INSTITUT FÜR INFORMATIK

Scheduling Parallel Jobs on a Network of Heterogeneous Platforms

Klaus Jansen, Denis Trystram

Bericht Nr. 1502
September 2015
ISSN 2192-6247

CHRISTIAN-ALBRECHTS-UNIVERSITÄT

ZU KIEL

Institut für Informatik der
Christian-Albrechts-Universität zu Kiel
Olshausenstr. 40
D – 24098 Kiel

Scheduling Parallel Jobs on a Network of Heterogeneous Platforms

Klaus Jansen, Denis Trystram

Bericht Nr. 1502
September 2015
ISSN 2192-6247

e-mail: kj@informatik.uni-kiel.de,
trystram@imag.fr

Scheduling Parallel Jobs on a Network of Heterogeneous Platforms *

Klaus Jansen
Department of Computer Science
University of Kiel
kj@informatik.uni-kiel.de

Denis Trystram
Laboratoire d'Informatique
University of Grenoble
trystram@imag.fr

September 23, 2015

Abstract

We consider the problem of scheduling parallel jobs on a network of heterogeneous platforms. Given a set $\mathcal{J}$ of n jobs where each job $j \in \mathcal{J}$ is described by a pair (p_j, q_j) with a processing time p_j and number q_j of processors required and a set of N heterogeneous platforms P_i with m_i processors, the goal is to find a schedule for all jobs on the platforms minimizing the maximum completion time. Unless $P = NP$ there is no approximation algorithm with absolute ratio better than 2 for the problem. We propose an approximation algorithm with absolute ratio 2 improving the previously best known algorithms. This closes the gap between the lower bound of 2 and the best approximation ratio.

1 Introduction

We study the problem of scheduling parallel jobs on a network of heterogeneous platforms. The input consists of a set $\mathcal{J} = \{1, \ldots, n\}$ of n jobs and a set $\mathcal{B}$ of N platforms $P_1, \ldots, P_N$, where each P_i consists of a set $M_i = \{1, \ldots, m_i\}$ of processors for $i \in [N] := \{1, \ldots, N\}$. The width of the platform P_i is the number of processors m_i. Each job $j \in \mathcal{J}$ is described by a pair (p_j, q_j) with a processing time (or height) $p_j \in \mathbb{N}$ and number of processors (or width) $q_j \in \mathbb{N}$ required to execute j. If all numbers m_i are equal, we have identical platforms. In the general case the numbers m_i may be different and the machines are called heterogeneous platforms. For simplification we suppose that $m_1 \geq m_2 \geq \ldots \geq m_N$.

A schedule is an assignment $a : \mathcal{J} \to \mathbb{Q}_{\geq 0} \times \cup_{i=1}^{N} 2^{M_i}$ that assigns every job j to a starting time $t_j = a_1(j)$ and to a subset $A_j = a_2(j) \subset M_i$ of processors of one platform P_i such that $|A_j| = q_j$. Clearly, a job j can only be executed in platform P_i if the width of the platform $m_i \geq q_j$. A schedule is feasible if every processor in every platform executes at most one job at any time. The goal is to find a feasible schedule with minimum length or makespan $\max_{i \in [N]} C_{max}(P_i)$ where $C_{max}(P_i) = \max_{j : A_j \subset M_i} t_j + p_j$ is the makespan on platform P_i (or height of platform P_i). The optimum value for an instance $(\mathcal{J}, \mathcal{B})$ is denoted by $OPT(\mathcal{J}, \mathcal{B})$.

By a reduction from 3-Partition, Zhuk [12] proved that there is no approximation algorithm with absolute approximation ratio better than 2 for packing rectangles with height 1 into multiple strips. This reduction shows also that there is no approximation algorithm with ratio better than 2 for scheduling parallel jobs on identical platforms, where $m_1 = \ldots = m_N$. For the general problem Tchernykh et al. presented in [10] an algorithm with absolute ratio 10. Earlier Remy claimed in [8] that the approximation ratio 2 of List Schedule is preserved when applied to the problem with identical platforms while in [10] and again later in [9] it is shown that List Schedule cannot even guarantee a constant approximation ratio for this problem.

On the other hand, several improved approximation algorithms for the scheduling problem have been proposed. In Table 1 we give an overview about the known approximation algorithms for heterogeneous platforms. Remark that in [9], the algorithm is an online non-clairvoyant algorithm where processing times are not available to the processor. One algorithm [2] works only under the constraint where the maximum required number of processors $\max q_j$ is at most the minimum number of processors $\min m_i$ among all platforms, while the algorithm in [11] works for the general problem with additional release dates.

Currently, the best known absolute ratio of an approximation algorithm [5] for the general problem with heterogeneous platforms is $(2 + \epsilon)$. The running time of the algorithm is $g(1/\epsilon)n^{O(f(1/\epsilon))}$ for some functions f and g. In this paper we propose a polynomial time algorithm with absolute ratio 2. This closes the gap between the inapproximability bound of < 2 and the currently best absolute ratio $(2 + \epsilon)$.

*Research supported by German Research Foundation (DFG), project Ja 612 / 12-2.

Table 1: Approximation algorithms for heterogeneous platforms.

		ratio	constraints
Tchernykh et al. [10]	2005	10	none
Schwiegelshohn et al. [9]	2008	3	non-clairvoyant
Tchernykh et al. [11]	2010	$2e+1$	release dates
Bougeret et al. [2]	2010	2.5	$\max q_j \leq \min m_i$
Dutot et al. [5]	2013	$(2+\epsilon)$	none
Jansen and Trystram**(new result)**	2015	2	none

Theorem 1.1 *There is an approximation algorithm that for a set $\mathcal{J}$ of n parallel jobs and a set $\mathcal{B}$ of N heterogeneous platforms generates a schedule for the jobs with makespan at most $2\ OPT(\mathcal{J},\mathcal{B})$. The running time is polynomial in n.*

Methods and Techniques. In order to obtain an approximation algorithm with absolute ratio 2, we use the following approach. Our new algorithm works in two phases. Suppose by scaling that $OPT(\mathcal{J},\mathcal{B}) \leq 1$. In the first phase we use a slight modification of the $(2+\epsilon)$-approximation algorithm in [5]. Depending on four cases, the algorithm in the first phase generates a solution where the makespan on some platforms is bounded by $(1+\epsilon)$ and on other platforms by 2 while a constant number of sets of jobs is non-assigned to the platforms (see also Section 2 and Figures 1-4). Our previous algorithm places these sets onto the first group of platforms causing a makespan of $(2+\epsilon)$. Instead of this approach, our new algorithm converts the approximate solution of the first phase with $\epsilon = 1/10$ into a 2-approximate solution. To achieve this goal we re-schedule in the second phase some jobs on the platforms and insert the sets of non-assigned jobs of the first phase.

Consider the two widest platforms P_1 and P_2 with makespan $\leq (1+\epsilon)$ where the ratio between the widths m_1 and m_2 of the platforms is bounded by a constant; i.e. $m_1/m_2 \leq O(1)$. In this case we can insert a set of jobs with height or processing time ≤ 1 and small width (or small total number of processors used) $w(B^*)$ into P_1 and P_2 such that the makespan on both platforms is bounded by 2. This is done by clever re-scheduling some jobs on P_1 and P_2. The second idea is to modify the schedule for three platforms P_1, P_2 and P_3 with makespan at most $(1+\epsilon)$. Here we can generate a modified schedule for these three platforms (by re-scheduling some jobs) such that the makespan can be bounded by $2, (1+4\epsilon)$ and 1, respectively. This idea helps us to insert a constant number of non-assigned sets of jobs of phase 1.

One difficulty occurs when the ratio between the two widest platforms m_1/m_2 is not bounded by a constant $O(1)$. In this case we generate a $(3/2+\epsilon)$ approximate schedule on platform P_1 with some additional properties; i.e. where all huge jobs with processing time $p_j > 1/2$ finish at the same time and which contains a gap of height $1/2$ and width which is at least a fraction of the entire width m_1. This idea was used also to obtain a $(3/2+\epsilon)$ approximation algorithm for $N = 1$ [6]. This additional structure helps us to re-schedule some jobs and to insert some non-assigned large jobs with processing time $p_j \in (\delta, 1/2]$ later into the gap. In addition we show the following result. Given a solution for the two widest platforms P_1 and P_2 with makespan $(3/2+\epsilon)$ and $(1+\epsilon)$ where P_1 has a gap of height $1/2$ and width $\geq m_2$, we are able to insert a set of non-assigned jobs with height ≤ 1 and small width $w(B^*)$ such that the makespan on both platforms is bounded by 2.

2 $(2+\epsilon)$-approximation algorithm

Via binary search in the interval $[p_{max}, np_{max}]$ we find a candidate T for the optimum makespan and test whether there is a approximate solution with makespan $2T$ or not. By scaling we may assume that $T = 1$ and $p_{max} \leq 1$. Similarly to the previous $(2+\epsilon)$-approximation algorithm [5] we partition the jobs into three types:

- small jobs with $p_j \leq \delta^5$,
- medium jobs with $p_j \in (\delta^5, \delta]$, and
- large jobs with $p_j > \delta$

where the total area $Area(\mathcal{J}_m) = \{p_j q_j | j \in \mathcal{J}_m\}$ of the medium jobs in $\mathcal{J}_m$ is bounded by $(\epsilon/10)m_1$ and $\delta \in (0, \epsilon/50]$ is a constant that depends on ϵ; see also [5] how to calculate δ.

Then we consider two main scenarios. For accuracy $\epsilon \in (0, 1/6]$ we use a value $\gamma \geq 1$ (that depends on $1/\delta$ and is specified later), and distinguish the following two scenarios:

(1) for all $i \in [N]$ we have $m_1/m_i \leq \gamma$.

(2) there is a number $K \in \{1, \ldots, N-1\}$ such that $m_1/m_i \leq \gamma$ for all $i \leq K$ and $m_1/m_{K+1} > \gamma$.

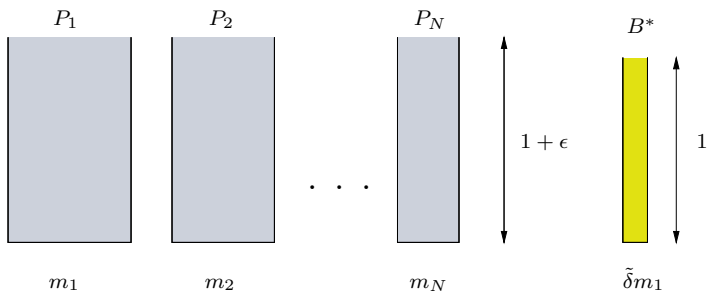

Figure 1: Solution generated by $(2+\epsilon)$ algorithm - case A

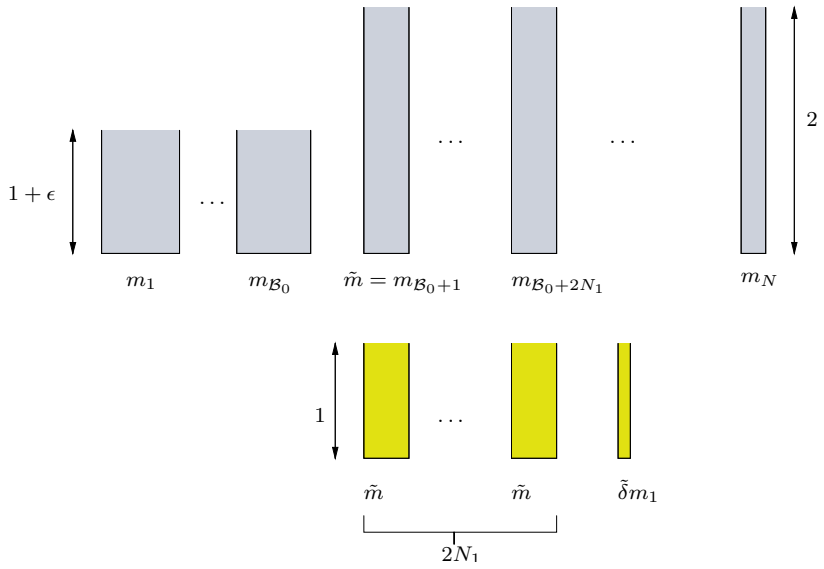

Figure 2: Solution generated by $(2+\epsilon)$ algorithm - case B

In the following we describe briefly the different scenarios and the solutions of the first phase. The details of the $(2+\epsilon)$-approximation algorithm are given in our appendix. In scenario (1) there are two subcases:

Case A: The number N of platforms is bounded by C/δ^4, where C is a constant. In this case, our algorithm computes in a first phase (see also Appendix A) an approximate solution with makespan at most $(1+\epsilon)$ on each platform, where a bin B^* (or a subset of jobs) with total height or execution time 1 and width or total number of processors $\leq \tilde{\delta}m_1$ (where $\tilde{\delta}$ is specified later) is not assigned to the platforms. Executing all jobs in B^* additionally on the first platform would give a $(2+\epsilon)$ - approximation. In Section 3 we show how to convert this solution in a second phase into a 2-approximate solution.

Case B: The number N of platforms is larger than C/δ^4. In this case the set $\mathcal{B}$ of all platforms is divided into two blocks of platforms:

- $\mathcal{B}_0$ with the widest platforms, where the cardinality $|\mathcal{B}_0| \in \{N_0, \ldots, N_0 + N_1 - 1\}$ and $N_0, N_1 = O(1/\delta^4)$,
- $\mathcal{B}_1$ with the remaining platforms $P_{|\mathcal{B}_0|+1}, \ldots, P_N$.

Afterwards, $\mathcal{B}_1$ is partitioned into $L = \max\{0, \lfloor \frac{N-N_0}{N_1} \rfloor\}$ groups each containing exactly N_1 platforms; using $|\mathcal{B}_0| \in [N_0, N_0+N_1]$. Our algorithm computes here in the first phase a solution where the makespan in the first $|\mathcal{B}_0|$ platforms is bounded by $(1+\epsilon)$ and where the makespan of the remaining platforms is bounded by 2. Again a bin B^* or subset of jobs, all with execution times ≤ 1 and total width $\leq \tilde{\delta}m_1$, is not assigned to the platforms. In addition, we have $2N_1$ bins of height ≤ 1 and width $\tilde{m} \leq m_{|\mathcal{B}_0|+1}$, that contain non-assigned jobs due to the rounding of the platform widths (see Appendix A for the details and Figure 2 for an illustration). Using $N_0 \geq 2N_1 + 1$, these non-assigned jobs

in these bins could be executed on the platforms $P_1, \ldots, P_{|\mathcal{B}_0|}$ (using $|\mathcal{B}_0| \geq N_0$). This would generate a schedule length or makespan $\leq (2+\epsilon)$ on the platforms. In Section 3 we show how to convert this solution in a second phase into a 2-approximate solution, too.

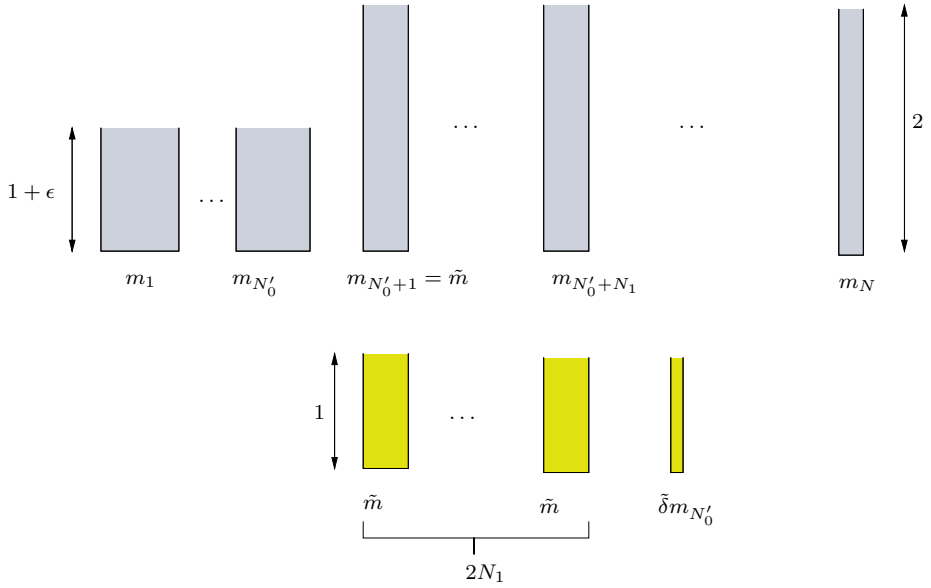

Figure 3: Solution generated by $(2+\epsilon)$ algorithm - case C

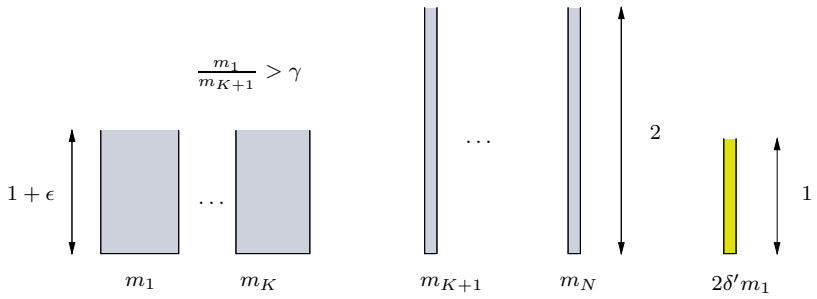

Figure 4: Solution generated by $(2+\epsilon)$ algorithm - case D

In the second scenario (2), we have to distinguish also two subcases depending on a constant $N'_0 = O(1/\delta^4)$:

Case C: $K \geq N'_0$. This case can be handled similar to case B. Our algorithm partitions the set $\mathcal{B}$ of N platforms into $\mathcal{B}_0 = \{P_1, \ldots, P_{N'_0}\}$ and $\mathcal{B}_1 = \{P_{N'_0+1}, \ldots, P_N\}$. The algorithm computes a solution in the first phase where the makespan in the first N'_0 platforms is at most $(1+\epsilon)$ and in the remaining platforms is at most 2; see Figure 3 for an illustration. Similar to case B above, there are $2N_1$ bins of height ≤ 1 and width $\tilde{m} \leq m_{N'_0+1}$ and an additional bin B^* of height 1 and width $\leq \tilde{\delta} m_{N'_0}$ that contain non-assigned jobs. Using $N'_0 \geq 2N_1 + 1$, these bins (or non-assigned jobs) could be executed on the first N'_0 platforms causing a makespan $\leq (2+\epsilon)$. Using our approach in Section 3 we are able to convert this in a second phase into a 2-approximate solution.

Case D: $K < N'_0$. Here our algorithm divides the set $\mathcal{B}$ of N platforms into $\mathcal{B}_0 = \{P_1, \ldots, P_K\}$ and $\mathcal{B}_1 = \{P_{K+1}, \ldots, P_N\}$. Then, the algorithm computes a solution in a first phase where the makespan in the first K platforms is bounded by $(1+\epsilon)$ and the makespan in the remaining platforms is bounded by 2. Similarly to the previous case C we obtain $2N_1$ additional bins of height ≤ 1 and width $\leq m_{K+1}$ plus one bin B^* of height 1 and bounded width $\leq \tilde{\delta} m_K$ that contain non-assigned jobs. The total width of the $2N_1 + 1$ bins can be bounded by $2\delta' m_1$ (where δ' is specified later); see also Figure 4. Our algorithm could place these bins of height 1 and total width $2\delta' m_1$ on platform P_1. This would imply a makespan of $(2+\epsilon)$ on the first platform. The converting step for case D is more complicated. In Section 4 we show how to modify the first phase and to improve the solution in a second phase to obtain a 2-approximate

schedule.

3 Cases A,B, and C: How to obtain a 2 approximate solution?

In this section we show how to convert the solutions of the first phase into a 2-approximate solution for the cases A, B and C. First we consider case A and B where $m_1/m_i \leq \gamma$ for all $i = 1, \ldots, N$ (where $\gamma = O(1)$). We may assume that $N \geq 2$; otherwise we get a $(3/2 + \epsilon) \leq 2$ solution [6] for $\epsilon \leq 1/2$. The execution times and starting times of the large jobs on the widest platforms are rounded up to multiples of δ^2 (see our appendix). This property is useful to convert the approximate solution.

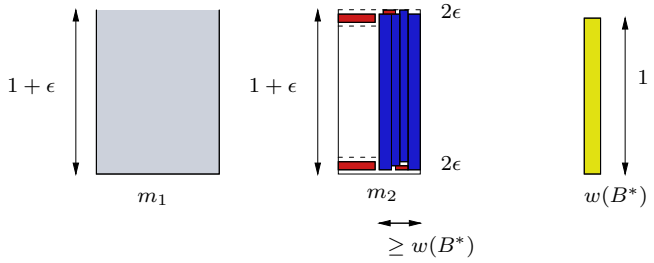

Figure 5: Algorithm to insert B^* - case 1, step 1

3.1 How to insert a thin bin of height 1?

The first step is to save some space for bin B^* with height 1 and width $w(B^*) \leq \tilde{\delta} m_1$. Notice in case A and B that the quotient $m_1/m_i \leq \gamma$ for all $i = 1, \ldots, N$. Using $\gamma = 2N_1/\delta'$ and $N_1 = \bar{C}/\delta^4 \leq C/\delta^4$ with $C \geq \bar{C} \geq 1$ (see also Lemma 2.6 in [5] for a sufficient bound for N_1), we obtain $1/\gamma = \delta'/(2N_1) \geq \delta'\delta^4/(2C) \geq (1/\delta^2 + 2)\tilde{\delta}$ (using $\tilde{\delta} \leq (1/(48C))\delta^{10}$ and $\delta' = (1/12)\delta^4$). This implies that $m_i \geq (1/\gamma)m_1 \geq (1/\delta^2 + 2)\tilde{\delta} m_1 \geq (1/\delta^2 + 2)w(B^*)$ for $i = 1, \ldots, N$. The following lemma shows that B^* can be inserted into the two widest platforms.

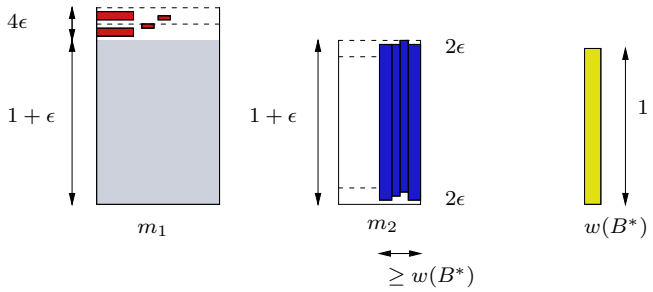

Figure 6: Algorithm to insert B^* - case 1, step 2

Lemma 3.1 *Let P_1 be the widest and P_2 be the second widest platform both with makespan $\leq 1 + \epsilon$. Furthermore, let B^* be an additional bin with height 1 and width $w(B^*)$. If $w(P_2) \geq (1/\delta^2 + 2)w(B^*)$ and $\epsilon \leq 1/5$, then B^* can be inserted into P_1 and P_2 such that the makespan $C_{max}(P_1)$ and $C_{max}(P_2)$ are both bounded by 2.*

Proof: Let X be the set of jobs in P_2 with height $> 1 - \epsilon$. In addition, let X_s be the set of jobs in P_2 with rounded starting time $s = a\delta^2$, $a \in \mathbb{N}_0$ and processing time $p_j \in (1/2, 1 - \epsilon]$.

Case 1: The total width $w(X) \geq w(B^*)$; i.e. the second widest platform P_2 has many jobs of height $> 1 - \epsilon$.

Study P_2 in more details; see Figure 5. Each job of height $> 1 - \epsilon$ intersects the horizontal lines at height $1 - \epsilon$ and 2ϵ. We remove all jobs that lie completely inside the horizontal layers of height 2ϵ at the beginning and end of the schedule. The removed horizontal layers of height 2ϵ can be both placed on top of P_1 since $w(P_1) \geq w(P_2)$. This generates in P_1 a makespan of $1 + 5\epsilon \leq 2$ (using $\epsilon \leq 1/5$); see Figure 6. Afterwards we move all large jobs with height

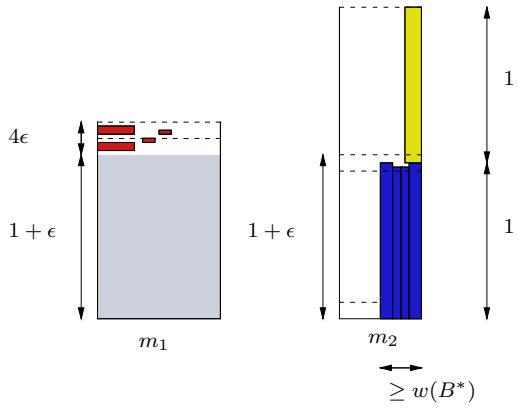

Figure 7: Algorithm to insert B^* - case 1, step 3

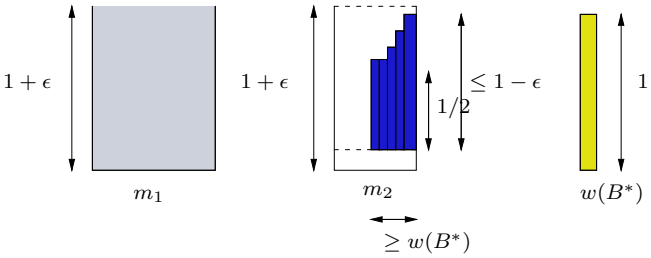

Figure 8: Algorithm to insert B^* - case 2, step 1

$> 1-\epsilon$ in P_2 down to starting time 0. This is possible, since all of them are started before time 2ϵ and now there are no jobs finishing before time 2ϵ. This generates a free block of width $\geq w(B^*)$ in P_2 from starting time 1 on. The bin B^* can be placed on P_2 at starting time 1. This generates a makespan in P_2 of 2; see Figure 7.

Case 2: The width $w(X_s) \geq w(B^*)$ for some rounding starting time s; i.e. the second widest platform P_2 has many jobs of height between $1/2$ and $1-\epsilon$ with the same rounded starting time $s = a\delta^2$; see Figure 8.

Moving a set X_s with this property from platform P_2 to P_1 generates a schedule on P_1 of makespan (or height) $\leq 1+\epsilon+1-\epsilon = 2$; see Figure 9. Now we have a gap in P_2 of height at least $1/2$ and width at least $w(B^*)$. In order to insert B^* we have to delay jobs that start after time $s+1/2$ by $1/2$. Then the gap has height at least 1. This implies that the makespan in P_2 after inserting B^* can be bounded by $1+1/2+\epsilon = 3/2+\epsilon \leq 2$. This is illustrated in Figure 10.

Case 3: For each rounded starting time $s \in \{0, \delta^2, 2\delta^2, ...\}$ the total width $w(X_s) < w(B^*)$ and $w(X) < w(B^*)$ in the second widest platform P_2.

Since the number of rounded starting times is at most $1/\delta^2$, the total width of $X' = X \cup \bigcup_s X_s$ is at most $(1/\delta^2+1)w(B^*)$. Using $m_2 \geq (1/\delta^2+2)w(B^*)$, at least $w(B^*)$ machines are not occupied by X'. Without loss of generality we may assume that X' uses the first $(1/\delta^2+1)w(B^*)$ machines. Then all other machines execute only jobs with processing time $\leq 1/2$; see also Figure 11. Consider now the horizontal line ℓ at height $3/4+\epsilon$ and remove all jobs that lie completely above this line and all jobs of height $\leq 1/2$ that intersect with line ℓ. Notice that a job j with $p_j \leq 1/2$ that intersects ℓ starts at or after time $1/4+\epsilon$. Therefore, the entire block of removed jobs has height at most $3/4$. Execute the removed set of jobs on the widest platform P_1. This generates a makespan of $\leq 1+\epsilon+3/4 = 7/4+\epsilon \leq 2$ on P_1 using $\epsilon \leq 1/4$; see Figure 12.

Now at least $w(B^*)$ of the machines in P_2 do not execute any job from step $3/4+\epsilon$ on. Therefore, bin B^* can be executed on P_2 from time step $3/4+\epsilon$ until $7/4+\epsilon$. This gives as illustrated in Figure 13 a makespan on P_2 of at most $7/4+\epsilon \leq 2$. □

3.2 How to insert $2N_1$ bins?

In the second step we show how to insert the $2N_1$ bins of height 1 and width $\tilde{m} \leq m_{|\mathcal{B}_0|}$. Let us consider here 3 platforms P, P' and P'' in $\mathcal{B}_0$ with makespan or height $\leq 1+\epsilon$. We may assume that $w(P) \geq w(P') \geq w(P'')$.

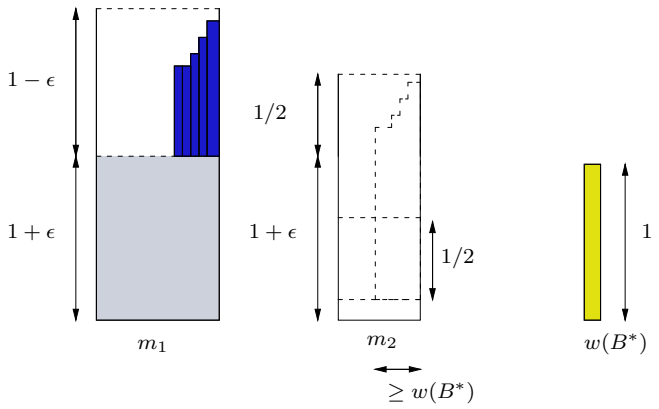

Figure 9: Algorithm to insert B^* - case 2, step 2

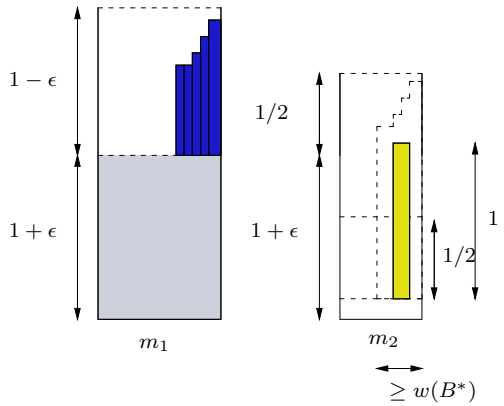

Figure 10: Algorithm to insert B^* - case 2, step 3

Lemma 3.2 *Let P, P', P'' be three platforms with makespan $\leq 1+\epsilon$ and $w(P) \geq w(P') \geq w(P'')$ and $\epsilon \leq 1/4$. Then we can generate a modified schedule of the jobs in P', P' and P' such that the makespan $C_{max}(P) \leq 2$, $C_{max}(P') \leq 1+4\epsilon \leq 2$ and $C_{max}(P'') \leq 1$.*

Proof: Consider now a horizontal layer of height ϵ in P'' between time 1 and $1+\epsilon$; see Figure 14. Take all jobs that lie completely in this layer plus jobs with height $\leq 1-2\epsilon$ that intersect the horizontal line ℓ at height 1. The total height of these jobs in the partial solution is at most $1-\epsilon$. Remove this part from P'' and place it on top of P. This generates a schedule on P with makespan $\leq 1+\epsilon+1-\epsilon = 2$; see also Figure 15. Next we remove the lower horizontal layer in P'' between time 0 and 3ϵ (including all jobs that lie completely inside the layer) and place the corresponding jobs on top of P'. This generates a schedule on P' with makespan $\leq 1+4\epsilon \leq 2$. Now each remaining job in P'' that intersects the horizontal line at time 1 and has length $> 1-2\epsilon$ intersects also the horizontal line at time 3ϵ. Since these jobs are executed all at time 3ϵ and there are no other jobs below the horizontal line at time 3ϵ, we can move all these long jobs down to starting time 0. After this step there are no jobs executed after time 1. Therefore, platform P'' now has makespan 1. This is illustrated in Figure 16. □

This procedure shows how to transform three platforms of height $\leq (1+\epsilon)$ into three platforms of height 2, $(1+4\epsilon) \leq 2$ and 1 for $\epsilon \leq 1/4$. If $|\mathcal{B}_0| \geq 2+6N_1$ (simply by specification $N_0 = 2+6N_1$ for case A and B), the widest two platforms can be used to insert B^* and the next $6N_1$ to insert $2N_1$ platforms of width $\tilde{m}_1 \leq m_{|\mathcal{B}_0|}$. This finishes the converting step for case A and B.

The same approach works also for case C using $N_0' \geq 2+6N_1$. The additional bin B^* of height 1 and width $w(B^*) \leq \tilde{\delta} m_{N_0'}$ fits on the two widest platforms using Lemma 3.1. Notice that the inequality $w(P_2) \geq (1/\delta^2+2)\tilde{\delta} w(P_2) \geq (1/\delta^2+2)\tilde{\delta} m_{N_0'} \geq (1/\delta^2+2) w(B^*)$ holds; simply by using $\tilde{\delta} = (1/(48C))\delta^{10}$ and $\tilde{\delta}(1/\delta^2+2) \leq 1$. Furthermore, the $2N_1$ platforms of width $\tilde{m} \leq m_{N_0'+1}$ can be merged with the $6N_1$ other platforms. This finishes case C.

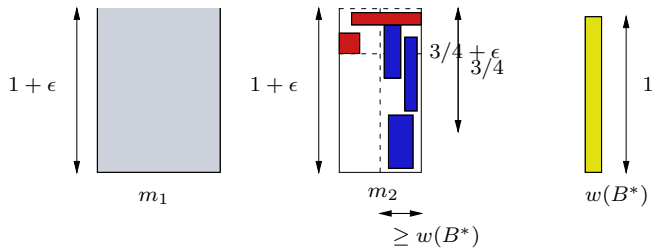

Figure 11: Algorithm to insert B^* - case 3, step 1

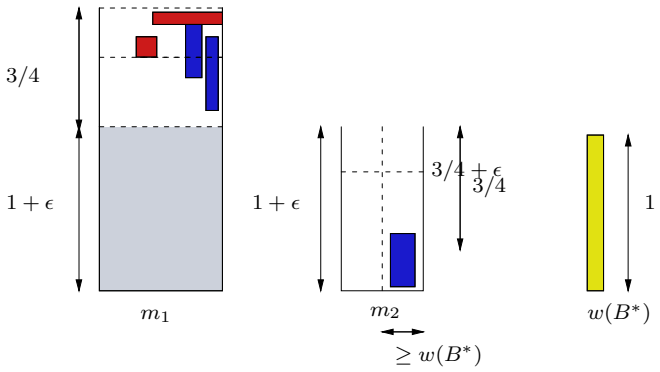

Figure 12: Algorithm to insert B^* - case 3, step 2

4 How to handle case D?

The main difficulty is case D. The case $K \geq 2$ can be handled using the approach above. If $m_2 \geq 2(1/\delta^2+2)\delta' m_1$ then there is enough space on the two widest platforms for the entire bin B^* (including the $2N_1$ additional bins) with height 1 and width $w(B^*) \leq 2\delta' m_1$. Here we can use Lemma 3.1 with $w(B^*) \leq 2\delta' m_1$.

Suppose that $m_2 \leq 2(1/\delta^2+2)\delta' m_1$. Since $K \geq 2$, we have $m_1/m_2 \leq \gamma$. Next we test whether there is an index K' with $m_2/m_{K'+1} > \gamma$ or not. If this is not the case, then $m_1/m_i = (m_1/m_2)(m_2/m_i) \leq \gamma^2 = O(1)$ for $i = 1,\ldots,N$ and we can use the approach in case A or B. In case A we obtain an additional bin B^* of height 1 and width $\leq 4N\alpha m_N/\delta^4 \leq \tilde{\delta} m_N$ (using $\alpha = \tilde{\delta}\delta^4/(4N)$) that can be inserted into the first and second platform using Lemma 3.1 and $m_2 \geq (1/\delta^2+2)\tilde{\delta} m_N$. Notice that $\tilde{\delta}$ is specified to obtain this inequality. In case B we get an additional bin of height 1 and width $\leq |\mathcal{B}_0| 4\alpha m_N/\delta^4 \leq \tilde{\delta} m_N$ (using $\alpha = \tilde{\delta}\delta^4/(4|\mathcal{B}_0|)$) and $2N_1$ bins of height 1 and width $\tilde{m} \leq m_{|\mathcal{B}_0|+1}$. Using $N_0' \geq 6N_1+2$ these $2N_1$ bins can be merged with the next widest $6N_1$ platforms. If there is an index K' with the property above then we run into a case similar to C or D. The case with $K' \geq N_0'$ can be handled in the same way as case B. Here we partition the set $\mathcal{B}$ of platforms into $\mathcal{B}_0 = \{P_1,\ldots,P_{N_0'}\}$ and $\mathcal{B}_1 = \{P_{N_0'+1},\ldots,P_N\}$. The other case with $K' < N_0'$ is the most interesting and difficult one. Here $\mathcal{B}_0 = \{P_1,\ldots,P_{K'}\}$ and $\mathcal{B}_1 = \{P_{K'+1},\ldots,P_N\}$. Using our algorithm we get an approximate solution with makespan $(1+\epsilon)$ for the first K' platforms and 2 for the remaining platforms. The unscheduled jobs are placed in $2N_1$ bins of height 1 and width $m_{K'+1}$ and an additional bin B^* of width $K'4\alpha m_{K'}/\delta^4 \leq N_0' 4\alpha m_{K'}/\delta^4 \leq \tilde{\delta} m_{K'} \leq \tilde{\delta} m_2$ (using $\alpha = \delta^4\tilde{\delta}/(4N_0')$). The total width of the other bins can be bounded by $2N_1 m_{K'+1} < 2N_1 m_2/\gamma = \delta' m_2$. Then, using Lemma 3.1 and $\tilde{\delta} \leq \delta'$ all these bins with total width $\leq 2\delta' m_2$ can be merged together with the two widest platforms. Here we use the property that $m_2 \geq 2(1/\delta^2+2)\delta' m_2$; this simply holds using our specification for δ'.

But if $K = 1$, then this approach above does not work. Here we have one wide platform and a large gap between the first and second platform. The main critical case occurs if the gap is larger than any constant. In this case we have to modify our previous approach as follows. We suppose that $N \geq 2$ and $m_1/m_2 \geq \gamma$. All jobs with width $> m_2$ have to placed on platform P_1. We guess here the large wide jobs for platform P_1; i.e. jobs with processing time $p_j > \delta$ and $q_j > \alpha_1 m_1$.

Similar to our previous approach we consider the following four scenarios.

Case D.A $m_1/m_2 > \gamma$, $m_2/m_i \leq \gamma'$ for $i = 3,\ldots,N$ and $N \leq C/\delta^4$.

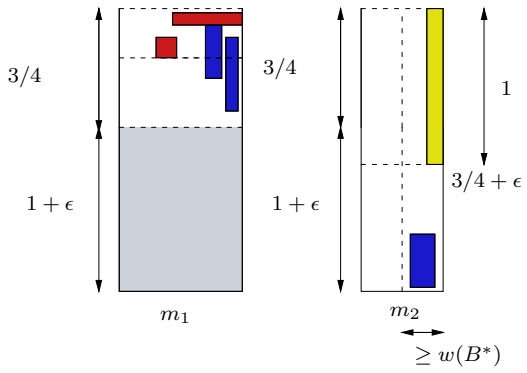

Figure 13: Algorithm to insert B^* - case 3, step 3

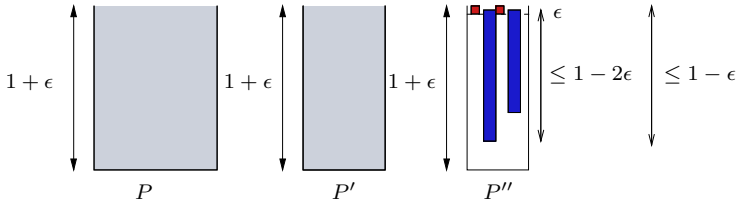

Figure 14: Algorithm to transform 3 platforms P, P', and P'' - step 1

Case D.B $m_1/m_2 > \gamma$, $m_2/m_i \leq \gamma'$ for $i = 3, \ldots, N$ and $N > C/\delta^4$.

Case D.C $m_1/m_2 > \gamma$, $m_2/m_K \leq \gamma'$, $m_2/m_{K+1} > \gamma'$ with $K \geq N_0' + 2$.

Case D.D $m_1/m_2 > \gamma$, $m_2/m_K \leq \gamma'$ and $m_2/m_{K+1} > \gamma'$ for $K < N_0' + 2$.

4.1 Case D.A

Suppose that $m_1/m_2 > \gamma$, $m_2/m_i \leq \gamma'$ for $i = 3, \ldots, N$ and $N \leq C/\delta^4$. In this case we guess the large wide jobs (with $p_j > \delta$ and $q_j \in [\alpha_2 m_2, m_2]$) for platforms $P_2, \ldots, P_N$ (i.e. the assigned platform and rounded starting time). Notice that there is only a constant number $O((1/\delta)(1/\alpha_2))$ of these jobs in each platform $P_2, \ldots, P_N$. Since $N \leq O(1/\delta^4)$ the total number of such jobs is at most $O((1/\delta^5)(1/\alpha_2))$. Notice that we may have a very large number of large wide jobs; due to the large gap between m_1 and m_2. But the number of possible guesses can be still bounded by a polynomial in n. In fact there are at most n candidates for each of the $O((1/\delta^5)(1/\alpha_2))$ jobs. After this guessing step we also know the remaining large wide jobs with $q_j > \alpha_2 m_2$ in P_1. Notice that the number of these jobs could be larger than $O(1)$. After shifting all huge jobs with $p_j > 1/2$ in P_1 (using the method in [6]) we obtain a $(3/2+\epsilon)$ makespan in P_1 where all huge jobs with $p_j > 1/2$ finish at the same time and there is a gap of height $1/2$ and width $\geq \frac{\delta^4}{8} m_1$ in P_1. This implies also that we know the starting and the finishing times of the huge jobs with width $q_j > \alpha_2 m_2$ in P_1. For all platforms $P_1, \ldots, P_N$ we guess an approximate load vector $\Pi_{i,a,h}$ for large narrow jobs; i.e. a multiple of $\alpha_2 m_N$ for each rounded starting time $a\delta^2$, rounded processing time $h\delta^2$ and platform P_i. Each large narrow job has width $\geq \alpha_2 m_2$. Since there are at most n such jobs for each triple (i, a, h), each multiple of $\alpha_2 m_2$ is bounded by n. Using $m_2/m_N \leq \gamma' = O(1)$, each multiple of $\alpha_2 m_N$ is bounded by $O(n)$. Since $N \leq C/\delta^4$, the number of triples is at most $O(N/\delta^4) \leq O(1/\delta^8)$. Using that each multiple is bounded by $O(n)$, the number of different vectors is also polynomial in n. Our algorithm produces a schedule with makespan $3/2 + \epsilon$ on P_1 and $1 + \epsilon$ on $P_2, \ldots, P_N$ for almost all jobs.

The jobs satisfy the following properties:

- huge jobs with $p_j > 1/2$ and $q_j > \alpha_2 m_2$ on P_1 finish at the same time.
- huge and large jobs with $p_j > \delta$ and $q_j > \alpha_2 m_2$ on $P_2, \ldots, P_N$ are placed in the guessing step.
- large jobs with $p_j \in (\delta, 1/2]$ and $q_j > \alpha_1 m_1$ on P_1 are placed in the guessing step.

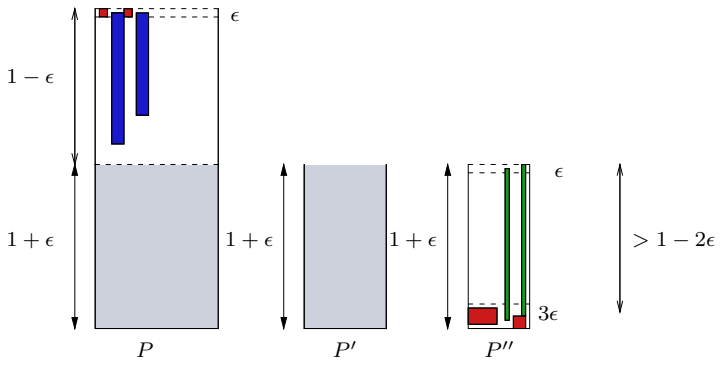

Figure 15: Algorithm to transform 3 platforms P, P', and P'' - step 2

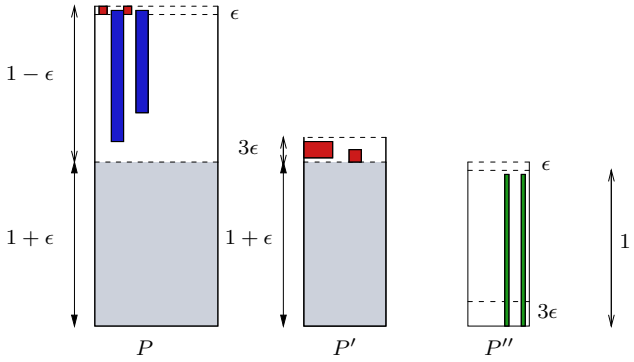

Figure 16: Algorithm to transform 3 platforms P, P', and P'' - step 3

- large jobs with $p_j \in (\delta, 1/2]$ and $q_j \leq \alpha_1 m_1$ are either assigned to a unique platform and starting time or fractionally to the platforms and starting times.
- huge jobs with $p_j > 1/2$ and $q_j \leq \alpha_2 m_2$ are also either assigned to a unique platform and starting time or fractionally to the platforms and starting times.
- small jobs are assigned via a linear program and strip packing approach into horizontal layers of the platforms.

The fractionally assigned large jobs in P_1 (including the amount to reduce the load values) have total width $\leq 4/\delta^4 \lfloor \alpha_1 m_1 \rfloor \leq (\delta^4/16) m_1$ using $\alpha_1 \leq \delta^8/40$ sufficiently small. This implies that these jobs fit all into the gap in P_1 and use at most half of the gap width $(\delta^4/8)m_1$. The fractionally assigned huge jobs in P_1 and large and huge jobs in the other platforms have a total width of $4N\frac{\lfloor \alpha_2 m_N \rfloor}{\delta^4}$ and can be packed into a bin B^* of height 1 and width $w(B^*) \leq \frac{4N\alpha_2 m_N}{\delta^4} \leq \tilde{\delta} m_N \leq \tilde{\delta} m_2$ (using $\alpha_2 \leq \tilde{\delta}\delta^8/(4C)$ small enough). For an illustration we refer to Figure 17.

The remaining half of the gap in P_1 has width at least $(\delta^4/16)m_1 > m_2$ using $m_1 > \gamma m_2$ and $\gamma > \frac{16}{\delta^4}$. The last inequality holds since $\gamma = 2N_1/\delta' > 16/\delta^4$ (by choosing $\delta' = \delta^4/12 < \delta^4/8$). This helps us now to obtain a 2-approximate solution.

How to insert a thin bin of height 1**?** As starting point we have a schedule for almost all jobs with makespan $\leq 3/2 + \epsilon$ on the first platform and $1 + \epsilon$ on the next platforms as in case *D.A.* In addition there is a gap of width $\geq (\delta^4/16)m_1 \geq m_2$ and height $1/2$ on the first platform and there is a bin with non-processed jobs with height 1 and width $w(B^*) \leq \tilde{\delta} m_2$. The following lemma shows how to modify our schedule. Notice that the inequality $m_2 \geq (1/\delta^2 + 2)\tilde{\delta} m_2 \geq (1/\delta^2 + 2)w(B^*)$ holds simply using our specification of $\tilde{\delta} = (1/(48C)\delta^{10}$.

Lemma 4.1 *Let P_1 and P_2 be the widest and second widest platform with makespan $\leq 3/2 + \epsilon$ on P_1 and $\leq 1 + \epsilon$ on P_2. In addition let B^* be a bin with height 1 and width $w(B^*)$.*

If there is a gap of height $1/2$ and width $\geq (\delta^4/16)m_1 \geq m_2$ on P_1, $w(P_2) \geq (1/\delta^2 + 2)w(B^*)$ and $\epsilon \leq 1/10$, then we can generate a schedule for the jobs in P_1, P_2 and B^* on platforms P_1 and P_2 with makespan $C_{max}(P_1), C_{max}(P_2) \leq 2$.

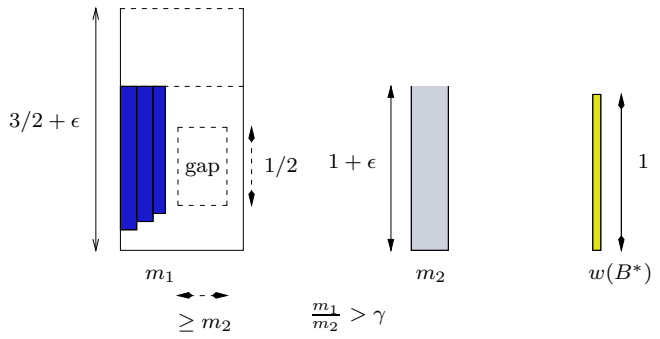

Figure 17: Scenario in lemma 4.1

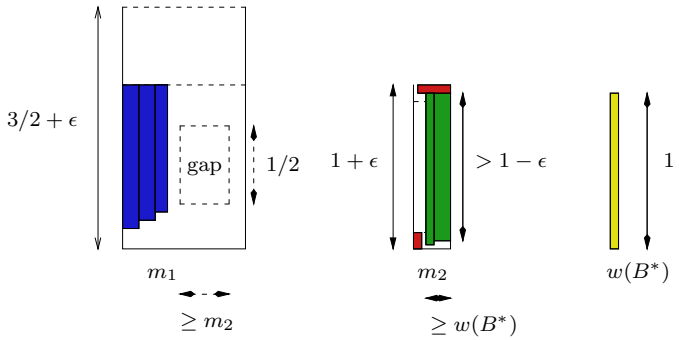

Figure 18: Algorithm for Lemma 4.1 - case 1, step 1

Proof: Depending on the second widest platform P_2 we have the following cases:

Case 1: The total width of the set X of jobs with processing time $> 1-\epsilon$ on platform P_2 is $\geq w(B^*)$; see Figure 18. Similar to Case 1 in Lemma 3.1 we remove two horizontal layers of height 2ϵ and move all $> 1-\epsilon$ jobs down to starting time 0. This generates a free block of width $\geq w(B^*)$ from starting time 1 on P_2. The bin B^* can be placed there, and, therefore we obtain a makespan of 2 on P_2. The removed horizontal layers are placed on top of P_1 and generate a makespan of $3/2+3\epsilon+2\epsilon = 3/2+5\epsilon \leq 2$ on P_1 (using $\epsilon \leq 1/10$); see also Figure 19.

Case 2: The total width of the set X_s of jobs with the same rounded starting time $s = a\delta^2$ and processing times $p_j \in (1/2, 1-\epsilon]$ is $\geq w(B^*)$; see Figure 20. First we enlarge the gap on P_1 by the additive value $1/2-\epsilon$; see Figure 21. Next we remove X_s from platform P_2 with total width at most m_2 and height $\leq 1-\epsilon$ and place X_s into the enlarged gap on platform P_1. This gives a makespan of $3/2+\epsilon+(1/2-\epsilon) = 2$ on platform P_1. Finally we use the space generated by X_s on platform P_2 to insert B^* (similar to case 2 in Section 3.1). Here we have to increase the new gap by the height $1/2$ and obtain a solution with makespan $1+\epsilon+1/2 = 3/2+\epsilon$ on P_2; see Figure 22.

Case 3: For each rounded starting time s, the total width $w(X_s) \leq w(B^*)$ and $w(X) \leq w(B^*)$ on platform P_2. The total width of $X' = X \cup \bigcup_s X_s$ is at most $(1/\delta^2+1)w(B^*)$. Using $m_2 \geq (1/\delta^2+2)w(B^*)$, at least $w(B^*)$ of the machines on P_2 are not used by X'. We may assume that the last $w(B^*)$ machines execute only jobs with processing time $\leq 1/2$; see Figure 23. We consider the horizontal line ℓ at height $3/4$. All jobs with processing time $p_j \leq 1/2$ that intersect ℓ can be moved from platform P_2 into the gap on P_1 with height $1/2$. This can be done since the total width of these jobs is at most $m_2 \leq (\delta^4/16)m_1$. Furthermore we can move all jobs that lie completely above ℓ also to platform P_1. This generates a makespan of $3/2+\epsilon+1/4+\epsilon \leq 7/4+2\epsilon \leq 2$ for $\epsilon \leq 1/8$. Removing both sets from P_2 generates a schedule where at least half of the machines on P_2 do not execute jobs from step $3/4$ on. Here we can place B^* and obtain a makespan of $3/4+1 = 7/4$ on P_2. For an illustration we refer to Figure 24. □

4.2 Case D.B

Suppose that $m_1/m_2 > \gamma$, $m_2/m_i \leq \gamma'$ for $i = 3, \ldots, N$ and $N > C/\delta^4$. In this case we partition the platforms into three groups $\{P_1\}$, $\mathcal{B}_0 = \{P_2, \ldots, P_{N^*+1}\}$ with $N^* \in \{N_0', \ldots, N_0'+N_1\}$ and $\mathcal{B}_1 = \{P_{N^*+2}, \ldots, P_N\}$ (where $\mathcal{B}_1$ is divided into groups with exactly N_1 platforms). Let $\bar{m} = m_{N^*+2}$ be the largest width in $\mathcal{B}_1$. All jobs with width

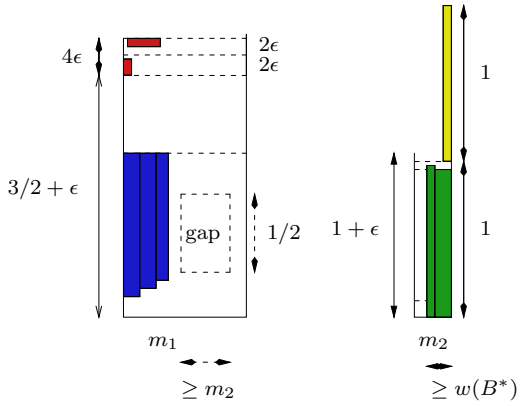

Figure 19: Algorithm to handle case D with $K = 1$ - case 1, step 2

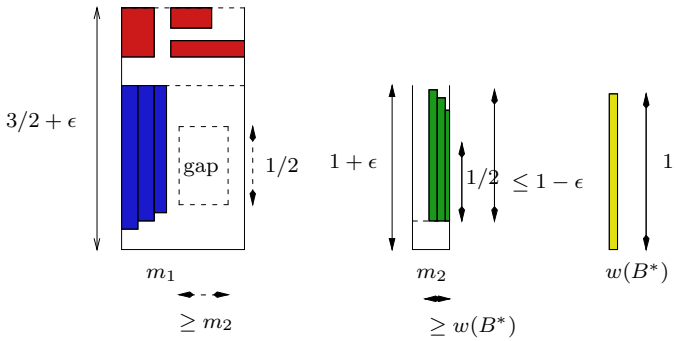

Figure 20: Algorithm for Lemma 4.1 - case 2, step 1

$> m_2$ go to P_1. After guessing the large wide jobs for $\mathcal{B}_0$ with processing time $> \delta$ and width in $[\alpha_2 m_N, m_2]$, the remaining large wide jobs are placed in P_1 or in $\mathcal{B}_1$. The number of guessed jobs for $\mathcal{B}_0$ is bounded by a constant $O((N_0' + N_1)(1/\delta)(m_2/(\alpha_2 m_N))) \leq poly(1/\delta)$. Notice all remaining large jobs with width $> \bar{m}$ go to P_1, using $\bar{m} \geq m_N \geq \alpha_2 m_N$. For the large wide jobs with width $> \alpha_1 m_1$ placed in P_1 we guess the starting times. The total number of these jobs is also bounded by a constant $O((1/\alpha_1)(1/\delta))$.

We guess here an approximate load vector $\Pi_{i,a,h}$ for $P_i \in \{P_1\} \cup \mathcal{B}_0$ with multiples of $\alpha_2 m_N$ for large and huge narrow jobs on platforms in $\mathcal{B}_0$ and multiples of $\alpha_1 m_1$ and $\bar{m}$ for large and huge narrow jobs on platform P_1, respectively. Again, the number of possible load vectors can be bounded by a polynomial in n. For platform P_1 large and huge narrow jobs with width $\leq \alpha_1 m_1$ and $\leq \bar{m}$) are placed fractionally via the linear program relaxation (see algorithm in [5]). The large wide jobs with width $> \alpha_1 m_1$ are placed during the guessing phase and the huge jobs with width $> \bar{m}$ finish all at the same time. Similar to case $D.A$, the algorithm in [5] computes a solution with makespan on P_1 of $(3/2 + \epsilon)$, where P_1 contains a gap of height $1/2$ and width $(\delta^4/8)m_1$, with makespan on $P_i \in \mathcal{B}_0$ of $(1 + \epsilon)$, and with makespan of 2 on the remaining platforms.

Almost all jobs are placed in the schedule with an exception of a bin B^+ with height $\leq 1/2$ and total width $\leq (4/\delta^4)\alpha_1 m_1 \leq (\delta^4/16)m_1$ using $\alpha_1 \leq \delta^8/64$ (with large narrow jobs in P_1), a bin B^* of height 1 and width $\leq |\mathcal{B}_0|\frac{4\alpha_2 m_N}{\delta^4} \leq \tilde{\delta} m_N \leq \tilde{\delta} m_2$ using $\alpha_2 \leq \tilde{\delta}\delta^4/(4(N_0' + N_1))$ for the other platforms, $2N_1$ bins of height 1 and width $\leq \bar{m}$ (due to the rounding in $\mathcal{B}_1$) plus additional bins for some huge narrow jobs assigned to P_1. Huge narrow jobs in P_1 have width $\leq \bar{m}$. In order to obtain a feasible solution in P_1, a set with huge narrow jobs of total size $\leq 3\bar{m}$ is removed from the schedule for each rounded starting time (otherwise these jobs do not fit there corresponding to the approximate load vector). These removed jobs fit into ≤ 5 bins of height 1 and width $\bar{m}$. Counting over all starting times, this gives at most $5(1 + 2\delta)/\delta^2 \leq 6/\delta^2$ bins. In total we obtain $6/\delta^2 + 2N_1$ bins of height 1 and width $\bar{m}$. Bin B^+ can be placed into the gap on platform P_1 and bin B^* can be inserted into platforms P_1 and P_2 using Lemma 4.1. For the other bins of width $\bar{m}$ we need $3(6/\delta^2 + 2N_1)$ bins of height $1 + \epsilon$ in $\mathcal{B}_0$ to apply Lemma 3.2. Using $N_0' \geq 3(6/\delta^2 + 2N_1) + 2$ we obtain a schedule for all jobs with makespan bounded by 2.

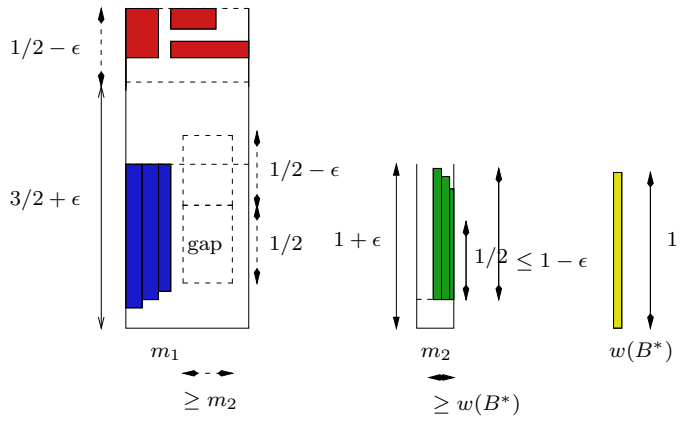

Figure 21: Algorithm for Lemma 4.1 - case 2, step 2

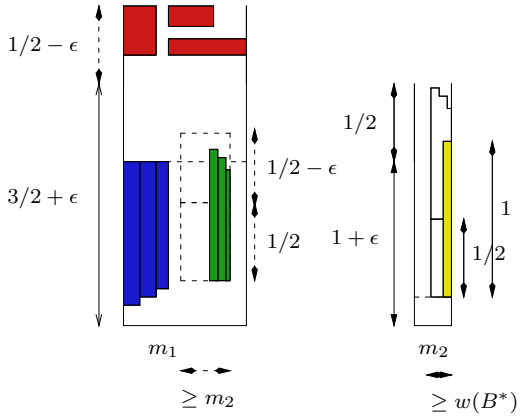

Figure 22: Algorithm for Lemma 4.1 - case 2, step 3

4.3 Case D.C

Here we suppose that $m_1/m_2 > \gamma$, $m_2/m_K \leq \gamma'$, $m_2/m_{K+1} > \gamma'$ with $K \geq N_0' + 2$. In this case we have also three groups $\{P_1\}$, $\mathcal{B}_0 = \{P_2, \ldots, P_{N_0'+1}\}$, and $\mathcal{B}_1 = \{P_{N_0'+2}, \ldots, P_N\}$. Let $\bar{m} = m_{N_0'+2}$ be the largest width in $\mathcal{B}_1$. All jobs with width $> m_2$ go to P_1. In this case we guess the positions of the large and huge jobs with width $> \alpha_1 m_1$ for P_1 and with width $\in [\alpha_2 m_{N_0'+2}, m_2]$ placed into $\mathcal{B}_0$. Notice that the number of large and huge wide jobs in $\mathcal{B}_0$ is at most $|\mathcal{B}_0| m_2/(\alpha_2 m_{N_0'+2}) \leq N_0'\gamma'/\alpha_2 = O(1)$.

The non-placed huge and large jobs with width $> \bar{m}$ go to P_1. Again we use an approximate load vector $\Pi_{i,a,h}$ for $\mathcal{B}_0 \cup \{P_1\}$ with multiples of $\alpha_2 m_{N_0'+2}$ for large and huge narrow jobs on platforms in $\mathcal{B}_0$ and multiples of $\alpha_1 m_1$ and $\bar{m}$ for large and huge narrow jobs on P_1, respectively. Notice that $\bar{m} = m_{N_0'+2} > \alpha_2 m_{N_0'+2}$. Again we obtain a schedule with a $(3/2 + \epsilon)$ approximate makespan on P_1 (including a gap of height $1/2$), an $(1 + \epsilon)$ approximate makespan on $P_2, \ldots, P_{N_0'+1}$ and a 2 approximate makespan on the remaining platforms. The non-assigned jobs corresponding to $\mathcal{B}_0$ fit into a bin B^* of width $|\mathcal{B}_0| 4\alpha_2 m_{N_0'+2}/\delta^4 \leq \tilde{\delta} m_2$ using $\alpha_2 \leq \tilde{\delta}\delta^4/(4N_0')$. For P_1 the large narrow jobs with $p_j \leq 1/2$ and $q_j \leq \alpha_1 m_1$ give an extra load $4/\delta^4 \alpha_1 m_1 \leq (\delta^4/16) m_1$ using $\alpha_1 \leq \delta^8/64$ and fit into the created gap of height $1/2$. The huge jobs in P_1 with $q_j \leq \bar{m}$ are placed via a linear program. For these jobs we obtain an extra load with 5 bins of width $\bar{m}$ for each rounded starting time. This gives again $6/\delta^2 + 2N_1$ bins of width $\bar{m}$ that can be merged with platforms in $\mathcal{B}_0$. Using Lemma 3.2 with $N_0' \geq 3(6/\delta^2 + 2N_1) + 2$ we obtain a schedule with makespan at most 2.

4.4 Case D.D

Here we suppose that $m_1/m_2 > \gamma$, $m_2/m_K \leq \gamma'$ and $m_2/m_{K+1} > \gamma'$ for $K < N_0' + 2$. In this case we use three groups $\{P_1\}$, $\mathcal{B}_0 = \{P_2, \ldots, P_K\}$ and $\mathcal{B}_1 = \{P_{K+1}, \ldots, P_N\}$. Again, all jobs with width $> m_2$ go to P_1. Here the algorithm

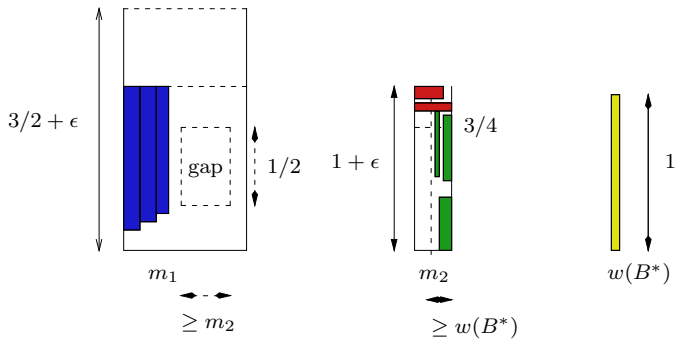

Figure 23: Algorithm for Lemma 4.1 - case 3, step 1

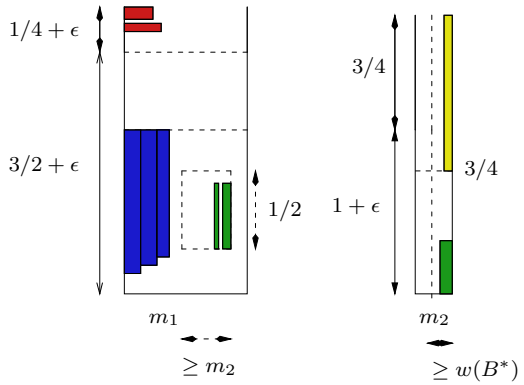

Figure 24: Algorithm for Lemma 4.1 - case 3, step 2

guesses the large and huge jobs with width $q_j > \alpha_1 m_1$ for P_1 and the large and huge jobs with width $q_j \in [\alpha_2 m_K, m_2]$ for $\mathcal{B}_0$. All remaining jobs with width $q_j > \max(m_{K+1}, \alpha_2 m_K)$ and $p_j > \delta$ go to P_1. Here our algorithm generates a schedule with makespan $\leq (3/2+\epsilon)$ for P_1 (including a gap of height $1/2$), makespan $\leq 1+\epsilon$ for each platform in $\mathcal{B}_0$ and makespan 2 for each platform in $\mathcal{B}_1$. On the other hand, there are several non-assigned jobs. For $\mathcal{B}_0$ there is a set of non-assigned jobs with width $\leq (K-1)\frac{4\alpha_2 m_K}{\delta^4} \leq N_0' \frac{4\alpha_2 m_2}{\delta^4} \leq \tilde{\delta} m_2$ and height 1; using $\alpha_2 \leq \tilde{\delta}\delta^8/(4N_0')$. For P_1 we have a set with load $(4/\delta^4)\alpha_1 m_1 \leq (\delta^4/16)m_1$ and height $1/2$ that fits again in the corresponding gap of P_1 (using $\alpha_1 = \delta^8/64$). In addition for P_1 we have a load of $3\max(\alpha_2 m_K, m_{K+1})$ with non-assigned huge jobs for each starting time. Counting over the starting times the total load is at most $3(1+2\delta)/\delta^2 \max(\alpha_2 m_K, m_{K+1}) \leq 4/\delta^2 \max(\alpha_2 m_K, m_{K+1})$. Finally we have $2N_1$ bins containing non-assigned jobs with width $\leq m_{K+1}$. Using $\gamma' = 2N_1/\delta'$ and $m_2/m_{K+1} > \gamma'$, we have $2N_1 m_{K+1} \leq 2N_1 m_2/\gamma' \leq \delta' m_2$.

Let us specify $\alpha_2 = \min(\delta'\delta^2/4, \tilde{\delta}\delta^8/(4N_0'))$.

Case 1: $m_{K+1} > \alpha_2 m_K$. In this case, the load $4/\delta^2 \max(\alpha_2 m_K, m_{K+1}) \leq (4/\delta^2)m_{K+1} < (4/\delta^2)(m_2/\gamma') = (2/(N_1\delta^2))\delta' m_2 = \delta'' m_2$ using $\delta'' = 2/(N_1\delta^2)\delta'$. To combine a bin with the width $(\delta' + \delta'' + \tilde{\delta})m_2$ with two platforms P_1 and P_2 of width m_1 and m_2 we need the property $m_2 \geq (1/\delta^2 + 2)(\delta' + \delta'' + \tilde{\delta})m_2$; see Lemma 4.1.

Case 2: $m_{K+1} \leq \alpha_2 m_K$. In this case, the load $4/\delta^2 \max(\alpha_2 m_K, m_{K+1}) \leq (4/\delta^2)\alpha_2 m_K \leq \delta' m_K \leq \delta' m_2$. To combine here a bin with width $(2\delta' + \tilde{\delta})m_2$ we need the property $m_2 \geq (1/\delta^2 + 2)(2\delta' + \tilde{\delta})m_2$; see Lemma 4.1.

In both cases above the extra load can be merged with the two widest platforms. Using $\delta' = (1/12)\delta^4$, $\tilde{\delta} = 1/(48C)\delta^{10}$ and $N_1 \geq 1$ we obtain $\tilde{\delta} \leq \delta' \leq \delta'' = 2/(N_1\delta^2)\delta'$ and $\delta'' \cdot (1/\delta^2 + 2) = 2\delta'/(N_1\delta^2)(2/\delta^2) \leq (4/12)(\delta^4/\delta^2)(1/\delta^2) = 1/3$. This implies that the properties in both cases above are satisfied and Lemma 4.1 can be applied. In total we obtain a schedule with makespan at most 2.

5 Conclusion

The paper here closes the gap between the best possible approximation ratio and the non-approximability bound for scheduling parallel jobs on identical and heterogeneous platforms. The main open question is to identify important

special cases where the running time and the absolute ratio of the approximation algorithm can be further improved.

References

[1] N. Bansal, X. Han, K. Iwama, M. Sviridenko, and G. Zhang. Harmonic algorithm for 3-dimensional strip packing problem. *ACM-SIAM Symposium on Discrete Algorithms (SODA 2007)*, 1197-1206.

[2] M. Bougeret, P.-F. Dutot, K. Jansen, C. Otte, and D. Trystram. A fast 5/2-approximation algorithm for hierarchical scheduling. *European Conference on Parallel and Distributed Computing (Euro-Par 2010)*, Ischia, Springer LNCS 6272, 157-167.

[3] M. Drozdowski. *Scheduling for Parallel Processing.* Computer Communications and Networks, Springer, 2009.

[4] J. Du and J.Y.T. Leung. Complexity of scheduling parallel task systems. *SIAM Journal on Discrete Mathematics*, 2(4):473–487, 1989.

[5] P.F. Dutot, K. Jansen, C. Robenek and D. Trystram. A $(2+\epsilon)$ - approximation for scheduling parallel jobs in platforms. *European Conference on Parallel and Distributed Computing (Euro-Par 2013)*, Aachen, Springer LNCS 8097, 78-89 and *Technical Report 1217 (2013)*, University of Kiel.

[6] K. Jansen. A $(3/2+\epsilon)$ approximation algorithm for scheduling malleable and non-malleable parallel tasks. *ACM Symposium on Parallelism in Algorithms and Architectures (SPAA 2012)*, Pittsburgh, 224-235.

[7] C.C. Lee and D.T. Lee. A simple on-line bin packing algorithm. *Journal of the ACM*, 32(3): 562-572, 1985.

[8] J. Remy. Resource constrained scheduling on multiple machines. *Information Processing Letters*, 91: 177-182, 2004.

[9] U. Schwiegelshohn, A. Tchernykh, and R. Yahyapour. Online scheduling in grids. *IEEE International Parallel and Distributed Processing Symposium (IPDPS 2008)*, Miami, 1-10.

[10] A. Tchernykh, J. Ramirez, A. Avetisyan, N. Kuzjurin, D. Grushin, and S. Zhuk. Two level job-scheduling strategies for a computational grid. *Conference on Parallel Processing and Applied Mathematics (PPAM 2005)*, Poznan, Springer LNCS 3911, 774-781.

[11] A. Tchernykh, U. Schwiegelshohn, R. Yahyapour, and N. Kuzjurin. On-line hierarchical job scheduling on grids with admissible allocation. *Journal of Scheduling* 13(5): 545-552, 2010.

[12] S.N. Zhuk. Approximation algorithms to pack rectangles into several strips. *Discrete Mathematics and Applications*, 16(1):73-85, 2006.

6 Appendix: $(2+\epsilon)$ approximation algorithm

In the appendix we give the details about the $(2+\epsilon)$ approximation algorithm. In the first scenario (1) where $m_1/m_i \leq \gamma$ for $i = 1, \ldots, N$ there are two subcases depending on the number N of platforms and a constant C:

6.1 Case A: $N \leq C/\delta^4$

First we round the starting and execution times of each large job to a multiple of δ^2. This increase the makespan on each platform from 1 to $1+2\delta$. Next we guess by enumeration the assignment for the large wide jobs with width $> \alpha m_N$ (i.e. the assigned platform and rounded starting time) and the approximate load of large narrow jobs with width $\leq \alpha m_N$ for each platform, rounded starting and execution time. Afterwards, the large narrow jobs are allocated via an assignment linear program to platforms and starting times. All fractional assigned large narrow jobs can be placed into a bin B^* of height 1 and width $\leq N\frac{4\lfloor \alpha m_N \rfloor}{\delta^4} \leq 4N\frac{\alpha m_N}{\delta^4}$ (see also Figure 1). Using $\alpha \leq \frac{\delta^8\tilde{\delta}}{4C}$ (where $\tilde{\delta} = 1/(48C)\delta^{10}$), the total width of B^* is at most $4N\frac{\alpha m_N}{\delta^4} \leq \frac{4C}{\delta^4}\frac{\delta^(\tilde{\delta}}{4C}\frac{m_N}{\delta^4} = \tilde{\delta} m_N \leq \tilde{\delta} m_1$.

Notice that we slightly modified (compared to the algorithm in [5]) the value of α in our new approach to reduce the total width of the additional bin B^*. In the algorithm we suppose that $m_N \geq 2/\alpha$. Using $m_N \geq 2/\alpha$ we obtain $\lfloor \alpha m_N \rfloor \geq \alpha m_N - 1 \geq \alpha m_N/2 \geq 1$. Using this property and the inequality $m_N \geq m_1/\gamma$, the number of large wide jobs within one platform is bounded by $\frac{1}{\delta}\frac{m_1}{\lfloor \alpha m_N \rfloor} \leq \frac{1}{\delta}\frac{m_1}{\alpha m_N/2} \leq \frac{1}{\delta}\frac{2\gamma}{\alpha} = O(1)$. Otherwise if $m_N \leq 2/\alpha$, the number of processors $m_i \leq m_1 \leq m_N\gamma \leq 2\gamma/\alpha$ is constant in every platform P_i. This implies that we can have only a constant number of large jobs in the entire instance. In this case we do not distinguish between large narrow and wide jobs and can guess the starting times for all large jobs. In both cases there are at most $poly(1/\delta)$ many large wide jobs in any instance corresponding to case A. Since $N = O(1/\delta^4)$, the number of possible assignments for the large wide jobs and the number of possible load vectors are bounded by a function in $1/\delta$ (see also [5]). Finally, the small jobs are also placed via a linear program and a 2D strip packing procedure to horizontal layers of height δ^2, where the width is given by m_i minus the total width of guessed large wide jobs and load values.

The medium jobs of total load $\leq (\epsilon/10)m_1$ are processed using list scheduling on top of the largest platform at the end. This will increase the length of the schedule on P_1 by at most $2\epsilon/10 \leq \epsilon/3$. The strip packing procedure places all small jobs into horizontal layers on the platforms by slightly increasing the height of each layer. For each layer we obtain an integral packing of height $\frac{(1+\epsilon')^2}{(1-\epsilon')}\delta^2 + (4M+1)\delta^5$. Using $\epsilon' = \epsilon/90$ this can be bounded by $(1+\epsilon/5)\delta^2 + (4M+1)\delta^5$. Since there are at most $(1+2\delta)/\delta^2$ layers, the makespan in every platform is increased by at most $(1+2\delta)/\delta^2((\epsilon/10)\delta^2 + (4/(\epsilon')^2+1)\delta^5) \leq (1+2\delta)(\epsilon/10+5\delta) \leq \epsilon/3$. This implies a schedule length of $1+2\delta+2\epsilon/3 \leq 1+\epsilon$ on each platform P_i. Executing B^* additionally on the first platform gives a $(2+\epsilon)$-approximation.

6.2 Case B: $N > C/\delta^4$

The set $\mathcal{B}$ of all platforms here is divided into two blocks of platforms:

- $\mathcal{B}_0$ with the widest platforms, where the cardinality $|\mathcal{B}_0| \in \{N_0, \ldots, N_0+N_1\}$ and $N_0, N_1 = O(1/\delta^4)$,
- $\mathcal{B}_1$ with the remaining platforms $P_{|\mathcal{B}_0|+1}, \ldots, P_N$.

Afterwards $\mathcal{B}_1$ is partitioned into $L = \max\{0, \lfloor \frac{N-N_0}{N_1} \rfloor\}$ groups each containing exactly $N_1 = \bar{C}/\delta^4$ platforms where $\bar{C} \leq C$. Again we round the starting and execution times of each large job allocated to platforms in $\mathcal{B}_0$ to multiples of δ^2. This generates a makespan bounded by $1+2\delta$ on these platforms. Next we guess by enumeration an assignment of large wide jobs with width $> \alpha m_N$ into $\mathcal{B}_0$. This enumeration can be done in polynomial time, since $\mathcal{B}_0$ contains at most $((1+2\delta)/\delta)(1/\alpha)(m_1/m_N)|\mathcal{B}_0| \leq O(1/(\delta\alpha\gamma\delta^4)) = O(1)$ many large wide jobs. Moverover, the approximate load for large narrow jobs allocated to rounded starting times, execution times and platforms in $\mathcal{B}_0$ are guessed by enumeration. These load values generate some gaps in $\mathcal{B}_0$ reserved for large narrow jobs. For large jobs assigned to $\mathcal{B}_1$ we use a harmonic rounding [1, 5, 7]; i.e. we round execution times $p_j \in (1/(i+1), 1/i]$ to $\tilde{p}_j = 1/i$ and obtain rounded execution times $\{1/K, \ldots, 1/3, 1/2, 1\}$. The harmonic rounding has the effect that the makespan increases from 1 to ≈ 1.691 in the platforms in $\mathcal{B}_1$.

Each group in $\mathcal{B}_1$ contains exactly N_1 platforms. For each group B_ℓ in $\mathcal{B}_1$ we round the number of processors of each platform up to the number of processors $\tilde{m}_\ell$ of the largest platform in B_ℓ. Then using a linear program we fractionally assign large narrow jobs into the reserved gaps of $\mathcal{B}_0$, small jobs into horizontal layers of $\mathcal{B}_0$ and the remaining jobs into the L rounded groups $\tilde{B}_\ell$ of $\mathcal{B}_1$. The fractional assigned large narrow jobs in $\mathcal{B}_0$ can be packed into a bin B^* with total height (schedule length) 1 and width $\leq |\mathcal{B}_0| 4\frac{\lfloor \alpha m_N \rfloor}{\delta^4} \leq \tilde{\delta} m_N \leq \tilde{\delta} m_1$ (using the specification $\alpha \leq \frac{\delta^4\tilde{\delta}}{4(N_0+N_1)} \leq \frac{\delta^4\tilde{\delta}}{4|\mathcal{B}_0|}$). Here we slightly modified the value for α again to reduce the total width of B^*.

The medium jobs can be scheduled again on top of the largest platform P_1 with total execution time $\leq \epsilon/3$. For each horizontal layer in $\mathcal{B}_0$ we get $M+1$ fractional small jobs (M wide and one narrow job). Therefore the strip packing procedure can pack all small jobs including these fractional assigned ones with height $\leq \frac{(1+\epsilon')^2}{(1-\epsilon')}\delta^2 + (5M+2)\delta^5$ (see also [5]). Therefore, the makespan in every platform in $\mathcal{B}_0$ increases by at most

$$(1+2\delta)/\delta^2((\epsilon/10)\delta^2 + (5/(\epsilon')^2+2)\delta^5) \leq (1+2\delta)(\epsilon/10+6\delta) \leq \epsilon/3.$$

This generates a makespan of $1+2\delta+2\epsilon/3 \leq 1+\epsilon$ on the plaforms in $\mathcal{B}_0$.

For each rounded group $\tilde{B}_\ell$ we get M fractional assigned wide jobs plus one fractional assigned narrow job. The integral assigned rounded jobs and the $M+1$ fractional assigned jobs for each group $\tilde{B}_\ell$ are placed via a strip packing procedure into $2N_1$ bins (or platforms) with $\tilde{m}_\ell$ processors and makespan 1. Here a specific strip packing procedure with a tall-not-sliced property (due to the harmonic rounding of the large jobs) and the property $N_1 = \Omega(1/\delta^4)$ is used; see also [1, 5]. The schedule can be converted into a schedule for the original N_1 platforms per group using a shifting argument; i.e. moving each bin with $\tilde{m}_\ell$ processors to the next group with a larger number of processors. Then all platforms in $\mathcal{B}_1$ have a makespan of 2. In addition we have $2N_1$ blocks of height ≤ 1 and width $\tilde{m} \leq m_{|\mathcal{B}_0|+1}$ that are not assigned yet due to the rounding of the platform widths. Therefore, our algorithm computes a solution where the makespan in the first $|\mathcal{B}_0|$ platforms is bounded by $(1+\epsilon)$ and in the remaining platforms is bounded by 2, respectively. In total, we have non-assigned jobs in a bin B^* with height 1 and width $\tilde{\delta}m_1$ and in $2N_1$ bins with height 1 and width $\tilde{m} \leq m_{|\mathcal{B}_0|+1}$ (see also Figure 2 for an illustration).

Using $N_0 \geq 2N_1+1$, the bin B^* and these $2N_1$ bins can be processed on top of the platforms $P_1, \ldots, P_{|\mathcal{B}_0|}$ (using $|\mathcal{B}_0| \geq N_0$) causing a makespan $\leq (2+\epsilon)$ on these platforms.

In the second scenario (2) there is a number $K \in \{1, \ldots, N-1\}$ such that $m_1/m_i \leq \gamma$ for all $i \leq K$ and $m_1/m_{K+1} > \gamma$. In this scenario we have to distinguish also two subcases depending on a constant $N_0' = O(1/\delta^4)$:

6.3 Case C: $K \geq N_0'$

This case can be handled similar to case B. Here a large job j is called wide, if $q_j \geq \lfloor \alpha m_{N_0'} \rfloor$; otherwise j is called narrow. Our algorithm partitions the set $\mathcal{B}$ of N platforms into $\mathcal{B}_0 = \{P_1, \ldots, P_{N_0'}\}$ and $\mathcal{B}_1 = \{P_{N_0'+1}, \ldots, P_N\}$. Afterwards $\mathcal{B}_1$ is partitioned into $L = \lceil \frac{N-N_0'}{N_1} \rceil$ groups where the first $L-1$ groups contain exactly N_1 platforms and the last group contains maybe less than N_1 platforms. The algorithm adds some dummy platforms with $\tilde{m}_L = m_{N_0'+(L-1)N_1+1}$ processors to the last group such that all groups have exactly N_1 platforms. Then it computes here a solution using the same method as in case B. It generates a solution where the makespan in the first N_0' platforms is at most $(1+\epsilon)$ and in the remaining platforms is at most 2; see Figure 3 for an illustration. Furthermore, there are non-assigned jobs in $2N_1$ bins of height ≤ 1 and width $\tilde{m} \leq m_{N_0'+1}$ and in an additional bin B^* of height 1 and width $N_0' 4\frac{\lfloor \alpha m_{N_0'} \rfloor}{\delta^4} \leq \tilde{\delta} m_{N_0'}$ (using $\alpha \leq \frac{\delta^4 \tilde{\delta}}{4N_0'}$). Using $N_0' \geq 2N_1+1$ these bins can be processed on the first N_0' platforms causing a makespan $\leq (2+\epsilon)$.

6.4 Case D: $K < N_0'$

.

Here our algorithm divides the set $\mathcal{B}$ of N platforms into $\mathcal{B}_0 = \{P_1, \ldots, P_K\}$ and $\mathcal{B}_1 = \{P_{K+1}, \ldots, P_N\}$ and, afterwards, partitions $\mathcal{B}_1$ into $L = \lceil \frac{N-K}{N_1} \rceil$ groups. Here a job is called wide if $q_j \geq \lfloor \alpha m_K \rfloor$. Then, the algorithm computes a solution via the method in case B where the makespan in the first K platforms is bounded by $(1+\epsilon)$ and the makespan in the remaining platforms is bounded by 2. Similarly to the previous case B we obtain non-assigned jobs in $2N_1$ bins of height ≤ 1 and width $\leq m_{K+1}$ plus in one bin B^* of height 1 and width $\leq K\frac{4\lfloor \alpha m_K \rfloor}{\delta^4} \leq N_0'\frac{4\lfloor \alpha m_K \rfloor}{\delta^4} \leq \tilde{\delta} m_K$ using $\alpha \leq \tilde{\delta}\frac{\delta^4}{4N_0'}$. In the worst case $K=1$ and we have to process the jobs in B^* and the $2N_1$ bins here only on platform P_1. Now the gap value $\gamma = 2N_1/\delta'$ with $\delta' = (1/12)\delta^4$ comes into the game.

Using this specification, the total width of the $2N_1$ bins is at most $2N_1 m_{K+1} \leq 2N_1 \frac{m_1}{\gamma} = \delta' m_1$. This together with B^* gives one additional bin to be placed of height 1 and width $\tilde{\delta} m_K + \delta' m_1 \leq 2\delta' m_1$; see also Figure 4. Our algorithm could places this bin of height 1 and width $2\delta' m_1$ on platform P_1. This implies a makespan of $(2+\epsilon)$ on the first platform. On the other hand, to convert this into a 2-approximate solution is more complicated. In Section 4 and Appendix B below we show how to modify the previous algorithm in case D to obtain a 2-approximate solution.