Algorithmica (2023) 85:3649–3679
https://doi.org/10.1007/s00453-023-01152-w

Peak Demand Minimization via Sliced Strip Packing

Max A. Deppert[1] · Klaus Jansen[1] · Arindam Khan[2] · Malin Rau[3] · Malte Tutas[1]

Received: 31 March 2022 / Accepted: 9 June 2023 / Published online: 27 July 2023

Abstract

We study the Non-preemptive Peak Demand Minimization (NPDM) problem, where we are given a set of jobs, specified by their processing times and energy requirements. The goal is to schedule all jobs within a fixed time period such that the peak load (the maximum total energy requirement at any time) is minimized. This problem has recently received significant attention due to its relevance in smart-grids. Theoretically, the problem is related to the classical strip packing problem (SP). In SP, a given set of axis-aligned rectangles must be packed into a fixed-width strip, such that the height of the strip is minimized. NPDM can be modeled as strip packing with slicing and stacking constraint: each rectangle may be cut vertically into multiple slices and the slices may be packed into the strip as individual pieces. The stacking constraint forbids solutions where two slices of the same rectangle are intersected by the same vertical line. Non-preemption enforces the slices to be placed in contiguous horizontal locations (but may be placed at different vertical locations). We obtain a $(5/3 + \varepsilon)$-approximation

An extended abstract of this paper was published in APPROX 2021.
Max A. Deppert, Klaus Jansen, Malte Tutas: Research supported by German Research Foundation (DFG) Projects JA 612/20-1, JA 612/25-1. Arindam Khan: Research supported by Pratiksha Trust Young Investigator Award, Google India Research Award, and Google ExploreCS Award.

✉ Max A. Deppert
made@informatik.uni-kiel.de

Klaus Jansen
kj@informatik.uni-kiel.de

Arindam Khan
arindamkhan@iisc.ac.in

Malin Rau
rau@informatik.uni-hamburg.de

Malte Tutas
mtu@informatik.uni-kiel.de

1 Kiel University, Kiel, Germany

2 Department of Computer Science and Automation, Indian Institute of Science, Bengaluru, India

3 Universität Hamburg, Hamburg, Germany

algorithm for the problem. We also provide an asymptotic efficient polynomial-time approximation scheme (AEPTAS) which generates a schedule for almost all jobs with energy consumption $(1+\varepsilon)$OPT. The remaining jobs fit into a thin container of height 1. This AEPTAS is used as a subroutine to acquire the $(5/3+\varepsilon)$-approximation algorithm. The previous best result for NPDM was a 2.7-approximation based on FFDH (Ranjan et al., in: 2015 IEEE symposium on computers and communication (ISCC), pp 758–763, IEEE, 2015). One of our key ideas is providing several new lower bounds on the optimal solution of a geometric packing, which could be useful in other related problems. These lower bounds help us to obtain approximative solutions based on Steinberg's algorithm in many cases. In addition, we show how to split schedules generated by the AEPTAS into few segments and to rearrange the corresponding jobs to insert the thin container mentioned above, such that it does not exceed the bound of $(5/3+\varepsilon)$OPT.

Keywords Peak demand minimization · Scheduling · Approximation · Algorithms

1 Introduction

Recent years have seen a substantial increase in the demand of electricity, due to rapid urbanization, economic growth, and new modes of electrical energy consumption (e.g., electric cars). Traditionally, electricity generation, transmission, and distribution relied on building infrastructure to support the peak load, when the demand for electricity is maximal. However, the peak is rarely achieved, and thus, more demands can be accommodated using the inherent flexibility of scheduling of certain jobs. For example, the energy requirements for HVAC units, electric vehicles, washers and dryers, water heaters, etc. can be met with a flexible scheduling of these appliances. Smart-grids [1–3] are next-generation cyber-physical systems that couple digital communication systems on top of the existing grid infrastructure for such efficient utilization of power, e.g., by shifting users' demand to off-peak hours in order to reduce peak load.

Future smart-grids are expected to obtain demand requirements for a time period and schedule the jobs such that the peak demand is minimized. Recently, this problem has received considerable attention [4–8]. Each job can also be modeled as a rectangle, with desired power demand as height and required running time as width. This gives a geometric optimization problem where the goal is to pack the slices of the rectangles into a strip of width as the time period. The goal is to minimize the maximum height of the packing. There is another additional *stacking constraint* requiring that no vertical line may intersect two slices from the same rectangle.

In this paper, we study this problem known as Non-preemptive Peak Demand Minimization (NPDM)). Formally, we are given a set of jobs $\mathcal{J}$. Each job $j \in \mathcal{J}$ has a processing time $p(j) \in \mathbb{N}$ and an energy requirement $e(j) \in \mathbb{N}$. Furthermore, we are given a deadline $D \in \mathbb{N}$. All the jobs are available from the time 0 and have to be finished before the deadline D. A schedule σ of the jobs $\mathcal{J}$ assigns each job a starting time $\sigma(j) \in \mathbb{N}$ such that it is finished before the deadline, i.e., $c(j) := \sigma(j) + p(j) \leq D$. The total energy consumption at a time $\tau \in \{0, \ldots, D\}$ is

given by $e(\tau) := \sum_{j \in \mathcal{J}, \sigma(j) \leq \tau < \sigma(j)+p(j)} e(j)$. The objective is to minimize the peak of energy consumption at any point in time, i.e., minimize $T_\sigma := \max_{\tau \in \{0,\ldots,D-1\}} e(\tau)$.

NPDM can be viewed as a variant of the strip packing problem, where we are allowed to slice the rectangles vertically, and the slices must be packed in contiguous horizontal positions (but may be placed at different vertical positions). In the classical strip packing problem, we are given a set of rectangles as well as a bounded-width strip, and the objective is to find a non-overlapping, axis-aligned packing of all rectangles into the strip such that the height of the packing is minimized. A simple reduction from the PARTITION problem shows a lower bound of $3/2$ for the ratio of polynomial-time approximation algorithms for the problem. In 1980, Baker et al. [9] first gave a 3-approximation algorithm. Later Coffman et al. [10] introduced two simple shelf-based algorithms: Next Fit Decreasing Height (NFDH), First Fit Decreasing Height (FFDH), with approximation ratios as 3 and 2.7, respectively. Sleator [11] gave a 2.5-approximation. Thereafter, Steinberg [12] and Schiermeyer [13] independently improved the approximation ratio to 2. Afterward, Harren and van Stee [14] obtained a 1.936-approximation. The state-of-the-art approximation ratio is $5/3 + \varepsilon$, due to Harren et al. [15].

Alamdari et al. [4] studied a variant where the preemption of jobs is allowed, also known as two-dimensional strip packing with slicing and stacking constraints (2SP-SSC), or preemptive offline cost optimal scheduling problem (P-OCOSP) [16]. They showed this variant to be NP-hard and obtained an FPTAS. They also studied several shelf-based algorithms and provide a practical polynomial time algorithm that allows only one preemption per job. Ranjan et al. [7] have proposed a practical 4/3-approximation algorithm for this problem.

For NPDM, Tang et al. [1] first proposed a 7-approximation algorithm. Yaw et al. [17] showed that NPDM is NP-hard to approximate within a factor better than $3/2$. They have given a 4-approximation for a special case where all jobs require the same execution time. Ranjan et al. [6] have proposed a 3-approximation algorithm for NPDM. They [16] also proposed an FFDH-based 2.7-approximation algorithm for a mixed variant where some jobs can be preempted, and some can not be preempted.

1.1 Our Contributions

In this paper we obtain improved approximation algorithms for NPDM. Note that optimal solutions of sliced strip packing/NPDM and strip packing can be quite different. In fact, in [18], an example with a ratio of $5/4$ is presented. Thus, the techniques from strip packing do not always translate directly to our problem. We exploit the property that, due to slicing, we can separately guess regions (*profile*) for packing of jobs with large energy demand (*tall* jobs) and jobs with large time requirements (*wide* jobs). We show that we can remove a small set of jobs with large energy demand so that we can approximately guess the optimal profile of jobs with large processing time so that their starting positions come from a set containing a constant number of values. This helps us to show the existence of a structured solution that we can pack near-optimally using linear programs. This shows the existence of an asymptotic efficient polynomial-time approximation scheme (AEPTAS):

Theorem 1 *For any $\varepsilon > 0$, there is an algorithm that schedules all jobs such that the peak load is bounded by $(1+\varepsilon)\mathrm{OPT} + e_{\max}$, where $e_{\max}$ denotes the maximal energy demand among the given jobs. The time complexity of this algorithm is bounded by $\mathcal{O}(n\log(n)) + 1/\varepsilon^{1/\varepsilon^{\mathcal{O}(1/\varepsilon)}}$.*

In fact, we show a slightly stronger result here, providing a schedule for almost all jobs $\mathcal{J} \setminus C$ with peak energy demand bounded by $(1+\mathcal{O}(\varepsilon))\mathrm{OPT}$ plus a schedule for the remaining jobs C with peak energy demand $e_{\max}$ and schedule length εD. Combining this algorithm with a repacking technique, or using Steinberg's algorithm [12] in some sub-cases, we obtain our main result:

Theorem 2 *For any $\varepsilon > 0$, there is a polynomial-time $(5/3+\varepsilon)$-approximation algorithm for NPDM.*

Previously, in strip packing (and related problems), the lower bound on the optimal packing height is given based on the height of the tallest job or the total area of all jobs [12, 13]. One of our main technical contributions is to show several additional lower bounds on the optimal load. These bounds may be helpful in other related geometric problems. In fact, these can be helpful to simplify some of the analyses of previous algorithms. Using these lower bounds, we show, intuitively, that if there is a large amount of energy-consuming jobs (or time-consuming jobs) we can obtain a good packing using Steinberg's algorithm. Otherwise, we start with the packing from AEPTAS and modify it to obtain a packing within $(5/3+\varepsilon)$-factor of the optimum. This repacking utilizes novel insights about the structure of the packing that precedes it, leading to a less granular approach when repacking.

Recently and independently , Gálvez et al. [19] presented similar results using a different methodology.

1.2 Related Work

Strip packing has also been studied under asymptotic approximation. The seminal work of Kenyon and Rémila [20] provided an APTAS with an additive term $O(h_{\max}/\varepsilon^2)$, where $h_{\max}$ is the height of the tallest rectangle. The latter additive term was subsequently improved to $h_{\max}$ by Jansen and Solis-Oba [21]. Pseudo-polynomial time algorithms for strip packing have received recent attention [22–25]. Finally, Jansen and Rau [26] gave an almost tight pseudo-polynomial time $(5/4+\varepsilon)$-approximation algorithm. Recently, Galvez et al. [27] gave a tight $(3/2+\varepsilon)$-approximation algorithm for a special case when all rectangles are skewed (each has either width or height $\leq \delta D$, where $\delta \in (0, 1]$ is a small constant).

A related problem is non-contiguous multiple organization packing [28], where the width of each rectangle represents a demand for a number of concurrent processors. This is similar to sliced strip packing. However, the slices need to be horizontally aligned to satisfy concurrency. Several important scheduling problems are related, such as multiple strip packing [26], malleable task scheduling [29], parallel task scheduling [30], moldable task scheduling [31, 32], etc.

Several geometric packing problems are well-studied in combinatorial optimization. In two-dimensional bin packing, we are given a set of rectangles, and the goal is

to pack all rectangles into the minimum number of unit square bins. This well-studied problem [33, 34] is known not to admit an APTAS [35] unless P=NP. The present best approximation ratio is 1.406 [36]. Another related problem is the two-dimensional geometric knapsack [37, 38], where each rectangle has an associated profit, and we wish to pack a maximum profit subset of rectangles in a given square knapsack. The present best approximation ratio for the problem is 1.89 [39]. These problems are also studied under guillotine cuts [40–42], where all jobs can be cut out by a recursive sequence of end-to-end cuts. There are several other important related problems, such as maximum independent set of rectangles [43], unsplittable flow on a path [44], storage allocation problem [45], etc. We refer the readers to [46] for a survey.

1.3 Notation

While this problem is, at its core, a scheduling problem, a visual representation greatly improves the intuitive understanding of the underlying concepts utilized in the following. As such, from here on out, we will use language reflective of the visualizations. Particularly, as energy demand is represented by a job's height, we will refer to it as such. Similarly, the processing time is represented by the width of a job, and the total work of a job can be seen as the area of the representative rectangle.

For an instance $I = (\mathcal{J}, D)$, we denote by $\mathrm{OPT}(I)$ (or just OPT) the optimal maximum height. For some set of jobs $\mathcal{J}$ we define the area as $\mathrm{area}(\mathcal{J}) = \sum_{i\in\mathcal{J}} w(i)h(i)$, the total width as $w(\mathcal{J}) = \sum_{i\in\mathcal{J}} w(i)$, as well as the total height as $h(\mathcal{J}) = \sum_{i\in\mathcal{J}} h(i)$. We use the additional notation of $\mathcal{J}_{P(i)} = \{i \in \mathcal{J} | P(i)\}$ as a restriction of $\mathcal{J}$ using the predicate P. E.g., we may express the height of jobs of $\mathcal{J}$ which have a width of at least $D/2$ by $h(\mathcal{J}_{w(i)\geq D/2})$. Furthermore, given a set of jobs $\mathcal{J}$, we denote $w_{\max}(\mathcal{J}) := \max_{i\in\mathcal{J}} w(i)$ and $h_{\max}(\mathcal{J}) := \max_{i\in\mathcal{J}} h(i)$ and write $h_{\max}$ and $w_{\max}$ if the set of jobs is clear from the context. We say a job i that is placed at $\sigma(i)$ overlaps any point in time τ if and only if $\sigma(i) \leq \tau < \sigma(i) + w(i)$. The set $\mathcal{J}(\tau)$ denotes the set of jobs that overlap the point in time τ. Additionally, we introduce segments S of the schedule, which refers to time intervals, and containers C which can be seen as sub schedules. The starting point of a time interval S will be denoted by $\sigma(S)$ and its endpoint as $c(S)$. On the other hand, a container C has a length (time), which is denoted as $w(C)$, and a bound on the height $h(C)$. If these containers are scheduled, they get a starting point $\sigma(C)$, which is added to the starting point of any job scheduled in C.

1.4 Steinberg's Algorithm

The following result from [12] will be a crucial subroutine in our algorithms.

Theorem 3 (Steinberg [12]) *Steinberg's algorithm packs a set of rectangular objects $\mathcal{R}$ into a rectangular container of height a and width b in polynomial time, if and only if the following inequalities hold:*

$$h_{\max} \leq a,\ w_{\max} \leq b,\ 2\sum_{r\in\mathcal{R}} h(r)w(r) \leq ab - (2h_{\max} - a)_+(2w_{\max} - b)_+, \quad \text{Steinberg's Cond.}$$

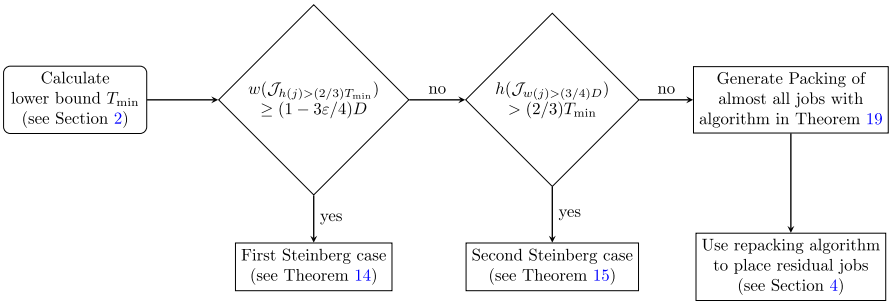


Fig. 1 An overview of the steps of the $(5/3+\varepsilon)$-approximation algorithm

where $x_+ = \max\{x, 0\}$*,* $w_{\max}$ *is the maximal width of a rectangle, and* $h_{\max}$ *is the maximal height of a rectangle,* $h(r)$ *represents the height of a rectangle and* $w(r)$ *represents the width of a rectangle.*

1.5 Overview of the $(5/3+\varepsilon)$-approximation

In the following, we present an overview of the steps of the $(5/3+\varepsilon)$-approximation algorithm, see Fig. 1. In the first step, we calculate a lower bound $T_{\min}$ on OPT using novel ideas, which can be found in Sect. 2. This lower bound guarantees that $T_{\min} \leq \text{OPT} \leq 2T_{\min}$.

If the jobs that are large in at least one of the two dimensions have a sufficiently large total area (with regard to the lower bound $T_{\min}$), a $(5/3+\varepsilon)$-approximation can be achieved by placing these jobs in a structured manner and using Steinberg's algorithms to place the remaining jobs. We describe two of these cases in Sect. 3.

Otherwise, we know that the total area of these jobs is not too large. In this case, we find a schedule σ_1 using the algorithm from Theorem 19 that schedules almost all the jobs resulting in a height of at most $(1+\mathcal{O}(\gamma))\text{OPT}$. The remaining jobs are contained in an extra schedule σ_2 of length γD and maximum height bounded by $T_{\min}$. How to find these two schedules is described in Sect. 5. Note that we have to choose $\gamma \in \mathcal{O}_\varepsilon(1)$ small enough, in order to meet the requirements for the next step.

Given these schedules, we present a rescheduling argument where we rearrange the schedule σ_1 such that we can add the schedule σ_2 while increasing the maximum height by at most $(2/3)\text{OPT}$. This rescheduling argument is described in Sect. 4 in the proof of Theorem 16.

2 Finding a Lower Bound on OPT

In this section, we first find a lower bound $T_{\min}$ on the optimal schedule height and then use Steinberg's algorithm to handle some types of instances.

There are three well-known lower bounds. First, the height of the tallest job $h_{\max} =: T_1$. Second, the bound given by the total area of the jobs, i.e., $\text{OPT} \geq \max\{h_{\max}, \text{area}(\mathcal{J})/D\} =: T_2$. Another simple lower bound is the total height of jobs wider than $D/2$ since they cannot be placed next to each other. We denote this lower bound as $T_3 := h(\mathcal{J}_{w(i)>D/2})$. We name the combination of these straightforward lower bounds $T_{\text{simple}} := \max\{T_1, T_2, T_3\}$. In the following, we will present three more complex lower bounds.

2.1 Fourth Bound

The bound described in this section depends on the combination of jobs with large heights.

Lemma 4 *It holds that* $w(\mathcal{J}_{h(i)>(1/3)\text{OPT}}) + w(\mathcal{J}_{h(i)>(2/3)\text{OPT}}) \leq 2D$ *and* $w(\mathcal{J}_{h(i)>(1/2)\text{OPT}}) \leq D$.

Proof Note that jobs with a height greater than $(1/3)\text{OPT}$ cannot intersect the same vertical line as jobs with a height greater than $(2/3)\text{OPT}$ in an optimal schedule. Furthermore, each vertical line through an optimal schedule can intersect at most two jobs with a height greater than $(1/3)\text{OPT}$. Moreover, no vertical line can intersect two jobs from the set $\mathcal{J}_{h(i)>(1/2)\text{OPT}}$. The claim is a consequence. □

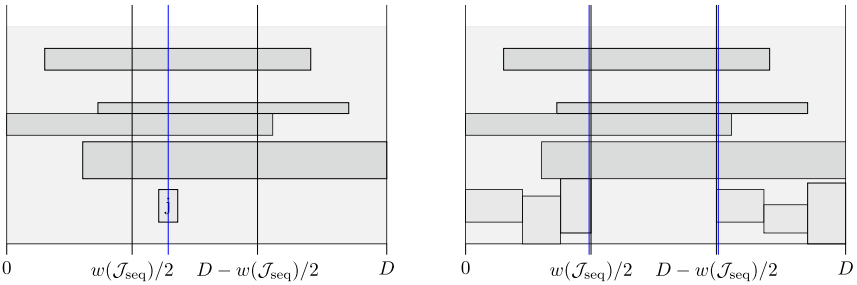
Corollary 5 (Lower bound T_4) *The value* $T_4 := \max\{T_{\text{simple}}, T_4'\}$ *is a lower bound on* OPT *where* T_4' *is the smallest value such that* $w(\mathcal{J}_{h(i)>T_4'/3}) + w(\mathcal{J}_{h(i)>(2/3)T_4'}) \leq 2D$ *and* $w(\mathcal{J}_{h(i)>T_4/2}) \leq D$.

Proof By Lemma 4, we know that $w(\mathcal{J}_{h(i)>(1/3)\text{OPT}}) + w(\mathcal{J}_{h(i)>(2/3)\text{OPT}}) \leq 2D$, and obviously we have $w(\mathcal{J}_{h(i)>\text{OPT}/2}) \leq D$. Therefore, the smallest value T_4' satisfying both inequalities, i.e. $w(\mathcal{J}_{h(i)>(1/3)T_4'}) + w(\mathcal{J}_{h(i)>(2/3)T_4'}) \leq 2D$ and $w(\mathcal{J}_{h(i)>T/2}) \leq D$, has to be a lower bound on OPT. Since T_{simple} and T_4' both are lower bounds on OPT, we know that T_4 is a lower bound as well. □

Note that we can find T_4 in $\mathcal{O}(n \log n)$ with the following algorithm:

- Start with $T_4 = T_{\text{simple}}$.
- As long as $w(\mathcal{J}_{h(i)\geq(1/3)T_4}) + w(\mathcal{J}_{h(i)\geq(2/3)T_4}) > 2D$ or $w(\mathcal{J}_{h(i)>T_4/2}) > D$ update T_4 as follows:
- For $\ell \in [0, 1]$, denote by h_ℓ the height of the smallest job in $\mathcal{J}_{h(i)>\ell \cdot T_4}$ and set $T_4 := \min\{3h_{1/3}, (3/2)h_{2/3}, 2h_{1/2}\}$. This iteratively excludes one job from one of the three sets.

As a consequence of this lower bound, we can link the total width of the jobs with a height greater than $(2/3)T_4$ to the total width of the jobs with a height between $(1/3)T_4$ and $(2/3)T_4$. We express this connection in the following lemma:



(a) An item from $\mathcal{J}_{\text{seq}}$ is scheduled in the central strip.

(b) No item from $\mathcal{J}_{\text{seq}}$ is scheduled in the central strip.

Fig. 2 An illustration of Lemma 7

Lemma 6 *Let* $w \in [0, 1/2)$. *Then*

$$w(\mathcal{J}_{h(i)>(2/3)T_4}) > (1-w)D \Rightarrow w(\mathcal{J}_{h(i)\in((1/3)T_4,(2/3)T_4]}) \leq 2wD.$$

Proof We know that $w(\mathcal{J}_{h(i)>(1/3)T_4}) + w(\mathcal{J}_{h(i)>(2/3)T_4}) \leq 2D$. Since we have $\mathcal{J}_{h(i)>(2/3)T_4} \subseteq \mathcal{J}_{h(i)>(1/3)T_4}$ and $w(\mathcal{J}_{h(i)>(2/3)T_4}) > (1-w)D$ it holds that $w(\mathcal{J}_{h(i)\in((1/3)T_4,(2/3)T_4]}) \leq 2wD$. □

2.2 Fifth Bound

Next, we obtain a bound based on a set of jobs that do not overlap vertically in a given optimal schedule (Fig. 2).

Lemma 7 *Consider an optimal schedule and let* $\mathcal{J}_{\text{seq}}$ *be a set of jobs such that no pair of jobs* $i, i' \in \mathcal{J}_{\text{seq}}$ *overlaps vertically, i.e.,* $\sigma(i) + w(i) \leq \sigma(i')$ *or* $\sigma(i') + w(i') \leq \sigma(i)$. *Furthermore, define* $\mathcal{J}_w := \mathcal{J}_{w(i)>(D-w(\mathcal{J}_{\text{seq}})/2)} \setminus \mathcal{J}_{\text{seq}}$. *Then there exists a vertical line through the schedule that intersects a job in* $\mathcal{J}_{\text{seq}}$ *and all the jobs in* $\mathcal{J}_w$.

Proof First note that $(D - w(\mathcal{J}_{\text{seq}})/2) \geq D/2$. Consider the vertical strip between $w(\mathcal{J}_{\text{seq}})/2$ and $(D - w(\mathcal{J}_{\text{seq}})/2)$. Each job in $\mathcal{J}_w$ completely overlaps this strip. Furthermore, either the strip itself contains a job in $\mathcal{J}_{\text{seq}}$, in which case the claim is trivially true, or on each position on both sides of the strip, there is a job from $\mathcal{J}_{\text{seq}}$. Assume the latter case. Since the jobs in $\mathcal{J}_w$ are strictly wider than $(D - w(\mathcal{J}_{\text{seq}})/2)$, there exists an $\sigma > 0$ such that the vertical line at $(D - w(\mathcal{J}_{\text{seq}})/2) + \sigma$ as well is overlapped by all the jobs in this set. Since this line also intersects a job from the set $\mathcal{J}_{\text{seq}}$, the claim follows. □

Corollary 8 *Let* $\mathcal{J}_{\text{seq}}$ *be a set of jobs such that* $w(\mathcal{J}_{\text{seq}}) \leq D$ *and consider* $\mathcal{J}_w := \mathcal{J}_{w(i)>D-w(\mathcal{J}_{\text{seq}})/2} \setminus \mathcal{J}_{\text{seq}}$. *Furthermore, let* $j_\perp \in \mathcal{J}_{\text{seq}}$ *be the job with the smallest height. Then it holds that* $\min\{h(j_\perp) + h(\mathcal{J}_w), 2h(j_\perp)\} \leq \text{OPT}$.

Proof Consider an optimal solution. We have to consider two cases. In the first case, two jobs from the set $\mathcal{J}_{\text{seq}}$ intersect the same vertical line. In this case, $2h(j_\perp)$ is obviously a lower bound on OPT.

On the other hand, if in any optimal schedule, there does not exist a pair of jobs from $\mathcal{J}_{\text{seq}}$ that overlap the same vertical line, we know by Lemma 7 that there exists a job in $\mathcal{J}_{\text{seq}}$ that overlaps with all the jobs in $\mathcal{J}_w$ and therefore $\text{OPT} \geq h(j_\perp) + h(\mathcal{J}_w)$ in this case. □

2.3 Lower Bound T_5

From Corollary 8, we derive a lower bound on OPT. For a given $k \in [n]$, we define $\mathcal{J}_k$ to be the set of the k jobs with the greatest height in $\mathcal{J}$ and $\mathcal{J}'_k$ to be the set of the k jobs with the greatest height in $\mathcal{J} \setminus \mathcal{J}_{w(i)>D/2}$. Let i_k and i'_k be the jobs with the smallest height in $\mathcal{J}_k$ and $\mathcal{J}'_k$, respectively. We define

$$T_{5,a} := \max_{k \leq n} \{\min\{h(i_k) + h(\mathcal{J}_{w(i)>D-w(\mathcal{J}_k)/2} \setminus \mathcal{J}_k), 2h(i_k)\} | w(\mathcal{J}_k) \leq D\},$$
$$T_{5,b} := \max_{k \leq n} \{\min\{h(i'_k) + h(\mathcal{J}_{w(i)>D-w(\mathcal{J}'_k)/2}), 2h(i'_k)\} | w(\mathcal{J}'_k) \leq D\},$$

and finally $T_5 = \max\{T_{5,a}, T_{5,b}\}$. Note that $\mathcal{J}'_k$ and $\mathcal{J}_{w(i)>D-w(\mathcal{J}'_k)/2}$ are disjoint, since $\mathcal{J}'_k$ contains only jobs with width at most $D/2$ and $\mathcal{J}_{w(i)>D-w(\mathcal{J}'_k)/2}$ contains only jobs with width larger than $D/2$, and hence, by Corollary 8, T_5 is a lower bound on OPT.

For this lower bound, we prove the following property.

Lemma 9 (Bounded height of wide jobs) *Let* $T = \max\{T_{\text{simple}}, T_4, T_5\}$, $w \in (0, 1/2)$ *and* $h \in (1/2, 1]$ *as well as* $\mathcal{J}_h := \mathcal{J}_{h(i)\geq hT}$ *and* $\mathcal{J}_{\text{wide}} := \mathcal{J}_{w(i)>(1/2+w/2)D} \setminus \mathcal{J}_h$. *It holds that*

$$w(\mathcal{J}_h) \geq (1-w)D \Rightarrow h(\mathcal{J}_{\text{wide}}) \leq (1-h)T.$$

Proof Let j be a job in $\mathcal{J}_h$. As such, it has a height of at least hT. Since $T \geq T_4$, it holds that $w(\mathcal{J}_h) \leq D$. By construction of T_5 for j, it holds that

$$h(j) + h(\mathcal{J}_{w(i)>D-w(\mathcal{J}_{h(i)\geq h(j)})/2} \setminus \mathcal{J}_{h(i)\geq h(j)}) \leq T_5.$$

Furthermore, note that $\mathcal{J}_h = \mathcal{J}_{h(i)\geq h(j)}$ for the job j with the smallest height in $\mathcal{J}_h$. Therefore, if $w(\mathcal{J}_h) \geq (1-w)D$, it holds that

$$\mathcal{J}_{w(i)>D-(1-w)D/2} \subseteq \mathcal{J}_{w(i)>D-w(\mathcal{J}_h)/2}$$

and hence,

$$\begin{aligned} hT + h(\mathcal{J}_{\text{wide}}) &= hT + h(\mathcal{J}_{w(i)>D-(1-w)D/2} \setminus \mathcal{J}_h) \\ &\leq h(j) + h(\mathcal{J}_{w(i)>D-w(\mathcal{J}_h)/2} \setminus \mathcal{J}_h) \leq T_5 \leq T \end{aligned}$$

which proves the claim. □

Lemma 10 *Let* $T = \max\{T_{\text{simple}}, T_4, T_5\}$, $w \in (1/2, 1]$ *and* $h \in (1/2, 1]$ *as well as* $\mathcal{J}_w := \mathcal{J}_{w(i)\geq wD}$ *and* $\mathcal{J}_h := \mathcal{J}_{h(i)>hT} \setminus \mathcal{J}_{w(i)>D/2}$. *It holds that*

$$h(\mathcal{J}_w) > (1-h)T \Rightarrow w(\mathcal{J}_h) \leq 2(1-w)D.$$

Proof Let $h(\mathcal{J}_w) > (1-h)T$. Since for each job in $\mathcal{J}_h$ it holds that $h(i) > T/2 \geq T_{3,b}/2$, by definition of $T_{3,b}$, for each $j \in \mathcal{J}_h$ it holds that

$$h(j) + h(\mathcal{J}_{w(i)>D-w(\mathcal{J}_{h(i)\geq h(j)})/2}) \leq T.$$

Therefore for the smallest job $j \in \mathcal{J}_h$, it holds that

$$h(j) + h(\mathcal{J}_{w(i)>D-w(\mathcal{J}_h)/2}) \leq T.$$

For contradiction assume that $w(\mathcal{J}_h) > 2(1-w)D$. Note that in this case $D - w(\mathcal{J}_h)/2 < wD$ and hence,

$$h(\mathcal{J}_{w(i)>D-w(\mathcal{J}_h)/2}) \geq h(\mathcal{J}_D) > (1-h)T.$$

As a consequence,

$$h(j) + h(\mathcal{J}_{w(i)>D-w(\mathcal{J}_h)/2}) > hT + (1-h)T = T,$$

a contradiction. □

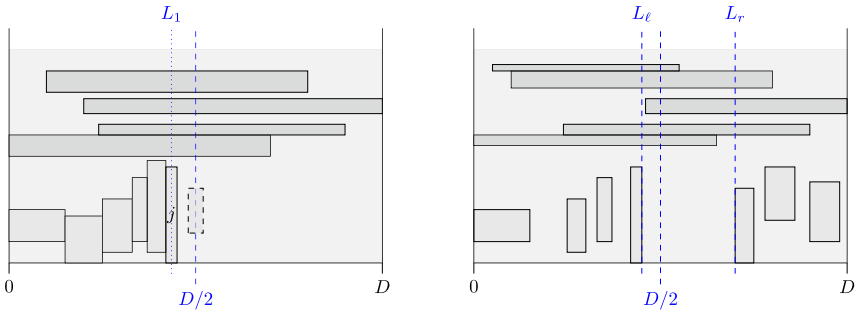
2.4 Sixth Bound

In this section, we present our final lower bound on OPT.

Lemma 11 *Consider an optimal schedule and let* $\mathcal{J}_{\text{seq}} \subseteq \mathcal{J}$ *be a set of jobs such that no pair of jobs* $j, j' \in \mathcal{J}_{\text{seq}}$ *overlaps vertically, i.e.,* $\sigma(j) + w(j) \leq \sigma(j')$ *or* $\sigma(j') + w(j') \leq \sigma(j)$. *Let* $\mathcal{J}_D \subseteq \mathcal{J}_{w(j)>(\max\{D-w(\mathcal{J}_{\text{seq}}), D/2\})} \setminus \mathcal{J}_{\text{seq}}$. *Then, there exists a vertical line through the schedule that intersects a job in* $\mathcal{J}_{\text{seq}}$ *and a subset* $\mathcal{J}'_D \subseteq \mathcal{J}_D$ *with* $h(\mathcal{J}'_D) \geq h(\mathcal{J}_D)/2$.

Proof First, we consider the trivial cases. If a job from $\mathcal{J}_{\text{seq}}$ overlaps the vertical line at $D/2$ the claim is trivially true, since all the jobs from $\mathcal{J}_D$ overlap $D/2$. On the other hand, if all the jobs in $\mathcal{J}_{\text{seq}}$ are left or right of $D/2$, it holds that $w(\mathcal{J}_{\text{seq}}) \leq D/2$ and one of the jobs has a distance of at most $D/2 - w(\mathcal{J}_{\text{seq}})$ from $D/2$. This job has to be overlapped by all the jobs from $\mathcal{J}_D$ since they have a width larger than $D - w(\mathcal{J}_{\text{seq}})$.

Otherwise, consider the vertical line L_ℓ through the right border of the rightmost job from $\mathcal{J}_{\text{seq}}$ that is left of $D/2$ and the vertical line L_r through the left border of the leftmost job from $\mathcal{J}_{\text{seq}}$ that is right of $D/2$. Note that L_ℓ and L_r have a distance of at most $(D - w(\mathcal{J}_{\text{seq}}))$. Consider the set $\mathcal{J}_{D,\ell} \subseteq \mathcal{J}_D$ that is intersected by the vertical line L_ℓ. Note that the remaining jobs in $\mathcal{J}_{D,r} := \mathcal{J}_D \setminus \mathcal{J}_{D,\ell}$ all overlap the vertical line



(a) An item from $\mathcal{J}_{\text{seq}}$ intersects $D/2$ or all items from $\mathcal{J}_{\text{seq}}$ are scheduled left of $D/2$.

(b) Not all items on the same side of $D/2$ or intersecting it.

Fig. 3 An illustration of Lemma 11

at $L_\ell + (D - w(\mathcal{J}_{\text{seq}})) \geq L_r$ and hence L_r as well. Since $\mathcal{J}_{D,r} \cup \mathcal{J}_{D,\ell} = \mathcal{J}_D$, one of the two sets has a height of at least $h(\mathcal{J}_D)/2$. Finally, note that there exists a small enough $\sigma > 0$ such that $L_\ell - \sigma$ and $L_r + \sigma$ overlap the same set of wide jobs as L_ℓ and L_r as well as the corresponding job in $\mathcal{J}_{\text{seq}}$ (Fig. 3). □

Corollary 12 *Let $\mathcal{J}_{\text{seq}}$ be a set of jobs such that $w(\mathcal{J}_{\text{seq}}) \leq D$ and consider $\mathcal{J}_D := \mathcal{J}_{w(i)>(\max\{D-w(\mathcal{J}_{\text{seq}}),D/2\})} \setminus \mathcal{J}_{\text{seq}}$. Furthermore, let $j_\perp \in \mathcal{J}_{\text{seq}}$ be the job with the smallest height. Then it holds that* $\min\{h(j_\perp) + h(\mathcal{J}_D)/2, 2h(j_\perp)\} \leq \text{OPT}$.

Proof Consider an optimal solution. We have to consider two cases. In the first case, two jobs from the set $\mathcal{J}_{\text{seq}}$ intersect the same vertical line. In this case, $2h(j_\perp)$ is obviously a lower bound on OPT.

On the other hand, if there does not exist a pair of jobs from $\mathcal{J}_{\text{seq}}$ that overlap the same vertical line in any optimal schedule, we know by Lemma 11 that there exists a job in $\mathcal{J}_{\text{seq}}$ that overlaps with a set $\mathcal{J}'_D \subseteq \mathcal{J}_D$ such that $h(\mathcal{J}'_D) \geq h(\mathcal{J}_D)/2$ and therefore we have $\text{OPT} \geq h(j_\perp) + h(\mathcal{J}_D)/2$ in this case. □

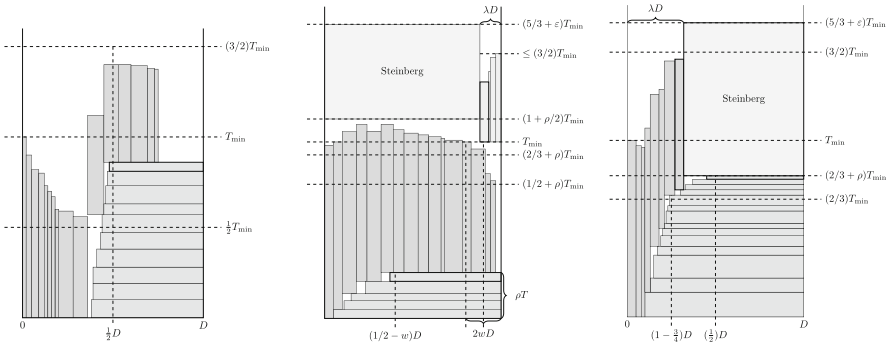
2.5 Lower Bound T_6

Define $\mathcal{J}_k$ as the set of the k jobs with largest height, $\mathcal{J}_{D,k} := \mathcal{J}_{w(i)>(\max\{D-w(\mathcal{J}_k),D/2\})} \setminus \mathcal{J}_k$. Let j_k be the job with the smallest height in $\mathcal{J}_k$, then define

$$T_6 := \max\{\min\{2h(j_k), h(j_k) + h(\mathcal{J}_{D,k})/2\} | k \in \{1, \ldots, n\}, w(\mathcal{J}_k) \leq D\}.$$

By Corollary 12, T_6 is a lower bound for OPT.

Note that in the following, we will denote the largest lower bound on the optimum, i.e., the minimum possible height of the schedule, as $T_{\min} = \max\{T_{\text{simple}}, T_4, T_5, T_6\}$.



(a) An L-shaped schedule (b) First Steinberg case (c) Second Steinberg case

Fig. 4 **a** One possible L-shaped schedule, where $\mathcal{J}_{\text{seq}}$ contains all the jobs with height larger than $T_{\min}/2$ and $\mathcal{J}_D$ contains all the jobs with width larger than $D/2$. **b** A schedule in the case that $w(\mathcal{J}_{h(i)>(2/3)T_{\min}}) \geq (1-w)D$, the first Steinberg case. **c** A schedule in the case that $h(\mathcal{J}_{w(i)>(3/4)D}) \geq (2/3)T_{\min}$, the second Steinberg case. Note that in this figure, we represent all jobs as rectangles since shifting them to the bottom will not decrease the peak energy demand

3 Instances Solved with Steinberg's Algorithm

In this section, we present two cases that can be solved by scheduling jobs that are large in at least one of the two dimensions in a sorted manner while the other jobs are scheduled using Steinberg's algorithm.

The sorted part uses the following shape. Given two disjoint sets of jobs $\mathcal{J}_{seq}$ and $\mathcal{J}_D$, we say they are placed *L-shaped*, if the jobs $i \in \mathcal{J}_D$ are placed such that $\sigma(i) + w(i) = D$, while the jobs in $\mathcal{J}_{seq}$ are sorted by height and placed left-aligned tallest to the left, see Fig. 4a.

Lemma 13 (L-shaped schedule) *Let* $T_{\min} = \max\{T_{\text{simple}}, T_4, T_5, T_6\}$. *If we place* $\mathcal{J}_{\text{seq}} := \mathcal{J}_{h(i)>T_{\min}/2}$ *and* $\mathcal{J}_D := \mathcal{J}_{w(i)>D/2} \setminus \mathcal{J}_{\text{seq}}$ *L-shaped, the schedule has a height of at most* $T_{\min} + h(\mathcal{J}_D)/2 \leq (3/2)\text{OPT}$.

Proof Consider a vertical line L through the generated schedule. If L does not intersect a job from $\mathcal{J}_{\text{seq}}$, the intersected jobs have a height of at most $h(\mathcal{J}_D) \leq T_{\min}$. Otherwise, let $i_L \in \mathcal{J}_{\text{seq}}$ and $\mathcal{J}_{W,L} \subseteq \mathcal{J}_D$ be the jobs intersected by L and define $\mathcal{J}_{\text{seq},L} := \mathcal{J}_{h(i)\geq h(i_L)}$. Note that by definition of the schedule, it holds that $\mathcal{J}_{W,L} \subseteq \mathcal{J}_{w(i)>(\max\{D-w(\mathcal{J}_{\text{seq},L}),D/2\})} \setminus \mathcal{J}_{\text{seq},L}$. Since $h(i_L) > T_6/2$, it holds that $T_6 \geq h(i_L) + h(\mathcal{J}_{W,L})/2$, by definition of T_6. As a consequence, $h(i_L) + h(\mathcal{J}_{W,L}) \leq T_{\min} + h(\mathcal{J}_{W,L})/2 \leq T + h(\mathcal{J}_D)/2 \leq (3/2)\text{OPT}$. □

Theorem 14 (First Steinberg Case)

Let $T_{\min}$ *be the lower bound on* OPT *as defined above and* $w \leq (3/4)\varepsilon$.

If $w(\mathcal{J}_{h(j)>(2/3)T_{\min}}) \geq (1-w)D$ *holds, there is a polynomial time algorithm to schedule all jobs with a peak demand (maximum height) at most* $(5/3+\varepsilon)T_{\min}$.

Proof We place jobs that are very wide or very tall in an ordered fashion, while the remaining jobs will be placed using Steinberg's Algorithm, see Fig. 4b. We define

$\mathcal{J}_D := \mathcal{J}_{w(j)>(1/2+w)D} \setminus \mathcal{J}_{h(j)>T_{\min}/2}$ to be the set of jobs with large widths excluding jobs with large heights. We place each job $j \in \mathcal{J}_D$ such that $\sigma(j) = D - w(j)$. All the jobs in $\mathcal{J}_{h(j)>T_{\min}/2}$ are sorted by height and placed left aligned, tallest first inside the schedule. Let $\rho := h(\mathcal{J}_D)/T_{\min}$ and let $h_{(1-2w)D}$ denote the height of the job in $\mathcal{J}_{h(j)>T_{\min}/2}$ at position $(1-2w)D$. Then $h_{(1-2w)D} \geq (2/3)T_{\min}$. By Lemma 9 and the choice of $T_{\min}$, we know that $h_{(1-2w)D} + h(\mathcal{J}_D) \leq T_{\min} \leq \mathrm{OPT}$ and hence $\rho \leq (1/3)$. Let L be a vertical line through the schedule that is at or strictly left of $(1/2 - w)D$ and intersects a job from $\mathcal{J}_{h(j)>T_{\min}/2}$ and all the jobs from $\mathcal{J}_D$. By Lemma 13, at L and left of L, the maximum height of the schedule is bounded by $(1 + \rho/2)T_{\min}$. On the other hand, right of L, the height of the schedule does not increase compared to L. As a consequence, the maximum height in the current schedule is bounded by $(1 + \rho/2)T_{\min} \leq (7/6)T_{\min}$. Furthermore, we know that right of $(1 - 2w)D$, the schedule has a maximum height of at most $T_{\min}$.

Consider the set of jobs $\mathcal{J}_{h(j)\in[(1/3)T,(1/2)T_{\min}]}$. By Lemma 6 we know that $w(\mathcal{J}_{h(j)\in[(1/3)T_{\min},(1/2)T_{\min}]}) \leq 2w \cdot D$, since $\mathcal{J}_{h(j)>(2/3)T_{\min}} \geq (1 - w)D$. Now we consider two cases.

Case A If $\varepsilon \leq \rho/2$, we place all the jobs in $\mathcal{J}_M := \mathcal{J}_{h(j)\in[(1/3)T_{\min},(1/2)T_{\min}]}$ right-aligned next to each other inside the strip. Since they have a height of at most $(1/2)T_{\min}$, and right of $(1 - 2w)D$, the schedule has a maximum height of at most $T_{\min}$, the maximum height of $(5/3)T_{\min}$ is not exceeded after adding these jobs. Define $\lambda := w(\mathcal{J}_M)/D$. Now at each point on the horizontal axis between 0 and $a := (1 - \lambda)D$, the schedule has a height of at most $(1 + \rho/2)T_{\min}$, and therefore, we can use a height of $b := (2/3 - \rho/2 + \varepsilon)T_{\min}$ to place the remaining jobs. Let $\mathcal{J}_{res}$ denote the set of residual jobs that still have to be placed. Note that each job in $\mathcal{J}_{res}$ has a height of at most $(1/3)T_{\min}$ and a width of at most $(1/2 + w)D$, and the total area of these jobs can be bounded by

$$\begin{aligned}\mathrm{area}(\mathcal{J}_{res}) &\leq DT_{\min} - (2/3)T_{\min} \cdot (1 - w)D - \rho T_{\min} \cdot (1/2 + w)D - (1/3)T \cdot \lambda D \\ &= (1/3 + (2/3)w - \rho(1/2 + w) - \lambda/3)DT_{\min},\end{aligned}$$

and hence $2\mathrm{area}(\mathcal{J}_{res}) \leq (2/3 + (4/3)w - \rho(1 + 2w) - (2/3)\lambda)DT_{\min}$. On the other hand, it holds that

$$\begin{aligned}&ab - (2w_{\max} - a)_+(2h_{\max} - b)_+ \\ &\quad= (2/3 - \rho/2 + \varepsilon)T_{\min}(1 - \lambda)D \\ &\qquad- (2(1/2 + w)D - (1 - \lambda)D)_+(2(1/3)T_{\min} - (2/3 - \rho/2 + \varepsilon)T_{\min})_+ \\ &\quad= (2/3 - \rho/2 + \varepsilon - (2/3)\lambda + (\rho/2 - \varepsilon)\lambda - (2w + \lambda)_+(\rho/2 - \varepsilon)_+)DT_{\min} \\ &\quad= (2/3 + \varepsilon(1 + 2w) - (1/2 + w)\rho - (2/3)\lambda)DT_{\min},\end{aligned}$$

since $\rho/2 - \varepsilon \geq 0$. Hence Steinberg's condition is fulfilled if $(4/3)w - \rho(w + 1/2) \leq \varepsilon(1 + 2w)$, which is true since $w \leq (3/4)\varepsilon$.

Case B On the other hand, if $\rho/2 < \varepsilon$, it holds that $(2/3 + \varepsilon - \rho/2)/2 \geq 1/3$, and we consider the set $\mathcal{J}_M := \mathcal{J}_{h(j)\in[((2/3+\varepsilon-\rho/2)/2)T_{\min},(1/2)T_{\min}]}$ instead of the set $\mathcal{J}_{h(j)\in[(1/3)T_{\min},(1/2)T_{\min}]}$, and place it right-aligned. Again, we define $\lambda := w(\mathcal{J}_M)$.

Now, each job in $\mathcal{J}_{res}$ has a height of at most $(1/3+\varepsilon/2-\rho/4)T_{\min}$ and a width of at most $(1/2+w)D$. The total area of these jobs can be bounded by

$$\begin{aligned}\operatorname{area}(\mathcal{J}_{res}) &\leq DT_{\min} - (2/3)T_{\min}\cdot(1-w)D - \rho T_{\min}\cdot(1/2+w)D \\ &\quad - (1/3+\varepsilon/2-\rho/4)T_{\min}\cdot\lambda D \\ &= (1/3+(2/3)w-\rho(1/2+w)-\lambda(1/3+\varepsilon/2-\rho/4))DT_{\min},\end{aligned}$$

and hence $2\operatorname{area}(\mathcal{J}_{res}) \leq (2/3+4/3w-\rho(1+2w)-(2/3+\varepsilon-\rho/2)\lambda)DT_{\min}$. On the other hand, it holds that

$$\begin{aligned}&ab-(2w_{\max}-a)_+(2h_{\max}-b)_+ \\ &\quad= (2/3-\rho/2+\varepsilon)T_{\min}(1-\lambda)D \\ &\qquad- (2(1/2+w)D-(1-\lambda)D)_+(2(1/3+\varepsilon/2-\rho/4)T_{\min}-(2/3-\rho/2+\varepsilon)T_{\min})_+ \\ &\quad= (2/3+\varepsilon-\rho/2-(2/3-\rho/2+\varepsilon)\lambda)DT_{\min}.\end{aligned}$$

Hence, Steinberg's condition is fulfilled if $(4/3-2\rho)w-\rho/2 \leq \varepsilon$, which is true since $w \leq (3/4)\varepsilon$.

Therefore, in both cases, we use Steinberg's algorithm to place the jobs $\mathcal{J}_{\text{res}}$ inside a rectangular container C of height $(2/3+\varepsilon-\rho)T_{\min}$ and width $(1-\lambda)D$, which in turn is positioned at $\sigma(C)=0$. □

Theorem 15 (Second Steinberg Case) *Let $T_{\min}$ be the lower bound on* OPT *defined as above. If $h(\mathcal{J}_{w(i)\geq(3/4)D}) > (2/3)T_{\min}$, then there is a polynomial time algorithm that schedules all the jobs inside the area* $[0, D]\times[0,(5/3)T_{\min}]$.

Proof In the first step, we place all the jobs in $\mathcal{J}_D := \mathcal{J}_{w(j)>D/2}$ and $\mathcal{J}_{seq} := \mathcal{J}_{h(j)>T/2}\setminus\mathcal{J}_D$ L-shaped. By Lemma 13 the resulting schedule has a maximum height of at most (3/2)OPT. Let $h(\mathcal{J}_D) := (2/3+\rho)T_{\min}$ and $w(\mathcal{J}_{seq}) := \lambda D$. By Lemma 10, we know that $\lambda D \leq 2(D-(3/4)D) = D/2$, since $h(\mathcal{J}_{w(j)\geq(3/4)D}) > (2/3)T_{\min}$.

The total area $\operatorname{area}(\mathcal{J}_{res})$ of the residual jobs is bounded by

$$\begin{aligned}\operatorname{area}(\mathcal{J}_{res}) &\leq DT_{\min} - (3/4)D\cdot(2/3)T_{\min} - (1/2)D\cdot\rho T_{\min} - \lambda D\cdot T_{\min}) \\ &= (1/2-\rho/2-\lambda/2)DT_{\min}.\end{aligned}$$

On the other hand, there is a rectangular area with width $a := (1-\lambda)D$ and height $b := ((5/3)-(2/3+T_{\min})) = (1-\rho)T_{\min} \geq (1/2)T$ where we can place the residual jobs. We will place the residual jobs into this area using Steinberg's algorithm. This is possible if Steinberg's condition $2\operatorname{area}(\mathcal{J}_{res}) \leq ab-(2\cdot w_{\max}(\mathcal{J}_{res})-a)_+(2\cdot h_{\max}(\mathcal{J}_{res})-b)_+$ is fulfilled, and each job fits inside the schedule area. Since $w_{\max}(\mathcal{J}_{res}) \leq D/2 \leq a$ and $h_{\max}(\mathcal{J}_{res}) \leq T_{\min}/2 < b$, it holds that

$$\begin{aligned}&ab-(2\cdot w_{\max}(\mathcal{J}_{res})-a)_+(2\cdot h_{\max}(\mathcal{J}_{res})-b)_+ \\ &\quad= (1-\lambda)D\cdot(1-\rho)T_{\min} - (D-(1-\lambda)D)_+(T_{\min}-(1-\rho)T_{\min})_+ \\ &\quad= (1-\lambda-\rho)D\cdot T_{\min}\end{aligned}$$

$$= 2(1/2 - \rho/2 - \lambda/2)DT_{\min} \geq 2\text{area}(\mathcal{J}_{res}).$$

The condition is fulfilled, and we can use the free rectangular area to place the residual jobs. □

4 Rearranging a Schedule that Fits Almost all Jobs

In this section, we assume that we are given a schedule that fits almost all jobs into a schedule of maximum height $T \leq (1+\varepsilon)\text{OPT}$. Left out of the schedule is a set of jobs that has a very small total width, where each job can have a height up to OPT. As such, this set of jobs can be fit into a strip of height OPT and width γD for some small $\gamma \in (0, \varepsilon]$. Later, we will see that such a schedule can be generated using the algorithm from Theorem 19 in Sect. 5.

Since we aim to generate a schedule of maximum height $(5/3+\varepsilon)\text{OPT}$, it does not suffice to simply place the non-scheduled jobs atop the generated schedule, as this would result in a maximum height of 2OPT. Instead, we must find some area in the generated schedule, inside of which we can remove jobs such that a height of OPT/3 for a width of λD is empty, where $\lambda \in [0, 1]$ is a small constant depending on ε (and γ). Once we have achieved this and placed the jobs removed by this procedure in a way that does not intersect this strip, we can then place the strip of height OPT at exactly that place, resulting in a schedule of maximum height $(5/3+\varepsilon)\text{OPT}$.

If none of the previously mentioned cases (as in Theorems 14 and 15, that can be solved using Steinberg's algorithm) apply, then the following Theorem combined with Theorem 19 proves Theorem 2.

Theorem 16 *Let $\varepsilon \in (0, 1/3]$, $\varepsilon' \leq (3/5)\varepsilon$ and $\gamma \leq (3/40)\varepsilon$. Furthermore, let I be an instance with $h(\mathcal{J}_{w(j)>(3/4)D}) \leq (2/3)T_{\min}$ and $w(\mathcal{J}_{h(j)>(2/3)T_{\min}}) \leq (1-(3/4)\varepsilon)D$ and a schedule σ with these properties:*

- *almost all jobs are scheduled with peak demand bounded by $T \leq (1+\varepsilon')\text{OPT}$;*
- *the remaining jobs are scheduled inside a box C_γ of height T and width γD.*

There is an Algorithm that finds a restructured schedule that places all the jobs up to a schedule with a maximum height of at most $(5/3+\varepsilon)\text{OPT}$.

Proof From the schedule σ, we will generate a new schedule σ'. Some jobs will be shifted to new starting positions σ'. Other jobs j (that are not explicitly shifted in this proof) keep their original starting positions, i.e., $\sigma'(j) = \sigma(j)$. □

4.1 Existence of a Suitable Job

If the schedule contains a job j with width $w(j) \in [\gamma, (1-2\gamma)D]$ and height $h(j) \in [(1/3)T, (2/3)T]$, we proceed as follows to make room for C_γ by shifting job j: Since $w(j) \leq (1-2\gamma)D$ it holds that $\max\{\sigma(j), D-(\sigma(j)+w(j))\} \geq \gamma D$. Let us, w.l.o.g., assume that $\sigma(j) \leq D-(\sigma(j)+w(j))$, otherwise we mirror the schedule at $D/2$. We shift the job j completely to the right (by at least γD) such that it is positioned at

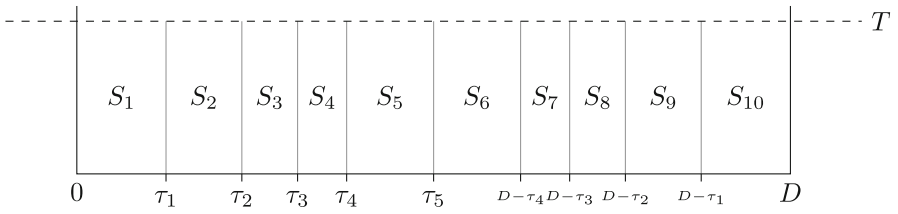


Fig. 5 Splitting the given schedule into segments at time points $\tau_1 = \frac{D}{8}, \tau_2 = \frac{(15-24\gamma)D}{64}, \tau_3 = \frac{(9+11\gamma)D}{32}$, $\tau_4 = \frac{(3+2\gamma)D}{8}$ and $\tau_5 = \frac{D}{2}$

$\sigma'(j) := D - w(j)$. This increases the maximum height to at most $(5/3)T$. Now the schedule between $\sigma(j)$ and $\sigma(j) + \gamma D$ has a height of at most $(2/3)T$. We place the box C_γ at $\sigma(j)$. Since the box has a height of at most T, the resulting schedule still has a maximum height of at most $(5/3)T \leq (5/3 + \varepsilon)\text{OPT}$.

4.2 No Suitable Job

If such a job does not exist, we search for segments that are not overlapped by jobs with a height greater than $(2/3)T$. We split the schedule into segments at the time steps (x-positions) $\tau_1 = \frac{D}{8}, \tau_2 = \frac{(15-24\gamma)D}{64}, \tau_3 = \frac{(9+11\gamma)D}{32}, \tau_4 = \frac{(3+2\gamma)D}{8}$ and $\tau_5 = \frac{D}{2}$ as well at $\tau_{10-i} = D - \tau_i$ for $i \in \{1, 2, 3, 4\}$ and set $\tau_0 = 0$. We number the resulting segments in increasing order from 0 to D. Note that segments S_i and S_{11-i} have exactly the same with and would be at the same position when mirroring the schedule at $D/2$, see Fig. 5. For $k \in \{1, \ldots, 10\}$, we denote by $\sigma(S_k) = \tau_{k-1})$ the start-time of a segment, by $c(S_k) = \tau_k)$ the end-time of the segment and by $w(S_k) = \tau_k - \tau_{k-1})$ the width of the segment S_k. Since $w(\mathcal{J}_{h(j)>(2/3)T}) \leq (1 - (3/4)\varepsilon)D$, we know, by the pigeonhole principle, that in one of these segments a total time of at least $(3/4)\varepsilon D/10 \geq (3/40)\varepsilon D \geq \gamma D$ is not overlapped by these jobs.

Let S_{l,k_1} be the earliest such strip, and S_{r,k_2} the latest such that $k_1, k_2 \in \{1, \ldots, 10\}$ represent the index of the strips $S_1, \ldots, S_{10}$. Note that it might happen that $S_{l,k_1} = S_{r,k_2}$. From now on, we assume that $\sigma(S_{l,k_1}) \leq D - c(S_{r,k_2})$ and otherwise mirror the schedule at $D/2$. As a consequence we now that $k_1 \in \{1, \ldots, 5\}$.

4.3 Modifying the Selected Strip

In the next step, we modify the start- and end-times of S_{l,k_1} such that it starts after the completion of a job with height at least $(2/3)T$ or at 0. We denote the shifted start times as $\sigma'(\cdot)$ and the shifted completion time as $c'(\cdot)$. If the start-time of S_{l,k_1}, i.e. τ_{k-1} intersects a job j with $h(j) \geq (2/3)T$, we define $\sigma'(S_{l,k_1}) := \sigma(j) + w(j) \leq c(S_{l,k_1}) - \gamma D$. Otherwise if $k_1 \neq 1$, we find the last job j ending before $\sigma(S_{l,k_1})$ with $h(j) \geq (2/3)T$ and define $\sigma'(S_{l,k_1}) := \sigma(j) + w(j)$ and shift the end-time of S_{l,k_1} by the same amount.

Note that $\sigma'(S_{l,k_1}) \geq \sigma(S_{l,k_1}) - \gamma D$: Since S_{l,k_1} is the first strip with at least γD time not occupied by jobs with a height greater than $(2/3)T$, the starting time

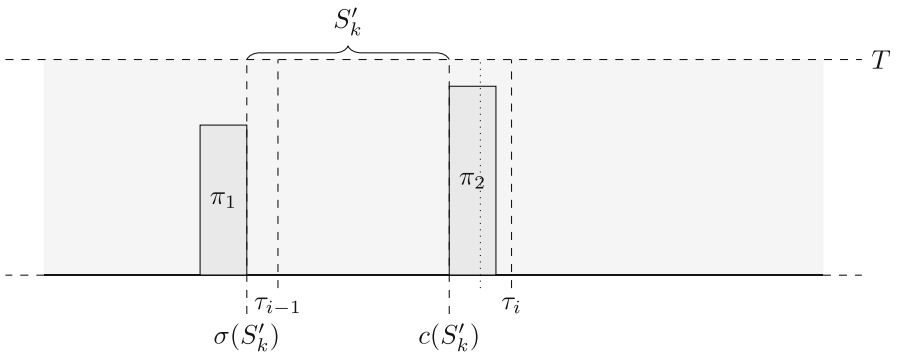


Fig. 6 An illustration of the border shifting procedure. The original borders are indicated by τ_{i-1} and τ_i. As τ_{i-1} is not intersected by a job with a height greater than $(2/3)T$, we shift $\sigma(S'_k)$ to an earlier point in time such that the job with a height greater than $(2/3)T$, π_1, ends at the exact same time. We then shift $c(S'_k)$ by the same amount. The shifted $c(S'_k)$ may intersect a job with a height greater than $(2/3)T$, π_2, indicated by the dotted line, and in this case, we shift the border further such that $c(S'_k) = \sigma(\pi_2)$ holds

of S_{l,k_1} is reduced by at most γD, while the width of the segment is not increased. The segment remains suitable because we only add width not utilized by jobs with a height greater than $(2/3)T$, and therefore removing the same amount of width at the end of the segment can not reduce the total width occupied by jobs with height less than $(2/3)T$ in the segment.

Finally, if the end-time of S_{l,k_1} intersects a job j that has a height greater than $(2/3)T$, we reduce it to $c'(S_{l,k_1}) := \sigma(j)$ and call the modified segment S'_{l,k_1}. These modifications never decrease the total width that is not overlapped by jobs with a height greater than $(2/3)T$ in S_{l,k_1}. We do the same but mirrored for S_{r,k_2} resulting in a modified segment S'_{r,k_2}. For an illustration of this procedure, see Fig. 6. If $D - c(S'_{r,k_2}) \leq \sigma(S'_{l,k_1})$, we mirror the schedule such that $\sigma'(j) = D - c(j)$. We denote by S'_k the segment in $\{S'_{l,k_1}, S'_{r,k_2}\}$ that appears first in this new schedule, where $k = \min\{k_1, k_2\}$ represents the original number of the chosen segment. As a consequence of this mirroring if $k \geq 2$, we ensured there exists a job j with $h(j) > (2/3)T$ and $c(S'_k) \leq \sigma(j) \leq D - \sigma(S'_k)$. Additionally, we know about the start and endpoints of this segment that $\tau_{k-1} - \gamma D \leq \sigma(S'_k)$ and $w(S'_k) \leq \tau_k - \tau_{k-1}$.

4.4 Rearranging Jobs in the Selected Strip

We aim to remove jobs from S'_k, such that the maximum height reached inside S'_k is bounded by $(2/3)T$. We categorize the jobs to be removed into three classes; first, the set of jobs $\mathcal{J}_{\text{cont}}$ that are wholly contained in S'_k due to the earlier shifting and have a height less than $(2/3)T$, second the set of jobs that have a height greater than $(2/3)T$, which must also be wholly contained in S'_k due to our border shifting, and finally the set of jobs intersecting one of the time points $\sigma(S'_k)$ or $c(S'_k)$.

First, we remove $\mathcal{J}_{\text{cont}}$ from the segment and schedule them inside a container that has a height of at most $(2/3)T$ and length at most $3w(S'_k)$.

Lemma 17 *The jobs $\mathcal{J}_{\text{cont}}$ can be scheduled inside a container C_{cont} of height $(2/3)T$ and width $3w(S'_k) \leq D/2$.*

Proof First note that $\text{area}(\mathcal{J}_{\text{cont}}) \leq w(S'_k)T$, since the maximum height in σ is bounded by T. We place these jobs using Steinberg's algorithm. Recall that this procedure allows us to place a set of rectangles $\mathcal{R}$ into a container of size $a \cdot b$ as long as the following conditions are met: $w_{max}(\mathcal{J}) \leq a, h_{\max}(\mathcal{J}) \leq b, 2 \cdot \text{area}(\mathcal{J}) \leq (ab - (2h_{\max}(\mathcal{J}) - T)_+(2w_{\max}(\mathcal{J}) - D)_+)$. Setting our values for $b = 3w(S'_k)$ and $a = (2/3)T$ yields the desired property. Clearly, no job wholly contained in a segment of width $w(S'_k)$ can have a width greater than $w(S'_k)$. Furthermore, the maximum height of any job in $\mathcal{J}_{\text{cont}}$ is $(2/3)T$. Finally, we have

$$\begin{aligned} 2 \cdot \text{area}(\mathcal{J}_{\text{cont}}) &\leq 2w(S'_k) \cdot T \\ &\leq 3w(S'_k) \cdot (2/3)T - (2w(S'_k) - 3w(S'_k))_+ \cdot (2(2/3)T - (2/3)T)_+ \\ &= (ab - (2h_{\max}(\mathcal{J}) - b)_+(2w_{\max}(\mathcal{J}) - a)_+) \end{aligned}$$

which proves the claim. □

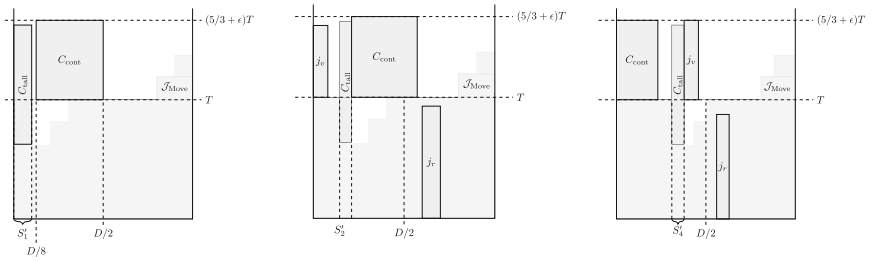
In the next step, we consider the jobs with heights greater than $(2/3)T$. By construction of the strip, we know that the total width of jobs with heights greater than $(2/3)T$ is bounded by $w(S'_k) - \gamma D$. We remove all these jobs from the strip and combine them with the extra container C_γ of height at most T and width γD to a new container called C_{tall}. It has a height of at most T and a width of at most $w(S'_k)$.

After this step, the only jobs remaining inside the area of S'_k are the jobs that overlap the borders of S'_k. If the maximum height in S'_k is lower than $(2/3)T$, we place the container C_{tall} inside the strip S'_k as well as the container C_{cont} right of $D/2$ and are done. Otherwise, we have to remove jobs that overlap the borders of the strip S'_k until the maximum height in S'_k is bounded by $(2/3)T$. The jobs we choose to remove are dependent on the position of the strip. The following lemma helps to see how these jobs can be shifted without increasing the maximum height of the schedule too much.

Lemma 18 *Consider a schedule σ with maximum height bounded by T and a time $\tilde{\tau}$, as well as a subset of jobs $\mathcal{J}_{\text{Move}} \subseteq \mathcal{J}(\tilde{\tau})$ with $h(\mathcal{J}_{\text{Move}}) \leq a \cdot T$ for some $a \in [0, 1]$. Let τ be the smallest value $\sigma(j)$ for $j \in \mathcal{J}_{\text{Move}}$. Consider the schedule σ', where all the jobs in $\mathcal{J}_{\text{Move}}$ are delayed such that they end at D, i.e., $\sigma'(j) = D - w(j)$ for all $\mathcal{J}_{\text{Move}}$ and $\sigma'(j) = \sigma(j)$ for all other jobs.*

In the schedule σ' before $D/2 + \tau/2$, the maximum height is bounded by T, while after $D/2 + \tau/2$, the maximum height is bounded by $(1 + a)T$.

Proof If the height of the schedule σ' is larger than T at a position τ', it has to be because one of the jobs in $\mathcal{J}_{\text{Move}}$ overlaps it. Hence, that maximum height is bounded by $(1+a)T$, since we shifted jobs with total height bounded by aT. Let j be one of the shifted jobs. If $\sigma'(j) > D/2+\tau/2$, the height of the schedule before $D/2+\tau/2$ cannot be influenced by this job. Therefore, assume that $D - w(j) = \sigma'(j) \leq D/2 + \tau/2$. As a consequence, $w(j) \geq D/2 - \tau/2$. Since $\sigma(j) \leq \tau$ it holds that $\sigma(j) + w(j) \geq D/2 + \tau/2$. Thus before time $D/2 + \tau/2$, the job j overlaps its previous positions and cannot increase the maximum height above T. □



(a) Repacking for $k = 1$ (b) Repacking for $k \in \{2, 3\}$ (c) Repacking for $k \in \{4, 5\}$

Fig. 7 Illustration of the steps in the proof of Theorem 16. Note that the set $\mathcal{J}_{\text{Move}}$ is delayed such that the jobs end at D. The containers C_{tall} and C_{cont} are placed such that they do not intersect. For **b** and **c**, the jobs j_v are placed in the same manner, and the job j_r is denoted

We choose which of the overlapping jobs to shift depending on if $k = 1$ or $k \neq 1$. Remember that none of the borders of S'_k overlap a job that has a height greater than $(2/3)T$, and assume, for the following, that there is a point inside S'_k where the total height of overlapping jobs is larger than $(2/3)T$.

Case 1: $k = 1$

Consider the time $\tau = c(S'_1) \leq D/8$ and the set of jobs $\mathcal{J}(\tau)$ that are intersected by this line. We know that the total height of jobs with a width greater than $(3/4)D$ is bounded by $(2/3)T$. Let $\mathcal{J}_{\text{Move}}$ be the set of jobs generated as follows: Greedily take the jobs with the greatest height from $\mathcal{J}(\tau) \setminus \mathcal{J}_{w(j)>(3/4)D}$, until either all the jobs from $\mathcal{J}(\tau)\setminus\mathcal{J}_{w(j)>(3/4)D}$ are contained in $\mathcal{J}_{\text{Move}}$ or $h(\mathcal{J}_{\text{Move}}) \in [(1/3)T, (2/3)T]$. In this process, we never exceed $(2/3)T$ since, if there is a job with a height greater than $(1/3)T$ in $\mathcal{J}(\tau)\setminus\mathcal{J}_{w(j)>(3/4)D}$, we choose it first and immediately stop. We delay the jobs in $\mathcal{J}_{\text{Move}}$ to new starting positions σ' such that for each job $j \in \mathcal{J}_{\text{Move}}$ we have that $c'(j) := \sigma'(j) + w(j) = D$. Note that $\sigma'(j) \geq (1/4)D$ for each $j \in \mathcal{J}_{\text{Move}}$ and therefore no longer overlaps $c(S'_1)$. Furthermore, we know by Lemma 18 that before $D/2$, the maximum height is bounded by T, while after $D/2$, the maximum height is bounded by $(5/3)T$. Furthermore, the maximum height inside S'_1 is bounded by $(2/3)T$.

Since S'_1 has a widthof at most $D/8$, we know by Lemma 17 that $\mathcal{J}_{\text{cont}}$ can be placed inside a container C_{cont} with height at most $(2/3)T$ and width bounded by $3D/8$. Therefore, we can schedule this container at $D/8$ and know that it ends before $D/2$. Finally, we schedule the container C_{tall} at $\sigma'(C_{\text{tall}}) = 0$. The peak height of the resulting schedule is bounded by $(5/3)T$. See Fig. 7a for the repacking procedure.

Case 2: $k \neq 1$ In this case, the borders of the considered strip can be overlapped from both sides. Furthermore, we know that the left border of S'_k is to the right of $D/8 - \gamma D \geq \gamma D$.

Consider the largest total height of jobs that are intersected by any vertical line through S'_k and denote this height as $T_{S'_k}$. Since there is a job j_l with height greater than $(2/3)T$ with $\sigma(j_l) + w(j_l) = \sigma(S'_k)$, we know that the total height of jobs intersecting $\sigma(S'_k)$ can be at most $(1/3)T$. Next, consider the closest job j_r that starts after $c(S'_k)$ and has a height greater than $(2/3)T$. By the choice of S'_k, we know that

such a job must exist and that $\sigma(j_r) \leq D - \sigma(S'_k)$, by the choice of S'_k out of $S'_{k_1,l}$ and $S'_{k_2,r}$. Furthermore, we know that the total height of jobs intersecting the vertical line at $\sigma(j_r)$ is bounded by $(1/3)T$.

Hence, the jobs that overlap the vertical line at $\sigma(j_r)$ and the jobs that overlap the vertical line at $\sigma(S'_k)$ add a total height of at most $(2/3)T$ to $T_{S'_k}$. Let us now consider the jobs $\mathcal{J}_M$ that overlap the time $c(S'_k)$ but neither the time $\sigma(S'_k)$ nor the time $\sigma(j_r)$. Each of them has a width of at most $D - \sigma(S'_k) - \sigma(j_r) \leq D - 2\sigma(S'_k)$. Hence when delaying their starting points such that $\sigma'(j) = D - w(j)$, they no longer overlap the time $c(S'_k)$ since $w(S'_k) \leq \sigma(S'_k)$ for each $k \in \{2, 3, 4, 5\}$.

We greedily take jobs from $\mathcal{J}_M$ that have the earliest starting point until we have all jobs from $\mathcal{J}_M$ or we have a total height of at least $(1/3)T$. If the total height of the chosen jobs is larger than $(2/3)T$, the last job j_v has a height of at least $(1/3)T$. Since it has a width less than $(1 - 2\gamma)D$, it has to have a width of less than γD. Otherwise, the job would have a width that allows us to shift only this job and make room for the strip C_γ. We remove this job and place it later, while we shift all the others to new positions σ' such that $\sigma'(j) = D - w(j)$ for each of the taken jobs j. We call the set of shifted jobs $\mathcal{J}_{\text{Move}}$.

Furthermore, since $\mathcal{J}_{\text{Move}}$ has a total height of at most $(2/3)T$ and a starting point right of $\sigma(S'_k)$, we know by Lemma 18 that the peak height right of $D/2 + \sigma(S'_k)/2$ is bounded by $(5/3)T$ while left of $D/2 + \sigma(S'_k)/2$ it is bounded by T.

Let $\sigma(j_l)$ be the starting time of the last taken job. Before $\sigma(j_l)$ (in S'_k) there is no longer a job from $\mathcal{J}_M$, and, hence inside in the strip S'_i that is left of $\sigma(j_l)$, the maximum height is bounded by $(2/3)T$. On the other hand, after $\sigma(j_l)$ (in S'_k), we either have removed jobs with total height at least $(1/3)T$, or all the jobs from $\mathcal{J}_M$ and hence the schedule there can have a total height of at most $(2/3)T$ as well. Therefore, we can place the container C_{tall} inside S'_k without increasing the total height above $(5/3)T$.

The container C_{cont} and the job j_v remain to be placed. For $k \in \{2, 3\}$, we set C_{cont} at $\sigma'(C_{\text{cont}}) = c(S'_k)$ and the job $\sigma'(j_v) = 0$, while for $k \in \{4, 5\}$, we set $\sigma'(C_{\text{cont}}) = 0$ and $\sigma'(j_v) = c(S'_k)$. We will now see, for each segment, that the maximum height of $(5/3)T$ is not exceeded by this new schedule.

First note that $w(j_v) \leq \gamma D \leq D/8 - \gamma D$ and hence does not intersect S'_2, when scheduled at $\sigma'(j_v) = 0$. Similarly, it is more narrow than S'_5 and $\sigma(S'_5)/2$, and hence fits right of S'_4 and S'_5 without increasing the schedule more than $(5/3)T$.

Let us now check the conditions for C_{cont}: For $k \in \{2, 3\}$, we have to ensure that

$$c(S'_k) + w(C_{\text{cont}})/2 \leq D/2 + \sigma(S'_k),$$

while for $k \in \{4, 5\}$ we have to prove that

$$w(C_{\text{cont}}) \leq \sigma(S'_k).$$

We start with $k = 2$: It holds that

$$w(S'_2) \leq \frac{(15 - 24\gamma)D}{64} - \frac{D}{8} = \frac{(7 - 24\gamma)D}{64}$$

and hence $w(C_{\text{cont}}) \leq 3(\frac{(7-24\gamma)D}{64})$. Therefore,

$$c(S_2') + w(C_{\text{cont}}) \leq \frac{(15-24\gamma)D}{64} + 3\left(\frac{(7-24\gamma)D}{64}\right) \leq \frac{D}{2} + \frac{D/8-\gamma D}{2} \leq \frac{D}{2} + \frac{\sigma(S_2')}{2}.$$

Furthermore for $k = 3$ we have

$$w(S_3') \leq \frac{(9+14\gamma)D}{32} - \frac{(15-24\gamma)D}{64} = (\frac{3}{64} + \frac{52\gamma}{32})D$$

and hence $w(C_{\text{cont}}) \leq 3(\frac{3}{64} + \frac{52\gamma}{32})D$. Therefore,

$$c(S_3') + w(C_{\text{cont}}) \leq \frac{(9+14\gamma)D}{32} + 3(\frac{3}{64} + \frac{52\gamma}{32})D = \frac{(27+184\gamma)D}{64},$$

while $D/2 + \sigma(S_3')/2 \geq D/2 + (\frac{(15-24\gamma)}{64} - \gamma)D/2 = (\frac{79}{128} - \frac{11}{8}\gamma)D$. As a consequence, $c(S_3') + w(C_{\text{cont}}) \leq D/2 + \sigma(S_3')/2$, since $\gamma \leq 1/40 \leq 25/392$.

Concerning $k = 4$, we have $w(S_4') \leq \frac{(3+2\gamma)D}{8} - \frac{(9+14\gamma)D}{32} = \frac{(3-6\gamma)D}{32}$. Hence,

$$w(C_{\text{cont}}) \leq 3\left(\frac{(3-6\gamma)D}{32}\right) = \frac{(9+14\gamma)D}{32} - \gamma D \leq \sigma(S_4').$$

Finally for $k = 5$, it holds that $w(S_5') \leq \frac{D}{2} - \frac{(3+2\gamma)D}{8} = \frac{(1+2\gamma)D}{8}$. Therefore,

$$w(C_{\text{cont}}) \leq 3 \cdot \frac{(1+2\gamma)D}{8} = \frac{(3+6\gamma)D}{8} = \frac{(1+2\gamma)D}{8} - \gamma D \leq \sigma(S_5').$$

For a visual representation of this repacking procedure, see Fig. 7. In all the cases, the generated schedule has a height of at most $(5/3)T \leq (5/3)(1+\varepsilon')\text{OPT} \leq (5/3+\varepsilon)\text{OPT}$. □

5 AEPTAS for NPDM

In this section, we will prove the following theorem.

Theorem 19 *Let $\varepsilon > 0$. There is an algorithm that places almost all jobs such that the maximum height is bounded by $T' := (1+\mathcal{O}(\varepsilon))\text{OPT}$. For the remaining jobs, we can choose one of the following containers for them to be placed in: C_1 with width εD and height T' or a container C_2 with width D and height $h_{\max}$. The time complexity of this algorithm is bounded by $\mathcal{O}(n\log(n)/\varepsilon) + 1/\varepsilon^{1/\varepsilon^{\mathcal{O}(1/\varepsilon)}}$.*

The statement, in fact, gives two variants of the algorithm. The first variant, where all remaining jobs are placed in C_1 is used in our $5/3+\varepsilon$ approximation algorithm, where the second variant with all remaining jobs in C_2 can be used to obtain the AEPTAS by setting $\sigma(C_2) = 0$. The described algorithm follows the dual-approximation

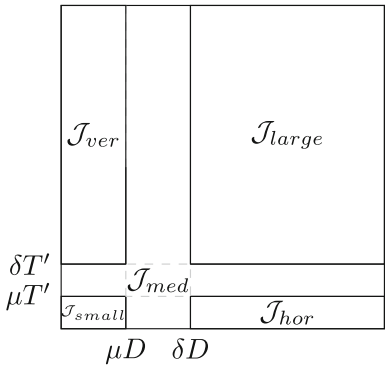


Fig. 8 Partition of the jobs. Each job is represented by a dot in this plane. The horizontal coordinate represents its width, while the vertical coordinate represents its height

framework. We describe an algorithm that, given a bound on the schedule maximum height T, computes a schedule with maximum height $(1 + \mathcal{O}(\varepsilon))T' + h_{\max}$, or decides correctly that there is no schedule with a maximum height at most T'. This algorithm can then be called in binary search fashion with values T between $T' = \max\{T_1, T_2, T_3, T_4\}$ and $\max\{2\text{area}(\mathcal{J})/D, 2h_{\max}\}$, using only multiples of $\varepsilon T'$.

Note that if $h_{\max} \leq \mathcal{O}(\varepsilon^3 T')$, we can use the algorithm in [47] to find an $(1 + \varepsilon)\text{OPT} + \mathcal{O}(\log(1/\varepsilon)/\varepsilon \cdot \varepsilon^3 T') = (1 + \mathcal{O}(\varepsilon))\text{OPT}$ approximation. Hence, we can assume that $h_{\max} > \mathcal{O}(\varepsilon^3 T')$.

5.1 Classification of Jobs

Given two values δ and μ with $\mu < \delta$, we partition the jobs into five sets: large, horizontal, vertical, small, and medium-sized jobs.

- $\mathcal{J}_{large} := \{i \in \mathcal{J} | h(i) \geq \delta T', w(i) > \delta D\}$
- $\mathcal{J}_{hor} := \{i \in \mathcal{J} | h(i) < \mu T', w(i) > \delta D\}$
- $\mathcal{J}_{ver} := \{i \in \mathcal{J} | h(i) \geq \delta T', w(i) < \mu D\}$
- $\mathcal{J}_{small} := \{i \in \mathcal{J} | h(i) < \mu T', w(i) < \mu D\}$
- $\mathcal{J}_{medium} := \mathcal{J} \setminus (\mathcal{J}_{large} \cup \mathcal{J}_{hor} \cup \mathcal{J}_{ver} \cup \mathcal{J}_{small})$

Lemma 20 *In $\mathcal{O}(n + 1/\varepsilon^2)$ operations it is possible to find values $\geq \varepsilon^{\mathcal{O}(1/\varepsilon^2)}$ for δ and μ such that $\text{area}(\mathcal{J}_{medium}) \leq (\varepsilon^2/4)DT$ and $\mu \leq c\varepsilon^5\delta$ for any given constant c.*

Proof Consider the sequence $\rho_0 := \varepsilon^5/4$, $\rho_{i+1} := c\rho_i\varepsilon^3$. Due to the pigeonhole principle, there exists an $i \in \{0, \ldots, 8/\varepsilon^2\}$ such that when defining $\delta := \rho_i$ and $\mu := \rho_{i+1}$, the total area of the medium-sized jobs is bounded by $(\varepsilon^2/4)DT$, because each job appears only in two possible sets of medium jobs. We have $\delta \geq \mu \geq \varepsilon^{\mathcal{O}(1/\varepsilon^2)}$ (Fig. 8). □

Lemma 21 [48] *We can round the heights $h(i)$ of the vertical and large jobs to multiples $k_i\varepsilon\delta T$ with $k_i \in \{1/\varepsilon, \ldots, 1/\varepsilon\delta\}$ such that the number of different demands is bounded by $O(1/\varepsilon^2 \log(1/\delta))$. This rounding increases the optimal height by at most $2\varepsilon T$.*

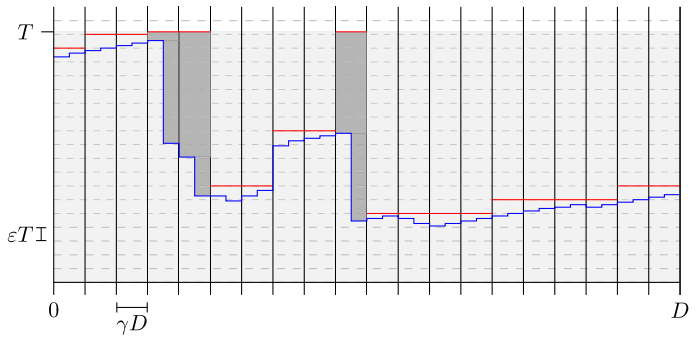


Fig. 9 In blue, we see the profile of horizontal and large items. This profile is rounded to multiples of εT (red). In segments where the profile jumps more than εT, we remove all vertical items (dark gray) and round the profile up to T (Color figure online)

This can be done via a standard combination of arithmetic and geometric rounding [48]. In the following, we will dismiss the medium jobs from the schedule.

5.2 Profile for Vertical Jobs

In the following, we will slice some vertical jobs into separated vertical pieces while repacking and structuring the schedule. Note that we allow this slicing since we only need a structural argument that all vertical jobs can be placed (as vertical slices) in the packing. Later, we will use an LP formulation to place vertical jobs. This will fix some of the sliced jobs, but still, leave us with a few jobs which are placed as vertical fractions. These fractionally placed jobs will contribute to the extra box mentioned in Theorem 19.

Given an optimal schedule, we partition the schedule into $1/\gamma$ segments of width γD, for a constant $\gamma \in \mathcal{O}_\varepsilon(1)$. Given a schedule of jobs J, we define profile of J to be $\{(x, y) | y = \sum_{j \in J | \sigma(j) \leq x \leq \sigma(j)+w(j)} h(j), 0 \leq x \leq D\}$. Height profile of jobs J at time t is $\mathcal{E}_J(t) := \sum_{j \in J | \sigma(j) \leq t \leq \sigma(j)+w(j)} h(j)$. You can imagine such a profile as the total height produced by placing the jobs from the set at their starting times in the schedule. Now consider the profile of large and horizontal jobs; see Fig. 9. Let $\tilde{J} := \mathcal{J}_{large} \cup \mathcal{J}_{hor}$. Next, we search for the segments where the maximal height of the profile of large and horizontal jobs and the minimal height of this profile differs more than εT, i.e., if in segment $S := (t_a, t_b)$, $|\max_{t \in S} \mathcal{E}_{\tilde{J}}(t) - \min_{t \in S} \mathcal{E}_{\tilde{J}}(t)| \geq \varepsilon T$, then we remove all vertical and small jobs from these segments fractionally, i.e., we vertically slice jobs, which are cut by the borders of the segment.

Claim 1 Let $\mathcal{J}_{rem}$ be the set of removed vertical and small jobs. Then area$(\mathcal{J}_{rem})$ is bounded by $\mathcal{O}(\gamma/\varepsilon\delta) \cdot D \cdot T$.

Proof Note that the height of the profile of horizontal or large jobs only changes, when horizontal or large jobs end or start. The large and horizontal jobs have a total height of at most T/δ since they have a width of at least δD, and the total area of the schedule is bounded by $T \cdot D$. Hence there can be at most $2(T/\delta)/\varepsilon T = \mathcal{O}(1/\varepsilon\delta)$ segments,

where the height of the profile changes more than εT. As a result, the total area of the removed vertical jobs can be bounded by $\mathcal{O}(1/\varepsilon\delta) \cdot (\gamma D \cdot T)$. □

Claim 2 (Size of γ) In the case of container C_1, we can choose $\gamma \in \mathcal{O}(\varepsilon\delta\lambda)$ such that we can schedule the removed vertical jobs fractionally inside a container $C_{1,1/4}$ of width $w(C_1)/4$ and height $h(C_1)$. Otherwise, we can choose $\gamma \in \mathcal{O}(\varepsilon^4\delta)$ such that we can schedule the removed vertical jobs fractionally inside a container $C_{2,1/4}$ of width $w(C_2)/4$ and height $h(C_2)$.

Proof Let $k \in \{1, 2\}$ depending on the chosen container. First we place all the jobs $\mathcal{J}_{rem,tall}$, i.e., jobs in $\mathcal{J}_{rem}$ with height greater than $h(C_{k,1/4})/2$ next to each other. The total width of these jobs is bounded by $2 \cdot \text{area}(\mathcal{J}_{rem,tall})/h(C_{k,1/4})$. Next, we place the residual jobs $\mathcal{J}_{rem,res} := \mathcal{J}_{rem} \backslash \mathcal{J}_{rem,tall}$, which have a height of at most $h(C_{k,1/4})/2$. We take slices of width 1 of the jobs and place them on top of each other until the height $h(C_{k,1/4})/2$ is reached. Since each job has a height of at most $h(C_{k,1/4})/2$, the height $h(C_{k,1/4})$ is not exceeded. The total width of this schedule is bounded by

$$2\text{area}(\mathcal{J}_{rem,res})/h(C_{k,1/4}) + 1 \leq \mathcal{O}(\gamma/(\varepsilon\delta) \cdot D \cdot T)/h(C_{k,1/4}).$$

Hence, for $C_{1,1/4}$ the total width is bounded by $\mathcal{O}(\gamma/(\varepsilon\delta) \cdot D \cdot T)/T = \mathcal{O}(\gamma/(\varepsilon\delta))D$. Therefore , when choosing $\gamma \in \mathcal{O}(\lambda\varepsilon\delta)$ for a suitable constant, the total width of this schedule is bounded by $w(C_1)/4$. Otherwise, the total width for container $C_{2,1/4}$ is bounded by $\mathcal{O}((\gamma/(\varepsilon\delta) \cdot D \cdot T)/\varepsilon^3 T) = \mathcal{O}(\gamma/\varepsilon^4\delta)D$. Thus, when choosing $\gamma \in \mathcal{O}(\varepsilon^4\delta)$ for a suitable constant, the total width of this schedule is bounded by $w(C_2)/4$. □

5.3 Algorithm to Place the Vertical, Small, and Medium Jobs

In the algorithm, we first round the heights of the vertical jobs to at most $\mathcal{O}(1/\varepsilon^2 \cdot \log(1/\delta)) = (1/\varepsilon)^{\mathcal{O}(1)}$ sizes using Lemma 21 (geometric rounding).

Afterward, we guess the height reserved for the vertical and small jobs rounding up to the next multiple of εT for each of the $1/\gamma$ segments, adding at most $2\varepsilon T$ to the height of the schedule. There are at most $\mathcal{O}((1/\varepsilon)^{1/\gamma})$ possible guesses. Furthermore, we introduce one segment $\top$ of height $\lceil h(C_k)/(\varepsilon T)\rceil \cdot \varepsilon T$ and width $w(C_k)/4\,(k \in \{1, 2\})$ for the set of removed vertical jobs. Let S_{ver} be the set of all introduced segments, and for each $s \in S_{ver}$ let $h_{s,ver}$ be the height reserved for vertical and small jobs. Note that for each $s \in S_{ver}$ there exists an $i \in \{0, \ldots, 1/\varepsilon + 3\}$ such that $h_{s,ver} = i\varepsilon T$. Furthermore, let $S_{ver,h}$ be the set of segments that have exactly height h and let $w(S_{ver,eh})$ be their total width.

To place the vertical jobs into the segments S_{ver}, we use a configuration LP. Let $C = \{a_\eta : \eta | \eta \in \{h(j) | j \in \mathcal{J}_{ver}\}\}$ be a configuration for vertical jobs, where a_η denotes the multiplicity with which the height η is contained in C. We denote by $h(C) := \sum_{\eta \in \{h(j) | j \in \mathcal{J}_{ver}\}} a_\eta \cdot \eta$ the total height of C, and by $\mathcal{C}_h$ the set of configurations with height at most h. Furthermore, for a given configuration C we denote by $a_\eta(C)$ the number of jobs contained in C that have a height of η. Since each vertical job has a

height of at least δT, there are at most $(1/\varepsilon)^{\mathcal{O}(1/\delta)}$ different configurations. Consider the following linear program:

$$\sum_{C \in \mathcal{C}_{i\varepsilon T}} x_{C,i} = w(S_{ver,i\varepsilon T}) \qquad \forall i \in \{1, \ldots, 1/\varepsilon + 3\} \quad (1)$$

$$\sum_{s \in S} \sum_{C \in \mathcal{C}_{h_{s,ver}}} a_\eta(C) x_{C,s} = \sum_{j \in \mathcal{J}_{ver}, h(j) = \eta} w(j) \qquad \forall \eta \in \{h(j) | j \in \mathcal{J}_{ver}\} \quad (2)$$

$$x_{C,i} \geq 0 \qquad \forall C \in \mathcal{C}, i \in \{1, \ldots, 1/\varepsilon\} \quad (3)$$

The variable $x_{C,i}$ represents the width of configuration C inside segments $s \in S_{ver}$ with reserved height $h_{s,ver} = i\varepsilon T$. The first equation handles configurations inside segments with a certain height capacity. It ensures that the total width of these configurations does not exceed the total width of the segments. The second equation ensures that the total width of jobs with a certain height is covered by the configurations. Doing so for all heights ensures every job is fully scheduled. A basic solution has at most $(1/\varepsilon + |\{h(j) | j \in \mathcal{J}_{ver}\}| + 1) = (1/\varepsilon)^3$ nonzero components.

We can solve the above linear program by guessing the set of nonzero components and then solving the resulting LP in $((1/\varepsilon)^{\mathcal{O}(1/\delta)})^{(1/\varepsilon)^3}$ time.

To place the vertical jobs, we first fill them greedily inside the configurations (slicing when the corresponding configuration slot is full) and afterward place the configurations inside the schedule, slicing the jobs at the segment borders. For each nonzero component, we have one configuration that contains at most $2/\delta$ fractionally placed vertical jobs on top of each other, which have a total height of at most $2T'$. Additionally, for each segment, we have the same amount of fractional jobs. Hence the total area of fractionally placed jobs can be bounded by $\mu D \cdot 2T \cdot ((1/\varepsilon)^3 + 1/\gamma)$. If we choose C_1 this can be bounded by $\mathcal{O}(\mu/(\lambda\varepsilon\delta))DT \leq \lambda DT/8$, since $\mu = c\varepsilon\delta\lambda^2$ and otherwise by $\in \mathcal{O}(\mu D \cdot T/(\varepsilon^5\delta)) \leq DT/8$, since $\mu = c\varepsilon^5\delta$ for a suitable small constant c. We remove the fractionally placed jobs $\mathcal{J}_{frac}$.

Next, we place the small jobs inside the empty area that can appear above each configuration for vertical jobs. Note that there are at most $((1/\varepsilon)^{\mathcal{O}(1)} + 1/\gamma)$ configurations, and the free area inside these configurations has at least the size of the total area of the small jobs. As a consequence, we have at most $2/\gamma$ rectangular areas to place the small jobs, which have a total area, which is at least the size of the small jobs. We use the NFDH algorithm to place these jobs inside the boxes until no other job fits inside.

Assume we could not place all the small jobs inside these boxes. When considering the free area in each box, there are three parts that contribute to it. First, each box can have a strip of free area later on in the schedule, which has a width of at most μD. The total free area contributed by this strip is bounded by $(2/\gamma)\mu D \cdot 2T$. Second, each box can have a strip of free area of height at most μT on the top, because otherwise, another line of jobs would have fit inside this box. Since there are no boxes on top of each other, we can bound the total free amount of work, which is a result of this strip, by $\mu T \cdot D$. Finally, there can be free area between the shelves of the jobs generated by the NFDH algorithm. This total free area is bounded by the height of the tallest job

times the width of the widest box, i.e. $\mu T \cdot \gamma D$. Hence the total free area inside the boxes is bounded by $5\mu D \cdot T \cdot \gamma + D \cdot \mu T$. Since $\gamma \in \mathcal{O}(1/\varepsilon^5\delta)$ and we have chosen $\mu \leq c\delta\varepsilon^6$ for a suitable small constant $c \in \mathbb{Q}$, the total area of the residual small jobs $\mathcal{J}_{small,res}$, which could not be placed is bounded by $\varepsilon T D$.

We place the residual small jobs $\mathcal{J}_{small,res}$ on top of the schedule using NFDH. This adds a height of at most $2\varepsilon T$ to the schedule. Next, we place the medium jobs. We schedule all the medium jobs that have a width larger than $w(C_k)/4$ with Steinberg's algorithm inside a box of height at most $\mathcal{O}(\varepsilon)T$ and width D. This is possible since they have a width in $(\varepsilon D, D]$ and therefore, each has a height of at most $\mathcal{O}(\varepsilon)T$ because their total area is bounded by $(\varepsilon^2/4)DT$. The residual jobs (that might have a width larger than εT) are placed inside the first half of the container using Steinberg's algorithm. The latter half of the container is filled with the extra box for vertical jobs defined for the LP and the fractionally scheduled jobs. The extra box has a width of at most $w(C_k)/4$. Since the fractionally placed vertical jobs $\mathcal{J}_{frac}$ have an area of at most $h(C_k) \cdot w(C_k)/8$ and each has a width of at most $\mu D < w(C_k)/8$, we can use Steinberg's algorithm to place them inside the last quarter of the container C_k.

5.4 Placement of Horizontal Jobs

In this section, we first reduce the number of possible starting points for horizontal jobs and then use a linear program to place the jobs in the schedule.

First step: use geometric grouping to reduce the number of widths of horizontal jobs. At a loss of at most $2\varepsilon T$ in the approximation ratio, we can reduce the number of widths of horizontal jobs to $\mathcal{O}(\log(1/\delta)/\varepsilon)$ using geometric grouping (see [49, Theorem 2] by Karmarkar and Karp).

These rounded jobs can be placed fractionally instead of the original jobs and an extra box of height at most $\mathcal{O}(\varepsilon)T$. In this fractional packing, the horizontal jobs are sliced vertically, i.e., different fractions of a job might have different starting points, but a fraction that is started, will not be interrupted and have the same height during its procession. We denote the rounded width of a job j as $w'(j)$.

In the next step, we will reduce the number of different starting points of the large and fractionally placed horizontal jobs without exceeding the given profile. Remember, we know the profile of large and horizontal jobs with precision εT for the segments of width γD.

Claim 3 Without loss in the approximation ratio, we can reduce the number of different starting points of rounded horizontal and large jobs to $(1/\varepsilon)^{(1/\varepsilon)^{\mathcal{O}(1/\varepsilon)}}$.

Proof Consider the large and horizontal jobs starting in the first segment. Since this segment has a width of $\gamma D \leq \delta D$, there can be no job ending in this segment. Hence this segment is maximally filled at the point γD. We can shift the starting point of each job in this segment to 0, and we will not change the maximal height of this segment.

Now consider a job $i \in \mathcal{J}_{hor} \cup \mathcal{J}_{large}$ starting in the second segment. If there is no horizontal or large job ending before the start of i, we can shift the starting point of i to γD without changing the maximal filling height in this segment. However, if there is a job $j \in \mathcal{J}_{hor} \cup \mathcal{J}_{large}$ ending before i in this segment, we can not shift this job

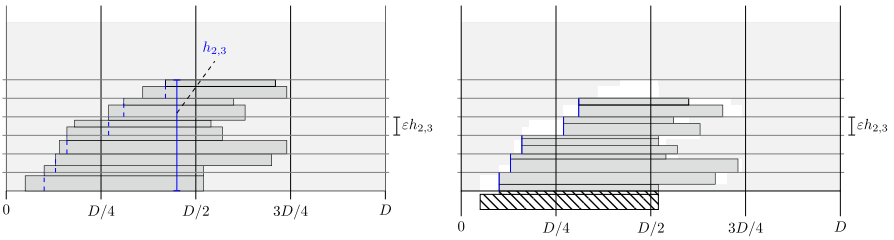


Fig. 10 An illustration of the reduction of *used* starting points for horizontal jobs. The right side shows the shifted version. The vertical lines represent the starting points after the shift. They are selected from the latest starting point within each segment of height $\varepsilon h_{2,3}$. The white area represents the cleared space from this procedure, while the removed jobs are shown as hatched below the schedule

to γD since then i and j overlap, which they did not before. This could change the maximal height of the profile in this segment. Nevertheless, if j is the last job ending before i, we can shift i to the left, such that i starts at the endpoint of j.

We iterate this shifting with all segments and all jobs in $\mathcal{J}_{hor} \cup \mathcal{J}_{large}$. As a result, all jobs start either at a multiple of γD, or they start at an endpoint of another job in $\mathcal{J}_{hor} \cup \mathcal{J}_{large}$. Therefore, we can describe the set of possible starting points for jobs in $\mathcal{J}_{hor} \cup \mathcal{J}_{large}$ as

$$S_{hor,large} := \left\{ l\gamma D + \sum_{j=1}^{1/\delta} w(i_j) | l \in \{0, 1, \ldots, 1/\gamma\}, i_j \in \mathcal{J}_{hor} \cup \mathcal{J}_{large} \forall j \in \{1, \ldots, 1/\delta_w\} \right\}.$$

It holds that $\left|S_{hor,large}\right| \leq (1/\gamma) \cdot (\log(1/\delta)/\epsilon)^{1/\delta} = (1/\varepsilon)^{(1/\varepsilon)^{\mathcal{O}(1/\varepsilon)}}$. □

Claim 4 At a loss of at most $\mathcal{O}(\varepsilon T)$ in the approximation ratio, we can reduce the number of *used* starting points for rounded horizontal jobs to $\mathcal{O}(1/\varepsilon\delta)$.

We partition the set of horizontal jobs by their width into $\mathcal{O}(\log(1/\delta))$ sets $\mathcal{J}^l_{hor} := \{i \in \mathcal{J}_{hor} | D/2^l < w(i) \leq D/2^{l-1}\}$. For each of these sets, we will reduce the number of starting positions to $2^l/\varepsilon^2$. We partition the schedule into 2^l segments of width $D/2^l$. Each job from the set $\mathcal{J}^l_{hor}$ has a width greater than $D/2^l$, and hence it starts in another segment as it ends. We consider for each segment all the horizontal jobs of the set $\mathcal{J}^l_{hor}$ ending in this segment and sort them by increasing starting position. Let $h_{l,i}$ be the height of the stack of jobs in $\mathcal{J}^l_{hor}$ ending in the i-th segment. We partition the stack into $1/\varepsilon$ layers of height $\varepsilon h_{l,i}$ and slice the horizontal jobs overlapping the layer borders. We remove all the jobs in the bottommost layer and shift the jobs from the layers above to the left, such that they start at the latest original starting position from the layer below. We repeat this procedure for each segment; see Fig. 10 for an illustration. By this shift, we reduce the total number of starting positions from jobs from the set $\mathcal{J}^l_{hor}$ to $2^l/\varepsilon$. The total height of the jobs we removed is bounded by $\varepsilon h(\mathcal{J}^l_{hor})$. Since these jobs have a width of at most $D/2^{l-1}$, we can schedule 2^{l-1} of these jobs after one another (horizontally), without violating the deadline. Hence, when scheduling these

jobs fractionally, we add at most $\varepsilon h(\mathcal{J}^l_{hor})/2^{l-1}$ to the schedule. Note that since all the jobs in set $\mathcal{J}^l_{hor}$ have a width of at least $D/2^l$, it holds that $\sum_{l=1}^{\lceil \log(1/\delta)\rceil} h(\mathcal{J}^l_{hor})/2^l \leq T$ and, hence, we add at most $\sum_{l=1}^{\lceil \log(1/\delta)\rceil} \varepsilon h(\mathcal{J}^l_{hor})/2^{l-1} \leq 2\varepsilon T$ to the height of the schedule, when scheduling the removed horizontal jobs.

The total number of starting positions is bounded by $\sum_{l=1}^{\lceil \log(1/\delta)\rceil} 2^l/\varepsilon = (2^{\lceil \log(1/\delta)\rceil+1} - 1)/\varepsilon \in \mathcal{O}(1/\delta_w\varepsilon)$. □

5.4.1 Algorithm to Place Horizontal and Large Jobs

To place the jobs in $\mathcal{J}_{hor} \cup \mathcal{J}_{large}$, we first guess the starting positions of the large jobs $\mathcal{J}_{large}$ in $\mathcal{O}(|S_{hor,large}|^{|\mathcal{J}_{large}|}) = (1/\varepsilon)^{(1/\varepsilon)^{\mathcal{O}(1/\varepsilon)}}$. Note that this guess affects the height that is left for horizontal jobs. Next, we guess which $\mathcal{O}(1/\varepsilon\delta)$ starting points in $S_{hor,large}$ will be used after the shifting due to Claim 4. There are at most $|S_{hor,large}|^{\mathcal{O}(1/\varepsilon\delta)} = (1/\varepsilon)^{(1/\varepsilon)^{\mathcal{O}(1/\varepsilon)}}$ possible guesses total. We call the set of guessed starting points $\bar{S}_{hor,large}$. For each starting point in $\bar{S}_{hor,large}$, we calculate the residual total height that is left after the guess for the large jobs. For a given $s \in \bar{S}_{hor,large}$ let $h_{s,hor}$ be this residual total height.

Consider the following linear program for horizontal jobs:

$$\begin{aligned}
&\sum_{\substack{\rho:=w'(j)\\ j\in\mathcal{J}_{hor}}} \sum_{\substack{s'\in\bar{S}_{hor,large}\\ s'\leq s<s'+\rho}} x_{\rho,s'} \leq h_{s,hor} && \forall s \in \bar{S}_{hor,large}\\
&\sum_{s\in S_{hor,large}} x_{\rho,s} = \sum_{\substack{j\in\mathcal{J}_{hor}\\ w'(j)=\rho}} h(j) && \forall \rho \in \{w'(j)|j \in \mathcal{J}_{hor}\}\\
&x_{\rho,s} \geq 0 && \forall s \in \bar{S}_{hor,large}, \rho \in \{w'(j)|j \in \mathcal{J}_{hor}\}
\end{aligned}$$

The variable $x_{\rho,s}$ denotes the total height of jobs with rounded width ρ starting at s. The first equation ensures that the height capacity at a starting time s is not exceeded by the jobs starting at or overlapping s. The second equation ensures that the total height requirement of jobs with rounded width ρ, $\sum_{\substack{j\in\mathcal{J}_{hor}\\ w'(j)=\rho}} h(j)$, is covered by the total height of jobs with this width started in the schedule $\sum_{s\in S_{hor,large}} x_{\rho,s}$.

A basic solution to this linear program has at most $|\bar{S}_{hor,large}| + |\mathcal{J}_{hor}| = \mathcal{O}(1/\varepsilon\delta)$ nonzero components. We can guess the nonzero components in at most $(|\bar{S}_{hor,large}| \cdot |\bar{\mathcal{J}}_{hor}|)^{|\bar{S}_{hor,large}|+|\bar{\mathcal{J}}_{hor}|} = (1/\varepsilon)^{(1/\varepsilon)^{\mathcal{O}(1/\varepsilon)}}$ guesses. Furthermore, we can guess their value with precision μT in at most $(1/\mu)^{|\bar{S}_{hor,large}|+|\bar{\mathcal{J}}_{hor}|} = (1/\varepsilon)^{(1/\varepsilon)^{\mathcal{O}(1/\varepsilon)}}$ guesses. Scheduling all the horizontal jobs integral and the error due to the precision add at most $2\mu T \cdot (|\bar{S}_{hor,large}| + |\bar{\mathcal{J}}_{hor}|)$ to the peak height. Note that $2\mu T \cdot (|\bar{S}_{hor,large}| + |\bar{\mathcal{J}}_{hor}|) \leq \mathcal{O}(\varepsilon)T'$ since $\mu \leq \mathcal{O}(\varepsilon^2\delta)$.

After this step, we have either scheduled all given jobs or have decided that it is not possible for the given guess of T and the profile. If it is not possible for any profile, we have to increase T. If we have found a schedule, we reduce the value of T. Each of the steps has increased the maximum by at most $\mathcal{O}(\varepsilon)T$ above T. Besides the job

classification and rounding, each time of each step of the algorithm is bounded by $(1/\varepsilon)^{1/\varepsilon^{\mathcal{O}(1/\varepsilon)}}$. Therefore, the described algorithm fulfills the claims of Theorem 19.

6 Conclusion

In this paper, we presented an AEPTAS with additive term $h_{\max}$ as well as a $(5/3+\varepsilon)$-approximation for Non-preemptive Peak Demand Minimization (NPDM). Since the lower bound for approximation algorithms for this problem is known to be $3/2$, this leaves a small gap between the lower bound and the approximation guarantee. Closing this gap is an interesting open question for further research, especially since, for the related strip packing problem, the same gap is yet to be resolved.

Funding Open Access funding enabled and organized by Projekt DEAL.

Declarations

Conflict of interest Arindam Khan declares that Prof. Susanne Albers was his postdoc mentor and Prof. Joseph Cheriyan was a coauthor. The other authors have no conflict of interest to disclose.

References

1. Tang, S., Huang, Q., Li, X., Wu, D.: Smoothing the energy consumption: Peak demand reduction in smart grid. In: 32nd IEEE International Conference on Computer Communications (INFOCOM), pp. 1133–1141. IEEE (2013). https://doi.org/10.1109/INFCOM.2013.6566904
2. Karbasioun, M.M., Shaikhet, G., Lambadaris, I., Kranakis, E.: Asymptotically optimal scheduling of random malleable demands in smart grid. Discrete Math. Algorithms Appl. **10**(02), 1850025 (2018)
3. Siano, P.: Demand response and smart grids—a survey. Renew. Sustain. Energy Rev. **30**, 461–478 (2014)
4. Alamdari, S., Biedl, T., Chan, T.M., Grant, E., Jampani, K.R., Keshav, S., Lubiw, A., Pathak, V.: Smart-grid electricity allocation via strip packing with slicing. In: Workshop on Algorithms and Data Structures, pp. 25–36. Springer (2013)
5. Liu, F.-H., Liu, H.-H., Wong, P.W.: Optimal nonpreemptive scheduling in a smart grid model. In: 27th International Symposium on Algorithms and Computation (ISAAC 2016). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2016)
6. Ranjan, A., Khargonekar, P., Sahni, S.: Offline preemptive scheduling of power demands to minimize peak power in smart grids. In: 2014 IEEE Symposium on Computers and Communications (ISCC), pp. 1–6. IEEE (2014)
7. Ranjan, A., Khargonekar, P., Sahni, S.: Smart grid power scheduling via bottom left decreasing height packing. In: 2016 IEEE Symposium on Computers and Communication (ISCC), pp. 1128–1133. IEEE (2016)
8. Chakraborty, N., Mondal, A., Mondal, S.: Efficient scheduling of nonpreemptive appliances for peak load optimization in smart grid. IEEE Trans. Ind. Inf. **14**(8), 3447–3458 (2017)

9. Baker, B.S., Coffman, E.G., Jr., Rivest, R.L.: Orthogonal packings in two dimensions. SIAM J. Comput. **9**(4), 846–855 (1980). https://doi.org/10.1137/0209064
10. Coffman, E.G., Jr., Garey, M.R., Johnson, D.S., Tarjan, R.E.: Performance bounds for level-oriented two-dimensional packing algorithms. SIAM J. Comput. **9**(4), 808–826 (1980). https://doi.org/10.1137/0209062
11. Sleator, D.D.: A 2.5 times optimal algorithm for packing in two dimensions. Inf. Process. Lett. **10**(1), 37–40 (1980). https://doi.org/10.1016/0020-0190(80)90121-0
12. Steinberg, A.: A strip-packing algorithm with absolute performance bound 2. SIAM J. Comput. **26**(2), 401–409 (1997). https://doi.org/10.1137/S0097539793255801
13. Schiermeyer, I.: Reverse-fit: a 2-optimal algorithm for packing rectangles. In: van Leeuwen, J. (ed.) Algorithms—ESA '94, Second Annual European Symposium, Utrecht, Proceedings. Lecture Notes in Computer Science, vol. 855, pp. 290–299. Springer (1994). https://doi.org/10.1007/BFb0049416
14. Harren, R., van Stee, R.: Improved absolute approximation ratios for two-dimensional packing problems. In: Dinur, I., Jansen, K., Naor, J., Rolim, J.D.P. (eds.) Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 12th International Workshop, APPROX 2009, and 13th International Workshop, RANDOM 2009, Berkeley. Proceedings. Lecture Notes in Computer Science, vol. 5687, pp. 177–189. Springer (2009). https://doi.org/10.1007/978-3-642-03685-9_14
15. Harren, R., Jansen, K., Prädel, L., van Stee, R.: A (5/3 + eps)-approximation for 2d strip packing. In: Brieden, A., Görgülü, Z., Krug, T., Kropat, E., Meyer-Nieberg, S., Mihelcic, G., Pickl, S.W. (eds.) 11th Cologne-Twente Workshop on Graphs and Combinatorial Optimization, Munich. Extended Abstracts, pp. 139– 142 (2012)
16. Ranjan, A., Khargonekar, P., Sahni, S.: Offline first fit scheduling in smart grids. In: 2015 IEEE Symposium on Computers and Communication (ISCC), pp. 758–763. IEEE (2015)
17. Yaw, S., Mumey, B., McDonald, E., Lemke, J.: Peak demand scheduling in the smart grid. In: 2014 IEEE International Conference on Smart Grid Communications (SmartGridComm), pp. 770–775. IEEE (2014)
18. Błądek, I., Drozdowski, M., Guinand, F., Schepler, X.: On contiguous and non-contiguous parallel task scheduling. J. Sched. **18**(5), 487–495 (2015)
19. Gálvez, W., Grandoni, F., Ameli, A.J., Khodamoradi, K.: Approximation algorithms for demand strip packing. CoRR arXiv:2105.08577 (2021)
20. Kenyon, C., Rémila, E.: A near-optimal solution to a two-dimensional cutting stock problem. Math. Oper. Res. **25**(4), 645–656 (2000). https://doi.org/10.1287/moor.25.4.645.12118
21. Jansen, K., Solis-Oba, R.: Rectangle packing with one-dimensional resource augmentation. Discret. Optim. **6**(3), 310–323 (2009). https://doi.org/10.1016/j.disopt.2009.04.001
22. Nadiradze, G., Wiese, A.: On approximating strip packing with a better ratio than 3/2. In: Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 1491–1510. SIAM (2016). https://doi.org/10.1137/1.9781611974331.ch102
23. Gálvez, W., Grandoni, F., Ingala, S., Khan, A.: Improved pseudo-polynomial-time approximation for strip packing. In: 36th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS), vol. 65, pp. 9– 1914. Schloss Dagstuhl—Leibniz-Zentrum für Informatik (2016). https://doi.org/10.4230/LIPIcs.FSTTCS.2016.9
24. Adamaszek, A., Kociumaka, T., Pilipczuk, M., Pilipczuk, M.: Hardness of approximation for strip packing. ACM Trans. Comput. Theory **9**(3), 14–1147 (2017). https://doi.org/10.1145/3092026
25. Henning, S., Jansen, K., Rau, M., Schmarje, L.: Complexity and inapproximability results for parallel task scheduling and strip packing. Theory Comput. Syst. **64**(1), 120–140 (2020). https://doi.org/10.1007/s00224-019-09910-6
26. Jansen, K., Rau, M.: Closing the gap for pseudo-polynomial strip packing. In: ESA, vol. 144, pp. 62–16214 (2019)
27. Gálvez, W., Grandoni, F., Ameli, A.J., Jansen, K., Khan, A., Rau, M.: A tight (3/2+ϵ) approximation for skewed strip packing. In: Byrka, J., Meka, R. (eds.) Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2020, Virtual Conference. LIPIcs, vol. 176, pp. 44–14418 (2020)
28. Bougeret, M., Dutot, P.F., Jansen, K., Otte, C., Trystram, D.: Approximating the non-contiguous multiple organization packing problem. In: IFIP International Conference on Theoretical Computer Science, pp. 316–327. Springer (2010)

29. Jansen, K.: Scheduling malleable parallel tasks: An asymptotic fully polynomial time approximation scheme. Algorithmica **39**(1), 59–81 (2004)
30. Jansen, K., Thöle, R.: Approximation algorithms for scheduling parallel jobs. SIAM J. Comput. **39**(8), 3571–3615 (2010). https://doi.org/10.1137/080736491
31. Jansen, K.: A (3/2+ ε) approximation algorithm for scheduling moldable and non-moldable parallel tasks. In: Proceedings of the Twenty-Fourth Annual ACM Symposium on Parallelism in Algorithms and Architectures, pp. 224–235 (2012)
32. Jansen, K., Land, F.: Scheduling monotone moldable jobs in linear time. In: 2018 IEEE International Parallel and Distributed Processing Symposium, IPDPS 2018, Vancouver, pp. 172–181. IEEE Computer Society (2018). https://doi.org/10.1109/IPDPS.2018.00027
33. Bansal, N., Caprara, A., Sviridenko, M.: A new approximation method for set covering problems, with applications to multidimensional bin packing. SIAM J. Comput. **39**(4), 1256–1278 (2009). https://doi.org/10.1137/080736831
34. Jansen, K., Prädel, L.: A new asymptotic approximation algorithm for 3-dimensional strip packing. In: 40th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM), vol. 8327, pp. 327– 338. Springer (2014). https://doi.org/10.1007/978-3-319-04298-5_29
35. Bansal, N., Correa, J.R., Kenyon, C., Sviridenko, M.: Bin packing in multiple dimensions: inapproximability results and approximation schemes. Math. Oper. Res. **31**(1), 31–49 (2006). https://doi.org/10.1287/moor.1050.0168
36. Bansal, N., Khan, A.: Improved approximation algorithm for two-dimensional bin packing. In: Chekuri, C. (ed.) Proceedings of SODA 2014, pp. 13–25. SIAM (2014)
37. Jansen, K., Zhang, G.: Maximizing the total profit of rectangles packed into a rectangle. Algorithmica **47**(3), 323–342 (2007). https://doi.org/10.1007/s00453-006-0194-5
38. Gálvez, W., Grandoni, F., Khan, A., Ramirez-Romero, D., Wiese, A.: Improved approximation algorithms for 2-dimensional knapsack: packing into multiple l-shapes, spirals and more. In: SoCG, pp. 39–13917 (2021)
39. Gálvez, W., Grandoni, F., Heydrich, S., Ingala, S., Khan, A., Wiese, A.: Approximating geometric knapsack via l-packings. In: FOCS, pp. 260–271 (2017)
40. Bansal, N., Lodi, A., Sviridenko, M.: A tale of two dimensional bin packing. In: FOCS, pp. 657–666 (2005)
41. Khan, A., Pittu, M.R.: On guillotine separability of squares and rectangles. In: APPROX, pp. 47–14722 (2020)
42. Khan, A., Maiti, A., Sharma, A., Wiese, A.: On guillotine separable packings for the two-dimensional geometric knapsack problem. In: SoCG, pp. 48–14817 (2021)
43. Adamaszek, A., Har-Peled, S., Wiese, A.: Approximation schemes for independent set and sparse subsets of polygons. J. ACM **66**(4), 29–12940 (2019)
44. Grandoni, F., Mömke, T., Wiese, A., Zhou, H.: A (5/3 + ε)-approximation for unsplittable flow on a path: placing small tasks into boxes. In: STOC, pp. 607–619 (2018)
45. Mömke, T., Wiese, A.: Breaking the barrier of 2 for the storage allocation problem. In: ICALP, pp. 86–18619 (2020)
46. Christensen, H.I., Khan, A., Pokutta, S., Tetali, P.: Approximation and online algorithms for multidimensional bin packing: a survey. Comput. Sci. Rev. **24**, 63–79 (2017)
47. Bougeret, M., Dutot, P., Jansen, K., Robenek, C., Trystram, D.: Approximation algorithms for multiple strip packing and scheduling parallel jobs in platforms. Discrete Math. Algorithms Appl. **3**(4), 553–586 (2011). https://doi.org/10.1142/S1793830911001413
48. Rau, M.: Useful structures and how to find them: hardness and approximation results for various variants of the parallel task scheduling problem. dissertation, Kiel University, Kiel (2019)
49. Karmarkar, N., Karp, R.M.: An efficient approximation scheme for the one-dimensional bin-packing problem. In: 23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, pp. 312–320. IEEE Computer Society (1982)