Journal of Combinatorial Optimization (2024) 47:34
https://doi.org/10.1007/s10878-024-01118-w

Randomized approximation schemes for minimizing the weighted makespan on identical parallel machines

Ruiqing Sun[1]

Accepted: 22 February 2024 / Published online: 31 March 2024

Abstract
In this paper, we discuss scheduling problems with m identical machines and n jobs where each job has to be assigned to some machine. The objective is to minimize the weighted makespan of jobs, i.e., the maximum weighted completion time of jobs. This scheduling problem is a generalization of minimizing the makespan on parallel machine scheduling problem. We present a $(2-\frac{1}{m})$-approximation algorithm and a randomized efficient polynomial time approximation scheme (EPTAS) for the problem. We also design a randomized fully polynomial time approximation scheme (FPTAS) for the special case when the number of machines is fixed.

Keywords Scheduling · Weighted makespan · Approximation algorithm · Efficient polynomial time approximation scheme · Fully polynomial time approximation scheme

1 Introduction

We consider a scheduling model with m identical machines $\mathcal{M} = \{M_1, M_2, \ldots, M_m\}$ and n jobs $\mathcal{J} = \{J_1, J_2, \ldots, J_n\}$. Each job has a processing time p_j and a weight w_j and each job has to be assigned to some machine. The objective is to minimize the maximum weighted completion time of jobs. Let C_j be the completion time of job J_j. For a parameter $p \in \mathbb{R}$, the L_p-norm of the job weighted completion time vector $(w_1C_1, \ldots, w_nC_n)$ can be defined as $(\sum_{j=1}^{n}(w_jC_j)^p)^{\frac{1}{p}}$. When $p = 1$, it is the classical total weighted completion time problem and can go back to the work of Smith (1956). For minimizing the weighted completion time, Sahni (1976) gave a fully polynomial time approximation scheme (FPTAS) for this problem on identical parallel machine setting with a fixed m. Kawaguchi and Kyan (1986) proved that list scheduling in order of nonincreasing order of w_j/p_j is an approximation algorithm with an

✉ Ruiqing Sun
ruiqing2020@126.com

[1] School of Mathematics and Statistics, Yunnan University, Kunming, People's Republic of China

approximation ratio of $\frac{1}{2}(1+\sqrt{2})$ for this problem when m is not fixed. Skutella and Woeginger (2000) designed a polynomial time approximation scheme (PTAS) for this problem when m is not fixed. For jobs with release date setting, Phillips et al. (1998) showed that a given preemptive schedule on single machine can produce a nonpreemptive schedule while increasing the total weighted completion time by at most a factor of 2. Afrati et al. (1999) proposed a PTAS for minimizing weighted completion time with release dates on identical parallel machines. For more general setting with unrelated parallel machines, i.e. the job J_j has a processing time p_{ij} on machine M_i, Skutella (2001) designed a 3/2-approximation algorithm based on the convex programming relaxation. Bansal et al. (2016) proposed the first $(3/2-c)$-approximation algorithm for this problem, for some constant $c = 10^{-7} > 0$ and they introduced a novel rounding scheme yielding strong negative correlation for the first time. Li (2020) used strong negative correlation as a black box and improved this approximation ratio to $(3/2-1/6000)$ based on the simpler time-indexed linear programming relaxation. Im and Shadloo (2020) via iterative fair contention-resolution techniques achieve a significantly stronger negative correlation and improved approximation ratio to 1.448. Baveja et al. (2023) improved the value of the negative correlation of Bansal et al. (2016) from 1/108 to 1/27, thus further improved the constant c in the algorithms of Bansal et al. (2016) for this problem. When $p = +\infty$, it is the maximum value of w_jC_j. For minimizing the weighted makespan (the maximum weighted completion time), Feng and Yuan (2007) first introduced the weighted makespan $WC_{\max}$ on a single-machine scheduling problem. Li (2015) studied the online single machine scheduling problem to minimize the weighted makespan $WC_{\max}$ in which jobs arrive over time. For the general online problem in Li (2015), Chai et al. (2018) provided two on-line algorithms with the best-possible competitive ratio 2. Recently, Lu et al. (2021) studied the single machine scheduling problem with rejection to minimize the weighted makespan. They showed that this problem was NP-hard and proposed an FPTAS for this problem.

When the weight of jobs is equal to 1, minimizing the weighted makespan on parallel machines is exactly minimizing the makespan. This is one of the most classical scheduling problems on parallel machines, Graham (1966), Graham (1969) provided the first approximation algorithm. For earlier approximation schemes for makespan minimization on parallel machines, see Hochbaum and Shmoys (1987), Hochbaum (1997), Alon et al. (1998). The efficient polynomial time approximation scheme (EPTAS) for scheduling on identical parallel machines with the asymptotic running time was presented by Jansen et al. (2020). Recently, Kones and Levin (2019) devised a unified framework for designing EPTAS for general makespan minimization on parallel machines.

A ρ-approximation algorithm for a minimization problem is a polynomial time algorithm that always finds a feasible solution with an objective value of at most ρ times the optimal value. The infimum value of ρ for which an algorithm is a ρ-approximation is called the approximation ratio or the performance guarantee of the algorithm. A PTAS for a given problem is a family of approximation algorithms such that the family has a $(1+\varepsilon)$-approximation algorithm for any $\varepsilon > 0$. An EPTAS is a PTAS whose time complexity is upper bounded by a value of $f(\frac{1}{\varepsilon})\cdot poly(n)$ where f is some computable (not necessarily polynomial) function and $poly(n)$ is a polynomial

of the length of the (binary) encoding of the input. An FPTAS is an EPTAS which satisfies that f must be a polynomial in $\frac{1}{\varepsilon}$.

Motivated by the study in Lu et al. (2021). In this paper, we first present a $(2-\frac{1}{m})$-approximation algorithm for the weighted makespan scheduling problem on identical parallel machines. Then, we develop a randomized EPTAS for this problem and we also design a randomized FPTAS for the special case when m is fixed. The remainder of this paper is organized as follows. In Sect. 2, we describe the definition of the weighted makespan scheduling problem and some preliminaries that will be used throughout the paper. In Sect. 3, we present a $(2-\frac{1}{m})$-approximation algorithm for the problem. In Sect. 4, we present the approximation schemes for the problem with a constant type of weight. In Sect. 5, we present the randomized approximation schemes for this problem. We present some conclusions and possible future research in the last section.

2 Problem formulation and preliminaries

In this paper, we consider the scheduling problem with m identical machines $\mathcal{M} = \{M_1, M_2, \ldots, M_m\}$ and n jobs $\mathcal{J} = \{J_1, J_2, \ldots, J_n\}$. Each job has a processing time p_j and a weight w_j. Without loss of generality, we assume that all p_j and w_j values are positive integers. A schedule is an assignment of n jobs to m machines, i.e. find a schedule $\sigma : \mathcal{J} \to \mathcal{M}$. Furthermore, let C_j be the completion time of J_j. Let $C_{\max} = \max\{C_j | J_j \in \mathcal{J}\}$ and $WC_{\max} = \max\{w_j C_j | J_j \in \mathcal{J}\}$ denote the makespan and weighted makespan (the maximum weighted completion time) of jobs, respectively. The objective is to minimize the weighted makespan of the jobs. Using the general notation for scheduling problems, our problem can be denoted by $P||WC_{\max}$ and let $P_m||WC_{\max}$ denote the special case when m is fixed. When $m = 1$, Li (2015) showed that the LW (Largest Weight first) rule yields an optimal schedule, i.e. if some job processing finished or the machine is idle, then schedule the job with the largest weight.

Clearly, our problem is NP-hard, since it is a general case of minimizing the makespan. We assume that $\mathcal{J}'$ is a set of some jobs. Let $p(\mathcal{J}') = \sum_{J_j \in \mathcal{J}'} p_j$ be the total processing time of $\mathcal{J}'$. Let $w_1 > w_2 > \cdots > w_\tau$ be the τ different job weights for job $J_j \in \mathcal{J}$, let $\mathcal{J}^t = \{J_j \in \mathcal{J} | w_j = w_t\}$, where $\mathcal{J}^t$ is the jobs with weight w_t. Then, we have a lower bound for our problem in the following:

$$WC_{\max} \geq w_t \frac{p(\mathcal{J}^t)}{m}, \quad \text{for each } t = 1, \ldots, \tau. \tag{1}$$

We demonstrate an example to explain our problem. We are given an instance I for $P||WC_{\max}$ with two machines and four jobs: $J_1 = (w_1, p_1) = (2, 1)$, $J_2 = (w_2, p_2) = (2, 3)$, $J_3 = (w_3, p_3) = (3, 2)$, $J_4 = (w_4, p_4) = (4, 1)$. We can construct an optimal solution that assigns J_4 followed by J_2 to the first machine and J_3 followed by J_1 to the second machine. The value of the objective function is $WC_{\max} = w_2 C_2 = 8$. In this paper, let $\mathbb{N}$ and $\mathbb{N}_0$ be the set of positive and nonnegative integers, respectively. Let σ^* and OPT be an optimal schedule and corresponding objective value of σ^*, respectively.

3 A $(2-\frac{1}{m})$-approximation algorithm

In this section, we provide a $(2-\frac{1}{m})$-approximation algorithm for problem $P||WC_{\max}$.

Algorithm A_1

Step 1: Sort all jobs such that $w_1 \geq w_2 \geq \cdots \geq w_n$.

Step 2: If a machine becomes idle, assign jobs to the machine according to the LS algorithm.

Let I be the original instance. Let σ be the schedule obtained from algorithm A_1 for instance I, and OUT be the corresponding objective values of σ. We have the following theorem.

Theorem 3.1 $OUT \leq (2-\frac{1}{m})OPT$.

Proof For any job J_j, let C_j and C_j^* be the completion time of J_j in σ and σ^*, respectively. Also, let $C_{\max} = \max_j C_j$ and $C_{\max}^* = \max_j C_j^*$ be the makespan of σ and σ^*, respectively. Let job J_c be the last job assigned to the maximum load machine. Clearly, we have

$$C_{\max} \leq \frac{\sum_{j=1}^{n} p_j - p_c}{m} + p_c \leq \frac{\sum_{j=1}^{n} p_j}{m} + (1-\frac{1}{m})p_c \leq \frac{\sum_{j=1}^{n} p_j}{m} + (1-\frac{1}{m})\max_j p_j \leq (2-\frac{1}{m})C_{\max}^*.$$

In the schedule σ, let J_l be the job such that $w_l C_l = OUT$. Without loss of generality, we suppose that all jobs start before S_l, where S_l is the starting time of job J_l. Otherwise, it can be deleted from I. Let the new instance be I^1, we have $OUT = OUT^1$ and $OPT \geq OPT^1$, where OUT^1 and OPT^1 be the value of schedule obtained from algorithm A_1 and optimal solution for instance I^1, respectively. That is, $OUT/OPT \leq OUT^1/OPT^1$, it dose not decreasing the value of OUT/OPT. See Fig. 1 for a visualization, where the shadow rectangle represents the set of jobs whose starting time after S_l. Thus, we can suppose that $w_l \leq w_j$ for any job J_j, since every job starts before S_l. Therefore, we have

$$OUT = w_l C_l \leq w_l C_{\max} \leq (2-\frac{1}{m})w_l C_{\max}^* \leq (2-\frac{1}{m})w_{l^*} C_{\max}^* \leq (2-\frac{1}{m})OPT,$$

where w_{l^*} is the weight of the job with largest completion time in schedule σ^*. This completes the proof of Theorem 3.1. □

4 Approximation schemes for a constant type of weight

In many real cases, some parameter may only constant types. For example, in natural scenarios, many machines are of the same type. Gehrke et al. (2018) consider the scheduling on unrelated machines of few different types. In this problem, each job J_j on machine M_i has a processing time p_{ij}. For the case where the number K of

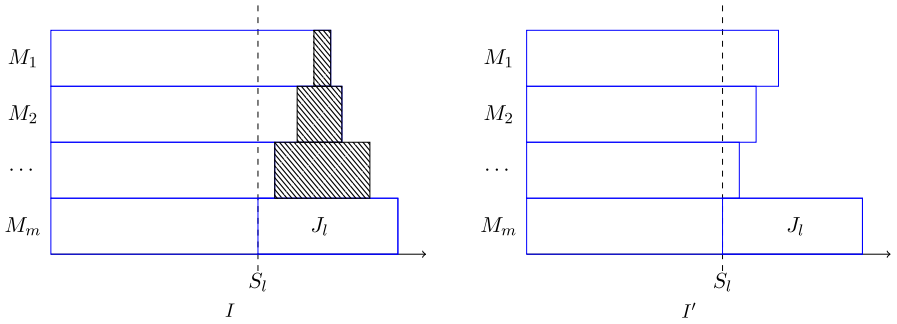

Fig. 1 A visualization for instance I and I'

machine types is constant, i.e. the machine $i \neq i'$ of the same type k satisfy $p_{ij} = p_{i'j}$, they presented a PTAS for this problem. Subsequently, Jansen and Maack (2019) presented an EPTAS for scheduling on unrelated machines of few different types. For the case $K = 2$, Imreh (2003) designed greedy algorithms with approximation rates $2 + (m_1 - 1)/m_2$ and $4 - 2/m_1$, Bleuse et al. (2015) presented an algorithm with approximation rate $4/3 + 3/m_2$, where m_i is the number of machines for type $i = 1, 2$. Moreover, Raravi and Nélis (2012) designed a PTAS for the case $K = 2$. In this section, we will develop an EPTAS and FPTAS for $P||WC_{\max}$ and $P_m||WC_{\max}$ such that jobs with a constant type of weight, respectively.

We can assume that $w_j \leq 1$ for all jobs $J_j \in \mathcal{J}$, since the weight of all the jobs can be divided by the maximum weight value. In this section, we consider all jobs with a constant type of weight. Let $0 < \delta < 1$ be a real constant and $\Delta \in \mathbb{N}$ be a constant. Without loss of generality, we assume that all jobs of weights w_j be the form δ^h with $h \in \{0, 1, \ldots, \Delta\}$ in instance I of problem $P||WC_{\max}$ and $Pm||WC_{\max}$. Let γ be any arbitrary small constant with $0 < \gamma < 1$, and let $\hat{\gamma} = \frac{\delta^{\Delta} \cdot \gamma}{4\Delta+5}$ such that $\frac{1}{\hat{\gamma}}$ is an integer.

The main idea of our approximation schemes for a constant type of weight is as follows. First, we round the processing time of each job and glue the small jobs in original instance I and the rounded instance is $\hat{I}$. Then, all the jobs with the same weight and rounded processing time are contained in the same job type. By the construction of $\hat{I}$ the number of different job types is a constant. Finally, we can determine an optimal solution to instance $\hat{I}$. Based on the optimal solution to $\hat{I}$, we construct a feasible solution to the original instance I. We can show that the ratio of the value of the constructed feasible solution for the original problem to the value of the optimal solution for the original problem is $(1 + \gamma)$.

4.1 EPTAS

In our EPTAS, we first apply algorithm A_1 to obtain a solution with value OUT. By Theorem 3.1, we have $OPT \leq OUT \leq (2 - \frac{1}{m})OPT \leq 2OPT$ or equivalently $OUT/2 \leq OPT \leq OUT$. Let $L = OUT$, then L is an upper bound of OPT. As a result, we have $t^* \leq \frac{1}{\delta^{\Delta}} WC_{t^*} \leq \frac{1}{\delta^{\Delta}} OPT \leq \frac{1}{\delta^{\Delta}} L$ in which t^*, WC_{t^*} are the corresponding makespan and weighted completion time values with respect to

σ^*, respectively. Let $f = \frac{1}{\delta^\Delta}$ and it is also a constant. Hence, we may assume that $p_j \le fL$, for $j = 1, \ldots, n$.

Let $\mathcal{J}_0 = \{J_j \in \mathcal{J} | p_j \le \hat{\gamma} fL\}$ be the set of **small** jobs and $\mathcal{J} \setminus \mathcal{J}_0$ be the set of **big** jobs. Furthermore, we divide the big jobs in $\mathcal{J} \setminus \mathcal{J}_0$ into κ subsets $\mathcal{J}_1, \mathcal{J}_2, \ldots, \mathcal{J}_\kappa$ such that

$$\kappa = \frac{1-\hat{\gamma}}{\hat{\gamma}^2} \text{ and } \mathcal{J}_k$$
$$= \{J_j \in \mathcal{J} | \hat{\gamma} fL + (k-1)\hat{\gamma}^2 fL < p_j \le \hat{\gamma} fL + k\hat{\gamma}^2 fL\}, \text{ for } k = 1, 2, \ldots, \kappa.$$

Based on the instance I, we construct a new instance $\hat{I}$. For each job $J_j \in \mathcal{J}_0$, we construct a job set $\hat{\mathcal{J}}_0$, which contains

$$\left\lceil \frac{\sum_{J_j \in \mathcal{J}_0} p_j}{\hat{\gamma} fL} \right\rceil$$

small job with processing time $\hat{\gamma} fL$. For each job $J_j \in \mathcal{J}_k (k = 1, 2, \ldots, \kappa)$, we construct a job $\hat{J}_j \in \hat{\mathcal{J}}_k$ such that

$$\hat{p}_j = \lceil \frac{p_j}{\hat{\gamma}^2 fL} \rceil \hat{\gamma}^2 fL.$$

By the construction of $\mathcal{J}_k$, we have $\frac{1}{\hat{\gamma}} + (k-1) < \frac{p_j}{\hat{\gamma}^2 fL} \le \frac{1}{\hat{\gamma}} + k$. Thus, we have

$$\hat{p}_j = \lceil \frac{p_j}{\hat{\gamma}^2 fL} \rceil \hat{\gamma}^2 fL = \hat{\gamma} fL + k\hat{\gamma}^2 fL$$

for job $\hat{J}_j \in \hat{\mathcal{J}}_k$, since $\frac{1}{\hat{\gamma}}$ is an integer.

Let $\hat{\mathcal{J}} = \cup_{k=0}^{\kappa} \hat{\mathcal{J}}_k$ and $\hat{n} = |\hat{\mathcal{J}}|$ be the new job set and corresponding the number of jobs in $\hat{\mathcal{J}}$, respectively. By the definition of $J_j \in \mathcal{J}_k (k = 1, 2, \ldots, \kappa)$, we also have

$$p_j \le \hat{p}_j \le p_j + \hat{\gamma}^2 fL \le (1 + \hat{\gamma}) p_j,$$

where the last inequality follows from $p_j > \hat{\gamma} fL$ for each job $J_j \in \mathcal{J}_k$. In the following, we will partition all of the small jobs into $\Delta + 1$ subsets $S_0, S_1, \ldots, S_\Delta$ ($\hat{S}_0, \hat{S}_1, \ldots, \hat{S}_\Delta$), where the jobs in S_h ($\hat{S}_h$) have the same weight $w_h (h = 0, 1, \ldots, \Delta)$ for instance I ($\hat{I}$).

Lemma 4.1 *Let $O\hat{P}T$ and OPT denote the optimal objective values for the corresponding instance $\hat{I}$ and I. Then the following holds*

$$OPT \le O\hat{P}T + (\Delta + 1)\hat{\gamma} fL.$$

Proof Let $\hat{\sigma}$ denote the optimal solution to instance $\hat{I}$. It suffices to show that there exists a feasible solution σ' such that $OPT' \leq (1+\hat{\gamma})O\hat{P}T$, where OPT' is the objective value of feasible solution σ'. Notice that replace every big job $\hat{J}_j$ by its corresponding job J_j in I, this cannot increase the machine completion time.

Let $\hat{S}_{i,h}$ denote the total processing time of the type h small jobs that are processed on machine M_i in $\hat{\sigma}$. Consider a small job set S_h, without loss of generality, we can assume that on each machine, the corresponding jobs are processed consecutively in schedule $\hat{\sigma}$. For $i = 1, 2, \ldots, m$, we schedule the jobs in $\mathcal{J}_0$ on machine M_i until the total processing time of the jobs just exceeds $\hat{S}_{i,h}$. Finally, we schedule the remaining jobs on any machine, and for the jobs assigned to the same machine we sequence them in an order of decreasing weights. This completes the construction of solution σ' and it is easy to verify that σ' is a feasible solution.

By the construction of solution σ', the completion time of each job on machine $M_i (i = 1, \ldots, m)$ in σ' will not exceed the completion time of each job on machine M_i in $\hat{\sigma}$ at most $(\Delta+1)\cdot\hat{\gamma} fL$. In solution σ', we assume that J_j with weight w_h for some fixed h is the job such that $w_j C_j = OPT'$ and processed on machine M_i. Recall that the last scheduled job in some subset $\mathcal{J}$ has the maximal weighted makespan among all jobs in the subset. Thus, in optimal solution $\hat{\sigma}$ of new instance $\hat{I}$, we can assume that job $J_{j'}$ has a similar weight w_h and is last processed in subset $\hat{S}_h$ on machine M_i. Thus, we have

$$\begin{aligned} OPT' &\leq w_{j'}(C_{j'}(\hat{\sigma}) + (\Delta+1)\hat{\gamma} fL) \\ &\leq O\hat{P}T + \max_{0\leq h\leq\Delta} \delta^h(\Delta+1)\hat{\gamma} fL \leq O\hat{P}T + (\Delta+1)\hat{\gamma} fL. \end{aligned}$$

By the construction of σ' and the above assumptions, job J_j has the same weight as $J_{j'}$ and the completion time of job J_j will not exceed the completion time of job $J_{j'}$ at most $(\Delta+1)\hat{\gamma} fL$, which means that the first inequality holds. The second inequality follows from the fact that $O\hat{P}T$ is the weighted makespan in $\hat{\sigma}$. This completes the proof. □

Lemma 4.2 *Let OPT and $O\hat{P}T$ denote the optimal objective values for the corresponding instance I and $\hat{I}$, respectively. We have*

$$O\hat{P}T \leq (1+\hat{\gamma})OPT + (\Delta+1)\hat{\gamma} fL.$$

Proof Let σ^* denote the optimal solution to instance I. Next we construct a new solution $\hat{\sigma}$ and the corresponding objective value is $O\hat{P}T$. Replace every big job J_j by its corresponding big job $\hat{J}_j$. This may increase every machine completion time by a factor of at most $(1+\hat{\gamma})$. Next, let S_{ih} denote the total size of small jobs of weight w_h that are processed on machine M_i in schedule σ^*. We schedule $\lceil \frac{S_{ih}}{\hat{\gamma} fL} \rceil$ jobs with length $\hat{\gamma} fL$ and weight w_h on machine M_i for instance $\hat{I}$. Finally, for the jobs assigned to the same machine, we sequence them in an order of decreasing weights. And it can accommodate all jobs in instance $\hat{I}$.

By the construction of solution $\hat{\sigma}$, the completion time of the each job on machine $M_i (i = 1, \ldots, m)$ in $\hat{\sigma}$ will not exceed the completion time of the each job on machine

M_i in σ^* by a factor of $(1+\hat{\gamma})$ and an amount of $(\Delta+1)\cdot\hat{\gamma} f L$. In solution $\hat{\sigma}$, we assume that J_j with weight w_h for some fix h be the job such that $w_j C_j = O\hat{P}T$ and processed on machine i. Recall that the last scheduled job in some subset $\mathcal{J}$ has the maximal weighted makespan among all jobs in the subset. Then, in optimal solution σ^* of original instance I, we can assume that the job $J_{j'}$ with similar weight w_h and last processed in subset S_h on machine M_i. Thus, we have

$$\begin{aligned} O\hat{P}T &\le w_{j'}((1+\hat{\gamma})C_{j'}(\sigma^*) + (\Delta+1)\hat{\gamma} f L) \\ &\le (1+\hat{\gamma})OPT + \max_{0\le h\le\Delta} \delta^h(\Delta+1)\hat{\gamma} f L \le (1+\hat{\gamma})OPT + (\Delta+1)\hat{\gamma} f L. \end{aligned}$$

By the construction of $\hat{\sigma}$ and the above assumptions, job J_j has the same weight as $J_{j'}$ and the completion time of job J_j will not exceed the completion time of job $J_{j'}$ by a factor of $(1+\hat{\gamma})$ and a amount of $(\Delta+1)\cdot\hat{\gamma} f L$, which means that the first inequality holds. The second inequality follows from fact that OPT is the weighted makespan in σ^*. This completes the proof. □

Lemma 4.3 *Let τ be the total number of different types of jobs, then $\tau \le (1+\Delta)\cdot(1+\kappa)$, where $\kappa = \frac{1-\hat{\gamma}}{\hat{\gamma}^2}$.*

We index the $(1+\Delta)\cdot(\kappa+1)$ job types as $1, 2, \ldots, (1+\Delta)\cdot(\kappa+1)$. We assume that the $(1+\Delta)\cdot(\kappa+1)$ job types are indexed in a non-increasing order of job weights. The job in each type are sorted in any order, since the jobs in each type have the same processing time and weight. We refer vector $\mathbf{n} = (n_{0,0}, n_{0,1}, \ldots, n_{\Delta,\kappa})$ represented the input jobs, where $n_{h,k}$ denotes the number of jobs of weight w_h and size $\hat{\gamma} f L + k\hat{\gamma}^2 f L$, for $h = 0, \ldots, \Delta$ and $k = 0, \ldots, \kappa$. Note that $\sum_{h=0}^{\Delta}\sum_{k=0}^{\kappa} n_{h,k} \le n$. An assignment to a machine is a vector $\mathbf{u} = (u_{0,0}, u_{0,1}, \ldots, u_{\Delta,\kappa})$ where $u_{h,k}$ is the number of jobs with weight δ^h and size $\hat{\gamma} f L + k\hat{\gamma}^2 f L$ that are assigned to the machine. The completion time of the machine is given by $C(\mathbf{u}) = \sum_{h=0}^{\Delta}\sum_{k=0}^{\kappa} u_{h,k}\cdot(\hat{\gamma} f L + k\hat{\gamma}^2 f L)$. Let $\hat{t}$ denote the optimal makespan value for the corresponding new instance $\hat{I}$. Since $t^* \le fOPT \le fL$ and the construction of instance $\hat{I}$, we have $\hat{t} \le f\hat{OPT} \le (1+\hat{\gamma})fL + (\Delta+1)\hat{\gamma} f^2 L$, which satisfies that

$$C(\mathbf{u}) = \sum_{h=0}^{\Delta}\sum_{k=0}^{\kappa} u_{h,k}\cdot(\hat{\gamma} f L + k\hat{\gamma}^2 f L) \le (1+\hat{\gamma})fL + (\Delta+1)\hat{\gamma} f^2 L,$$

implying that

$$\sum_{h=0}^{\Delta}\sum_{k=0}^{\kappa} u_{h,k} \le 1 + \frac{1}{\hat{\gamma}} + (\Delta+1)f.$$

Denote by U the set of vectors $\mathbf{u}$ with $C(\mathbf{u}) \le (1+\hat{\gamma})fL + (\Delta+1)\hat{\gamma} f^2 L$. For a vector $\mathbf{u} \in U$, each entry $u_{h,k}$ is bounded by a number that only depends on the constants $\hat{\gamma}$ and is thus independent of the input. Therefore, the set $|U| \le (\frac{1}{\hat{\gamma}} + 1 + (1+\Delta)f)^{\tau}$ is of constant size.

For each vector $\mathbf{u} \in U$, let $g(\mathbf{u}) = \max_{k_1=0,\ldots,\Delta} \delta^{k_1} \sum_{h=0}^{k_1} \sum_{k=0}^{\kappa} u_{h,k} \cdot (\hat{\gamma} f L + k\hat{\gamma}^2 f L)$ denote the contribution to the objective function of a machine that is scheduled according to $\mathbf{u}$. Since the weighted makespan is determined by the last job of the subset that have the same weight.

We can now formulate the problem of finding an optimal schedule as an integer linear program with a constant number of variables. For each vector $\mathbf{u} \in U$ we introduce a variable $x_{\mathbf{u}}$ which denotes the number of machines that are assigned jobs according to $\mathbf{u}$. An optimal schedule is then given by the following program:

$$
\begin{aligned}
\min \quad & z \\
s.t. \quad & \sum_{\mathbf{u} \in U} x_{\mathbf{u}} = m \\
& \sum_{\mathbf{u} \in U} x_{\mathbf{u}} \cdot \mathbf{u} = \mathbf{n} \\
& y_{\mathbf{u}} \leq x_{\mathbf{u}} \leq m \cdot y_{\mathbf{u}} && \forall \mathbf{u} \in U \\
& z \geq y_{\mathbf{u}} \cdot g(\mathbf{u}) && \forall \mathbf{u} \in U \\
& x_{\mathbf{u}} \in \{0, 1, \ldots, m\} && \forall \mathbf{u} \in U \\
& y_{\mathbf{u}} \in \{0, 1\} && \forall \mathbf{u} \in U.
\end{aligned}
$$

The 0–1 variables $y_{\mathbf{u}}$ indicates whether the assignment $\mathbf{u}$ is used ($y_{\mathbf{u}} = 1$) or not ($y_{\mathbf{u}} = 0$). In case the assignment $\mathbf{u}$ is used, then the term $y_{\mathbf{u}} \cdot g(\mathbf{u})$ contributes to the objective function. Since the number of variables of this integer linear program is $2|U| + 1$. Therefore, the above integer linear program can be solved optimally within time

$$O(|U|^{O(|U|)} \log^{O(1)} n) = O(g(\frac{1}{\hat{\gamma}})n),$$

where $|U| = O((\frac{1}{\hat{\gamma}} + \Delta f)^{\frac{\Delta}{\hat{\gamma}^2}})$, f and Δ are three constants. It has been shown by Lenstra (1983) that an integer linear program in constant dimension can be solved in polynomial time, whose running time is exponential in the dimension of the program but polynomial in the logarithms of the coefficients, where $g(\frac{1}{\hat{\gamma}})$ is an exponential function of $\frac{1}{\hat{\gamma}}$.

Based on Lemma 4.1 and Lemma 4.2, we can obtain a feasible schedule σ with objective value OUT_γ is at most $(1 + \gamma)OPT$, i.e.

Lemma 4.4 $OUT_\gamma \leq (1 + \gamma)OPT$.

Recall that $\hat{\gamma} = \frac{\delta^{\Delta} \cdot \gamma}{4\Delta + 5}$. Hence,

$$
\begin{aligned}
OUT_\gamma &\leq \hat{OPT} + (\Delta + 1)\hat{\gamma} f L \\
&\leq (1 + \hat{\gamma})OPT + (\Delta + 1)\hat{\gamma} f L + (\Delta + 1)\hat{\gamma} f L
\end{aligned}
$$

$$\leq (1+\gamma)OPT,$$

where the last inequality follows from $L \leq 2OPT$ and $f = \frac{1}{\delta^{\Delta}}$.

We now consider the time complexity of $P||WC_{\max}$ in our EPTAS. We first need to apply our $O(n \log n)$ algorithm A_1 to determine the value of L. It only takes $O(n)$ time to construct instances $\hat{I}$. An optimal solution for instance $\hat{I}$ can be found within time $O(g(\frac{1}{\hat{\gamma}})n)$, by solving the above integer program. Based on the optimal solution to $\hat{I}$, solution σ is easily generated in $O(n \log n)$ time. Therefore, the overall running time is $O(n \log n + g(\frac{1}{\hat{\gamma}})n)$, where $g(\frac{1}{\hat{\gamma}})$ is an exponential function of $\frac{1}{\hat{\gamma}}$. We thus have the following theorem.

Theorem 4.5 *There exists an EPTAS with running time $O(n \log n + g(\frac{1}{\gamma})n)$ for problem $P||WC_{\max}$ with a constant type of weight, where $g(\frac{1}{\gamma})$ is an exponential function of $\frac{1}{\gamma}$.*

4.2 FPTAS when *m* is fixed

In the following we present an efficient FPTAS for $P_m||WC_{\max}$. Following from the fact that the processing time of each job in $\hat{\mathcal{J}}$ is no less than $\hat{\gamma} f L$ and $C(\mathbf{u}) \leq (1+\hat{\gamma}) f L + (\Delta+1)\hat{\gamma} f^2 L$, we have the following lemma.

Lemma 4.6 *The number of jobs in $\hat{\mathcal{J}}$ is at most*

$$\hat{n} \leq \tau \cdot (\frac{1}{\hat{\gamma}} + 1 + (1+\Delta) f)m.$$

We now develop a different exact algorithm to solve problem instance $\hat{I}$. We present a dynamic programming algorithm for $P_m||WC_{\max}$ as follows.

Sort all jobs such that $w_1 \geq \cdots \geq w_n$. Let $f_j(t_1, \ldots, t_m)$ be the minimum $WC_{\max}$ value when (1) the jobs in consideration are $\hat{J}_1, \ldots, \hat{J}_j$; and (2) the makespan of the jobs assigned to M_i is exactly t_i. It is clear that $t_i = 0, \hat{\gamma} f L, \hat{\gamma} f L + \hat{\gamma}^2 f L, \ldots, (1+\hat{\gamma}) f L + (\Delta+1)\hat{\gamma} f^2 L$.

If $\hat{J}_j$ is selected to be processed by machine M_i, then we have the makespan of $\hat{J}_1, \ldots, \hat{J}_j$ is $t_i - p_j$. Thus, we have

$$f_j(t_1, \ldots, t_m) = \max\{f_{j-1}(t_1, \ldots, t_{i-1}, t_i - \hat{p}_j, t_{i+1}, \ldots, t_m), w_j \cdot t_i\}.$$

We have the following dynamic programming DP:
Boundary condition:

$$f_1(t_1, \ldots, t_m) = \begin{cases} w_1 \hat{p}_1, & \text{if } t_i = \hat{p}_1 \text{ for some } i \text{ and } t_j = 0 \text{ for all } j \neq i; \\ +\infty, & \text{if } t_1 = t_2 = \cdots = t_m = 0; \\ +\infty, & \text{otherwise.} \end{cases}$$

The recursive function:

$$f_j(t_1,\ldots,t_m)=\min\begin{cases}\max\{f_{j-1}(t_1-\hat{p}_j,\ldots,t_m),w_j\cdot t_1\};\\ \max\{f_{j-1}(t_1,t_2-\hat{p}_j,\ldots,t_m),w_j\cdot t_2\};\\ \ldots\\ \max\{f_{j-1}(t_1,\ldots,t_m-\hat{p}_j),w_j\cdot t_m\}.\end{cases}$$

The Optimal Value:

$$\begin{aligned}&\min\{f_{\hat{n}}(t_1,\ldots,t_m)\}|t_i\in\{0,\hat{\gamma}fL,\hat{\gamma}fL\\&\quad+\hat{\gamma}^2fL,\ldots,(1+\hat{\gamma})fL+(\Delta+1)\hat{\gamma}f^2L\},i=1,\ldots,m\}.\end{aligned}$$

We now consider the time complexity of our FPTAS. We first need to apply our $O(n\log n)$ algorithm A_1 to determine the value of L. It only takes $O(n)$ time to construct instances $\hat{I}$. An optimal solution for instance $\hat{I}$ by using algorithm DP, whose running time is

$$\begin{aligned}&O\left(\hat{n}\left(1+\frac{1+(\Delta+1)\hat{\gamma}f}{\hat{\gamma}^2}\right)^m\right)\\&=O\left(\tau\cdot\left(\frac{1}{\hat{\gamma}}+(\Delta+1)f\right)m\left(1+\frac{1+(\Delta+1)\hat{\gamma}f}{\hat{\gamma}^2}\right)^m\right)\\&=O\left(\frac{1}{\hat{\gamma}^{2m+3}}\right),\end{aligned}$$

where m, Δ, f are three constants, $\tau\le(1+\Delta)(1+\kappa)$ and $\kappa=\frac{1-\hat{\gamma}}{\hat{\gamma}^2}$. Based on the optimal solution to $\hat{I}$, solution σ' is easily generated in $O(n\log n)$ time. Therefore, the overall running time is $O(n\log n+\frac{1}{\hat{\gamma}^{2m+3}})$. We thus have the following theorem.

Theorem 4.7 *There exists an FPTAS with running time $O(n\log n+\frac{1}{\gamma^{2m+3}})$ for problem $P_m||WC_{\max}$ with a constant type of weight.*

5 The randomized polynomial time approximation schemes

The randomized approximation schemes for $P||WC_{\max}$ and $P_m||WC_{\max}$ will be shown in this section. The randomized approximation schemes use analysis methods that are based on ideas from Skutella and Woeginger (2000). However, we use some novel lower bounds on the optimal solution value of our problem. Our randomized approximation schemes are divided into three steps. The first step is to divide the job set according to the weight value of w_j. The second step is to find the corresponding approximate schemes for the subsets of the job divided in the previous step. In the third step, we glue the approximate solution obtained in the second step to a feasible solution to the original problem.

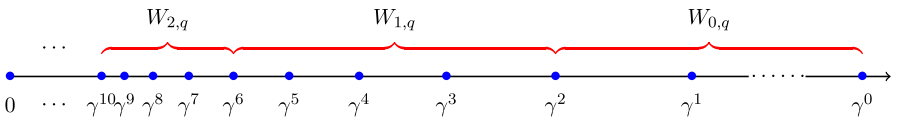

Fig. 2 A visualization of the weight distribution with $\Gamma = 4$, $q = 3$

For the sake of approximation ratio analysis, we can assume that $w_j \leq 1$ for all jobs $J_j \in \mathcal{J}$, since the weight of all the jobs can be divided by the maximum weight value. In the following analysis, if we use Theorem 4.5, we obtain the EPTAS for $P||WC_{\max}$ with a constant type of weight, and if we use Theorem 4.7, we obtain the FPTAS for $P_m||WC_{\max}$ with a constant type of weight. Thus, we only consider the results of randomized EPTAS for $P||WC_{\max}$ and the results of randomized FPTAS for $P_m||WC_{\max}$ which can be analogously and simply obtained.

5.1 The randomized approximation scheme

Our randomized approximation scheme is divided into the following steps.

(I) Let Γ be a positive integer and let $\gamma = \frac{1}{\Gamma}$. We will select Γ too large such that γ is small. We divide the job set $\mathcal{J}$ as follows: First, let $\mathcal{J}^h := \{J_j \in \mathcal{J} | \gamma^h < w_j \leq \gamma^{h-1}\}$, for $h \in \mathbb{N}$. Then, pick q of $\{1, 2, \ldots, \Gamma\}$ uniformly at random, set $W_{0,q} = \{1, 2, \ldots, q-1\}$, and for $s \in \mathbb{N}$:

$$W_{s,q} = \{(s-1) \cdot \Gamma + q, \ldots, s \cdot \Gamma - 1 + q\};$$

for $s \in \mathbb{N}_0$, let $\mathcal{J}_{s,q} = \cup_{h \in W_{s,q}} \mathcal{J}^h$.

For fixed q, the number of nonempty subsets $\mathcal{J}_{s,q}, s \in \mathbb{N}_0$ is at most n. We only consider those subsets in this randomized approximation scheme. A visualization of the weight distribution is shown in Fig. 2.

(II) For any nonempty sets of jobs $\mathcal{J}_{s,q}$, the ratio of minimum weight to maximum weight is bounded by γ^Γ. Thus, we can assume that the weights are in the $[\gamma^\Gamma \rho, \rho]$ range, where arbitrary $\rho > 0$. Note that by rescaling the weights of jobs we can restrict to the case $\rho = 1$. Then, we choose the constant $0 < \delta < 1$ arbitrarily close to 1, which leads to $\frac{1}{\delta}$ arbitrary small. This is similar to the case that discussed in Section 4, we can round up the weights of all the jobs to the form δ^h in job set $\mathcal{J}_{s,q}$ where h is an integer and bounded by a constant, since Γ, γ, δ are constants and it will increase the value of the objective function by an arbitrary small factor of $\frac{1}{\delta}$. Thus, we can compute a polynomial time approximation scheme according to Theorems 4.5 and let the objective value of the approximation scheme is $OUT_{s,q}$. Let $OUT_{s,q*}$ denotes the value of an optimal schedule for $\mathcal{J}_{s,q}$. The scheduling produced by the second step is called the (job) subset scheduling.

(III) These schedules of job subsets are glued in the algorithm's final step. We first randomly and uniformly permute the machines in each job subset schedule and then gluing those schedules. Since there could be a situation in which each subset of jobs in $\mathcal{J}_{s,q}$ only contains one job, which is always scheduled to process on machine 1 in the corresponding job subset schedule and gluing the schedules will lead to only one

machine processing job. Finally, the probability that two jobs from different subsets will be processed on the same machine in this randomly generated schedule is equal to $\frac{1}{m}$.

5.2 The analysis of the randomized approximation scheme

Note that the objective value of the randomized approximation scheme schedule is equal to some job's weighted completion time in the subset schedule plus the extra loss caused by the delay of this job in the gluing step. The value of some job's weighted completion time is less than or equal to the weighted makespan of the same job subset in the subset schedule. Additionally, the weighted makespan in this subset schedule is less than or equal to the maximum weighted makespan in all job subset schedules. Finally, we can show that the value of the maximum weighted makespan value of the job subset schedule cannot exceeded the optimal schedule value, and we will also show that the job delays in the gluing step are too small to ignore.

Lemma 5.1 *For each $q \in \{1, \ldots, \Gamma\}$ we have*

$$OPT \geq \max_{s \in \mathbb{N}_0} OUT_{s,q*} \text{ and } OPT \geq \gamma^h \frac{p(\mathcal{J}^h)}{m}.$$

Proof Take an optimal schedule of job set $\mathcal{J}$ and denote the completion time of job J_j as C_j^*. For each subset $\mathcal{J}_{s,q}$, this optimal subset schedule can be viewed as processed on m machines starting at time 0. However, the completion time C_j^* of all jobs $J_j \in \mathcal{J}_{s,q}$ in the optimal schedule also defines a feasible schedule for job subset $\mathcal{J}_{s,q}$. This yields $OPT = \max_{s \in \mathbb{N}_0, J_j \in \mathcal{J}_{s,q}} w_j C_j^* \geq \max_{s \in \mathbb{N}_0} OUT_{s,q*}$. Thus, OPT is not less than optimal schedule of value for each subset $\mathcal{J}_{s,q}$. For the second lower bound, if we round the weights of jobs $J_j \in \mathcal{J}^k$ down to $w_j = \gamma^k$, for $k \in \mathbb{N}$, it will decrease the value of an optimal schedule. The result then follows from (1). □

Without loss of generality, we assume job $J_j \in \mathcal{J}^k$ achieves the weighted makespan in the schedule of the randomized approximation scheme, for some fixed $k \in \mathbb{N}$. Let $E[WC_{\max}]$ be the expected value of the schedule in the randomized approximation scheme. We have the following lemma.

Lemma 5.2 *The expected value of the schedule in randomized approximation scheme is*

$$E[WC_{\max}] \leq E[\max_{s \in \mathbb{N}_0} OUT_{s,q}] + w_{j\,J_j \in \mathcal{J}^k} \sum_{h<k} \min\left\{1, \frac{k-h}{\Gamma}\right\} \frac{p(\mathcal{J}^h)}{m}. \quad (2)$$

Proof We first keep q fixed and analyze the conditional expectation $E_q[WC_{\max}]$ when some job $J_j \in \mathcal{J}^k$ achieves the weighted makespan in the schedule of the randomized approximation scheme. Recall that the value of some job's weighted completion time in the subset schedule is less than or equal to the weighted makespan of the same job subset. Additionally, the weighted makespan in the subset schedule is less than

or equal to the maximum weighted makespan in all job subset schedules. Thus, this conditional expectation is less than or equal to the value of the job subset schedule $\max_{s\in\mathbb{N}_0} OUT_{s,q}$ plus the expected loss caused by the delayed job J_j in the gluing step.

For fixed q, we next analyze the expected delay of some job $J_j \in \mathcal{J}$. Let $J_j \in \mathcal{J}^k \subseteq \mathcal{J}_{t,q}$ be the corresponding job subset for J_j such that $t \in \mathbb{N}_0$ and $k \in W_{t,q}$. The machines are permuted uniformly at random in the gluing step. Thus, the expected loss of job J_j delay is equal to the average load produced by the previous set of jobs, i.e., $\cup_{s=0}^{t-1}\mathcal{J}_{s,q}$, that completed on the machine that job J_j processed. We know that the expected loss of job delay is $\frac{p(\mathcal{J}^h)}{m}$ if indices h and k fall in different subset $W_{s,q}$ for $s \in \mathbb{N}_0$. The expected loss of job delay is 0 if indices h and k fall in the same subset $W_{s,q}$ for $s \in \mathbb{N}_0$.

Since we chose q uniformly at random, then the expected value of job delay loss is equal to the probability that indices h and k lie in different subset $W_{s,q}$ for $s \in \mathbb{N}_0$. By the random choice of q, this probability is equal to $\frac{k-h}{\Gamma}$ for $k-h \leq \Gamma$ and it is 1 for $k-h > \Gamma$. Thus, we have

$$E[WC_{\max}] \leq E[\max_{s\in\mathbb{N}_0} OUT_{s,q}] + w_{j\,J_j\in\mathcal{J}^k} \sum_{h<k} \min\{1, \frac{k-h}{\Gamma}\} \frac{p(\mathcal{J}^h)}{m}.$$

□

Theorem 5.3 *For a given $0 < \gamma < 1$, let $\Gamma = \lceil \frac{4}{\varepsilon} \rceil$. Then, the expected value of the randomized approximation scheme schedule is*

$$E[WC_{\max}] \leq (1+\gamma)OPT.$$

Proof Without loss of generality, we also assume that job $J_j \in \mathcal{J}^k$ achieves the weighted makespan in the randomized approximation scheme schedule, for some fixed $k \in \mathbb{N}$. Based on Theorem 4.5, we have $OUT_{s,q} \leq (1+\gamma)\cdot OUT_{s,q*}$ for each q, where $OUT_{s,q}$ and $OUT_{s,q*}$ are the computed weighted makespan value using Theorem 4.5 and the optimal value in subset $\mathcal{J}_{s,q}$, respectively. Based on Lemma 5.1, we have $\max_{s\in\mathbb{N}_0} OUT_{s,q*} \leq OPT$. Thus

$$E[\max_{s\in\mathbb{N}_0} OUT_{s,q}] \leq (1+\gamma)\cdot OPT.$$

In the last part of (2), we sum h for a fixed k. Note that for $k \in \mathbb{N}$, we have

$$w_{j\,J_j\in\mathcal{J}^k} \leq \gamma^{k-1}.$$

For fixed $k \in \mathbb{N}$, we divide $h(h < k)$ into two partial sums $Sum_1 + Sum_2$. The first partial Sum_1 is $h = k-1$ and it is bounded by

$$Sum_1 \leq \gamma^{k-1-h} \cdot \frac{1}{\Gamma} \cdot \frac{p(\mathcal{J}^h)}{m} \gamma^h \leq \gamma OPT.$$

The second partial Sum_2 is $k \geq h+2$. In this case we simply set $\min\{1, \frac{k-h}{\Gamma}\}$ to 1 and it is bounded by

$$Sum_2 \leq \sum_{h \leq k-2} \gamma^{k-1-h} \cdot \frac{p(\mathcal{J}^h)}{m} \gamma^h \leq \sum_{h \in \mathbb{N}} \gamma^h OPT \leq \frac{\gamma}{1-\gamma} OPT.$$

We sum these two partial sums, i.e. $Sum_1 + Sum_2$, then the expected value of the randomized approximation scheme schedule is

$$\begin{aligned} E[WC_{\max}] &\leq E[\max_{s \in \mathbb{N}_0} OUT_{s,q}] + w_{j\,J_j \in \mathcal{J}^k} \sum_{h<k} \min\{1, \frac{k-h}{\Gamma}\} \frac{p(\mathcal{J}^h)}{m} \\ &\leq (1 + 2\gamma + \frac{\gamma}{1-\gamma}) OPT \leq (1+\varepsilon) OPT. \end{aligned}$$

The second inequality follows from $\gamma = \frac{1}{\Gamma}$ and $\Gamma = \lceil \frac{4}{\varepsilon} \rceil$. □

6 Conclusion

In this paper, we consider problem $P||WC_{\max}$. We first design a $(2-\frac{1}{m})$-approximation algorithm for this problem. Then, we design an EPTAS and an FPTAS for the problem with a constant type of weight. Finally, based on the above approximation schemes, we propose a randomized EPTAS for the problem, and a randomized FPTAS for the special case when m is fixed. Moreover, it is also interesting to consider the online or semi-online versions of this problem. Finally, we will consider designing an EPTAS for this problem in the future.

Funding The work is supported in part by the National Natural Science Foundation of China [No. 12071417].

Data availability Data sharing not applicable to this article as no datasets were generated or analysed during the current study.

Declarations

Conflict of interest The author has no conflict of interest.

References

Afrati FN, Bampis E, Chekuri C, Karger DR, Kenyon C, Khanna S, Milis I, Queyranne M, Skutella M, Stein C, Sviridenko M (1999) Approximation schemes for minimizing average weighted completion time with release dates. In: Proceedings of the 40th annual IEEE symposium on foundations of computer science. IEEE Computer Society Press, Los Alamitos, CA, pp 32–43

Alon N, Azar Y, Woeginger GJ, Yadid T (1998) Approximation schemes for scheduling on parallel machines. J Sched 1(1):55–66

Bansal N, Srinivasan A, Svensson O (2016) Lift-and-round to improve weighted completion time on unrelated machines. In: Proceedings of 48th annual ACM symposium theory computing (STOC). ACM, pp 156–167
Baveja A, Qu X, Srinivasan A (2023) Approximating weighted completion time via stronger negative correlation. J Sched 1–10
Bleuse R, Kedad-Sidhoum S, Monna F, Mounié G, Trystram D (2015) Scheduling independent tasks on multi-cores with gpu accelerators. Concurr Comput Pract Exp 27(6):1625–1638
Chai X, Lu LF, Li WH, Zhang LQ (2018) Best-possible online algorithms for single machine scheduling to minimize the maximum weighted completion time. Asia-Pac J Oper Res 35:1850048
Feng Q, Yuan JJ (2007) NP-hardness of a multicriteria scheduling on two families of jobs. OR Trans 11:121–126
Gehrke JC, Jansen K, Kraft SE, Schikowski J (2018) A PTAS for scheduling unrelated machines of few different types. Int J Found Comput Sci 29(4):591–621
Graham RL (1966) Bounds for certain multiprocessing anomalies. Bell Syst Tech J 45(9):1563–1581
Graham RL (1969) Bounds on multiprocessing timing anomalies. SIAM J Appl Math 17(2):416–429
Hochbaum DS (1997) Various notions of approximations: good, better, best and more. In: Hochbaum DS (ed) Approximation algorithms. PWS Publishing Company, Boston
Hochbaum DS, Shmoys DB (1987) Using dual approximation algorithms for scheduling problems theoretical and practical results. J ACM 34(1):144–162
Imreh C (2003) Scheduling problems on two sets of identical machines. Computing 70(4):277–294
Im S, Shadloo M (2020) Weighted completion time minimization for unrelated machines via iterative fair contention resolution. In: Symposium on discrete algorithms, SODA, pp 2790–2809
Jansen K, Maack M (2019) An EPTAS for scheduling on unrelated machines of few different types. Algorithmica 81:4134–4164
Jansen K, Klein KM, Verschae J (2020) Closing the gap for makespan scheduling via sparsification techniques. Math Oper Res 45(4):1371–1392
Kawaguchi T, Kyan S (1986) Worst case bound of an LRF schedule for the mean weighted flow-time problem. SIAM J Comput 15(4):1119–1129
Kones I, Levin A (2019) A unified framework for designing EPTAS for load balancing on parallel machines. Algorithmica 81(7):3025–3046
Lenstra HW Jr (1983) Integer programming with a fixed number of variables. Math Oper Res 8(4):538–548
Li WJ (2015) A best possible online algorithm for the parallel-machine scheduling to minimize the maximum weighted completed time. Asia-Pac J Oper Res 32:1550030
Li S (2020) Scheduling to minimize total weighted completion time via time-indexed linear programming relaxations. SIAM J Comput 49(4):FOCS 17-409
Lu L, Zhang L, Ou J (2021) Single machine scheduling with rejection to minimize the weighted makespan. In: Wu W, Du H (eds) AAIM 2021, vol 13153. LNCS. Springer, Cham, pp 96–110
Phillips C, Stein C, Wein J (1998) Minimizing average completion time in the presence of release dates. Math Program 82:199–223
Raravi G, Nélis V (2012) A PTAS for assigning sporadic tasks on two-type heterogeneous multiprocessors. In: 2012 IEEE 33rd real-time systems symposium (RTSS). IEEE, pp 117–126
Sahni SK (1976) Algorithms for scheduling independent tasks. J ACM 23(1):116–127
Skutella M (2001) Convex quadratic and semidefinite programming relaxations in scheduling. J ACM 48(2):206–242
Skutella M, Woeginger GJ (2000) A PTAS for minimizing the total weighted completion time on identical parallel machines. Math Oper Res 25(1):63–75
Smith WE (1956) Various optimizers for single-stage production. Nav Res Logist Quart 3:59–66