Theoretical Computer Science 803 (2020) 57–70

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Approximation algorithms for the three-machine proportionate mixed shop scheduling ☆

Longcheng Liu [a,b], Yong Chen [c,b], Jianming Dong [d,b], Randy Goebel [b], Guohui Lin [b,*], Yue Luo [a], Guanqun Ni [e,b], Bing Su [f], Yao Xu [b], An Zhang [c,b]

[a] School of Mathematical Sciences, Xiamen University, Xiamen, China
[b] Department of Computing Science, University of Alberta, Edmonton, AB T6G 2E8, Canada
[c] Department of Mathematics, Hangzhou Dianzi University, Hangzhou, China
[d] Department of Mathematics, Zhejiang Sci-Tech University, Hangzhou, China
[e] College of Management, Fujian Agriculture and Forestry University, Fuzhou, China
[f] School of Economics and Management, Xi'an Technological University, Xi'an, China

ARTICLE INFO

Article history:
Received 26 January 2019
Received in revised form 27 March 2019
Accepted 19 May 2019
Available online 2 July 2019

Keywords:
Scheduling
Mixed shop
Proportionate
Approximation algorithm
Fully polynomial-time approximation scheme

ABSTRACT

A mixed shop is a manufacturing infrastructure designed to process a mixture of a set of flow-shop jobs and a set of open-shop jobs. Mixed shops are in general much more complex to schedule than flow-shops and open-shops, and have been studied since the 1980's. We consider the three machine proportionate mixed shop problem denoted as $M3 \mid prpt \mid C_{\max}$, in which by "proportionate" each job has equal processing times on all three machines. Koulamas and Kyparisis (2015) [6] showed that the problem is solvable in polynomial time in some very special cases; for the non-solvable case, they proposed a 5/3-approximation algorithm. In this paper, we first present an improved 4/3-approximation algorithm and show that this ratio of 4/3 is asymptotically tight; when the largest job is a flow-shop job, we then present a fully polynomial-time approximation scheme (FPTAS). On the negative side, while the $F3 \mid prpt \mid C_{\max}$ problem is polynomial-time solvable, we show an interesting hardness result that adding one open-shop job to the job set makes the problem NP-hard if this open-shop job is larger than any flow-shop job. We are able to design an FPTAS for this special case too.

1. Introduction

We study the following three-machine proportionate mixed shop problem, denoted as $M3 \mid prpt \mid C_{\max}$ in the three-field notation [4]. Given three machines M_1, M_2, M_3 and a set $\mathcal{J} = \mathcal{F} \cup \mathcal{O}$ of jobs, where $\mathcal{F} = \{J_1, J_2, \ldots, J_\ell\}$ and $\mathcal{O} = \{J_{\ell+1}, J_{\ell+2}, \ldots, J_n\}$, each job $J_i \in \mathcal{F}$ needs to be processed non-preemptively through M_1, M_2, M_3 sequentially with a processing time p_i on each machine and each job $J_i \in \mathcal{O}$ needs to be processed non-preemptively on M_1, M_2, M_3 in any machine order, with a processing time q_i on each machine. The scheduling constraint is as usual that at every time point a

☆ An extended abstract appears in the *Proceedings of AAIM 2018*. LNCS 11343, pages 268–280.

* Corresponding author.
E-mail addresses: longchengliu@xmu.edu.cn (L. Liu), chenyong@hdu.edu.cn (Y. Chen), djm226@163.com (J. Dong), rgoebel@ualberta.ca (R. Goebel), guohui@ualberta.ca (G. Lin), guanqunni@163.com (G. Ni), subing684@sohu.com (B. Su), xu2@ualberta.ca (Y. Xu), anzhang@hdu.edu.cn (A. Zhang).

https://doi.org/10.1016/j.tcs.2019.05.036
0304-3975/

job can be processed by at most one machine and a machine can process at most one job. The objective is to minimize the maximum job completion time, *i.e.*, the makespan.

The jobs of $\mathcal{F}$ are referred to as *flow-shop jobs* and the jobs of $\mathcal{O}$ are called *open-shop jobs*. The mixed shop problem is to process such a mixture of a set of flow-shop jobs and a set of open-shop jobs. We assume without loss of generality that $p_1 \geq p_2 \geq \ldots \geq p_\ell$ and $q_{\ell+1} \geq q_{\ell+2} \geq \ldots \geq q_n$.

Mixed shops have many real-life applications and have been studied since the 1980's. The scheduling of medical tests in an outpatient health care facility and the scheduling of classes/exams in an academic institution are two typical examples, where the patients (students, respectively) must complete a number of medical tests (academic activities, respectively); some of these activities must be done in a specified sequential order while the others can be finished in any order; and the time-spans for all these activities should not overlap with each other. The *proportionate* shops were also introduced in the 1980's [9] and they are one of the most specialized shops with respect to the job processing times which have received many studies [10].

Masuda et al. [8] and Strusevich [14] considered the two-machine mixed shop problem to minimize the makespan, *i.e.*, $M2 \mid\mid C_{\max}$; they both showed that the problem is polynomial-time solvable. Shakhlevich and Sotskov [11] studied mixed shops for processing two jobs with an arbitrary regular objective function. Brucker [1] surveyed the known results on the mixed shop problems either with two machines or for processing two jobs. Shakhlevich et al. [12] studied the mixed shop problems with more than two machines for processing more than two jobs, with or without preemption. Shakhlevich et al. [13] reviewed the complexity results on the mixed shop problems with three or more machines for processing a constant number of jobs.

When $\mathcal{O} = \emptyset$, the $M3 \mid prpt \mid C_{\max}$ problem reduces to the $F3 \mid prpt \mid C_{\max}$ problem, which is solvable in polynomial time [2]. When $\mathcal{F} = \emptyset$, the problem reduces to the $O3 \mid prpt \mid C_{\max}$ problem, which is ordinary (or called weakly) NP-hard [7]. It follows that the $M3 \mid prpt \mid C_{\max}$ problem is at least ordinary NP-hard. Recently, Koulamas and Kyparisis [6] showed that for some very special cases, the $M3 \mid prpt \mid C_{\max}$ problem is solvable in polynomial time; for the non-solvable case, they showed an absolute performance bound of $2\max\{p_1, q_{\ell+1}\}$ and presented a 5/3-approximation algorithm.

In this paper, we first design an improved 4/3-approximation algorithm for (the non-solvable case of) the $M3 \mid prpt \mid C_{\max}$ problem, and show that the performance ratio of 4/3 is asymptotically tight. When the largest job is a flow-shop job, that is $p_1 \geq q_{\ell+1}$, we then present a *fully polynomial-time approximation scheme* (FPTAS). On the negative side, while the $F3 \mid prpt \mid C_{\max}$ problem is polynomial-time solvable, we show an interesting hardness result that adding one single open-shop job to the job set makes the problem NP-hard if this open-shop job is larger than any flow-shop job (that is, $\mathcal{F} = \{J_1, J_2, \ldots, J_{n-1}\}$ and $\mathcal{O} = \{J_n\}$, and $q_n > p_1$). We construct the reduction from the well-known PARTITION problem [3]. Denote the special case in which $|\mathcal{F}| = n-1$ and $|\mathcal{O}| = 1$ as $M3 \mid prpt, (n-1, 1) \mid C_{\max}$. We propose an FPTAS for this special case $M3 \mid prpt, (n-1, 1) \mid C_{\max}$.

The rest of the paper is organized as follows. In Section 2, we introduce some notation and present a lower bound on the optimal makespan $C^*_{\max}$. We present in Section 3 the FPTAS for the $M3 \mid prpt \mid C_{\max}$ problem when $p_1 \geq q_{\ell+1}$. The 4/3-approximation algorithm for the case where $p_1 < q_{\ell+1}$ is presented in Section 4, and the performance ratio of 4/3 is shown to be asymptotically tight. We show in Section 5 that, when the open-shop job J_n is strictly larger than all the other jobs, the special case $M3 \mid prpt, (n-1, 1) \mid C_{\max}$ is NP-hard, through a reduction from the PARTITION problem. Section 6 contains an FPTAS for the special case $M3 \mid prpt, (n-1, 1) \mid C_{\max}$. We conclude the paper with some remarks in Section 7.

2. Preliminaries

For any subset of jobs $\mathcal{X} \subseteq \mathcal{F}$, the *total processing time* of the jobs of $\mathcal{X}$ on one machine is denoted as

$$P(\mathcal{X}) = \sum_{J_i \in \mathcal{X}} p_i.$$

For any subset of jobs $\mathcal{Y} \subseteq \mathcal{O}$, the *total processing time* of the jobs of $\mathcal{Y}$ on one machine is denoted as

$$Q(\mathcal{Y}) = \sum_{J_i \in \mathcal{Y}} q_i.$$

The set minus operation $\mathcal{J} \setminus \{J\}$ for a single job $J \in \mathcal{J}$ is abbreviated as $\mathcal{J} \setminus J$ throughout the paper.

In a schedule π, we use S^i_j and C^i_j to denote the start time and the finish time of the job J_j on the machine M_i, respectively, for $i = 1, 2, 3$ and $j = 1, 2, \ldots, n$.

Given that the *load* (*i.e.*, the total job processing time) of each machine is $P(\mathcal{F}) + Q(\mathcal{O})$, the job $J_{\ell+1}$ has to be processed by all three machines, and one needs to process all the flow-shop jobs of $\mathcal{F}$, the following lower bound on the optimum $C^*_{\max}$ is established [2,6]:

$$C^*_{\max} \geq \max\{P(\mathcal{F}) + Q(\mathcal{O}),\ 3q_{\ell+1},\ 2p_1 + P(\mathcal{F})\}. \tag{1}$$

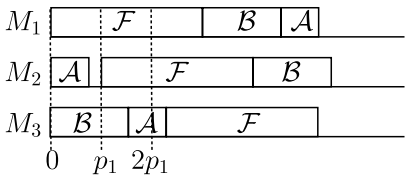

Fig. 3.1. An illustration of the schedule π produced by Proc$(\mathcal{A}, \mathcal{B}, \mathcal{F})$, where $\{\mathcal{A}, \mathcal{B}\}$ is a bipartition of the set $\mathcal{O}$ and the jobs of each of $\mathcal{A}, \mathcal{B}, \mathcal{F}$ are processed in the same LPT order on all three machines.

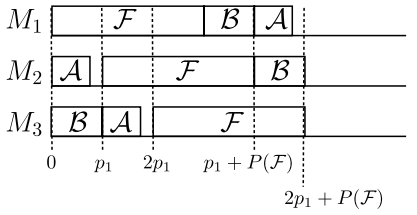

Fig. 3.2. An illustration of the schedule π produced by Proc$(\mathcal{A}, \mathcal{B}, \mathcal{F})$ when both $Q(\mathcal{A}) \le p_1$ and $Q(\mathcal{B}) \le p_1$.

3. An FPTAS for the case where $p_1 \ge q_{\ell+1}$

In this section, we design an approximation algorithm $A(\epsilon)$ for the $M3 \mid prpt \mid C_{\max}$ problem when $p_1 \ge q_{\ell+1}$, for any given $\epsilon > 0$. The algorithm $A(\epsilon)$ produces a schedule π with its makespan $C^{\pi}_{\max} < (1+\epsilon)C^{*}_{\max}$, and its running time polynomial in both n and $1/\epsilon$.

Consider a bipartition $\{\mathcal{A}, \mathcal{B}\}$ of the job set $\mathcal{O} = \{J_{\ell+1}, J_{\ell+2}, \ldots, J_n\}$, *i.e.*, $\mathcal{A} \cup \mathcal{B} = \mathcal{O}$ and $\mathcal{A} \cap \mathcal{B} = \emptyset$. Throughout the paper, a part of the bipartition is allowed to be empty. The following six-step *procedure* Proc$(\mathcal{A}, \mathcal{B}, \mathcal{F})$ produces a schedule π:

1. the jobs of $\mathcal{F}$ are processed in the same *longest processing time* (LPT) order on all three machines, and every job is processed first on M_1, then on M_2, lastly on M_3;
2. the jobs of $\mathcal{A}$ are processed in the same LPT order on all three machines, and every one is processed first on M_2, then on M_3, lastly on M_1;
3. the jobs of $\mathcal{B}$ are processed in the same LPT order on all three machines, and every one is processed first on M_3, then on M_1, lastly on M_2; and
4. the machine M_1 processes (the jobs of) $\mathcal{F}$ first, then $\mathcal{B}$, lastly $\mathcal{A}$, denoted as $\langle \mathcal{F}, \mathcal{B}, \mathcal{A} \rangle$;
5. the machine M_2 processes $\mathcal{A}$ first, then $\mathcal{F}$, lastly $\mathcal{B}$, denoted as $\langle \mathcal{A}, \mathcal{F}, \mathcal{B} \rangle$;
6. the machine M_3 processes $\mathcal{B}$ first, then $\mathcal{A}$, lastly $\mathcal{F}$, denoted as $\langle \mathcal{B}, \mathcal{A}, \mathcal{F} \rangle$.

Proc$(\mathcal{A}, \mathcal{B}, \mathcal{F})$ runs in $O(n \log n)$ time to produce the schedule π, of which an illustration is shown in Fig. 3.1.

Lemma 3.1. *If both $Q(\mathcal{A}) \le p_1$ and $Q(\mathcal{B}) \le p_1$, then the schedule π produced by* Proc$(\mathcal{A}, \mathcal{B}, \mathcal{F})$ *is optimal, with its makespan $C^{\pi}_{\max} = C^{*}_{\max} = 2p_1 + P(\mathcal{F})$.*

Proof. The schedule depicted in Fig. 3.2 is feasible since we have the following inequalities: $Q(\mathcal{A}) \le p_1$, $Q(\mathcal{B}) \le p_1$. The makespan is achieved on M_3 and $C^{\pi}_{\max} = 2p_1 + P(\mathcal{F})$, which meets the lower bound in Eq. (1). □

Lemma 3.2. *If both $Q(\mathcal{A}) \ge p_1$ and $Q(\mathcal{B}) \ge p_1$, then the schedule π produced by* Proc$(\mathcal{A}, \mathcal{B}, \mathcal{F})$ *is optimal, with its makespan $C^{\pi}_{\max} = C^{*}_{\max} = P(\mathcal{F}) + Q(\mathcal{O})$.*

Proof. In this case, in the schedule π produced by Proc$(\mathcal{A}, \mathcal{B}, \mathcal{F})$, the machine M_1 does not idle since $p_1 \ge q_{\ell+1}$ and the jobs of $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{F}$ are processed in the LPT order. The machine M_2 does not idle either since $Q(\mathcal{A}) \ge p_1$ and the jobs of $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{F}$ are processed in the LPT order. The machine M_3 does not idle since $Q(\mathcal{B}) \ge p_1 \ge q_{\ell+1}$, $Q(\mathcal{A}) + Q(\mathcal{B}) \ge 2p_1$, and the jobs of $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{F}$ are processed in the LPT order. Therefore, the makespan of this schedule is $C^{\pi}_{\max} = P(\mathcal{F}) + Q(\mathcal{O})$, which meets the lower bound in Eq. (1) (see for an illustration in Fig. 3.3). This finishes the proof of the lemma. □

Now we are ready to present the approximation algorithm $A(\epsilon)$, for any $\epsilon > 0$.

In the first step, we check whether $Q(\mathcal{O}) \le p_1$ or not. If $Q(\mathcal{O}) \le p_1$, then we run Proc$(\mathcal{O}, \emptyset, \mathcal{F})$ to construct a schedule π and terminate the algorithm. The schedule π is optimal by Lemma 3.1. Otherwise, we next check whether $Q(\mathcal{O}) \ge 3p_1$

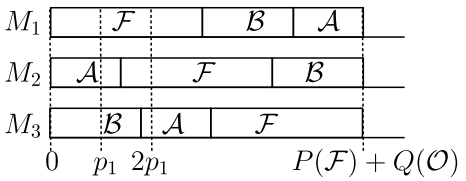

Fig. 3.3. An illustration of the schedule π produced by PROC$(\mathcal{A}, \mathcal{B}, \mathcal{F})$ when both $Q(\mathcal{A}) \geq p_1$ and $Q(\mathcal{B}) \geq p_1$.

ALGORITHM $A(\epsilon)$:

1. If $Q(\mathcal{O}) \leq p_1$, then run PROC$(\mathcal{O}, \emptyset, \mathcal{F})$ to produce a schedule π;
   output the schedule π;
   if $Q(\mathcal{O}) \geq 3p_1$, then greedily bipartition $\mathcal{O}$ into $\mathcal{A}$ and $\mathcal{B}$ such that both $Q(\mathcal{A}) \geq p_1$ and $Q(\mathcal{B}) \geq p_1$, and run PROC$(\mathcal{A}, \mathcal{B}, \mathcal{F})$ to produce a schedule π;
   output the schedule π.
2. Construct an instance of SUBSET-SUM,
   in which a number q_i corresponds to the job $J_i \in \mathcal{O}$ and the bound is p_1.
   2.1. Run a $(1+\epsilon)$-approximation for MIN-SUBSET-SUM to obtain a job subset $\mathcal{A}$.
   2.2. Run PROC$(\mathcal{A}, \mathcal{O} \setminus \mathcal{A}, \mathcal{F})$ to construct a schedule π^1.
3. 3.1. Run a $(1-\epsilon)$-approximation for MAX-SUBSET-SUM to obtain a job subset $\mathcal{B}$.
   3.2. Run PROC$(\mathcal{O} \setminus \mathcal{B}, \mathcal{B}, \mathcal{F})$ to construct a schedule π^2.
4. Output the schedule with a smaller makespan between π^1 and π^2.

Fig. 3.4. A high-level description of the algorithm $A(\epsilon)$.

or not. If $Q(\mathcal{O}) \geq 3p_1$, then we may greedily bipartition $\mathcal{O}$ into $\mathcal{A}$ and $\mathcal{B}$ such that both $Q(\mathcal{A}) \geq p_1$ and $Q(\mathcal{B}) \geq p_1$ (we remark that this is doable since $p_1 \geq q_{\ell+1}$), followed by running PROC$(\mathcal{A}, \mathcal{B}, \mathcal{F})$ to construct a schedule π and terminate the algorithm. The schedule π is optimal by Lemma 3.2.

In the second step ($p_1 < Q(\mathcal{O}) < 3p_1$ holds), the algorithm $A(\epsilon)$ constructs an instance of the SUBSET-SUM problem [3], where every job $J_i \in \mathcal{O}$ gives a number q_i and the bound is set to p_1. The MIN-SUBSET-SUM problem is to find a subset of $\{q_i \mid J_i \in \mathcal{O}\}$ of minimum sum that *is greater than* the bound p_1, and it admits an FPTAS which is derived from an FPTAS for the MAX-SUBSET-SUM problem [5]. Using the fact that $p_1 < Q(\mathcal{O}) < 3p_1$, the algorithm $A(\epsilon)$ runs an $O(\min\{n/\epsilon, n + 1/\epsilon^2 \log(1/\epsilon)\})$-time $(1+\epsilon)$-approximation algorithm for the MIN-SUBSET-SUM problem to obtain a job subset $\mathcal{A}$. (We remark that $A(\epsilon)$ can actually run an $O(\min\{n/\epsilon, n + 1/\epsilon^2 \log(1/\epsilon)\})$-time $(1-\epsilon/2)$-approximation algorithm for the MAX-SUBSET-SUM objective with the bound set to $Q(\mathcal{O}) - p_1$, then take the complement as $\mathcal{A}$.) It then runs PROC$(\mathcal{A}, \mathcal{O} \setminus \mathcal{A}, \mathcal{F})$ to construct a schedule, denoted as π^1.

The MAX-SUBSET-SUM problem is to find a subset of $\{q_i \mid J_i \in \mathcal{O}\}$ of maximum sum that *is less than or equal to* the bound p_1, and it admits an FPTAS [5]. In the third step, the algorithm $A(\epsilon)$ runs the $O(\min\{n/\epsilon, n + 1/\epsilon^2 \log(1/\epsilon)\})$-time $(1-\epsilon)$-approximation algorithm for the MAX-SUBSET-SUM problem to obtain a job subset $\mathcal{B}$. Then it runs PROC$(\mathcal{O} \setminus \mathcal{B}, \mathcal{B}, \mathcal{F})$ to construct a schedule, denoted as π^2.

Lastly, the algorithm $A(\epsilon)$ outputs the schedule with a smaller makespan between π^1 and π^2. A high-level description of $A(\epsilon)$ is provided in Fig. 3.4.

In the following performance analysis, we assume without of loss of generality that $Q(\mathcal{O}) > p_1$. We have the following (in-)equalities inside the algorithm $A(\epsilon)$:

$$\text{OPT}^1 = \min\{Q(\mathcal{X}) \mid \mathcal{X} \subseteq \mathcal{O},\ Q(\mathcal{X}) > p_1\}; \tag{2}$$

$$p_1 < Q(\mathcal{A}) \ \leq \ (1+\epsilon)\text{OPT}^1; \tag{3}$$

$$\text{OPT}^2 = \max\{Q(\mathcal{Y}) \mid \mathcal{Y} \subseteq \mathcal{O},\ Q(\mathcal{Y}) \leq p_1\}; \tag{4}$$

$$p_1 \geq Q(\mathcal{B}) \ \geq \ (1-\epsilon)\text{OPT}^2, \tag{5}$$

where OPT^1 (OPT^2, respectively) is the optimum to the constructed MIN-SUBSET-SUM (MAX-SUBSET-SUM, respectively) problem.

Lemma 3.3. *In the algorithm $A(\epsilon)$, if $Q(\mathcal{O} \setminus \mathcal{A}) \leq p_1 - \epsilon\text{OPT}^1$, then for any bipartition $\{\mathcal{X}, \mathcal{Y}\}$ of the job set $\mathcal{O}$, $Q(\mathcal{X}) > p_1$ implies $Q(\mathcal{Y}) \leq p_1$.*

Proof. Note that the job subset $\mathcal{A}$ is computed in Step 2.1 of the algorithm $A(\epsilon)$, and it satisfies Eq. (3). By the definition of OPT^1 in Eq. (2) and using Eq. (3), we have $Q(\mathcal{X}) \geq \text{OPT}^1 \geq Q(\mathcal{A}) - \epsilon\text{OPT}^1$. Furthermore, from the fact that $Q(\mathcal{O}) = Q(\mathcal{X}) + Q(\mathcal{Y}) = Q(\mathcal{A}) + Q(\mathcal{O} \setminus \mathcal{A})$ and the assumption that $Q(\mathcal{O} \setminus \mathcal{A}) \leq p_1 - \epsilon\text{OPT}^1$, we have

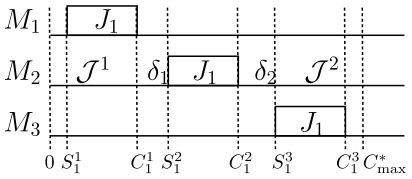

Fig. 3.5. An illustration of an optimal schedule π^*, in which $\mathcal{J}^1$ and $\mathcal{J}^2$ are the subsets of jobs processed on M_2 before J_1 and after J_1, respectively; δ_1 and δ_2 are the total amount of machine idle time for M_2 before processing J_1 and after processing J_1, respectively.

$$\begin{aligned} Q(\mathcal{Y}) &= Q(\mathcal{A}) + Q(\mathcal{O}\setminus\mathcal{A}) - Q(\mathcal{X}) \\ &\le Q(\mathcal{A}) + Q(\mathcal{O}\setminus\mathcal{A}) - (Q(\mathcal{A}) - \epsilon\mathrm{OPT}^1) \\ &= Q(\mathcal{O}\setminus\mathcal{A}) + \epsilon\mathrm{OPT}^1 \\ &\le p_1 - \epsilon\mathrm{OPT}^1 + \epsilon\mathrm{OPT}^1 \\ &= p_1. \end{aligned}$$

This finishes the proof of the lemma. □

Lemma 3.4. *In the algorithm $A(\epsilon)$, if $Q(\mathcal{O}\setminus\mathcal{A}) \le p_1 - \epsilon\mathrm{OPT}^1$, then $C^*_{\max} \ge P(\mathcal{F}) + Q(\mathcal{O}) + p_1 - \mathrm{OPT}^2$.*

Proof. Consider an arbitrary optimal schedule π^* that achieves the makespan $C^*_{\max}$. Note that the flow-shop job J_1 is first processed on the machine M_1, then on machine M_2, and last on machine M_3.

On the machine M_2, let $\mathcal{J}^1 = \mathcal{O}^1 \cup \mathcal{F}^1$ denote the subset of jobs processed before J_1, and $\mathcal{J}^2 = \mathcal{O}^2 \cup \mathcal{F}^2$ denote the subset of jobs processed after J_1, where $\{\mathcal{O}^1, \mathcal{O}^2\}$ is a bipartition of the job set $\mathcal{O}$ and $\{\mathcal{F}^1, \mathcal{F}^2\}$ is a bipartition of the job set $\mathcal{F}\setminus J_1$. Also, let δ_1 and δ_2 denote the total amount of machine idle time for M_2 before processing J_1 and after processing J_1, respectively (see Fig. 3.5 for an illustration).

Note that $\mathcal{F} = J_1 \cup \mathcal{F}^1 \cup \mathcal{F}^2$ is the set of flow-shop jobs. The job J_1 and the jobs of $\mathcal{F}^1$ should be finished on the machine M_1 before time S_1^2, and the job J_1 and the jobs of $\mathcal{F}^2$ can only be started on the machine M_3 after time C_1^2. That is,

$$p_1 + P(\mathcal{F}^1) \le S_1^2 \tag{6}$$

and

$$p_1 + P(\mathcal{F}^2) \le C^*_{\max} - C_1^2. \tag{7}$$

If $Q(\mathcal{O}^1) \le p_1$, then we have $Q(\mathcal{O}^1) \le \mathrm{OPT}^2$ by the definition of OPT^2 in Eq. (4). Combining this with Eq. (6), we achieve that $\delta_1 = S_1^2 - P(\mathcal{F}^1) - Q(\mathcal{O}^1) \ge p_1 - \mathrm{OPT}^2$.

If $Q(\mathcal{O}^1) > p_1$, then we have $Q(\mathcal{O}^2) \le p_1$ by Lemma 3.3. Hence, $Q(\mathcal{O}^2) \le \mathrm{OPT}^2$ by the definition of OPT^2 in Eq. (4). Combining this with Eq. (7), we achieve that $\delta_2 = C^*_{\max} - C_1^2 - P(\mathcal{F}^2) - Q(\mathcal{O}^2) \ge p_1 - \mathrm{OPT}^2$.

The last two paragraphs prove that $\delta_1 + \delta_2 \ge p_1 - \mathrm{OPT}^2$. Therefore,

$$\begin{aligned} C^*_{\max} &\ge Q(\mathcal{O}^1) + P(\mathcal{F}^1) + \delta_1 + p_1 + Q(\mathcal{O}^2) + P(\mathcal{F}^2) + \delta_2 \\ &= P(\mathcal{F}) + Q(\mathcal{O}) + \delta_1 + \delta_2 \\ &\ge P(\mathcal{F}) + Q(\mathcal{O}) + p_1 - \mathrm{OPT}^2. \end{aligned}$$

This finishes the proof of the lemma. □

Lemma 3.5. *In the algorithm $A(\epsilon)$, if $Q(\mathcal{O}\setminus\mathcal{A}) \le p_1 - \epsilon\mathrm{OPT}^1$, then $C^{\pi^2}_{\max} < (1+\epsilon)C^*_{\max}$.*

Proof. Denote $\overline{\mathcal{B}} = \mathcal{O}\setminus\mathcal{B}$. Note that the job set $\mathcal{B}$ computed in Step 3.1 of the algorithm $A(\epsilon)$ satisfies $p_1 \ge Q(\mathcal{B}) \ge (1-\epsilon)\mathrm{OPT}^2$, and the schedule π^2 is constructed by $\textsc{Proc}(\overline{\mathcal{B}}, \mathcal{B}, \mathcal{F})$. We distinguish the following two cases according to the value of $Q(\overline{\mathcal{B}})$.

Case 1. $Q(\overline{\mathcal{B}}) \le p_1$. In this case, the schedule π^2 is optimal by Lemma 3.1.

Case 2. $Q(\overline{\mathcal{B}}) > p_1$. The schedule π^2 constructed by $\textsc{Proc}(\overline{\mathcal{B}}, \mathcal{B}, \mathcal{F})$ has the following properties (see Fig. 3.6 for an illustration):

1. The jobs are processed consecutively on the machine M_1 since J_1 is the largest job. The completion time of M_1 is thus $C_1^{\pi^2} = Q(\mathcal{O}) + P(\mathcal{F})$.

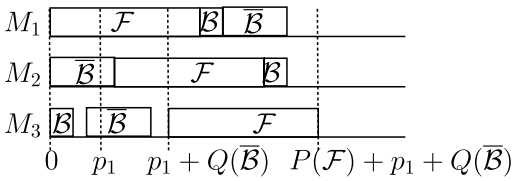

Fig. 3.6. An illustration of the schedule π^2 constructed by Proc$(\overline{\mathcal{B}}, \mathcal{B}, \mathcal{F})$ in Case 2, where $Q(\mathcal{B}) \le p_1$ and $Q(\overline{\mathcal{B}}) > p_1$. The machines M_1 and M_2 do not idle; the machine M_3 may idle between processing the job set $\mathcal{B}$ and the job set $\overline{\mathcal{B}}$ and may idle between processing the job set $\overline{\mathcal{B}}$ and the job set $\mathcal{F}$. M_3 starts processing the job set $\mathcal{F}$ at time $p_1 + Q(\overline{\mathcal{B}})$.

2. The jobs are processed consecutively on the machine M_2 due to $Q(\mathcal{B}) \le p_1$ and $Q(\overline{\mathcal{B}}) > p_1$. The completion time of M_2 is thus $C_2^{\pi^2} = Q(\mathcal{O}) + P(\mathcal{F})$.
3. The machine M_3 starts processing the job set $\mathcal{F}$ consecutively at time $p_1 + Q(\overline{\mathcal{B}})$ due to $Q(\mathcal{B}) \le p_1$. The completion time of M_3 is $C_3^{\pi^2} = P(\mathcal{F}) + p_1 + Q(\overline{\mathcal{B}})$.

Note that $C_3^{\pi^2} = P(\mathcal{F}) + p_1 + Q(\overline{\mathcal{B}}) \ge P(\mathcal{F}) + Q(\mathcal{B}) + Q(\overline{\mathcal{B}}) = Q(\mathcal{O}) + P(\mathcal{F})$, implying $C_{\max}^{\pi^2} = P(\mathcal{F}) + p_1 + Q(\overline{\mathcal{B}})$. Combining Eq. (5) with Lemma 3.4, we have

$$\begin{aligned} C_{\max}^{\pi^2} &= P(\mathcal{F}) + p_1 + Q(\overline{\mathcal{B}}) \\ &= P(\mathcal{F}) + Q(\mathcal{O}) + p_1 - Q(\mathcal{B}) \\ &\le P(\mathcal{F}) + Q(\mathcal{O}) + p_1 - (1-\epsilon)\mathrm{OPT}^2 \\ &\le C_{\max}^* + \epsilon \mathrm{OPT}^2 \\ &< (1+\epsilon) C_{\max}^*, \end{aligned}$$

where the last inequality is due to $\mathrm{OPT}^2 \le p_1 < C_{\max}^*$. This finishes the proof of the lemma. □

Lemma 3.6. *In the algorithm $A(\epsilon)$, if $p_1 - \epsilon\mathrm{OPT}^1 < Q(\mathcal{O} \setminus \mathcal{A}) < p_1$, then $C_{\max}^{\pi^1} < (1+\epsilon)C_{\max}^*$.*

Proof. Denote $\overline{\mathcal{A}} = \mathcal{O} \setminus \mathcal{A}$. Note that the job set $\mathcal{A}$ computed in Step 2.1 of the algorithm $A(\epsilon)$ satisfies $p_1 < Q(\mathcal{A}) \le (1+\epsilon)\mathrm{OPT}^1$, and the schedule π^1 is constructed by Proc$(\mathcal{A}, \overline{\mathcal{A}}, \mathcal{F})$.

By a similar argument as in Case 2 in the proof of Lemma 3.5, replacing the two job sets $\mathcal{B}, \overline{\mathcal{B}}$ by the two job sets $\overline{\mathcal{A}}, \mathcal{A}$, we conclude that the makespan of the schedule π^1 is achieved on the machine M_3, $C_{\max}^{\pi^1} = P(\mathcal{F}) + Q(\mathcal{O}) + p_1 - Q(\overline{\mathcal{A}})$. Combining Eq. (1) with the assumption that $p_1 - \epsilon\mathrm{OPT}^1 < Q(\overline{\mathcal{A}})$, we have

$$C_{\max}^{\pi^1} < P(\mathcal{F}) + Q(\mathcal{O}) + \epsilon\mathrm{OPT}^1 \le C_{\max}^* + \epsilon\mathrm{OPT}^1 \le (1+\epsilon)C_{\max}^*,$$

where the last inequality follows from $\mathrm{OPT}^1 \le Q(\mathcal{O}) \le C_{\max}^*$.

This finishes the proof of the lemma. □

Theorem 3.7. *The algorithm $A(\epsilon)$ is an $O(\min\{n/\epsilon, n + 1/\epsilon^2 \log(1/\epsilon)\})$-time $(1+\epsilon)$-approximation for the $M3 \mid prpt \mid C_{\max}$ problem when $p_1 \ge q_{\ell+1}$.*

Proof. First of all, the procedure Proc$(\mathcal{X}, \mathcal{Y}, \mathcal{F})$ on a bipartition $\{\mathcal{X}, \mathcal{Y}\}$ of the job set $\mathcal{O}$ takes $O(n \log n)$ time. Recall that the job set $\mathcal{A}$ is computed by a $(1+\epsilon)$-approximation for the Min-Subset-Sum problem, in $O(\min\{n/\epsilon, n + 1/\epsilon^2 \log(1/\epsilon)\})$ time; the other job set $\mathcal{B}$ is computed by a $(1-\epsilon)$-approximation for the Max-Subset-Sum problem, also in $O(\min\{n/\epsilon, n + 1/\epsilon^2 \log(1/\epsilon)\})$ time. The total running time of the algorithm $A(\epsilon)$ is thus polynomial in both n and $1/\epsilon$.

When $Q(\mathcal{O}) \le p_1$, the schedule constructed in the algorithm $A(\epsilon)$ is optimal by Lemma 3.1; when $Q(\mathcal{O}) \ge 3p_1$, or the job set $\mathcal{O} \setminus \mathcal{A}$ computed in Step 2.1 of the algorithm $A_1(\epsilon)$ has total processing time not less than p_1, the schedule constructed in the algorithm $A(\epsilon)$ is optimal by Lemma 3.2. When $Q(\mathcal{O} \setminus \mathcal{A}) < p_1$, the smaller makespan between the two schedules π^1 and π^2 constructed by the algorithm $A(\epsilon)$ is less than $(1+\epsilon)$ of the optimum by Lemmas 3.5 and 3.6. Therefore, the algorithm $A(\epsilon)$ has a worst-case performance ratio of $(1+\epsilon)$. This finishes the proof of the theorem. □

4. A 4/3-approximation for the case where $p_1 < q_{\ell+1}$

In this section, we present a 4/3-approximation algorithm for the $M3 \mid prpt \mid C_{\max}$ problem when $p_1 < q_{\ell+1}$, and we show that this ratio of 4/3 is asymptotically tight.

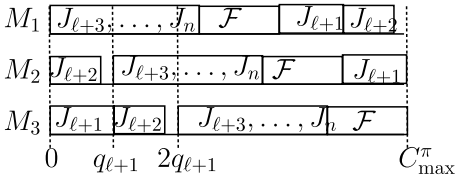

Fig. 4.1. A feasible schedule π for the $M3 \mid prpt \mid C_{\max}$ problem when there are at least two open-shop jobs and $p_1 < q_{\ell+1}$.

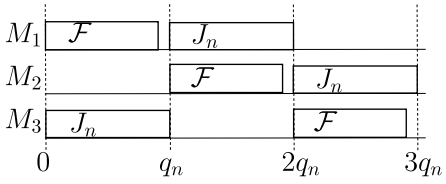

Fig. 4.2. A feasible schedule π for the $M3 \mid prpt \mid C_{\max}$ problem when there is only one open-shop job J_n and $p_1 < q_n$; the configuration shown here corresponds to $P(\mathcal{F}) \le q_n$ and thus π is optimal.

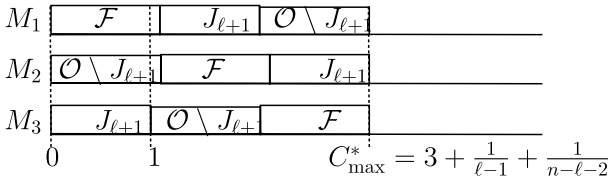

Fig. 4.3. An optimal schedule for the constructed instance of the $M3 \mid prpt \mid C_{\max}$ problem, in which $p_i = \frac{1}{\ell-1}$ for all $i = 1, 2, \ldots, n$, $q_{\ell+1} = 1$ and $q_i = \frac{1}{n-\ell-2}$ for all $i = \ell+2, \ell+3, \ldots, n$.

Theorem 4.1. *When $p_1 < q_{\ell+1}$, the $M3 \mid prpt \mid C_{\max}$ problem admits an $O(n \log n)$-time 4/3-approximation algorithm.*

Proof. Consider first the case where there are at least two open-shop jobs. Construct a permutation schedule π in which the job processing order for M_1 is $\langle J_{\ell+3}, \ldots, J_n, \mathcal{F}, J_{\ell+1}, J_{\ell+2}\rangle$, where the jobs of $\mathcal{F}$ are processed in the LPT order; the job processing order for M_2 is $\langle J_{\ell+2}, J_{\ell+3}, \ldots, J_n, \mathcal{F}, J_{\ell+1}\rangle$; the job processing order for M_3 is $\langle J_{\ell+1}, J_{\ell+2}, J_{\ell+3}, \ldots, J_n, \mathcal{F}\rangle$. See Fig. 4.1 for an illustration, where the start time for $J_{\ell+3}$ on M_2 is $q_{\ell+1}$, and the start time for $J_{\ell+3}$ on M_3 is $2q_{\ell+1}$. One can check that the schedule π is feasible when $p_1 < q_{\ell+1}$, and it can be constructed in $O(n \log n)$ time.

The makespan of the schedule π is $C^{\pi}_{\max} = P(\mathcal{F}) + Q(\mathcal{O}) + q_{\ell+1} - q_{\ell+2}$. Combining this with Eq. (1), we have

$$C^{\pi}_{\max} \le P(\mathcal{F}) + Q(\mathcal{O}) + q_{\ell+1} \le \frac{4}{3} C^*_{\max}.$$

When there is only one open-shop job J_n (that is, $\mathcal{F} = \{J_1, J_2, \ldots, J_{n-1}\}$ and $\mathcal{O} = \{J_n\}$), construct a permutation schedule π in which the job processing order for M_1 is $\langle \mathcal{F}, J_n\rangle$, where the jobs of $\mathcal{F}$ are processed in the LPT order; the job processing order for M_2 is $\langle \mathcal{F}, J_n\rangle$; the job processing order for M_3 is $\langle J_n, \mathcal{F}\rangle$ (see for an illustration in Fig. 4.2). If $P(\mathcal{F}) \le q_n$, which is shown in Fig. 4.2, then π has makespan $3q_n$ and thus is optimal. If $P(\mathcal{F}) > q_n$, then π has makespan $C^{\pi}_{\max} \le 2q_n + P(\mathcal{F}) \le \frac{4}{3} C^*_{\max}$ by Eq. (1). This finishes the proof of the theorem. □

Remark 4.2. Construct an instance in which $p_i = \frac{1}{\ell-1}$ for all $i = 1, 2, \ldots, \ell$, $q_{\ell+1} = 1$ and $q_i = \frac{1}{n-\ell-2}$ for all $i = \ell+2, \ell+3, \ldots, n$. Then for this instance, the schedule π constructed in the proof of Theorem 4.1 has makespan $C^{\pi}_{\max} = 4 + \frac{1}{\ell-1}$; an optimal schedule has makespan $C^*_{\max} = 3 + \frac{1}{\ell-1} + \frac{1}{n-\ell-2}$ (see for an illustration in Fig. 4.3). This suggests that the approximation ratio of 4/3 is asymptotically tight for the algorithm presented in the proof of Theorem 4.1.

5. NP-hardness for $M3 \mid prpt, (n-1, 1) \mid C_{\max}$ when $p_1 < q_n$

Recall that we use $M3 \mid prpt, (n-1, 1) \mid C_{\max}$ to denote the special of $M3 \mid prpt \mid C_{\max}$ where there is only one open-shop job, *i.e.*, $\mathcal{F} = \{J_1, J_2, \ldots, J_{n-1}\}$ and $\mathcal{O} = \{J_n\}$. In this section, we show that this special case $M3 \mid prpt, (n-1, 1) \mid C_{\max}$ is already NP-hard if the unique open-shop job is larger than any flow-shop job, *i.e.*, $p_1 < q_n$. We prove the NP-hardness through a reduction from the Partition problem [3], which is a well-known NP-complete problem.

Theorem 5.1. *The $M3 \mid prpt, (n-1, 1) \mid C_{\max}$ problem is NP-hard if the open-shop job is strictly larger than any flow-shop job.*

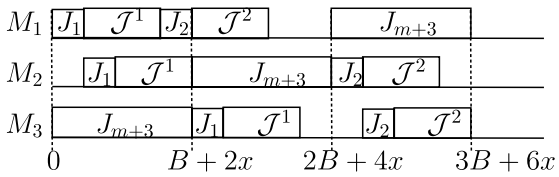

Fig. 5.1. A feasible schedule π for the constructed instance of the $M3 \mid prpt \mid C_{\max}$ problem, when the set S can be partitioned into two equal parts S_1 and S_2. The partition of the flow-shop jobs $\{\mathcal{J}^1, \mathcal{J}^2\}$ is correspondingly constructed. In the schedule, the jobs of $\mathcal{J}^1$ and the jobs of $\mathcal{J}^2$ are processed in the LPT order, respectively.

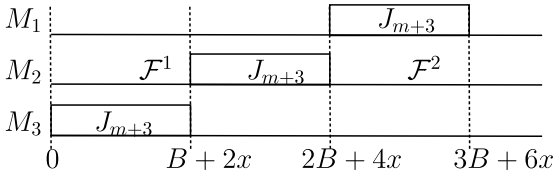

Fig. 5.2. An illustration of an optimal schedule for the constructed instance of the $M3 \mid prpt, (n-1,1) \mid C_{\max}$ problem with $\mathcal{O} = \{J_{m+3}\}$ and $q_{m+3} = B + 2x$. Its makespan is $C^*_{\max} = 3B + 6x = 3q_{m+3}$.

Proof. An instance of the PARTITION problem consists of a set $S = \{a_1, a_2, a_3, \ldots, a_m\}$ where each a_i is a positive integer and $a_1 + a_2 + \ldots + a_m = 2B$, and the query is whether or not S can be partitioned into two parts such that each part sums to exactly B.

Let $x > B$, and we assume that $a_1 \geq a_2 \geq \ldots \geq a_m$.

We construct an instance of the $M3 \mid prpt \mid C_{\max}$ problem as follows: there are in total $m+2$ flow-shop jobs, and their processing times are $p_1 = x$, $p_2 = x$, and $p_{i+2} = a_i$ for $i = 1, 2, \ldots, m$; there is only one open-shop job with processing time $q_{m+3} = B + 2x$. Note that the total number of jobs is $n = m + 3$, and one sees that the open-shop job is larger than any flow-shop job.

If the set S can be partitioned into two parts S_1 and S_2 such that each part sums to exactly B, then we let $\mathcal{J}^1 = J_1 \cup \{J_i \mid a_i \in S_1\}$ and $\mathcal{J}^2 = J_2 \cup \{J_i \mid a_i \in S_2\}$. We construct a permutation schedule π in which the job processing order for M_1 is $\langle \mathcal{J}^1, \mathcal{J}^2, J_{m+3} \rangle$, where the jobs of $\mathcal{J}^1$ and the jobs of $\mathcal{J}^2$ are processed in the LPT order, respectively; the job processing order for M_2 is $\langle \mathcal{J}^1, J_{m+3}, \mathcal{J}^2 \rangle$; the job processing order for M_3 is $\langle J_{m+3}, \mathcal{J}^1, \mathcal{J}^2 \rangle$. See Fig. 5.1 for an illustration, in which J_1 starts at time 0 on M_1, starts at time x on M_2, and starts at time $B + 2x$ on M_3; J_2 starts at time $B + x$ on M_1, starts at time $2B + 4x$ on M_2, and starts at time $2B + 5x$ on M_3; J_{m+3} starts at time 0 on M_3, starts at time $B + 2x$ on M_2, and starts at time $2B + 4x$ on M_1. The feasibility is trivial and its makespan is $C^{\pi}_{\max} = 3q_{m+3} = 3B + 6x$, suggesting the optimality.

Conversely, if the optimal makespan for the constructed instance is $C^*_{\max} = 3B + 6x = 3q_{m+3}$, then we will show next that S admits a partition into two equal parts.

Firstly, we see that the second machine processing the open-shop job J_{m+3} cannot be M_1, since otherwise M_1 has to process all the jobs of $\mathcal{F}$ before J_{m+3}, leading to a makespan larger than $3B + 6x$; the second machine processing the open-shop job J_{m+3} cannot be M_3 either, since otherwise M_3 has no room to process any job of $\mathcal{F}$ before J_{m+3}, leading to a makespan larger than $3B + 6x$ too. Therefore, the second machine processing the open-shop job J_{m+3} has to be M_2, see Fig. 5.2 for an illustration.

Denote the job subsets processed before and after the job J_{m+3} on M_2 as $\mathcal{F}^1$ and $\mathcal{F}^2$, respectively. Then all these flow-shop jobs of $\mathcal{F}^1$ are started by M_1 and M_2 at time 0 and must be finished by them by time $B + 2x$, and likewise all those flow-shop jobs of $\mathcal{F}^2$ are started by M_2 and M_3 at time $2B + 4x$ and finished by them by time $3B + 6x$. Let y and z be the longest individual job processing time among the jobs of $\mathcal{F}^1$ and among the jobs of $\mathcal{F}^2$, respectively. We have $y + P(\mathcal{F}^1) \leq B + 2x$ and $z + P(\mathcal{F}^2) \leq B + 2x$. Since $x > B$ and $\max\{y, z\} = x$, we conclude that $P(\mathcal{F}^1) = P(\mathcal{F}^2) = B + x$; then again due to $x > B$, neither of $\mathcal{F}^1$ and $\mathcal{F}^2$ may contain both J_1 and J_2, which have processing times x. That is, $\mathcal{F}^1$ and $\mathcal{F}^2$ each contains exactly one of J_1 and J_2. Therefore, the jobs of $\mathcal{F}^1 \setminus \{J_1, J_2\}$ have a total processing time of exactly B, suggesting a subset of S sums to exactly B. This finishes the proof of the theorem. □

6. An FPTAS for $M3 \mid prpt, (n-1,1) \mid C_{\max}$

Recall that the FPTAS in Theorem 3.7 designed for the general $M3 \mid prpt, (n-1,1) \mid C_{\max}$ problem when $p_1 \geq q_n$ also works for the special case $M3 \mid prpt, (n-1,1) \mid C_{\max}$ when $p_1 \geq q_n$. In this section, we design another FPTAS for the $M3 \mid prpt, (n-1,1) \mid C_{\max}$ problem when $p_1 < q_n$, which is denoted as $M3 \mid prpt, (n-1,1), p_1 < q_n \mid C_{\max}$ for simplicity. That is, we design a $(1+\epsilon)$-approximation algorithm $C(\epsilon)$ for this case, for any given $\epsilon > 0$, and its running time is polynomial in both n and $1/\epsilon$.

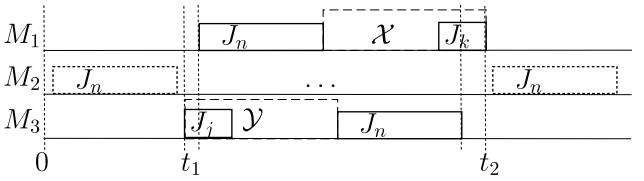

Fig. 6.1. Two possible values for t_1: if the job J_j as shown exists, then $t_1 = S_j^3$, otherwise $t_1 = S_n^1$; symmetrically, two possible values for t_2: if the job J_k as shown exists, then $t_2 = C_k^1$, otherwise $t_2 = C_n^3$.

6.1. A polynomial-time solvable case

The following lemma states that if the total processing time of all the flow-shop jobs is no greater than q_n, then we can easily construct an optimal schedule in linear time.

Lemma 6.1. *If $P(\mathcal{F}) \le q_n$, then the $M3 \mid prpt, (n-1,1), p_1 < q_n \mid C_{\max}$ problem is solvable in linear time and $C^*_{\max} = 3q_n$.*

Proof. In this case, we construct a permutation schedule π in which the job processing order for M_1 is $\langle \mathcal{F}, J_n \rangle$, where the jobs of $\mathcal{F}$ are processed in the given (arbitrary, no need to be sorted) order; the job processing order for M_2 is $\langle \mathcal{F}, J_n \rangle$; the job processing order for M_3 is $\langle J_n, \mathcal{F} \rangle$. As depicted in Fig. 4.2, the jobs of $\mathcal{F}$ are processed consecutively on each machine, starting at time 0 on M_1, starting at time q_n on M_2, and starting at time $2q_n$ on M_3; the unique open-shop job J_n starts at time 0 on M_3, starts at time q_n on M_1, and starts at time $2q_n$ on M_2. The schedule is feasible due to $P(\mathcal{F}) \le q_n$ and its makespan is $3q_n$, and thus by Eq. (1) it is an optimal schedule.

This proves the lemma. □

6.2. Structural properties of optimal schedules

We assume in the rest of the section that $P(\mathcal{F}) > q_n$, from which we conclude that $\mathcal{F}$ contains at least two jobs. We explore the structural properties of optimal schedules for designing our approximation algorithm.

Lemma 6.2. *There exists an optimal schedule for the $M3 \mid prpt, (n-1,1), p_1 < q_n \mid C_{\max}$ problem in which the open-shop job J_n is processed on M_3 before it is processed on M_1.*

Proof. We prove the lemma by construction. Suppose π^* is an optimal schedule in which the open-shop job J_n is processed on M_3 after it is processed on M_1. (That is, the machine order for J_n is $\langle -, M_1, -, M_3, - \rangle$, with exactly one of the $-$'s replaced by M_2.) Recall that we use S_j^i (C_j^i, respectively) to denote the start (finish, respectively) time of the job J_j on M_i in π^*. Clearly, $C_n^1 \le S_n^3$.

We determine in the following two time points t_1 and t_2.

We see that if J_n is first processed on M_2, *i.e.*, $C_n^2 \le S_n^1$, then all the jobs of $\mathcal{F}$ processed before J_n on M_2 can be finished on M_3 by time C_n^2, due to $p_1 < q_n$. It follows that if J_n is first processed on M_2, then we may assume without loss of generality that in π^*, there is no job J_j such that $S_j^3 < C_n^2 < C_j^3$ (that is, the processing periods of J_j on M_3 and of J_n on M_2 intersect at time C_n^2). If there is a job J_j such that $S_j^3 < S_n^1 < C_j^3$ (that is, the processing periods of J_j on M_3 and of J_n on M_1 intersect at time S_n^1), then we set $t_1 = S_j^3$, otherwise we set $t_1 = S_n^1$ (see Fig. 6.1 for an illustration).

Symmetrically, if J_n is last processed on M_2, *i.e.*, $C_n^3 \le S_n^2$, then all the jobs of $\mathcal{F}$ processed after J_n on M_2 can be started on M_1 by time S_n^2, due to $p_1 < q_n$. It follows that if J_n is last processed on M_2, then we may assume without loss of generality that in π^*, there is no job J_k such that $S_k^1 < S_n^2 < C_k^1$ (that is, the processing periods of J_k on M_1 and of J_n on M_2 intersect at time S_n^2). If there is a job J_k such that $S_k^1 < C_n^3 < C_k^1$ (that is, the processing periods of J_k on M_1 and of J_n on M_3 intersect at time C_n^3), then we set $t_2 = C_k^1$, otherwise we set $t_2 = C_n^3$ (see Fig. 6.1 for an illustration).

We note that both t_1 and t_2 are well defined. On the machine M_2, the job J_n is either finished before time t_1 (*i.e.*, $C_n^2 \le t_1$), or started after time t_2 (*i.e.*, $t_2 \le S_n^2$), or is processed in between the closed time interval $[C_n^1, S_n^3]$. We thus obtain from π^* another schedule π by 1) moving the job subset $\mathcal{X}$ originally processed on M_1 in between the closed time interval $[C_n^1, t_2]$ to start exactly q_n time units earlier, while moving J_n on M_1 to start at time $t_2 - q_n$, and 2) moving the job subset $\mathcal{Y}$ originally processed on M_3 in between the closed time interval $[t_1, S_n^3]$ to start exactly q_n time units later, while moving J_n on M_3 to start at time t_1. See for an illustration in Fig. 6.2 and how it is obtained from the configuration in Fig. 6.1. The schedule π is feasible because the processing times of J_n on all three machines are still non-overlapping and all the other jobs are flow-shop jobs which can only be processed earlier on M_1 and/or be processed later on M_3. Since no other job is moved, one clearly sees that π maintains the same makespan as π^* and thus π is also an optimal schedule, in which J_n is processed on M_3 before it is processed on M_1. This finishes the proof of the lemma. □

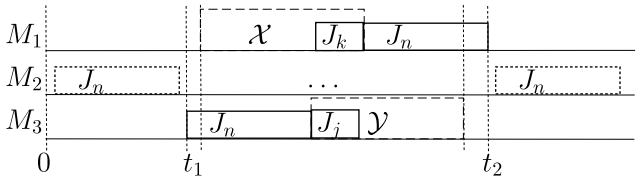

Fig. 6.2. The schedule π obtained from the optimal schedule π^* by swapping the job subset $\mathcal{X}$ processed on M_1 in between the closed time interval $[C_n^1, t_2]$ with J_n on M_1, and swapping the job subset $\mathcal{Y}$ processed on M_3 in between the closed time interval $[t_1, S_n^3]$ with J_n on M_3.

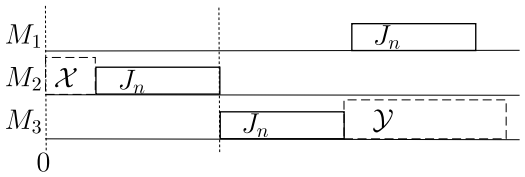

Fig. 6.3. The schedule π in which the machine order for J_n is $\langle M_2, M_3, M_1\rangle$. The job J_n starts on M_3 immediately after it is finished on M_2. The job subset $\mathcal{X}$ is processed on M_2 before J_n and the job subset $\mathcal{Y}$ is processed on M_3 after J_n.

From Lemma 6.2, we conclude that in the optimal schedules the machine order for J_n is $\langle -, M_3, -, M_1, -\rangle$, with exactly one of the $-$'s replaced by M_2. The following two lemmas discuss the optimal schedules constrained to the machine order for J_n.

Lemma 6.3. *If the machine order for J_n is forced to be $\langle M_2, M_3, M_1\rangle$ or $\langle M_3, M_1, M_2\rangle$, then an optimal schedule π^* can be constructed in $O(n \log n)$ time and its makespan is $C^*_{\max} = 2q_n + P(\mathcal{F})$.*

Proof. We prove the lemma for the case where the machine order for J_n is $\langle M_2, M_3, M_1\rangle$; the case where the machine order for J_n is $\langle M_3, M_1, M_2\rangle$ can be almost identically argued, by swapping M_3 with M_1.

Let π be a schedule in which the machine order for J_n is $\langle M_2, M_3, M_1\rangle$. Since J_n is last processed by M_1 while all the other jobs are flow-shop jobs, we may swap J_n with the job subset processed on M_1 after J_n, if any. This swapping certainly does not increase the makespan and thus we may assume that, on M_1, all the jobs of $\mathcal{F}$ are processed before J_n. We prove the lemma by showing that $C^{\pi}_{\max} \geq 2q_n + P(\mathcal{F})$ and constructing a concrete schedule with its makespan equal to $2q_n + P(\mathcal{F})$. Recall that we use S^i_j (C^i_j, respectively) to denote the start (finish, respectively) time of the job J_j on M_i in π.

Similar to the place where we determine the time point t_1 in the proof of Lemma 6.2, all the jobs of $\mathcal{F}$ processed before J_n on M_2, which form the subset $\mathcal{X}$, can be finished on M_3 by time C^2_n, due to $p_1 < q_n$. It follows that J_n may immediately start on M_3 once it is finished on M_2, *i.e.*, $S^3_n = C^2_n$, while all the other jobs of $\mathcal{F}$ not processed by time C^2_n on M_3, which form the subset $\mathcal{Y}$, can be moved to be processed after J_n (in their original processing order on M_3 in π). See for an illustration in Fig. 6.3. Since $\mathcal{X} \cup \mathcal{Y} = \mathcal{F}$, the completion time for the machine M_3 is at least $P(\mathcal{X}) + q_n + q_n + P(\mathcal{Y}) \geq 2q_n + P(\mathcal{F})$, and consequently $C^{\pi}_{\max} \geq 2q_n + P(\mathcal{F})$.

Clearly, if $\mathcal{X} = \emptyset$ in the schedule π, and the jobs of $\mathcal{F}$ are processed on all the three machines in the same LPT order, then the completion times for M_1, M_2, M_3 are $\max\{P(\mathcal{F}), 2q_n\} + q_n$, $q_n + P(\mathcal{F})$, and $2q_n + P(\mathcal{F})$, respectively. From the fact that $P(\mathcal{F}) > q_n$, we conclude that the makespan of such a schedule is exactly $2q_n + P(\mathcal{F})$. Note that the schedule can be constructed in $O(n \log n)$ time. This finishes the proof of the lemma. □

Lemma 6.4. *If the machine order for J_n is forced to be $\langle M_3, M_2, M_1\rangle$, and $\mathcal{F}^1, \mathcal{F}^2 \subseteq \mathcal{F}$ are the subsets of jobs forced to be processed before and after J_n on the machine M_2, respectively, then an optimal schedule π^* can be constructed in $O(n \log n)$ time and its makespan is $C^*_{\max} = \max\{p_{i_1} + P(\mathcal{F}^1), q_n\} + q_n + \max\{p_{i_2} + P(\mathcal{F}^2), q_n\}$, where J_{i_1} is the largest job of $\mathcal{F}^1$ and J_{i_2} is the largest job of $\mathcal{F}^2$.*

Proof. Consider a feasible π in which the machine order for J_n is $\langle M_3, M_2, M_1\rangle$, and $\mathcal{F}^1, \mathcal{F}^2 \subseteq \mathcal{F}$ are the subsets of jobs processed before and after J_n on M_2, respectively.

Similarly as in the proof of Lemma 6.3, since J_n is last processed by M_1 while all the other jobs are flow-shop jobs, we may swap J_n with the job subset processed on M_1 after J_n, if any, in the schedule π. This swapping certainly does not increase the makespan and thus we may assume that, on M_1, all the jobs of $\mathcal{F}$ are processed before J_n. Symmetrically, we may also assume that, on M_3, all the jobs of $\mathcal{F}$ are processed after J_n.

Note that $\mathcal{F}^1, \mathcal{F}^2 \subseteq \mathcal{F}$ are the subsets of jobs processed before and after J_n in the schedule π, respectively. We can also assume that $\mathcal{F}^1 \neq \emptyset$, as otherwise we may swap to process J_n first on M_2 and then to process J_n on M_3 to convert π into a feasible schedule for the case where the machine order for J_n is $\langle M_2, M_3, M_1\rangle$. It follows that the optimal schedule π^* constructed in $O(n \log n)$ time in Lemma 6.3 serves, with its makespan $C^*_{\max} = 2q_n + p(\mathcal{F})$. For the same reason, we can assume that $\mathcal{F}^2 \neq \emptyset$.

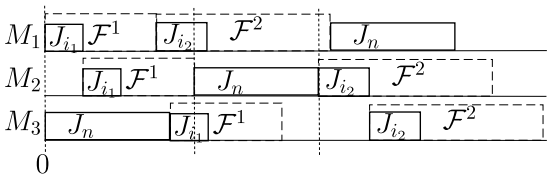

Fig. 6.4. The schedule π in which the machine order for J_n is $\langle M_3, M_2, M_1\rangle$. The job subsets $\mathcal{F}^1$ and $\mathcal{F}^2$ are processed on M_2 before and after J_n, respectively. The jobs J_{i_1} and J_{i_2} are the largest job of $\mathcal{F}^1$ and $\mathcal{F}^2$, respectively.

ALGORITHM $B(\epsilon)$:

1. Construct an instance of SUBSET-SUM, in which a number p_j corresponds to the job $J_j \in \{J_{i+1}, J_{i+2}, \ldots, J_{n-1}\}$ and the bound is $q_n - 2p_i$.
   1.1. Run a $(1+\epsilon)$-approximation for MIN-SUBSET-SUM to obtain a job subset $\mathcal{C}$.
   1.2. Use Lemma 6.4 to construct a schedule $\pi^i(\mathcal{C}, \overline{\mathcal{C}})$.
2. 2.1. Run a $(1-\epsilon)$-approximation for MAX-SUBSET-SUM to obtain a job subset $\mathcal{D}$.
   2.2. Use Lemma 6.4 to construct a schedule $\pi^i(\mathcal{D}, \overline{\mathcal{D}})$.
3. Output the better schedule between $\pi^i(\mathcal{C}, \overline{\mathcal{C}})$ and $\pi^i(\mathcal{D}, \overline{\mathcal{D}})$.

Fig. 6.5. A high-level description of the algorithm $B(\epsilon)$.

Since all the jobs of $\mathcal{F}^1$ have to be finished by time S_n^2 on the two machines M_1 and M_2, we have $S_n^2 \geq p_{i_1} + P(\mathcal{F}^1)$. From $C_n^3 \leq S_n^2$, we conclude that $S_n^2 \geq \max\{p_{i_1} + P(\mathcal{F}^1), q_n\}$. Similarly, since all the jobs of $\mathcal{F}^2$ have to be processed on the two machines M_2 and M_3, and the earliest starting time is C_n^2, the completion time for the machine M_3 is at least $C_n^2 + p_{i_2} + P(\mathcal{F}^2)$. Note that the earliest starting time for processing J_n on M_1 is C_n^2 too. It follows that the makespan of the schedule π is $C_{\max}^{\pi} \geq \max\{p_{i_1} + P(\mathcal{F}^1), q_n\} + q_n + \max\{p_{i_2} + P(\mathcal{F}^2), q_n\}$. See for an illustration in Fig. 6.4.

On the other hand, if the jobs of $\mathcal{F}^1$ are processed in the same LPT order on M_1 and M_2 in the schedule π, then they are finished by time S_n^2 on M_1 and M_2. Furthermore, due to $p_{i_1} \leq p_1 < q_n$, all the jobs of $\mathcal{F}^1$ can be processed in the same LPT order on M_3 and finished by time C_n^2. Also, if the jobs of $\mathcal{F}^2$ are processed in the LPT order on M_1 following the jobs of $\mathcal{F}^1$, then by time C_n^2, the job J_{i_2} will be finished on M_1 and thus can start on M_2 at time C_n^2. It follows that if the jobs of $\mathcal{F}^2$ are processed in the same LPT order on M_2 and M_3 in the schedule π, then they are finished by time $C_n^2 + p_{i_2} + P(\mathcal{F}^2)$ on M_2 and M_3. This way, the makespan of such a schedule π is $C_{\max}^{\pi} = \max\{p_{i_1} + P(\mathcal{F}^1), q_n\} + q_n + \max\{p_{i_2} + P(\mathcal{F}^2), q_n\}$. Note that the schedule can be constructed in $O(n \log n)$ time. This finishes the proof of the lemma. □

6.3. The $(1+\epsilon)$-approximation algorithm $C(\epsilon)$

From each job $J_i \in \mathcal{F}$ with $i > 1$ and a bipartition $\{\mathcal{C}, \overline{\mathcal{C}}\}$ of the job subset $\{J_{i+1}, J_{i+2}, \ldots, J_{n-1}\}$, we obtain a bipartition $\{\mathcal{F}^1, \mathcal{F}^2\}$ for $\mathcal{F}$ where $\mathcal{F}^1 = \{J_i\} \cup \mathcal{C}$ and $\mathcal{F}^2 = \{J_1, J_2, \ldots, J_{i-1}\} \cup \overline{\mathcal{C}}$. Note that J_i and J_1 are the largest job of $\mathcal{F}^1$ and $\mathcal{F}^2$, respectively. We may then use Lemma 6.4 to construct in $O(n \log n)$ time an optimal schedule π^* in which the machine order for J_n is forced to be $\langle M_3, M_2, M_1\rangle$, and $\mathcal{F}^1, \mathcal{F}^2 \subset \mathcal{F}$ are the subsets of jobs forced to be processed before and after J_n on the machine M_2, respectively. The makespan of π^* is $C_{\max}^* = \max\{p_i + P(\mathcal{F}^1), q_n\} + q_n + \max\{p_1 + P(\mathcal{F}^2), q_n\}$ (see for an illustration in Fig. 6.4). Apparently the schedule π^* relies on the bipartition $\{\mathcal{F}^1, \mathcal{F}^2\}$ of $\mathcal{F}$, and it eventually relies on the job J_i and the bipartition $\{\mathcal{C}, \overline{\mathcal{C}}\}$ of $\{J_{i+1}, J_{i+2}, \ldots, J_{n-1}\}$. We therefore re-denote this schedule as $\pi^i(\mathcal{C}, \overline{\mathcal{C}})$.

Let π^i denote the best schedule among all $\pi^i(\mathcal{C}, \overline{\mathcal{C}})$'s, over all possible bipartitions $\{\mathcal{C}, \overline{\mathcal{C}}\}$ of $\{J_{i+1}, J_{i+2}, \ldots, J_{n-1}\}$. Correspondingly, its makespan is denoted as $C_{\max}^i$.

When $2p_i \geq q_n$, we have $2p_1 \geq q_n$ too; consequently for any bipartition $\{\mathcal{C}, \overline{\mathcal{C}}\}$, the makespan of the schedule $\pi^i(\mathcal{C}, \overline{\mathcal{C}})$ is $p_i + P(\mathcal{F}^1) + q_n + p_1 + P(\mathcal{F}^2) = p_i + p_1 + q_n + P(\mathcal{F})$. We thus have proved the following lemma.

Lemma 6.5. *When $2p_i \geq q_n$, the best schedule π^i can be constructed in $O(n \log n)$ time and its makespan is $C_{\max}^i = p_i + p_1 + q_n + P(\mathcal{F})$.*

When $2p_i < q_n$, the best schedule π^i or the best bipartition $\{\mathcal{C}, \overline{\mathcal{C}}\}$ of $\{J_{i+1}, J_{i+2}, \ldots, J_{n-1}\}$ is hard to locate (see Theorem 5.1). Nonetheless, we are able to design a fully polynomial time algorithm $B(\epsilon)$ that constructs a feasible schedule $\pi^{i,\epsilon}$ with its makespan $C_{\max}^{i,\epsilon} \leq (1+\epsilon)C_{\max}^i$, for any $\epsilon > 0$.

The algorithm $B(\epsilon)$ first constructs an instance of the SUBSET-SUM problem, in which every job J_j gives a number p_j $(j = i+1, i+2, \ldots, n-1)$ and the bound is set to $q_n - 2p_i$. It then calls an $O(\min\{n/\epsilon, n + 1/\epsilon^2 \log(1/\epsilon)\})$-time $(1+\epsilon)$-approximation for the MIN-SUBSET-SUM problem to obtain a job subset $\mathcal{C}$; and calls an $O(\min\{n/\epsilon, n + 1/\epsilon^2 \log(1/\epsilon)\})$-time $(1-\epsilon)$-approximation for the MAX-SUBSET-SUM problem to obtain another job subset $\mathcal{D}$. In another $O(n \log n)$ time, the algorithm constructs two schedules $\pi^i(\mathcal{C}, \overline{\mathcal{C}})$ and $\pi^i(\mathcal{D}, \overline{\mathcal{D}})$, and returns the better one. A high-level description of the algorithm $B(\epsilon)$ is provided in Fig. 6.5.

Lemma 6.6. *When $2p_i < q_n$, the algorithm $B(\epsilon)$ constructs a feasible schedule $\pi^{i,\epsilon}$ in $O(\min\{n/\epsilon, n + 1/\epsilon^2 \log(1/\epsilon)\})$ time, with its makespan $C^{i,\epsilon}_{\max} \le (1+\epsilon)C^i_{\max}$, for any $\epsilon > 0$.*

Proof. The running time of the algorithm $B(\epsilon)$ is dominated by the two calls to the approximation algorithms for the Subset-Sum problem, which are in $O(\min\{n/\epsilon, n + 1/\epsilon^2 \log(1/\epsilon)\})$.

Let OPT^3 (OPT^4, respectively) denote the optimal solution to the Min-Subset-Sum (Max-Subset-Sum, respectively) problem and also abuse it to denote the sum of the numbers in the solution. We therefore have the following (in-)equalities inside the algorithm $B(\epsilon)$:

$$\text{OPT}^3 = \min\{P(\mathcal{X}) \mid \mathcal{X} \subseteq \{J_{i+1}, J_{i+2}, \ldots, J_{n-1}\},\ P(\mathcal{X}) > q_n - 2p_i\}; \tag{8}$$

$$q_n - 2p_i < P(\mathcal{C}) \ \le\ (1+\epsilon)\text{OPT}^3; \tag{9}$$

$$\text{OPT}^4 = \max\{P(\mathcal{Y}) \mid \mathcal{Y} \subseteq \{J_{i+1}, J_{i+2}, \ldots, J_{n-1}\},\ P(\mathcal{Y}) \le q_n - 2p_i\}; \tag{10}$$

$$q_n - 2p_i \ge P(\mathcal{D}) \ \ge\ (1-\epsilon)\text{OPT}^4. \tag{11}$$

Let $C^1_{\max}$ ($C^2_{\max}$, respectively) denote the makespan of the schedule $\pi^i(\mathcal{C}, \overline{\mathcal{C}})$ ($\pi^i(\mathcal{D}, \overline{\mathcal{D}})$, respectively). That is,

$$C^1_{\max} = 2p_i + P(\mathcal{C}) + q_n + \max\left\{p_1 + \sum_{j=1}^{i-1} p_j + P(\overline{\mathcal{C}}), q_n\right\}; \tag{12}$$

$$C^2_{\max} = 2q_n + \max\left\{p_1 + \sum_{j=1}^{i-1} p_j + P(\overline{\mathcal{D}}), q_n\right\}. \tag{13}$$

We distinguish three cases:

In the first case where $p_1 + \sum_{j=1}^{i-1} p_j + P(\overline{\mathcal{C}}) \ge q_n$, Eq. (12) becomes

$$C^1_{\max} = 2p_i + P(\mathcal{C}) + q_n + p_1 + \sum_{j=1}^{i-1} p_j + P(\overline{\mathcal{C}}) = p_i + p_1 + P(\mathcal{F}) + q_n \le C^i_{\max},$$

where the last inequality holds by Lemma 6.4, suggesting that the schedule $\pi^i(\mathcal{C}, \overline{\mathcal{C}})$ is optimal.

In the second case where $p_1 + \sum_{j=1}^{i-1} p_j + P(\overline{\mathcal{D}}) \le q_n$, Eq. (13) becomes

$$C^2_{\max} = 3q_n \le C^i_{\max},$$

where the last inequality holds by Lemma 6.4 again, suggesting that the schedule $\pi^i(\mathcal{D}, \overline{\mathcal{D}})$ is optimal.

In the last case, we consider $p_1 + \sum_{j=1}^{i-1} p_j + P(\overline{\mathcal{C}}) < q_n$ and $p_1 + \sum_{j=1}^{i-1} p_j + P(\overline{\mathcal{D}}) > q_n$. Assume in the best schedule π^i the bipartition of $\{J_{i+1}, J_{i+2}, \ldots, J_{n-1}\}$ is $\{\mathcal{C}^*, \overline{\mathcal{C}^*}\}$.

If $P(\mathcal{C}^*) > q_n - 2p_i$, *i.e.*, $\mathcal{C}^*$ is a feasible solution to the constructed Min-Subset-Sum instance, then we have

$$\text{OPT}^3 \le P(\mathcal{C}^*). \tag{14}$$

Using Eqs. (9, 14), Eq. (12) becomes

$$\begin{aligned} C^1_{\max} &= 2q_n + 2p_i + P(\mathcal{C}) \\ &\le 2q_n + 2p_i + (1+\epsilon)\text{OPT}^3 \\ &\le 2q_n + 2p_i + (1+\epsilon)P(\mathcal{C}^*) \\ &\le (1+\epsilon)(2q_n + 2p_i + P(\mathcal{C}^*)) \\ &\le (1+\epsilon)C^i_{\max}, \end{aligned} \tag{15}$$

where the last inequality holds by Lemma 6.4 again.

Otherwise we have $P(\mathcal{C}^*) \le q_n - 2p_i$, *i.e.*, $\mathcal{C}^*$ is a feasible solution to the constructed Max-Subset-Sum instance; we have

$$\text{OPT}^4 \ge P(\mathcal{C}^*). \tag{16}$$

Using Eqs. (11, 16), Eq. (13) becomes

$$\begin{aligned} C^2_{\max} &= 2q_n + p_1 + \textstyle\sum_{j=1}^{i-1} p_j + P(\overline{\mathcal{D}}) \\ &= 2q_n + p_1 + \textstyle\sum_{j=1}^{i-1} p_j + P(\overline{\mathcal{C}^*}) + P(\overline{\mathcal{D}}) - P(\overline{\mathcal{C}^*}) \\ &= 2q_n + p_1 + \textstyle\sum_{j=1}^{i-1} p_j + P(\overline{\mathcal{C}^*}) + P(\mathcal{C}^*) - P(\mathcal{D}) \\ &\le 2q_n + p_1 + \textstyle\sum_{j=1}^{i-1} p_j + P(\overline{\mathcal{C}^*}) + \epsilon P(\mathcal{C}^*) \\ &\le C^i_{\max} + \epsilon P(\mathcal{C}^*) \\ &\le (1+\epsilon)C^i_{\max}, \end{aligned} \tag{17}$$

ALGORITHM $C(\epsilon)$:

1. If $p_1 \geq q_n$, run the algorithm $A(\epsilon)$ and return the produced schedule.
2. If $P(\mathcal{F}) \leq q_n$, use Lemma 6.1 to construct an optimal schedule π^0 and return π^0.
3. Use Lemma 6.3 to construct two optimal schedules $\pi^{2,3,1}$ and $\pi^{3,1,2}$.
4. For each $i = 2, 3, \ldots, n-1$,
   4.1. if $2p_i > q_n$, use Lemma 6.5 to construct the schedule π^i;
   4.2. otherwise run the algorithm $B(\epsilon)$ to construction a schedule $\pi^{i,\epsilon}$.
   Let $\pi^{3,2,1}$ denote the best schedule among these $n-2$ schedules.
5. Return the best schedule among $\pi^{2,3,1}$, $\pi^{3,1,2}$ and $\pi^{3,2,1}$.

Fig. 6.6. A high-level description of the algorithm $C(\epsilon)$.

where the second last inequality holds by Lemma 6.4 and the last inequality holds due to the trivial fact that $P(\mathcal{C}^*) \leq C^i_{\max}$.

Therefore, in the last case, combining Eqs. (15, 17) we have

$$C^{i,\epsilon}_{\max} = \min\{C^1_{\max}, C^2_{\max}\} \leq (1+\epsilon)C^i_{\max}.$$

This finishes the proof of the lemma. □

The $(1+\epsilon)$-approximation algorithm $C(\epsilon)$ for the $M3 \mid prpt, (n-1,1) \mid C_{\max}$ problem takes advantage of the $(1+\epsilon)$-approximation algorithm $A(\epsilon)$ for the $M3 \mid prpt \mid C_{\max}$ problem when $p_1 \geq q_{\ell+1}$ (presented in Section 3, see Theorem 3.7), the optimal schedule constructed in Lemma 6.1 for the $M3 \mid prpt, (n-1,1), p_1 < q_n \mid C_{\max}$ problem when $P(\mathcal{F}) \leq q_n$, the two optimal schedules constructed in Lemma 6.3 for the $M3 \mid prpt, (n-1,1), p_1 < q_n \mid C_{\max}$ problem when the machine order for J_n is forced to be $\langle M_2, M_3, M_1\rangle$ or $\langle M_3, M_1, M_2\rangle$, respectively, denoted as $\pi^{2,3,1}$ and $\pi^{3,1,2}$, respectively, and the $n-2$ schedules π^i, $i = 2, 3, \ldots, n-1$, constructed either in Lemma 6.5 or by the algorithm $B(\epsilon)$ (see Lemma 6.6). A high-level description of $C(\epsilon)$ is depicted in Fig. 6.6.

Theorem 6.7. *The algorithm $C(\epsilon)$ is an $O(\min\{n^2/\epsilon, n^2 + n/\epsilon^2 \log(1/\epsilon)\})$-time $(1+\epsilon)$-approximation for the $M3 \mid prpt, (n-1,1) \mid C_{\max}$ problem.*

Proof. Recall from Theorem 3.7 that the running time of the algorithm $A(\epsilon)$ is in $O(\min\{n/\epsilon, n + 1/\epsilon^2 \log(1/\epsilon)\})$; Lemma 6.1 constructs the optimal schedule π^0 in $O(n)$ time; Lemma 6.3 constructs each of the two optimal schedules $\pi^{2,3,1}$ and $\pi^{3,1,2}$ in $O(n \log n)$ time; Lemma 6.5 constructs the schedule π^i also in $O(n \log n)$ time; and the running time of the algorithm $B(\epsilon)$ is in $O(\min\{n/\epsilon, n + 1/\epsilon^2 \log(1/\epsilon)\})$. Since $A(\epsilon)$ is called only once, while $B(\epsilon)$ could be called for $O(n)$ times, the overall time complexity of $C(\epsilon)$ is an order higher, that is, $O(\min\{n^2/\epsilon, n^2 + n/\epsilon^2 \log(1/\epsilon)\})$.

The worst-case performance ratio of $1+\epsilon$ is implied by Theorem 3.7 and Lemmas 6.1–6.6. In more details, if $p_1 \geq q_n$, then the ratio is guaranteed by Theorem 3.7; if $P(\mathcal{F}) \leq q_n$, then the optimality is guaranteed by Lemma 6.1; otherwise, Lemma 6.2 states that in the optimal schedule, the machine order for the unique open-shop job J_n is one of $\langle M_2, M_3, M_1\rangle$, $\langle M_3, M_1, M_2\rangle$, and $\langle M_3, M_2, M_1\rangle$: in the first two cases, the optimality is guaranteed by Lemma 6.3, while in the last case, the performance ratio is guaranteed by Lemmas 6.5 and 6.6 together. This finishes the proof of the theorem. □

7. Concluding remarks

In this paper, we studied the three-machine proportionate mixed shop problem $M3 \mid prpt \mid C_{\max}$. We presented first an FPTAS for the case where $p_1 \geq q_{\ell+1}$; and then proposed a 4/3-approximation algorithm for the other case where $p_1 < q_{\ell+1}$, for which we also showed that the performance ratio of 4/3 is asymptotically tight. The $F3 \mid prpt \mid C_{\max}$ problem is polynomial-time solvable; we showed an interesting hardness result that adding only one open-shop job to the job set makes the problem NP-hard if the open-shop job is larger than any flow-shop job. The special case in which there is only one open-shop job is denoted as $M3 \mid prpt, (n-1,1) \mid C_{\max}$. Lastly we proposed an FPTAS for this special case $M3 \mid prpt, (n-1,1) \mid C_{\max}$.

We believe that when $p_1 < q_{\ell+1}$, the $M3 \mid prpt \mid C_{\max}$ problem can be better approximated than 4/3, and an FPTAS is perhaps possible. Our last FPTAS for the special case $M3 \mid prpt, (n-1,1) \mid C_{\max}$ can be considered as the first successful step towards such an FPTAS.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

The authors would like to thank the anonymous reviewers for their many suggestions and comments that help improve the presentation.

LL was supported by the China Scholarship Council Grant 201706315073 and the Fundamental Research Funds for the Central Universities. YC and AZ were supported by the NSFC Grants 11771114 and 11571252; YC was also supported by the China Scholarship Council Grant 201508330054. JD was supported by the NSFC Grant 11501512 and the Zhejiang Provincial Natural Science Foundation Grant No. LY18A010029. RG, GL and YX were supported by the NSERC Canada; LL, YC, JD, GN and AZ were all partially supported by the NSERC Canada during their visits to Edmonton. GN was supported by the NSFC Grant 71501045, the Natural Science Foundation of Fujian Province Grant 2016J01332 and the Education Department of Fujian Province.

References

[1] P. Brucker, Scheduling Algorithms, Springer, New York, 2007.
[2] F.Y. Chin, L.L. Tsai, On j-maximal and j-minimal flow shop schedules, J. ACM 28 (1981) 462–476.
[3] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman and Company, San Francisco, 1979.
[4] R.L. Graham, E.L. Lawler, J.K. Lenstra, R. Kan, Optimization and approximation in deterministic sequencing and scheduling: a survey, Ann. Discrete Math. 5 (1979) 287–326.
[5] H. Kellerer, U. Pferschy, M.G. Speranza, An efficient approximation scheme for the subset-sum problem, in: Proceedings of the 8th International Symposium on Algorithms and Computation (ISAAC'97), in: LNCS, vol. 1350, 1997, pp. 394–403.
[6] C. Koulamas, G.J. Kyparisis, The three-machine proportionate open shop and mixed shop minimum makespan problems, Eur. J. Oper. Res. 243 (2015) 70–74.
[7] C.Y. Liu, R.L. Bulfin, Scheduling ordered open shops, Comput. Oper. Res. 14 (1987) 257–264.
[8] T. Masuda, H. Ishii, T. Nishida, The mixed shop scheduling problem, Discrete Appl. Math. 11 (1985) 175–186.
[9] P.S. Ow, Focused scheduling in proportionate flowshops, Manag. Sci. 31 (1985) 852–869.
[10] S.S. Panwalkar, M.L. Smith, C. Koulamas, Review of the ordered and proportionate flow shop scheduling research, Nav. Res. Logist. 60 (2013) 46–55.
[11] N.V. Shakhlevich, Y.N. Sotskov, Scheduling two jobs with fixed and nonfixed routes, Computing 52 (1994) 17–30.
[12] N.V. Shakhlevich, Y.N. Sotskov, F. Werner, Shop-scheduling problems with fixed and non-fixed machine orders of the jobs, Ann. Oper. Res. 92 (1999) 281–304.
[13] N.V. Shakhlevich, Y.N. Sotskov, F. Werner, Complexity of mixed shop scheduling problems: a survey, Eur. J. Oper. Res. 120 (2000) 343–351.
[14] V.A. Strusevich, Two-machine super-shop scheduling problem, J. Oper. Res. Soc. 42 (1991) 479–492.