Theoretical Computer Science 657 (2017) 64–72

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

An FPTAS for the parallel two-stage flowshop problem

Jianming Dong [a,1], Weitian Tong [b,c,1], Taibo Luo [d,b,1], Xueshi Wang [a], Jueliang Hu [a], Yinfeng Xu [d,e], Guohui Lin [a,b,*]

[a] Department of Mathematics, Zhejiang Sci-Tech University, Hangzhou, Zhejiang 310018, China
[b] Department of Computing Science, University of Alberta, Edmonton, Alberta T6G 2E8, Canada
[c] Department of Computer Sciences, Georgia Southern University, Statesboro, GA 30458, USA
[d] Business School, Sichuan University, Chengdu, Sichuan 610065, China
[e] State Key Lab for Manufacturing Systems Engineering, Xi'an, Shaanxi 710049, China

ARTICLE INFO

Article history:
Received 8 September 2015
Accepted 27 April 2016
Available online 8 June 2016

Keywords:
Two-stage flowshop scheduling
Multiprocessor scheduling
Makespan
Dynamic programming
Fully polynomial-time approximation scheme

ABSTRACT

We consider the NP-hard *m-parallel two-stage flowshop* problem, abbreviated as the $(m, 2)$-PFS problem, where we need to schedule n jobs to m parallel identical two-stage flowshops in order to minimize the makespan, *i.e.* the maximum completion time of all the jobs on the m flowshops. The $(m, 2)$-PFS problem can be decomposed into two subproblems: to assign the n jobs to the m parallel flowshops, and for each flowshop to schedule the jobs assigned to the flowshop. We first present a pseudo-polynomial time dynamic programming algorithm to solve the $(m, 2)$-PFS problem optimally, for any fixed m, based on an earlier idea for solving the $(2, 2)$-PFS problem. Using the dynamic programming algorithm as a subroutine, we design a fully polynomial-time approximation scheme (FPTAS) for the $(m, 2)$-PFS problem.

1. Introduction

In the *m-parallel k-stage flowshop* problem, denoted as (m, k)-PFS, there are m parallel identical k-stage flowshops $F_1, F_2, \ldots, F_m$. Each of these *classic* k-stage flowshop contains exactly one machine at every stage, or equivalently k sequential machines. An input job has k tasks, and it can be assigned to one and only one of the m flowshops for processing; once it is assigned to a flowshop, its k tasks are then processed on the k sequential machines in the flowshop, respectively. Let $M_{\ell,1}, M_{\ell,2}, \ldots, M_{\ell,k}$ denote the k sequential machines in the flowshop F_ℓ, for every ℓ. Let $\mathcal{J}$ denote a set of n input jobs $J_1, J_2, \ldots, J_n$. The job J_i is represented as a k-tuple $(p_{i,1}, p_{i,2}, \ldots, p_{i,k})$, where $p_{i,j}$ is the processing time for the j-th task, that is, the j-th task needs to be processed non-preemptively on the j-th machine in the flowshop to which the job J_i is assigned. For all i, j, $p_{i,j}$ is a non-negative integer. The objective of this problem is to minimize the makespan, that is the completion time of the last job.

Clearly, when $m = 1$, the problem reduces to the classic k-stage flowshop (*flowshop scheduling* in [4]); when $k = 1$, the problem reduces to another classic m-parallel identical machine scheduling problem (*multiprocessor scheduling* in [4]). When only the two-stage flowshops are involved, *i.e.* $(m, 2)$-PFS, the problem has been previously studied in [11,20,23].

* Corresponding author at: Department of Computing Science, University of Alberta, Edmonton, Alberta T6G 2E8, Canada.
E-mail address: guohui@ualberta.ca (G. Lin).

[1] Co-first authors.

http://dx.doi.org/10.1016/j.tcs.2016.04.046
0304-3975/

Table 1
Known results for the hybrid k-stage flowshop problem.

		m_j machines in stage j		
		$m_j = 1$	m_j fixed	m_j arbitrary
k stages	$k = 1$	Polynomial time	FPTAS [18]	PTAS [12]
	$k = 2$	Polynomial time [15]	PTAS [10]	PTAS [19]
	$k \geq 3$ fixed	PTAS [10]	PTAS [10]	PTAS [14]
	k arbitrary	Not be approximated within 1.25 [22]		

The (m, k)-PFS problem is closely related to the well studied *hybrid k-stage flowshop* problem [16], which also includes the classic k-stage flowshop and the classic parallel identical machine scheduling problem as special cases. The *hybrid k-stage flowshop* problem is a flexible manufacturing system model, and it contains $m_j \geq 1$ parallel identical machines in the j-th stage, $j = 1, 2, \ldots, k$. The problem is abbreviated as $(m_1, m_2, \ldots, m_k)$-HFS. A job J_i is again represented as a k-tuple $(p_{i,1}, p_{i,2}, \ldots, p_{i,k})$, where $p_{i,j}$ is the processing time for the j-th task, that can be processed non-preemptively on any one of the m_j machines in the j-th stage. The objective of the $(m_1, m_2, \ldots, m_k)$-HFS problem is also to minimize the makespan. One clearly sees that when $m_1 = m_2 = \ldots = m_k = 1$, the problem reduces to the classic k-stage flowshop problem; when $k = 1$, the problem reduces to the classic m-parallel identical machine scheduling problem.

We next review some of the most important and relevant results on the k-stage flowshop problem and on the m-parallel identical machine scheduling problem. For the k-stage flowshop problem, it is known that for $k \in \{2, 3\}$, there exists an optimal schedule that is a *permutation schedule* for which all the k machines process the jobs in the same order; but for $k \geq 4$, there may exist no optimal schedule which is a permutation schedule [3]. When $k = 2$, the two-stage flowshop problem is polynomial time solvable, by Johnson's algorithm [15]; the k-stage flowshop problem becomes *strongly* NP-hard when $k \geq 3$ [5]. After several efforts [15,5,6,2], Hall presented a polynomial-time approximation scheme (PTAS) for the k-stage flowshop problem, for any fixed integer $k \geq 3$ [10]. Note that due to the strongly NP-hardness, a PTAS is the best possible result unless P = NP. When k is a part of input (*i.e.* an arbitrary integer), the problem cannot be approximated within 1.25 [22]; nevertheless, it remains unknown whether the problem can be approximated within a constant factor.

For the m-parallel identical machine scheduling problem, it is NP-hard when $m \geq 2$ [4]. When m is a fixed integer, the problem admits a pseudo-polynomial time exact algorithm [4] that can be used to construct an FPTAS [18]; when m is a part of input, the problem becomes strongly NP-hard, but admits a PTAS by Hochbaum and Shmoys [12].

The literature on the hybrid k-stage flowshop problem $(m_1, m_2, \ldots, m_k)$-HFS is also rich [17], especially for the hybrid two-stage flowshop problem (m_1, m_2)-HFS. First, $(1, 1)$-HFS is the classic two-stage flowshop problem which can be solved optimally in polynomial time [15]. When $\max\{m_1, m_2\} \geq 2$, the (m_1, m_2)-HFS problem becomes strongly NP-hard [13]. The special cases $(m_1, 1)$-HFS and $(1, m_2)$-HFS have attracted many researchers' attention [7,9,1,8]; the interested reader might refer to [21] for a survey on the hybrid two-stage flowshop problem with a single machine in one stage.

For the general hybrid k-stage flowshop problem $(m_1, m_2, \ldots, m_k)$-HFS, when all the m_1, m_2, ..., m_k are fixed integers, Hall claimed that the PTAS for the classic k-stage flowshop problem can be extended to a PTAS for the $(m_1, m_2, \ldots, m_k)$-HFS problem [10]. Later, Schuurman and Woeginger presented a PTAS for the hybrid two-stage flowshop problem (m_1, m_2)-HFS, even when the numbers of machines m_1 and m_2 in the two stages are a part of input [19]. Jansen and Sviridenko generalized this result to the hybrid k-stage flowshop problem $(m_1, m_2, \ldots, m_k)$-HFS, where k is a fixed integer while $m_1, m_2, \ldots, m_k$ can be a part of input [14]. Due to the inapproximability of the classic k-stage flowshop problem, when k is arbitrary, the $(m_1, m_2, \ldots, m_k)$-HFS problem can not be approximated within 1.25 either unless P = NP [22]. Table 1 summarizes the results we reviewed earlier. Besides, there are plenty of heuristic algorithms in the literature for the general hybrid k-stage flowshop problem, and the interested readers can refer to the survey by Ruiz et al. [17].

Compared to the rich literature on the hybrid k-stage flowshop problem, the (m, k)-PFS problem is much less studied. In fact, the general (m, k)-PFS problem is almost untouched, except only the two-stage flowshops are involved [11,20,23]. He et al. first proposed the m parallel identical two-stage flowshop problem $(m, 2)$-PFS, motivated by an application from the glass industry [11]. In their work, the $(m, 2)$-PFS problem is formulated as a mixed-integer programming and an efficient heuristics is proposed [11]. Vairaktarakis and Elhafsi [20] also studied the $(m, 2)$-PFS problem, in order to investigate the hybrid k-stage flowshop problem. Among other results, Vairaktarakis and Elhafsi observed that the $(2, 2)$-PFS problem can be broken down into two subproblems, a job partition problem and a classic two-stage flowshop problem [20]. Note that the second subproblem can be solved optimally by Johnson's algorithm [15]. The NP-hardness of the first subproblem [4] implies the NP-hardness of $(2, 2)$-PFS, simply by setting all $p_{i,2}$'s to zeros. One of the major contributions in [20] is an $O(nP^3)$-time dynamic programming algorithm for solving the $(2, 2)$-PFS problem optimally, where n is the number of jobs and P is the sum of all processing times. That is, this exact algorithm runs in pseudo-polynomial time.

The NP-hardness of $(2, 2)$-PFS implies that the general $(m, 2)$-PFS problem is NP-hard, either m is a part of input (arbitrary) or m is a fixed integer greater than one. Recently, Zhang et al. [23] studied the $(m, 2)$-PFS problem from the approximation algorithm perspective, more precisely only for the special case where $m = 2$ or 3. They designed a 3/2-approximation algorithm for the $(2, 2)$-PFS problem and a 12/7-approximation algorithm for the $(3, 2)$-PFS problem [23]. Both algorithms are variations of Johnson's algorithm and the main idea is first to sort all the jobs using Johnson's algorithm into a sequence, then to cut this sequence into two (three, respectively) parts for the two (three, respectively) two-stage flowshops