asymptotic $(2 + \varepsilon)$-approximation algorithms for both Round-UFP and Round-SAP, and absolute $(2.5+\varepsilon)$ and 3-approximation algorithms for the two problems, respectively. This improves upon the previous absolute 3- and 240-approximation algorithms mentioned above. Note that for both problems our factor of 2 is a natural threshold: in many algorithms for UFP and SAP [10, 3, 33, 34, 45, 46], the

[1] For basic definitions related to approximation algorithms, we refer to Section 2.

input jobs are partitioned into jobs that are relatively small and relatively large (compared to the edge capacities). Then, both sets are handled separately with very different sets of techniques. This inherently loses a factor of 2. Our algorithms are based on a connection of our problems to the dynamic storage allocation (DSA) problem and we show how to use some known deep results for DSA [11] in our setting. In DSA the goal is to place some given jobs as non-overlapping rectangles, minimizing the height of the resulting packing. This problem might seem totally unrelated at first glance, in particular, unrelated to Round-UFP. The mentioned result produces an alignment of rectangles that correspond to our input jobs, based on which we assign jobs to the different rounds in Round-SAP and Round-UFP. In one of the two cases that we distinguish, we additionally use a dynamic program that finds an optimal solution for the relatively large jobs in the input. In comparison, the previously best-known algorithm for Round-UFP is just a greedy routine that sorts the input jobs and then assigns them greedily into the rounds. The previously best result for Round-SAP takes this packing (for Round-UFP) and applies a reduction from [45] as a black-box, using a (quite large) factor of 80, yielding a 240-approximation algorithm. In particular, our algorithm for Round-SAP yields a much smaller approximation ratio and uses techniques that are much better tailored to Round-SAP.

For the general cases of Round-UFP and Round-SAP, we give an $O(\log\log\min\{m,n\})$-approximation algorithm. Depending on the concrete values of n, m, and $c_{\max}$, this constitutes an up to exponential improvement compared to the best known result for Round-UFP [35] and for Round-SAP (by the reasoning via [45] above). As done previously in the literature, we represent these large jobs by rectangles that are drawn as high as possible underneath the capacity profile and we seek a solution such that in each round, the rectangles corresponding to the jobs in the round are pairwise non-overlapping. In contrast to prior work, we formulate this problem as a configuration-LP and we show that we can solve it in polynomial time via a suitable separation oracle. The integral parts of its solution immediately yield an assignment of some input jobs into at most OPT many rounds. With an additional step, we show that the remaining problem (corresponding to the fractional part of the solution of the configuration-LP) can be reduced to several instances of Round-UFP/ Round-SAP with the property that each point is overlapped by at most $O(\log m)$ rectangles. This is crucial to be able to apply another novel step: we employ a recent result by Chalermsook and Walczak [12] that yields a bound on the coloring number of rectangle intersection graphs, and we put the rectangles from each color class into a separate round. Since each point is overlapped by at most $O(\log m)$ rectangles, this yields an approximation ratio of $O(\log\log m)$ only, and since w.l.o.g. $m \leq 2n$, also of $O(\log\log n)$. In comparison, the previously best-known algorithm for the general case of Round-UFP also uses the viewpoint of the rectangles above, but it pays up to a factor of $O(\log m)$ in order to reduce the general case to the special case in which each rectangle is stabbed by one or two lines from a suitably defined set of lines. For the latter case, it employs a polynomial-time 3-approximation algorithm. In particular, this algorithm neither employs a configuration-LP nor the mentioned result in [12].

Then we study the setting of resource augmentation, i.e., where we can increase the edge capacities by a factor of $1+\delta$, while still comparing our solution with the optimal solution with original capacities. In this case, we show that we can reduce the given problem to the setting in which the edge capacites are in the range $[1, 1/\delta)$. Applying the algorithm from [35] then yields a $O(\log\log\frac{1}{\delta})$-approximation for this case for Round-UFP, and with a similar argumentation as before also for Round-SAP.

Furthermore, for the case of the NBA we improve the absolute approximation ratio from 24 to 12 for Round-UFP, and from 1920 to 17 for Round-SAP, and we even obtain an asymptotic $(16+\varepsilon)$-approximation for Round-SAP. For Round-SAP we give a black-box

Table 1 Overview of our results. We distinguish the settings according to uniform edge capacities, the no-bottleneck-assumption (NBA), general edge capacities, and general edge capacities with $(1+\delta)$-resource augmentation (r.a.). Also, we distinguish between absolute approximation ratios and asymptotic approximation ratios. All listed previous results are absolute approximation ratios.

Problem	Edge capacites	Previous approximation	Improved approximation
Round-UFP	uniform	3 [49]	asymp. $2+\varepsilon$, abs. $2.5+\varepsilon$
Round-SAP	uniform	240 [49, 45]	asymp. $2+\varepsilon$, abs. 3
Round-UFP	NBA	24 [21]	abs. 12
Round-SAP	NBA	1920 [21, 45]	asymp. $16+\varepsilon$, abs. 17
Round-UFP	general	$O(\log\min\{n,m,\log c_{\max}\})$ [35]	abs. $O(\log\log\min\{n,m\})$
Round-SAP	general	$O(\log\min\{n,m,\log c_{\max}\})$ [35]	abs. $O(\log\log\min\{n,m\})$
Round-UFP	general with r.a.	$O(\log\min\{n,m,\log c_{\max}\})$ [35]	abs. $O(\log\log(1/\delta))$
Round-SAP	general with r.a.	$O(\log\min\{n,m,\log c_{\max}\})$ [35]	abs. $O(\log\log(1/\delta))$
Round-Tree	uniform	6 [23]	asymp. 5.1, abs. 5.5
Round-Tree	NBA	64 [21]	asymp. 49, abs. 55

reduction to the case of uniform edge capacities. We first round the job sizes and the edge capacities to powers of 2. Then we partition the input jobs into classes according to their *bottleneck capacities* (the bottleneck capacity of a job is the minimum capacity of an edge through which the job passes). For each class of bottleneck capacities, we define a corresponding instance of Round-SAP with uniform edge capacities. The number of different job classes can be super-constant. However, we show that we can combine the solutions for this super-constant number of classes to a global solution for the given instance so that overall we obtain a constant asymptotic approximation ratio of $16+\varepsilon$ for Round-SAP. We remark that our black-box reduction loses a fixed factor of 8, so any improvements for the case of uniform edge capacities would yield an improved approximation ratio for the case of the NBA as well. In comparison, the previously best-known result for Round-SAP under the NBA uses the known 24-approximation algorithm for Round-UFP and then invokes the mentioned result in [45] as a black-box, yielding a quite large approximation ratio of 1920.

For the setting of round-UFP under the NBA, the previously best-known 24-approximation algorithm in [21] loses a large factor of 16 for packing the small jobs. This algorithm is a simple greedy routine that sorts jobs by their left endpoints and places them appropriately into rounds. In contrast, we first scale up the job demands to integral powers of $1/2$ (assuming the minimum edge capacity to be 1) and then consider jobs based on their bottleneck capacities. We then devise a novel way of classifying the jobs based on the *density* of the jobs on various edges: for one class of jobs we show that a simple greedy algorithm loses only a factor of 4; for the other, our classification scheme ensures that the *congestion* on *every* edge is small which enables us to apply a result in [48], again losing a factor of 4. Also for the large jobs, we improve the best-known 8-approximation to a 4-approximation, thus getting an 12-approximation overall for Round-UFP under the NBA.

If in Round-UFP we are given a tree instead of a path, we obtain the Round-Tree problem. The best known result for it under the NBA is a 64-approximation [21] and a 6-approximation is known for uniform edge capacities [23]. We improve the best known approximation ratio under the NBA to 55 and also provide a 5.5-approximation algorithm for the case of uniform edge capacities.

See Table 1 for an overview of our results.

1.2 Other Related Work

Without the NBA, Epstein et al. [22] showed that no deterministic online algorithm can achieve a competitive ratio better than $\Omega(\log\log n)$ or $\Omega(\log\log\log(c_{\max}/c_{\min}))$ for ROUND-UFP. They also gave a $O(\log c_{\max})$-competitive algorithm, where $c_{\max}$ is the largest edge capacity. Without the NBA, recently Jahanjou et al. [35] gave a $O(\min(\log m, \log\log c_{\max}))$-competitive algorithm. For the special case of ROUND-UFP for uniform edge capacities in the online setting, Adamy and Erlebach [2] gave a 195-competitive algorithm, which was subsequently improved to 10 [47, 5].

ROUND-UFP and ROUND-SAP are related to many fundamental optimization problems. For example, ROUND-SAP can be interpreted as an intermediate problem between two-dimensional bin packing (2BP) and the rectangle coloring problem (RC). In 2BP, the goal is to find an axis-parallel nonoverlapping packing of a given set of rectangles (which we can translate in both dimensions) into minimum number of unit square bins. If all edges have the same capacity then ROUND-SAP can be seen as a variant of 2BP in which the horizontal coordinate of each item is fixed and we can choose only the vertical coordinate. For 2BP, the present best asymptotic approximation guarantee is 1.406 [8]. On the other hand, in RC, all rectangles are fixed and the goal is to color the rectangles using a minimum number of colors such that no two rectangles of the same color intersect. For RC, recently Chalermsook and Walczak [12] have given a polynomial-time algorithm that uses only $O(\omega \log \omega)$ colors, where ω is the clique number of the corresponding intersection graph (and hence a lower bound on the number of needed colors). Another related problem is Dynamic Storage Allocation (DSA), where the objective is to pack the given jobs (with fixed horizontal location) such that the maximum vertical height, $\max_j(h_j + d_j)$ (called the *makespan*) is minimized. The current best known approximation ratio for DSA is $(2 + \varepsilon)$ [11].

In a sense ROUND-UFP and ROUND-SAP are 'BIN PACKING-type' problems, and their corresponding 'KNAPSACK-type' problems are UFP and SAP, respectively, where each job has an associated profit and the goal is to select a subset of jobs which can be packed into one single round satisfying the corresponding valid packing constraints. There is a series of work [33, 7, 9, 3, 34, 32] in UFP, culminating in a PTAS [31]. It is maybe surprising that ROUND-UFP does not admit an APTAS, even though UFP admits a PTAS. For SAP, the currently best polynomial time approximation ratio is $2 + \varepsilon$ [45], which has been recently improved to $1.969 + \varepsilon$ [46] for the case of uniform capacities, and also a quasi-polynomial time $(1.997 + \varepsilon)$-approximation is known for quasi-polynomially bounded input data.

There are many other related problems, such as two-dimensional knapsack [26, 37, 28, 36], strip packing [27, 25, 20, 39], maximum independent set of rectangles [1, 12, 44, 29], guillotine separability of rectangles [41, 40, 42], weighted bipartite edge coloring [43], maximum edge disjoint paths [18], etc. We refer the readers to [38, 17] for an overview of these problems.

2 Preliminaries

Let OPT_{UFP} and OPT_{SAP} denote the optimal number of rounds required to pack all jobs of a given instance of ROUND-UFP and ROUND-SAP, respectively. By simple preprocessing, we can assume that each vertex in V corresponds to endpoint(s) of some job(s) in J, and hence $m \leq 2n-1$. Job j is said to *pass through* edge e if $e \in P_j$. The *load* on edge e is defined as $l_e := \sum_{e \in P_j} d_j$, the total sum of demands of all jobs passing through e. Let $L := \max_e l_e$ denote the maximum *load*.

We now define some notions related to approximation algorithms. Consider a minimization problem Π. An algorithm $\mathcal{A}$ has approximation guarantee of α ($\alpha > 1$), if $\mathcal{A}(I) \leq \alpha\, OPT(I)$ for all input instances I of Π. This is also known as *absolute approximation guarantee*. As common in bin packing literature, we also study asymptotic approximation which intuitively is the approximation ratio when OPT tends to infinity. An algorithm $\mathcal{A}$ has asymptotic approximation guarantee of α, if $\mathcal{A}(I) \leq \alpha\, OPT(I) + o(OPT(I))$ for all input instances I of Π. A problem admits polynomial time approximation scheme or PTAS (resp. asymptotic polynomial time approximation scheme or APTAS) if for every constant $\varepsilon > 0$, there exists a $(1+\varepsilon)$-approximation (resp. $(1+\varepsilon)$-asymptotic approximation) algorithm with running time $O\left(n^{f(1/\varepsilon)}\right)$, for any function f that depends only on ε.

3 Lower Bounds

A simple reduction from the Partition problem shows that it is NP-hard to obtain a better approximation ratio than 3/2 for the classical Bin Packing problem. However, in the resulting instances, the optimal solutions use only two or three bins. On the other hand, Bin Packing admits an APTAS [19] and thus, for any $\varepsilon > 0$, a $(1+\varepsilon)$-approximation algorithm for instances in which OPT is sufficiently large. Since Round-SAP and Round-UFP are generalizations of Bin Packing (even if G has only a single edge), the lower bound of 3/2 continues to hold. However, maybe surprisingly, we show that unlike Bin Packing, Round-SAP and Round-UFP do not admit APTASes, even in the case of uniform edge capacities. More precisely, we provide a lower bound of $(1 + 1/1398)$ on the asymptotic approximation ratio for Round-SAP and Round-UFP via a reduction from the 2-Bounded Occurrence Maximum 3-Dimensional Matching (2-B-3-DM) problem.

In 2-B-3-DM, we are given as input three pairwise disjoint sets $X := \{x_1, x_2, \ldots, x_q\}$, $Y := \{y_1, y_2, \ldots, y_q\}$, and $Z := \{z_1, z_2, \ldots, z_q\}$ and a set of triplets $\mathcal{T} \subseteq X \times Y \times Z$ such that each element of $X \cup Y \cup Z$ occurs in exactly two triplets in $\mathcal{T}$. Note that $|X| = |Y| = |Z| = q$ and $|\mathcal{T}| = 2q$. A *matching* is a subset $M \subseteq \mathcal{T}$ such that no two triplets in M agree in any coordinate. The goal is to find a matching of maximum cardinality (denoted by OPT_{3DM}). Chlebik and Chlebikova [15] gave the following hardness result.

▶ **Theorem 1** ([15]). *For 2-B-3-DM there exists a family of instances such that for each instance K of the family, either $OPT_{3DM}(K) < \alpha(q) := \lfloor 0.9690082645q \rfloor$ or $OPT_{3DM}(K) \geq \beta(q) := \lceil 0.979338843q \rceil$, and it is NP-hard to distinguish these two cases.*

Hardness of 2-B-3-DM has been useful in inapproximability results for various (multidimensional) packing, covering, and scheduling problems, e.g. vector packing [52], geometric bin packing [6], geometric bin covering [16], generalized assignment problem [13], etc. Similar to these results, we also use gadgets based on a reduction from 2-B-3-DM to the 4-Partition problem. However, the previous techniques are not directly transferable to our problem due to the inherent differences between these problems. Therefore, we first use the technique from [52] to associate certain integers with the elements of $X \cup Y \cup Z$ and $\mathcal{T}$ and then adapt the numeric data in a different way to obtain the hard instances.

Let $\rho = 32q$ and let $\mathcal{V}$ be the set of $5q$ integers defined as follows: $x'_i = i\rho + 1$, for $1 \leq i \leq |X|$, $y'_j = j\rho^2 + 2$, for $1 \leq j \leq |Y|$, $z'_k = k\rho^3 + 4$, for $1 \leq k \leq |Z|$, $\tau'_l = \rho^4 - k\rho^3 - j\rho^2 - i\rho + 8$, for each triplet $\tau_l = (x_i, y_j, z_k) \in \mathcal{T}$. Define $\gamma = \rho^4 + 15$. The following result is due to Woeginger [52] [2].

[2] There was a minor bug in [52], which was fixed by Ray [50]. See Lemma 2 in [50] for the proof of the lemma.

▶ **Lemma 2** ([52]). *Four integers in $\mathcal{V}$ sum up to the value γ if and only if (i) one of them corresponds to some element $x_i \in X$, one to some element $y_j \in Y$, one to an element $z_k \in Z$, and one to some triplet $\tau_l \in \mathcal{T}$, and if (ii) $\tau_l = (x_i, y_j, z_k)$ holds for these four elements.*

Now, we create a hard instance, tailor-made for our problems. We define that our path $G = (V, E)$ has 40000γ vertices that we identify with the numbers $0, 1, ..., 40000\gamma$. For each $x_i \in X$ (respectively $y_j \in Y$, $z_k \in Z$), we specify two jobs $a_{X,i}$ and $a'_{X,i}$ (respectively $a_{Y,j}$, $a'_{Y,j}$ and $a_{Z,k}$, $a'_{Z,k}$), which will be called *peers* of each other. Each job j is specified by a triplet (s_j, t_j, d_j). We define

- $a_{X,i} = (0, 20000\gamma - 4x'_i, 999\gamma + 4x'_i)$ and $a'_{X,i} = (20000\gamma - 4x'_i, 40000\gamma, 1001\gamma - 4x'_i)$,
- $a_{Y,j} = (0, 20000\gamma - 4y'_j, 999\gamma + 4y'_j)$ and $a'_{Y,j} = (20000\gamma - 4y'_j, 40000\gamma, 1001\gamma - 4y'_j)$, and
- $a_{Z,k} = (0, 20000\gamma - 4z'_k, 999\gamma + 4z'_k)$ and $a'_{Z,k} = (20000\gamma - 4z'_k, 40000\gamma, 1001\gamma - 4z'_k)$.

For each $\tau_l \in \mathcal{T}$, we define two jobs b_l and b'_l (also *peers*) by:

- $b_l = (0, 19001\gamma - 4\tau'_l, 999\gamma + 4\tau'_l)$ and $b'_l = (19001\gamma - 4\tau'_l, 40000\gamma, 1001\gamma - 4\tau'_l)$.

Finally let D be a set of $5q - 4\beta(q)$ *dummy* jobs each specified by $(0, 40000\gamma, 2997\gamma)$. We define that each edge $e \in E$ has a capacity of $c_e := c^* := 4000\gamma$. This completes the reduction. For any job $j = (s_j, t_j, d_j)$ we define its *width* $w_j := t_j - s_j$.

Let $A_X := \{a_{X,i} \mid 1 \leq i \leq q\}$ and $A'_X := \{a'_{X,i} \mid 1 \leq i \leq q\}$. The sets A_Y, A'_Y, A_Z, A'_Z are defined analogously. Let $A := A_X \cup A_Y \cup A_Z$ and $A' := A'_X \cup A'_Y \cup A'_Z$. Finally let $B := \{b_l \mid 1 \leq l \leq 2q\}$ and $B' := \{b'_l \mid 1 \leq l \leq 2q\}$.

To provide some intuition, we first give an upper bound on the number of jobs that can be packed in a round. All following lemmas, statements, and constructions hold for both Round-SAP and Round-UFP.

▶ **Lemma 3.** *In any feasible solution any round can contain at most 8 jobs.*

We say that a round is *nice* if it contains exactly 8 jobs. It turns out that such a round corresponds exactly to one element $\tau_l = (x_i, y_j, z_k) \in \mathcal{T}$. We say that the jobs $a_{X,i}, a'_{X,i}, a_{Y,j}, a'_{Y,j}, a_{Z,k}, a'_{Z,k}, b_l$, and b'_l *correspond* to $\tau_l = (x_i, y_j, z_k)$.

▶ **Lemma 4.** *We have that a round is nice if and only if there is an element $\tau_l = (x_i, y_j, z_k) \in \mathcal{T}$ such that the round contains exactly the jobs that correspond to $\tau_l = (x_i, y_j, z_k)$.*

Given an optimal solution OPT_{3DM} to 2-B-3-DM with $|OPT_{3DM}| \geq \beta(q)$, we construct a solution as follows:

1. Let $\mathcal{M}$ be any subset of OPT_{3DM} with $|\mathcal{M}| = \beta(q)$. Create $\beta(q)$ nice rounds corresponding to the elements in $\mathcal{M}$, i.e., for each element $\tau_l = (x_i, y_j, z_k) \in \mathcal{M}$, create a round containing the jobs that correspond to $\tau_l = (x_i, y_j, z_k)$.
2. For each $\tau_l \in \mathcal{T} \setminus \mathcal{M}$, create a round containing b_l and b'_l along with a dummy job.
3. For each $x_i \in X$ (respectively $y_j \in Y$, $z_k \in Z$) not covered by $\mathcal{M}$, pack $a_{X,i}$ and $a'_{X,i}$ (respectively $a_{Y,j}$, $a'_{Y,j}$ and $a_{Z,k}$, $a'_{Z,k}$) together with one dummy job in one round.

▶ **Lemma 5.** *If $|OPT_{3DM}| \geq \beta(q)$ then the constructed solution is feasible and it uses at most $5q - 3\beta(q)$ rounds.*

Proof. One can easily check that all constructed rounds are feasible. In step (1) we construct exactly $\beta(q)$ rounds. In step (2), we construct $|\mathcal{T}| - \beta(q) = 2q - \beta(q)$ rounds, since $|\mathcal{T}| = 2q$. In step (3), we construct $3|\mathcal{T} \setminus \mathcal{M}| = 3q - 3|OPT_{3DM}|$ rounds. Hence, overall we construct at most $5q - 3|OPT_{3DM}| \leq 5q - 3\beta(q)$ rounds. ◀

Conversely, assume that $|OPT_{3DM}| < \alpha(q)$ and that we are given any feasible solution to our constructed instance. We want to show that it uses at least $5q - 3\beta(q) + \frac{1}{7}(\beta(q) - \alpha(q))$ rounds. For this, a key property of our construction is given in the following lemma.

▶ **Lemma 6.** *If a round contains a dummy job, then it can have at most three jobs: at most one dummy job, at most one job from $A \cup B$, and at most one job from $A' \cup B'$.*

Let n_g denote the number of nice rounds in our solution, n_d the number of rounds with a dummy job, and n_b the number of remaining rounds. Note that each of the latter rounds can contain at most 7 jobs each. Since all jobs in $A \cup A' \cup B \cup B'$ need to be assigned to a round, we have that $8n_g + 7n_b + 2n_d \geq 6q + 2|\mathcal{T}| = 10q$. Since the nice rounds correspond to a matching of the given instance of 2-B-3-DM, we have that $n_g \leq \alpha(q)$. Using this, we lower-bound the number of used rounds in the following lemma.

▶ **Lemma 7.** *If $|OPT_{3DM}| < \alpha(q)$ then the number of rounds in our solution is $n_d+n_g+n_b \geq (5q - 3\beta(q)) + \frac{1}{7}(\beta(q) - \alpha(q))$.*

Proof. Since $8n_g + 7n_b + 2n_d \geq 6q + 2|\mathcal{T}| = 10q$ and $n_d = 5q - 4\beta(q)$, we obtain that $8n_g + 7n_b \geq 8\beta(q)$. Thus $n_g + n_b \geq \frac{8}{7}\beta(q) - \frac{1}{7}n_g$. Since $n_g \leq \alpha(q)$ the number of rounds is at least $n_d + n_g + n_b \geq 5q - 4\beta(q) + \frac{8}{7}\beta(q) - \frac{1}{7}\alpha(q) = (5q - 3\beta(q)) + \frac{1}{7}(\beta(q) - \alpha(q))$. ◀

Now Lemmas 5 and 7 yield our main theorem.

▶ **Theorem 8.** *There exists a constant $\delta_0 > 1/1398$, such that it is NP-hard to approximate* Round-UFP *and* Round-SAP *in the case of uniform edge capacities with an asymptotic approximation ratio less than $1 + \delta_0$.*

4 Algorithms for Uniform Capacity Case

In this section, we provide asymptotic $(2 + \varepsilon)$-approximation for Round-SAP and Round-UFP for the case of uniform edge capacities.

We distinguish two cases, depending on the value of $d_{\max} := \max_{j \in J} d_j$ compared to L.

4.1 Case 1: $d_{\max} \leq \varepsilon^7 L$

First, we invoke an algorithm from [11] for the dynamic storage allocation (DSA) problem. Recall that in DSA the input consists of a set of jobs like in Round-SAP and Round-UFP, but without upper bounds of the edge capacities. Instead, we seek to define a height h_j for each job j such that the resulting rectangles for the jobs are non-overlapping and the *makespan* $\max_j(h_j + d_j)$ is minimized. The maximum load L is defined as in our setting.

We invoke the following theorem on our input jobs J with $\delta := \varepsilon$.

▶ **Theorem 9** ([11])**.** *Assume that we are given a set of jobs J' such that $d_j \leq \delta^7 L$ for each job $j \in J'$. Then there exists an algorithm that produces a DSA packing of J' with makespan at most $(1 + \kappa\delta)L$, where $\kappa > 0$ is some global constant independent of δ.*

Let ξ denote the makespan of the resulting solution to DSA and let c^* denote the (uniform) edge capacity. For each $h \in \mathbb{R}$, we define the horizontal line $\ell_h := \mathbb{R} \times \{h\}$. A job j is said to be *sliced* by ℓ_h if for the computed packing of the jobs it holds that $h_j < h < h_j + d_j$. Now we will transform this into Round-SAP or Round-UFP packing.

We define a set of rounds Γ_1. The set Γ_1 contains a round for each integer i with $0 \leq i \leq \lfloor \xi/c^* \rfloor$ and this round contains all jobs lying between ℓ_{ic^*} and $\ell_{(i+1)c^*}$. In this case, we define *congestion* $r = \lceil L/c^* \rceil$ (clearly, $r \leq OPT_{UFP} \leq OPT_{SAP}$). Thus, $|\Gamma_1| \leq \lfloor (1 + \kappa\varepsilon)L/c^*) \rfloor + 1 \leq (1 + \kappa\varepsilon)r + 1$. There are two subcases.

Subcase A: Assume that $r > 1/(2\kappa\varepsilon)$. In this case $|\Gamma_1| \leq (1 + 3\kappa\varepsilon)r$. We define a set of rounds Γ_2 as follows. For each integer i with $1 \leq i \leq \lfloor \xi/c^* \rfloor$, Γ_2 has a round containing all jobs that are sliced by ℓ_{ic^*}. Thus $|\Gamma_2| \leq \lfloor (1 + \kappa\varepsilon)L/c^* \rfloor \leq (1 + \kappa\varepsilon)r$. Hence, the total number of rounds is bounded by $(2 + 4\kappa\varepsilon)r \leq (2 + O(\varepsilon))OPT_{UFP} \leq (2 + O(\varepsilon))OPT_{SAP}$.

Subcase B: Assume that $r \leq 1/(2\kappa\varepsilon)$. Now $\xi \leq (1+\kappa\varepsilon)L$, and therefore $\xi - L \leq \kappa\varepsilon L \leq c^*/2$. Hence, we have $|\Gamma_1| \leq r+1$ and the $(r+1)^{\text{th}}$ round is filled up to a capacity of at most $c^*/2$ on each edge. Now the total load of the set of jobs that are sliced by $(\ell_{ic^*})_{1\leq i\leq r}$ is at most $r \cdot \varepsilon^7 L$. We now invoke the following result on DSA to this set of jobs.

▶ **Theorem 10** ([30]). *Let J' be a set of jobs with load L. Then a DSA packing of J' of makespan at most $3L$ can be computed in polynomial time.*

Thus the makespan of the computed solution is at most $3r \cdot \varepsilon^7 L \leq 3 \cdot \frac{1}{2\kappa\varepsilon} \cdot \varepsilon^7 \cdot \frac{c^*}{2\kappa\varepsilon} \leq c^*/2$, if ε is small enough. Hence these jobs can be added to the $(r+1)^{\text{th}}$ round of Γ_1. Therefore, we get a packing of J using at most $r+1 \leq OPT_{UFP}+1 \leq OPT_{SAP}+1$ rounds.

4.2 Case 2: $d_{\max} > \varepsilon^7 L$

For this case, we have $c^* \geq d_{\max} > \varepsilon^7 L$ and therefore $r = \lceil L/c^* \rceil \leq 1/\varepsilon^7$. We partition the input jobs into *large* and *small* jobs by defining $J_{\text{large}} := \{\mathrm{j} \in J | d_{\mathrm{j}} > \varepsilon^{56} L\}$ and $J_{\text{small}} := \{\mathrm{j} \in J | d_{\mathrm{j}} \leq \varepsilon^{56} L\}$.

We start with the small jobs J_{small}. First, we apply Theorem 9 to them with $\delta := \varepsilon^8$ and obtain a DSA packing $\mathcal{P}$ for them. We transform it into a solution to Round-SAP with at most $r+1$ rounds as follows: we introduce a set Γ_1 consisting of $r+1$ rounds exactly as in the previous case (when $r \leq 1/(2\kappa\varepsilon)$). The $(r+1)^{\text{th}}$ round would be filled up to a capacity of at most $\kappa\varepsilon^8 L \leq \kappa\varepsilon c^*$. Again applying Theorem 10 to the remaining jobs, we get a DSA packing of makespan at most $3r \cdot \varepsilon^{56} L \leq 3 \cdot (1/\varepsilon^7) \cdot \varepsilon^{56} \cdot (c^*/\varepsilon^7) \leq c^*/2$, and therefore these jobs can be packed inside the $(r+1)^{\text{th}}$ round. Hence, there exists a packing of J_{small} using at most $r+1 \leq OPT_{UFP}+1 \leq OPT_{SAP}+1$ rounds.

Now we consider the large jobs J_{large}. Our strategy is to compute an optimal solution for them via dynamic programming (DP). Intuitively, our DP orders the jobs in J_{large} non-decreasingly by their respective source vertices and assigns them to the rounds in this order. Since the jobs are large, each edge is used by at most $1/\varepsilon^{56}$ large jobs, and using interval coloring one can show easily that at most $1/\varepsilon^{56} = O_\varepsilon(1)$ rounds suffice (e.g., we can color the jobs with $1/\varepsilon^{56}$ colors such that no two jobs with intersecting paths have the same color). This already gives a packing into at most $(2+\varepsilon)OPT_{UFP} + O_\varepsilon(1)$ rounds. However, our DP will return an improved packing with at most $(2+\varepsilon)OPT_{UFP}+1$ rounds. In our DP we have a cell for each combination of an edge e and the assignment of all jobs passing through e to the rounds. Given this, the corresponding subproblem is to assign additionally all jobs to the rounds whose paths lie completely on the right of e.

For Round-SAP we additionally want to bound the number of possible heights h_{j}. To this end, we restrict ourselves to packings that are normalized which intuitively means that all jobs are pushed up as much as possible. Formally, we say that a packing for a set of jobs J' inside a round is *normalized* if for every $\mathrm{j} \in J'$, either $h_{\mathrm{j}} + d_{\mathrm{j}} = c^*$ or $h_{\mathrm{j}} + d_{\mathrm{j}} = h_{\mathrm{j}'}$ for some $\mathrm{j}' \in J'$ such that $P_{\mathrm{j}} \cap P_{\mathrm{j}'} \neq \emptyset$.

▶ **Lemma 11.** *Consider a valid packing of a set of jobs $J' \subseteq J_{\text{large}}$ inside one round. Then there is also a packing for J' that is normalized.*

Now the important insight is that in a normalized packing of large jobs, the height h_{j} of a job j is the difference of (the top height level) c^* and the sum of at most $1/\varepsilon^{56}$ jobs in J_{large}. Thus, the number of possible heights is bounded by $n^{O\left(1/\varepsilon^{56}\right)}$ and we can compute all these possible heights before starting our DP.

▶ **Lemma 12.** *Given J_{large} we can compute a set $\mathcal{H}$ of $n^{O(1/\varepsilon^{56})}$ values such that in any normalized packing of a set $J' \subseteq J_{\text{large}}$ inside one round, the height $h_{\mathfrak{j}}$ of each job $\mathfrak{j} \in J'$ is contained in $\mathcal{H}$.*

Now we can compute the optimal packing via a dynamic program as described above, which yields the following lemma.

▶ **Lemma 13.** *Consider an instance of* Round-UFP *or* Round-SAP *with a set of jobs J' satisfying the following conditions:*

(i) *The number of jobs using any edge is bounded by ω.*

(ii) *In the case of* Round-SAP *there is a given set $\mathcal{H}'$ of allowed heights for the jobs.*

Then we can compute an optimal solution to the given instance in time $(n|\mathcal{H}'|)^{O(\omega)}$.

We invoke Lemma 13 with $J' := J_{\text{large}}$, $\omega = 1/\varepsilon^{56}$, and in the case of Round-SAP we define $\mathcal{H}'$ to be the set $\mathcal{H}$ due to Lemma 12. This yields at most OPT_{UFP} rounds in total for the large jobs J_{large}. Hence, we obtain a packing of J using at most $2 \cdot OPT_{UFP} + 1$ rounds.

Case 1 and 2 together imply a packing into at most $(2+\varepsilon)OPT_{UFP} + 1$ rounds. This yields our main theorem for the case of uniform edge capacities.

▶ **Theorem 14.** *For any $\varepsilon > 0$, there exist asymptotic $(2+\varepsilon)$-approximation algorithms for* Round-SAP *and* Round-UFP*, assuming uniform edge capacities.*

We now derive some bounds on the absolute approximation ratios. If $OPT_{SAP} = 1$, our algorithm would return a packing using at most $(2+\varepsilon)\cdot 1 + 1$ rounds, and hence at most 3 rounds. If $OPT_{SAP} \geq 2$, then our algorithm uses at most $(2+\varepsilon)\cdot OPT_{SAP} + OPT_{SAP}/2 = (2.5+\varepsilon)OPT_{SAP}$ rounds. Hence, we obain the following result.

▶ **Theorem 15.** *There exists a polynomial time 3-approximation algorithm for* Round-SAP*, assuming uniform edge capacities.*

For Round-UFP, it is easy to check whether $OPT_{UFP} = 1$ by checking whether $L \leq c^*$. Otherwise, $OPT_{UFP} \geq 2$ and similar as above, the number of rounds used would be at most $(2.5+\varepsilon)\cdot OPT_{UFP}$. This gives an improvement over the result of Pal [49].

▶ **Theorem 16.** *For any $\varepsilon > 0$, there exists a polynomial time $(2.5+\varepsilon)$-approximation algorithm for* Round-UFP*, assuming uniform edge capacities.*

5 General Case

In this section, we present our algorithms for the general cases of Round-UFP and Round-SAP. We begin with our $O(\log\log\min\{n,m\})$-approximation algorithms where we consider Round-UFP first and describe later how to extend our algorithm to Round-SAP. We split the input jobs into large and small jobs. For each job $\mathfrak{j}$, we denote by $\mathfrak{b}_{\mathfrak{j}}$ the minimum capacity of the edges in $P_{\mathfrak{j}}$, i.e., $\min\{c_e : e \in P_{\mathfrak{j}}\}$. We call $\mathfrak{b}_{\mathfrak{j}}$ the *bottleneck capacity* of the job $\mathfrak{j}$. We define $J_{\text{large}} := \{\mathfrak{j} \in J | d_{\mathfrak{j}} > \mathfrak{b}_{\mathfrak{j}}/4\}$ and $J_{\text{small}} := \{\mathfrak{j} \in J | d_{\mathfrak{j}} \leq \mathfrak{b}_{\mathfrak{j}}/4\}$. For the small jobs, we invoke a result by Elbassioni et al. [21] that yields a 16-approximation.

▶ **Theorem 17** ([21])**.** *We are given an instance of* Round-UFP *with a set of jobs J' such that $d_{\mathfrak{j}} \leq \frac{1}{4}\mathfrak{b}_{\mathfrak{j}}$ for each job $\mathfrak{j} \in J'$. Then there is a polynomial time algorithm that computes a 16-approximate solution to J'.*

(a) **(b)**

Figure 2 (a) The capacity profile along with the sets of lines $\mathcal{H}$ and $\mathcal{V}$ and some top-drawn jobs; **(b)** The jobs after processing;

Now consider the large jobs J_{large}. For each job $\mathsf{j} \in J_{\text{large}}$, we define a rectangle $R_{\mathsf{j}} = (s_{\mathsf{j}}, \mathfrak{b}_{\mathsf{j}} - d_{\mathsf{j}}) \times (t_{\mathsf{j}}, \mathfrak{b}_{\mathsf{j}})$. Note that R_{j} corresponds to the rectangle for j in Round-SAP if we assign j the maximum possible height h_{j} (which is $h_{\mathsf{j}} := \mathfrak{b}_{\mathsf{j}} - d_{\mathsf{j}}$). We say that a set of jobs $J' \subseteq J_{\text{large}}$ is *top-drawn* (underneath the capacity profile), if their rectangles are pairwise non-overlapping, i.e., if $R_{\mathsf{j}} \cap R_{\mathsf{j}'} = \emptyset$ for any $\mathsf{j}, \mathsf{j}' \in R_{\mathsf{j}}$. If a set of jobs J' is top-drawn, then it clearly forms a feasible round in Round-UFP. However, not every feasible round of Round-UFP is top-drawn. Nevertheless, we look for a solution to Round-UFP in which the jobs in each round are top-drawn. The following lemma implies that this costs only a factor of 8 in our approximation ratio.

▶ **Lemma 18** ([10]). *Let $J' \subseteq J_{\text{large}}$ be a set of jobs packed in a feasible round for a given instance of Round-UFP. Then J' can be partitioned into at most 8 sets such that each of them is top-drawn.*

Let $\mathcal{R}_{\text{large}} := \{R_{\mathsf{j}} | \mathsf{j} \in J_{\text{large}}\}$ denote the rectangles corresponding to the large jobs and let ω_{large} be their clique number, i.e., the size of the largest set $\mathcal{R}' \subseteq \mathcal{R}_{\text{large}}$ such that all rectangles in $\mathcal{R}'$ pairwise overlap. As a consequence of Helly's theorem [3] for such a set of axis-parallel rectangles $\mathcal{R}'$ there must be a point in which all rectangles in $\mathcal{R}'$ overlap. Note that we need at least ω_{large} rounds since we seek a solution with only top-drawn jobs in each round.

We first reduce the original instance to the case where there are only $O\left(m^2\right)$ many distinct job demands. For each $i \in \{1, \ldots, m\}$, let p_i denote the point (i, c_{e_i}). We draw a horizontal and a vertical line segment passing through p_i and lying completely under the capacity profile (see Figure 2(a)). This divides the region underneath the capacity profile into at most m^2 regions. Let $\mathcal{H}$ denote the set of horizontal lines and $\mathcal{V}$ denote the set of vertical lines drawn. Thus, the top edge of any rectangle corresponding to a large job must touch a line in $\mathcal{H}$. Now consider any rectangle R_{j} corresponding to a job j. Let $h \in \mathcal{H}$ be the horizontal line segment lying just below the bottom edge of R_{j}. We increase the value of d_{j} so that the the bottom edge of R_{j} now touches the line segment h (see Figure 2(b)). Since the rectangles were top-drawn, the clique number of this new set of large rectangles (denoted by $\mathcal{R}'_{\text{large}}$) does not change. Also any feasible packing of $\mathcal{R}'_{\text{large}}$ is a feasible packing of $\mathcal{R}_{\text{large}}$.

Note that $\mathcal{R}'_{\text{large}}$ contains at most m^4 distinct types of jobs: the endpoints s_{j} and t_{j} can be chosen in $\binom{m+1}{2} \leq m^2$ ways and the top and bottom edges of R_{j} must coincide with two lines from $\mathcal{H}$, which can be again chosen in $\binom{m}{2} \leq m^2$ ways. Let U denote the number of types of job of the given instance and let $\mathcal{R}'_{\text{large}} = J_1 \cup J_2 \cup \ldots \cup J_U$ be the decomposition of $\mathcal{R}'_{\text{large}}$ into the U distinct job types.

[3] Project the family of rectangles onto both x and y dimensions. These projections are intervals that are pairwise intersecting. Helly's theorem states that a pairwise intersecting set of intervals share a common point. Then the product of each of these 1 dimensional common intersections is shared by each of the rectangles.

We now formulate the configuration LP for this instance. Let $\mathcal{C}$ denote the set of all possible configurations of a round containing jobs from $\mathcal{R}'_{\text{large}}$, drawn as top-drawn sets. For each $C \in \mathcal{C}$, we introduce a variable x_C, which stands for the number of rounds having configuration $C \in \mathcal{C}$. We write $J_k \lhd C$ if configuration C contains a job from J_k (note that C can contain at most one job from J_k). Then the relaxed configuration LP and its dual (which contains a variable y_k for each set J_k) are as follows.

$$\begin{aligned} &\text{minimize} && \sum_{C \in \mathcal{C}} x_C \\ &\text{subject to} && \sum_{C: J_k \lhd C} x_C \geq |J_k|, \quad k = 1, \dots, U \\ & && x_C \geq 0, \quad C \in \mathcal{C} \end{aligned} \qquad \begin{aligned} &\text{maximize} && \sum_{k=1}^{U} |J_k| y_k \\ &\text{subject to} && \sum_{k: J_k \lhd C} y_k \leq 1, \quad C \in \mathcal{C} \\ & && y_k \geq 0, \quad k = 1, \dots, U \end{aligned}$$

The dual LP can be solved via the ellipsoid method with a suitable separation oracle. We interpret y_k as the weight of each job in J_k. Given $(y_k)_{k \in \{1,\dots,U\}}$, the separation problem asks whether there exists a configuration where jobs are drawn as top-drawn sets and the total weight of all the jobs in the configuration exceeds 1. For this, we invoke the following result of Bonsma et al. [10].

▶ **Theorem 19** ([10]). *Given an instance of UFP with a set of jobs J', the maximum-weight top-drawn subset of J' can be computed in $O\left(nm^3\right)$ time.*

Let $(x^*_C)_{C \in \mathcal{C}}$ be an optimal basic solution of the primal LP. By the rank lemma, there are at most U configurations C for which x^*_C is non-zero. For each non-zero x^*_C, we introduce $\lfloor x^*_C \rfloor$ rounds with configuration C, thus creating at most $8 \cdot OPT_{UFP}$ rounds (due to Lemma 18). Now let $\mathcal{R}''_{\text{large}} \subseteq \mathcal{R}'_{\text{large}}$ be the large jobs that are yet to be packed and let ω''_{large} be their clique number (and note that $\omega''_{\text{large}} \leq 8 \cdot OPT_{UFP}$). In particular, a feasible solution to the configuration LP for the rectangles in $\mathcal{R}''_{\text{large}}$ is to select one more round for each configuration C with $x^*_C > 0$. Therefore, we conclude that $\omega''_{\text{large}} \leq U \leq m^4$ since there are at most U configurations C with $x^*_C > 0$ and for each point, each configuration contains at most one rectangle covering this point.

Our strategy is to invoke the following theorem on $\mathcal{R}''_{\text{large}}$.

▶ **Theorem 20** ([12]). *Given a set of rectangles with clique number ω, in polynomial time, we can compute a coloring of the rectangles using $O(\omega \log \omega)$ colors such that no two rectangles of the same color intersect.*

Thus, if $\omega''_{\text{large}} = O(\log m)$ then we obtain an $O(\log \log m)$-approximation as desired. However, it might be that ω''_{large} is larger. In that case, we partition $\mathcal{R}''_{\text{large}}$ into $\omega''_{\text{large}}/\log m$ sets, such that each of them has a clique size of $O(\log m)$.

▶ **Lemma 21.** *There is a randomized polynomial time algorithm that w.h.p. computes a partition $\mathcal{R}''_{\text{large}} = \mathcal{R}_1 \dot\cup \dots \dot\cup \mathcal{R}_{\omega''_{\text{large}}/\log m}$ such that for each set $\mathcal{R}_i$, the corresponding clique size is at most $O(\log m)$.*

Proof. We split the rectangles $\mathcal{R}''_{\text{large}}$ uniformly at random into $\omega''_{\text{large}}/\log m$ sets $\mathcal{R}_1, \dots, \mathcal{R}_{\omega''_{\text{large}}/\log m}$. Thus the expected clique size in each set $\mathcal{R}_i$ at any point p under the profile is at most $\log m$. Using the Chernoff bound, the probability that the clique size at p is more than $8 \log m$ is at most $2^{-8 \log m} = 1/m^8$. As before, we draw the set of horizontal and vertical lines $\mathcal{H}$ and $\mathcal{V}$, respectively, under the capacity profile, dividing the region underneath the profile into at most m^2 regions. Clearly, the clique number must be the same at all points inside any such region. Thus the probability that there exists a point p under the capacity profile where the clique size is more than $8 \log m$ is at most $m^2/m^8 \leq 1/m^6$. Hence using union bound, probability that clique size is more than $8 \log m$ at some point in some set $\mathcal{R}_i$ is at most $1/m^2$ (since $\omega''_{\text{large}} \leq m^4$). ◀

We apply Theorem 20 to each set $\mathcal{R}_i$ separately and thus obtain a coloring with $O(\log m \log\log m)$ colors. Thus, for all sets $\mathcal{R}_i$ together we use at most $\frac{\omega''_{\text{large}}}{\log m} O(\log m \log\log m) = O(\omega''_{\text{large}} \log\log m)$ colors. We pack the jobs from each color class to a separate round for our solution to Round-UFP. This yields an $O(\log\log m)$-approximation, together with Theorem 17. Since $m \leq 2n-1$ after our preprocessing, our algorithms are also $O(\log\log n)$-approximation algorithms.

▶ **Theorem 22.** *There exists a randomized $O(\log\log\min\{n,m\})$-approximation algorithm for Round-UFP for general edge capacities.*

In order to obtain an algorithm for Round-SAP, we invoke the following lemma due to [45] to each round of the computed solution to Round-UFP.

▶ **Lemma 23** ([45]). *Let J' be the set of jobs packed in a feasible round for a given instance of Round-UFP. Then in polynomial time we can partition J' into $O(1)$ sets and compute a height $h_{\mathfrak{j}}$ for each job $\mathfrak{j} \in J'$ such that each set yields a feasible round of Round-SAP.*

This yields a solution to Round-SAP with only $O(OPT_{UFP} \log\log m) \leq O(OPT_{SAP} \log\log m)$ many rounds.

▶ **Theorem 24.** *There exists a randomized $O(\log\log\min\{n,m\})$-approximation algorithm for Round-SAP for general edge capacities.*

5.1 An $O(\log\log\frac{1}{\delta})$-approximation algorithm with $(1+\delta)$-resource augmentation

We show that if we are allowed a resource augmentation of a factor of $1+\delta$ for some $\delta > 0$, we can get an $O(\log\log\frac{1}{\delta})$-approximation for both Round-SAP and Round-UFP.

Consider Round-UFP first. Recall that $\mathfrak{b}_{\mathfrak{j}}$ denotes the bottleneck capacity of job $\mathfrak{j}$, i.e. $\min\{c_e : e \in P_{\mathfrak{j}}\}$. For each $i \in \mathbb{N}$ we define the set $J^{(i)} := \{\mathfrak{j} \in J \mid \mathfrak{b}_{\mathfrak{j}} \in [1/\delta^i, 1/\delta^{i+1})\}$ and consider one resulting set $J^{(i)}$. By definition no job in $J^{(i)}$ uses an edge whose capacity is less than $1/\delta^i$. Also, we can assume that the capacity of each edge is at most $2/\delta^{i+1}$ since for each edge e with a capacity of more than $1/\delta^{i+1}$, each jobs using it must also use the closest edge on the left or on the right of e with a capacity of at most $1/\delta^{i+1}$. Thus, in a feasible round, e is used by jobs from $J^{(i)}$ with a total demand of at most $2/\delta^{i+1}$. Thus, via scaling we can assume that the edge capacities are in the range $[1, 2/\delta)$ if our input consists of $J^{(i)}$ only.

▶ **Lemma 25.** *Let $J^{(i)} := \{\mathfrak{j} \in J \mid \mathfrak{b}_{\mathfrak{j}} \in [1/\delta^i, 1/\delta^{i+1})\}$. For packing jobs in $J^{(i)}$, it can be assumed that the capacity of each edge lies in the range $[1/\delta^i, 2/\delta^{i+1})$.*

Hence using the following theorem, we get a $O(\log\log\frac{1}{\delta})$-approximate solutions for each $J^{(i)}$, which in particular uses at most $O(OPT_{UFP} \log\log\frac{1}{\delta})$ rounds.

▶ **Theorem 26** ([35]). *There is a polynomial time $O(\log\log\frac{c_{\max}}{c_{\min}})$-approximation algorithm for Round-UFP.*

Next, we argue that we can combine the rounds computed for the sets $J^{(i)}$. More precisely, we show that if we take one round from each set $J^{(0)}, J^{(2)}, J^{(4)}, \ldots$ and form their union, then they form a feasible round for the given instance under $(1+\delta)$-resource augmentation: as argued above, each set $J^{(i)}$ uses at most a capacity of $2/\delta^{i+1}$ on each edge e in any feasible round. Also, if an edge e is used by a job from a set $J^{(i')}$ for an even $i' \in \mathbb{N}$ in a feasible round, then e has a capacity of at least $1/\delta^{i'}$. Therefore, if we have $(1+2\delta)$-resource augmentation available, then by a geometric sum argument the gained capacity on e is enough for one round of each of the sets $J^{(i'-2)}, J^{(i'-4)}, J^{(i'-6)}, \ldots$.

With a similar argumentation we can show that we obtain a feasible solution if we take one round from each set $J^{(1)}, J^{(3)}, J^{(5)}, \ldots$.

▶ **Lemma 27.** *Take one computed round for each set $J^{(2k)}$ with $k \in \mathbb{N}$ or one computed round from each set $J^{(2k+1)}$ with $k \in \mathbb{N}$, and let J' be their union. Then J' is a feasible round for the given instance of* Round-UFP *under $(1+\delta)$-resource augmentation.*

Thus, due to Lemma 27 we obtain a solution with at most $O(OPT_{UFP} \log\log \frac{1}{\delta})$ rounds for the overall instance. As earlier, we take the given Round-UFP solution and apply Lemma 23 to it, which yields a solution to Round-SAP with at most $O(OPT_{SAP} \log\log \frac{1}{\delta})$ rounds.

▶ **Theorem 28.** *There exists an $O(\log\log \frac{1}{\delta})$-approximation algorithm for* Round-SAP *and* Round-UFP *for general edge capacities and $(1+\delta)$-resource augmentation.*

6 Algorithms for the no-bottleneck-assumption

In this section, we present a $(16+\varepsilon)$-approximation algorithm for Round-SAP and a 12-approximation for Round-UFP, both under the no-bottleneck-assumption (NBA).

6.1 Algorithm for Round-SAP

For our algorithm for Round-SAP under NBA, we first scale down all job demands and edge capacities so that $c_{\min} := \min_{e \in E} c_e = 1$ (note that this implies that $d_j \le 1$ for each job $j \in J$). Then, we scale down all edge capacities to the nearest power of 2 and let $(c'_e)_{e \in E}$ denote the new edge capacities. Define a set of horizontal lines $\mathcal{L} := \{\ell_{2^k} | k \in \mathbb{N}\}$. Let OPT'_{SAP} denote the optimal solution for the rounded down capacities $(c'_e)_{e \in E}$ under the additional constraint that there must be no job whose rectangle intersects a line in $\mathcal{L}$.

It can be shown that given a valid Round-SAP packing $\mathcal{P}$ of a set of jobs J' for the edge capacities $(c_e)_{e \in E}$, there exists a valid packing of J' into 4 rounds $\mathcal{R}^1, \mathcal{R}^2, \mathcal{R}^3, \mathcal{R}^4$, under profile $(c'_e)_{e \in E}$ such that no job is sliced by a line in $\mathcal{L}$. This implies that $OPT'_{SAP} \le 4 \cdot OPT_{SAP}$.

We shall now obtain a valid packing of J for the edge capacities $(c'_e)_{e \in E}$. Let $c'_{\max} := \max_{e \in E} c'_e$ and for each $i \in \{0, 1, \ldots, \log c'_{\max}\}$ let $J^{(i)} \subseteq J$ denote the set of jobs with bottleneck capacity 2^i according to $(c'_e)_{e \in E}$. For each set $J^{(i)}$ we create a new (artificial) instance with uniform edge capacities: in the instance for $J^{(0)}$ all edges have capacity 1, and for each $i \in \{1, 2, \ldots, \log c'_{\max}\}$ in the instance for $J^{(i)}$ all edges have capacity 2^{i-1}. For each $i \in \{0, 1, 2, \ldots, \log c'_{\max}\}$, denote by $OPT^{(i)}$ the number of rounds needed in the optimal solution to the instance for $J^{(i)}$. Since in the solution OPT'_{SAP}, no rectangle is intersected by a line in $\mathcal{L}$, for each set $J^{(i)}$ we can easily rearrange the jobs in $J^{(i)}$ in OPT'_{SAP} such that we obtain a solution for $J^{(i)}$ with at most $2 \cdot OPT'_{SAP}$ rounds.

For each set $J^{(i)}$ we invoke our asymptotic $(2+\varepsilon)$-approximation algorithm for Round-SAP for uniform edge capacities (see Section 4) and obtain a solution $\Gamma^{(i)}$ which hence uses $\Gamma^{(i)} \le (2+\varepsilon) \cdot OPT^{(i)} + 1 \le (4 + O(\varepsilon)) \cdot OPT'_{SAP} + 1 \le (16 + O(\varepsilon)) \cdot OPT_{SAP} + 1$ rounds. Finally, we combine the solutions $\Gamma^{(i)}$ for all $i \in \{0, 1, 2, \ldots, \log c'_{\max}\}$ to one global solution of J, which uses at most $\max^{(i)}\{\Gamma^{(i)}\}$ many rounds. Hence we have the following theorem.

▶ **Theorem 29.** *For any $\varepsilon > 0$, there exists a polynomial-time asymptotic $(16+\varepsilon)$-approximation and an absolute 17-approximation algorithm for* Round-SAP *under the NBA.*

6.2 Algorithm for Round-UFP

In this section, we present a 12-approximation for Round-UFP under NBA. In Round-UFP, it is not clear how to bootstrap the algorithm for the uniform case as we did for Round-SAP, since in the optimal solution it might not be possible to draw the jobs as non-overlapping rectangles. Instead, our algorithm refines combinatorial properties from [21] to obtain an improved approximation ratio.

For any edge e, we define the *congestion* $r_e := \lceil l_e/c_e \rceil$, and $r = \max_e r_e$ denotes the maximum congestion of any edge of the path. A result in [48] states that for the special case when all jobs have identical demands and all edge capacities are integral multiples of the demand, a Round-UFP packing using r rounds can be found efficiently.

Via scaling, we assume that $c_{\text{min}} = 1$ and the demand of each job is at most 1. Let $J_{\text{large}} := \{\mathrm{j} \in J \mid d_\mathrm{j} > 1/2\}$ and $J_{\text{small}} := J \setminus J_{\text{large}}$. For jobs in J_{large}, applying the result due to [48] after scaling up all the demands to 1 and scaling down the capacity of each edge to the nearest integer directly yields a packing using at most $4r$ rounds.

Now for each job $\mathrm{j} \in J_{\text{small}}$, we round up its demand to the next larger power of 1/2. For each $i \in \mathbb{N}$, let $J^{(i)}$ denote the set of jobs whose demands after rounding equal $\frac{1}{2^i}$. For each edge e and each $i \in \mathbb{N}$, we define $n_{e,i} := |\{\mathrm{j} \in J^{(i)} \mid e \in P_\mathrm{j}\}|$. We partition each set $J^{(i)}$ into the sets $J'^{(i)} = \{\mathrm{j} \in J^{(i)} \mid \exists e \in P_\mathrm{j} : n_{e,i} < 2r\}$ and $J''^{(i)} = J \setminus J'^{(i)}$. Let $n'_{e,i} := |\{\mathrm{j} \in J'^{(i)} \mid e \in P_\mathrm{j}\}|$ and $n''_{e,i} := |\{\mathrm{j} \in J''^{(i)} \mid e \in P_\mathrm{j}\}|$. Clearly, $n'_{e,i} < 4r$ for each edge e and each i.

Since there are at most $4r$ jobs from each $J'^{(i)}$ over any edge, the jobs in $\bigcup_i J'^{(i)}$ can be easily packed into $4r$ rounds using interval coloring. For the jobs in $\bigcup_i J''^{(i)}$, we partition the available capacity inside each round among the sets $J''^{(i)}$; formally for each edge e, we reserve a capacity of $\frac{1}{2^i} \cdot \left\lfloor \frac{n_{e,i}}{2r} \right\rfloor$ for $J''^{(i)}$ (note $\sum_i \frac{1}{2^i} \left\lfloor \frac{n_{e,i}}{2r} \right\rfloor \le c_e$). The resulting congestion of any edge having non-zero capacity is $\frac{\frac{1}{2^i} n''_{e,i}}{\frac{1}{2^i}\left\lfloor \frac{n_{e,i}}{2r} \right\rfloor} \le \frac{\frac{n_{e,i}}{2r}}{\left\lfloor \frac{n_{e,i}}{2r} \right\rfloor} \cdot 2r \le 4r$. Thus using the algorithm due to [48], jobs in $J''^{(i)}$ can be packed into at most $4r$ rounds with these capacities. Hence we obtain a valid packing of J_{small} using at most $4r + 4r = 8r$ rounds.

We thus have the following theorem.

▶ **Theorem 30.** *There exists a polynomial-time 12-approximation algorithm for* Round-UFP *under the NBA.*

7 Algorithms for Round-Tree

Extending the results for Round-UFP, using results on path coloring [24] and multicommodity demand flow [14], we obtain the following results for Round-Tree (we refer the reader to the full version of this paper for details).

▶ **Theorem 31.** *For* Round-Tree*, there exists a polynomial-time asymptotic (resp. absolute) 5.1- (resp. 5.5-) approximation algorithm for uniform edge capacities and an asymptotic (resp. absolute) 49- (resp. 55-) approximation algorithm for the general case under the NBA.*

References

1 Anna Adamaszek, Sariel Har-Peled, and Andreas Wiese. Approximation schemes for independent set and sparse subsets of polygons. *Journal of the ACM*, 66(4):29:1–29:40, 2019.

2 Udo Adamy and Thomas Erlebach. Online coloring of intervals with bandwidth. In *WAOA*, pages 1–12. Springer, 2003.

3 Aris Anagnostopoulos, Fabrizio Grandoni, Stefano Leonardi, and Andreas Wiese. A mazing $2+\varepsilon$ approximation for unsplittable flow on a path. In *SODA*, pages 26–41, 2014.

4 Matthew Andrews and Lisa Zhang. Bounds on fiber minimization in optical networks with fixed fiber capacity. In *24th Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 1, pages 409–419, 2005.

5 Yossi Azar, Amos Fiat, Meital Levy, and NS Narayanaswamy. An improved algorithm for online coloring of intervals with bandwidth. *Theoretical Computer Science*, 363(1):18–27, 2006.

6 Nikhil Bansal, José R Correa, Claire Kenyon, and Maxim Sviridenko. Bin packing in multiple dimensions: inapproximability results and approximation schemes. *Mathematics of operations research*, 31(1):31–49, 2006.

7 Nikhil Bansal, Zachary Friggstad, Rohit Khandekar, and Mohammad R Salavatipour. A logarithmic approximation for unsplittable flow on line graphs. *ACM Transactions on Algorithms (TALG)*, 10(1):1–15, 2014.

8 Nikhil Bansal and Arindam Khan. Improved approximation algorithm for two-dimensional bin packing. In *SODA*, pages 13–25, 2014.

9 Jatin Batra, Naveen Garg, Amit Kumar, Tobias Mömke, and Andreas Wiese. New approximation schemes for unsplittable flow on a path. In *SODA*, pages 47–58, 2015.

10 Paul S. Bonsma, Jens Schulz, and Andreas Wiese. A constant factor approximation algorithm for unsplittable flow on paths. In *FOCS*, pages 47–56, 2011.

11 Adam L. Buchsbaum, Howard J. Karloff, Claire Kenyon, Nick Reingold, and Mikkel Thorup. OPT versus LOAD in dynamic storage allocation. In *STOC*, pages 556–564, 2003.

12 Parinya Chalermsook and Bartosz Walczak. Coloring and maximum weight independent set of rectangles. In *SODA*, pages 860–868, 2021.

13 Chandra Chekuri and Sanjeev Khanna. A polynomial time approximation scheme for the multiple knapsack problem. *SIAM Journal on Computing*, 35(3):713–728, 2005.

14 Chandra Chekuri, Marcelo Mydlarz, and F. Bruce Shepherd. Multicommodity demand flow in a tree and packing integer programs. *ACM Transactions on Algorithms (TALG)*, 3(3):27, 2007.

15 Miroslav Chlebík and Janka Chlebíková. Complexity of approximating bounded variants of optimization problems. *Theoretical Computer Science*, 354(3):320–338, 2006.

16 Miroslav Chlebík and Janka Chlebíková. Hardness of approximation for orthogonal rectangle packing and covering problems. *Journal of Discrete Algorithms*, 7(3):291–305, 2009.

17 Henrik I. Christensen, Arindam Khan, Sebastian Pokutta, and Prasad Tetali. Approximation and online algorithms for multidimensional bin packing: A survey. *Computer Science Review*, 24:63–79, 2017.

18 Julia Chuzhoy and Shi Li. A polylogarithmic approximation algorithm for edge-disjoint paths with congestion 2. *Journal of the ACM*, 63(5):1–51, 2016.

19 W Fernandez De La Vega and George S. Lueker. Bin packing can be solved within $1+\varepsilon$ in linear time. *Combinatorica*, 1(4):349–355, 1981.

20 Max A Deppert, Klaus Jansen, Arindam Khan, Malin Rau, and Malte Tutas. Peak demand minimization via sliced strip packing. In *APPROX/RANDOM*, volume 207, pages 21:1–21:24, 2021.

21 Khaled M. Elbassioni, Naveen Garg, Divya Gupta, Amit Kumar, Vishal Narula, and Arindam Pal. Approximation algorithms for the unsplittable flow problem on paths and trees. In *FSTTCS*, pages 267–275, 2012.

22 Leah Epstein, Thomas Erlebach, and Asaf Levin. Online capacitated interval coloring. *SIAM Journal on Discrete Mathematics*, 23(2):822–841, 2009.

23 Thomas Erlebach and Klaus Jansen. Off-line and on-line call-scheduling in stars and trees. In *International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 199–213. Springer, 1997.

24 Thomas Erlebach and Klaus Jansen. The complexity of path coloring and call scheduling. *Theoretical Computer Science*, 255(1-2):33–50, 2001.

25 Waldo Gálvez, Fabrizio Grandoni, Afrouz Jabal Ameli, Klaus Jansen, Arindam Khan, and Malin Rau. A tight $(3/2+\epsilon)$ approximation for skewed strip packing. In *APPROX/RANDOM*, pages 44:1–44:18, 2020.

26 Waldo Gálvez, Fabrizio Grandoni, Sandy Heydrich, Salvatore Ingala, Arindam Khan, and Andreas Wiese. Approximating geometric knapsack via l-packings. In *FOCS*, pages 260–271, 2017.

27 Waldo Gálvez, Fabrizio Grandoni, Salvatore Ingala, and Arindam Khan. Improved pseudo-polynomial-time approximation for strip packing. In *FSTTCS*, pages 9:1–9:14, 2016.

28 Waldo Gálvez, Fabrizio Grandoni, Arindam Khan, Diego Ramírez-Romero, and Andreas Wiese. Improved approximation algorithms for 2-dimensional knapsack: Packing into multiple l-shapes, spirals, and more. In *SoCG*, volume 189, pages 39:1–39:17, 2021.

29 Waldo Gálvez, Arindam Khan, Mathieu Mari, Tobias Mömke, Madhusudhan Reddy, and Andreas Wiese. A $(2+\varepsilon)$-approximation algorithm for maximum independent set of rectangles. *arXiv preprint*, 2021. `arXiv:2106.00623`.

30 Jordan Gergov. Algorithms for compile-time memory optimization. In *SODA*, pages 907–908, 1999.

31 Fabrizio Grandoni, Tobias Mömke, and Andreas Wiese. A PTAS for unsplittable flow on a path. In *STOC*, pages 289–302, 2022.

32 Fabrizio Grandoni, Tobias Mömke, and Andreas Wiese. Unsplittable flow on a path: The game! In *SODA*, pages 906–926, 2022.

33 Fabrizio Grandoni, Tobias Mömke, Andreas Wiese, and Hang Zhou. To augment or not to augment: Solving unsplittable flow on a path by creating slack. In *SODA*, pages 2411–2422, 2017.

34 Fabrizio Grandoni, Tobias Mömke, Andreas Wiese, and Hang Zhou. A $(5/3 + \epsilon)$-approximation for unsplittable flow on a path: placing small tasks into boxes. In *STOC*, pages 607–619, 2018.

35 Hamidreza Jahanjou, Erez Kantor, and Rajmohan Rajaraman. Improved algorithms for scheduling unsplittable flows on paths. In *ISAAC*, pages 49:1–49:12, 2017.

36 Klaus Jansen, Arindam Khan, Marvin Lira, and K. V. N. Sreenivas. A PTAS for packing hypercubes into a knapsack. In *ICALP*, volume 229, pages 78:1–78:20, 2022.

37 Klaus Jansen and Guochuan Zhang. On rectangle packing: maximizing benefits. In *SODA*, pages 204–213, 2004.

38 Arindam Khan. *Approximation algorithms for multidimensional bin packing.* PhD thesis, Georgia Institute of Technology, 2015.

39 Arindam Khan, Aditya Lonkar, Arnab Maiti, Amatya Sharma, and Andreas Wiese. Tight approximation algorithms for two-dimensional guillotine strip packing. In *ICALP*, volume 229, pages 80:1–80:20, 2022.

40 Arindam Khan, Arnab Maiti, Amatya Sharma, and Andreas Wiese. On guillotine separable packings for the two-dimensional geometric knapsack problem. In *SoCG*, volume 189, pages 48:1–48:17, 2021.

41 Arindam Khan and Madhusudhan Reddy Pittu. On guillotine separability of squares and rectangles. In *APPROX/RANDOM*, pages 47:1–47:22, 2020.

42 Arindam Khan and Eklavya Sharma. Tight approximation algorithms for geometric bin packing with skewed items. In *APPROX/RANDOM*, volume 207, pages 22:1–22:23, 2021.

43 Arindam Khan and Mohit Singh. On weighted bipartite edge coloring. In *FSTTCS*, pages 136–150, 2015.

44 Joseph SB Mitchell. Approximating maximum independent set for rectangles in the plane. In *FOCS*, pages 339–350, 2022.

45 Tobias Mömke and Andreas Wiese. A (2+ε)-approximation algorithm for the storage allocation problem. In *ICALP*, pages 973–984, 2015.

46 Tobias Mömke and Andreas Wiese. Breaking the barrier of 2 for the storage allocation problem. In *ICALP*, pages 86:1–86:19, 2020.

47 NS Narayanaswamy. Dynamic storage allocation and on-line colouring interval graphs. In *COCOON*, pages 329–338. Springer, 2004.

48 Christos Nomikos, Aris Pagourtzis, and Stathis Zachos. Routing and path multicoloring. *Inf. Process. Lett.*, 80(5):249–256, 2001.

49 Arindam Pal. Approximation algorithms for covering and packing problems on paths. *arXiv preprint*, 2014. `arXiv:1402.1107`.

50 Arka Ray. There is no aptas for 2-dimensional vector bin packing: Revisited. *arXiv preprint*, 2021. `arXiv:2104.13362`.

51 Peter Winkler and Lisa Zhang. Wavelength assignment and generalized interval graph coloring. In *SODA*, pages 830–831, 2003.

52 Gerhard J Woeginger. There is no asymptotic ptas for two-dimensional vector packing. *Information Processing Letters*, 64(6):293–297, 1997.