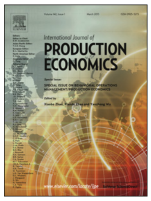
Int. J. Production Economics 167 (2015) 88–96

Contents lists available at ScienceDirect

Int. J. Production Economics

journal homepage: www.elsevier.com/locate/ijpe



Scheduling of jobs with cross families in two stage manufacturing systems

Nooshin Nekoiemehr [a,*], Esaignani Selvarajah [b,1], Guoqing Zhang [a,2]

[a] Industrial and Manufacturing System Department, University of Windsor, 401 Sunset Avenue, Windsor, Ontario, Canada N9B 3P4
[b] Odette School of Business, University of Windsor, 401 Sunset Avenue, Windsor, Ontario, Canada N9B 3P4

ARTICLE INFO

Article history:
Received 22 March 2014
Accepted 11 May 2015
Available online 20 May 2015

Keywords:
Batching
Scheduling
Flow shop
Makespan
Cross family
Production

ABSTRACT

This paper studies scheduling of n jobs in a two stage manufacturing system to minimize makespan when each stage has its own job families and families in each stage require sequence independent setups. In a manufacturing system that each stage has its own family; jobs belonging to different families in one stage may belong to the same family in another stage, and we call it cross families. To the best of our knowledge, there is no study on cross families in scheduling literature. Our problem is NP-hard for arbitrary number of families, and therefore we study the problem with fixed number of families. We first analyze some properties of the optimal schedule and show that Johnson sequence is optimal for jobs belonging to the same family on both machines. We develop an efficient branch and bound algorithm with complexity of $O(n^c)$, where c is a constant and a hybrid genetic algorithm for large scale problems, using the properties of the optimal schedule. Finally, we present computational experiment to demonstrate the effectiveness of our algorithms.

1. Introduction

In this paper, we study scheduling of jobs in a two stage production system where jobs have family sequence-independent setup times to minimize makespan. We assume that job allocations to families are machine based and therefore there are two family classes: family class for stage 1 and family class for stage 2. Thus each job will have two different family memberships and we call this problem jobs with *cross families*, i.e., two jobs belonging to two different families in one stage may belong to the same family in another stage. This system can be represented by a two machine permutation flow shop, where each stage is represented by a machine with cross family setup. Since the problem is NP-hard for arbitrary number of families, we study the problem when there is fixed number of family in each stage.

Past studies on scheduling with family setups have assumed that jobs belonging to the same family in one stage belong to the same family in other stages. However, this assumption may not be applicable for all production cases. For example, consider an automobile manufacturing system where all jobs are processed first in the body shop and then in the paint shop. In the body shop, jobs are categorized into two-door or four-door families and the same jobs are re-categorized in the paint shop based on paint requirements. For instance, a two-door job and a four-door job belong to the same family in the paint shop if they both require the same color. To the best of our knowledge in scheduling literature, there is no study on jobs with cross families.

The permutation flow shop scheduling problem with makespan minimization has been well studied due to its important applications in manufacturing systems, assembly lines and information service facilities. Two machine flow shop scheduling problem was first studied by Johnson (1954). Yoshida and Hitomi (1979) studied the problem when setups are required before processing jobs. Special cases of flow shop scheduling problems are studied by Nouweland et al. (1992), Wlodzimierz (1977), Johnny et al. (1992), Chuanli and Hengyong (2012) and Lin-hui Sun et al. (2012). When there are family setups, jobs of the same family are grouped into batches in order to minimize resource required for setups. There are several studies considering batching in flow shops with different assumptions, such as Logendran et al. (2006), Hendizadeha et al. (2008), Voxa and Wittb (2007) and Bozorgirad and Logendran (2013). Readers may refer to studies of Allahverdi et al. (2008) and Edwin et al. (2000) for a complete survey on batching. The problem addressed in this paper is closely related to studies on two machine flow shop scheduling with family-sequence-independent setups to minimize makespan. There have been many studies on this problem over the past four decades.

* Corresponding author. Tel.: +1 226 246 7666.
E-mail addresses: nekoiem@uwindsor.ca (N. Nekoiemehr), selvare@uwindsor.ca (E. Selvarajah), gzhang@uwindsor.ca (G. Zhang).
[1] Tel.: +1 519 253 3000x3105.
[2] Tel.: +1 519 253 3000x2637.

http://dx.doi.org/10.1016/j.ijpe.2015.05.022
0925-5273/© 2015 Elsevier B.V. All rights reserved.

Scheduling problems with family setups can be classified into two main classes: (i) scheduling with group technology (GT) assumption, and (ii) scheduling without GT assumption.

In group technology, jobs of the same family are scheduled into single batch and thus it simplifies the problem. Therefore, for problems with fixed number of families, researchers were able to develop polynomially bounded algorithms with GT assumption. Sekiguchi (1983) proved the optimality of Johnson job sequence within each batch and used composite job approach to schedule families under series–parallel precedence constraint. Ham et al. (1985) presented a two-step procedure to find an optimal solution for this problem and Baker (1990) developed a polynomially bounded algorithm based on the results obtained by Ham et al. (1985). Logendran and Sriskandarajah (1993), Marco (2004), Lee and Mirchandani (1988), and Cheng and Wang (1998, 1999) studied two machine flow shop problems with setups under special conditions such as zero-buffer, limited-buffer, identical versatile machines, one-setup problem, discrete or batch processor machines, and provided different solution techniques. Yang and Chern (2000) considered job removal time and transportation time between machines and proposed polynomial-time algorithm.

Two machine scheduling problem to minimize makespan without GT assumption is more difficult than the problem with GT assumptions. Therefore, researchers developed heuristic algorithms, approximation algorithms, and for some special cases polynomial time algorithms. Kleinau (1993) showed that this problem is NP-hard for both anticipatory (setup can start on the second machine before the processing of the job on the first machine is finished) and non-anticipatory (setup cannot start on the second machine until the processing of the job is finished on the first machine) setups when there are arbitrary number of families. Zdrzalka (1995) developed heuristic algorithms and investigated their worst-case performances. Danneberg et al. (1998) compared several heuristic algorithms for the problem with limited buffer between machines. Lin and Cheng (2001) studied the problem with no-wait and batch availability assumptions, and proved that the problem is strongly NP-hard. They proposed an optimal batch size formulation when jobs have identical processing time. Chen et al. (1998) proposed two heuristic algorithms with $O(n\log n)$ time to solve the problem with arbitrary number of job families. The first algorithm, with GT assumption and applying Johnson's algorithm and the second one with relaxing GT assumption and applying open shop scheduling technique in order to improve the worst case ratio. Cheng et al. (2000) proved that the problem with batch availability assumption is strongly NP-hard, and presented a heuristic algorithm while investigating some special cases. Agnieszka and Rudek (2013) developed meta-heuristic algorithms using tabu search and simulated annealing when processing times follow aging effect function.

In this paper, we study scheduling of jobs with cross families and sequence independent setups in two machine flow shop to minimize makespan when there is fixed number of families. We first investigate some properties of the optimal schedule and show that Johnson sequence is optimal for jobs belonging to the same family on both machines. We develop an efficient branch and bound algorithm with complexity of $O(n^c)$, where c is the total number of families and is a constant, and a hybrid genetic algorithm for large scale problems using the properties of the optimal schedule. The paper is organized as follows. Section 2 defines the problem with notations and assumptions. Section 3 discusses some preliminaries for our problem and analyzes properties of optimal schedule. Section 4 presents a branch and bound algorithm to sequence given batches optimally. Section 5 discusses on optimal scheduling, and Section 6 presents a hybrid genetic algorithm for large scale problems. Finally computational experiment is provided in Section 7 and conclusion and future research are given in Section 8.

2. Problem definition

We are given a set of n jobs $\{J_1, J_2, \ldots, J_n\}$ that has to be scheduled in a two machine flow shop to minimize the makespan ($C_{\max}$). There are two family types, families on machine $1(M_1)$ and families on machine $2(M_2)$. Each job has two family memberships, its family on M_1 and its family on M_2. An anticipatory sequence independent setup is required on each machine when switching from one family to another. Job J_j requires a processing time of $p_{j,l}$ and a setup time of $s_{j,l}$ on M_l. When any group of jobs having the same family on both machines are scheduled consecutively, no setup is required on either machine except for the first job in that group. We call such group a batch, and without loss of generality, we interchangeably use $s_{r,l}$ to denote the setup time of J_r on M_l or the setup time of β_r (the rth batch) on M_l.

There is a fixed number of families on each machine; K families on M_1 and L families on M_2. All jobs are available at time zero, and processing of a job on the second machine can be started immediately after the completion of that job on the first machine. Jobs follow the same processing order on both machines and a machine can process at most one job at a time. Processing of a job cannot be interrupted and there is an unlimited buffer capacity between machines. We describe this scheduling problem using the three-field notation of Graham et al. (1979) as $F2/ST, SI, CB/C_{\max}$, where $F2$ stands for a two machine flow shop, ST for setup time, $C_{\max}$ for makespan, SI and CB stands for sequence independent setups and cross families respectively.

We use the following additional notations:

- $\tau(f,g)$: The set of all jobs belonging to the fth family on M_1 and gth family on M_2, i.e., jobs having the same family on both machines.
- $I_j(\varphi)$: The idle time on M_2 immediately before starting processing (after setup if required) of job J_j in any given sequence φ.
- $A_j(\varphi)$: The total idle time on M_2 before processing job J_j, in any given sequence φ.
- $T_{j,l}(\varphi)$: The start time of job J_j (or batch β_j) on M_l after any setup, if required, in any given sequence φ.
- $$a_{i,j} = a_{j,i} = \begin{cases} 0 & \text{if jobs } J_i \text{ and } J_j \text{ (or batches } \beta_i \text{ and } \beta_j) \\ & \text{belong to the same family on machine } M_1 \\ 1 & \text{otherwise} \end{cases}$$
- $$b_{i,j} = b_{j,i} = \begin{cases} 0 & \text{if jobs } J_i \text{ and } J_j \text{ (or batches } \beta_i \text{ and } \beta_j) \\ & \text{belong to the same family on machine } M_2 \\ 1 & \text{otherwise} \end{cases}$$
- $U = [a_{i,j}]_{n\times n}$
- $V = [b_{i,j}]_{n\times n}$

Note that there will be at most $K\times L$, $\tau(f,g)$ sets in this problem. For illustration consider an example with job set $\{1,2,\ldots,7\}$ and $K=L=2$. Let job set in families 1 and 2 on M_1 be $\{1,2,3,4\}$ and $\{5,6,7\}$ respectively, and job set in families 1 and 2 on M_2 be $\{2,4,7\}$ and $\{1,3,5,6\}$. Then, $\tau(1,1)=\{2,4\}$, $\tau(1,2)=\{1,3\}$, $\tau(2,1)=\{7\}$, and $\tau(2,2)=\{5,6\}$. For jobs 1 and 2, $a_{1,2}=a_{2,1}=0$, $b_{1,2}=b_{2,1}=1$ because jobs 1 and 2 belong to the same family on machine M_1 and to different families on machine M_2.

Let σ represent job sequence in the order of job index, i.e., $\sigma = \{J_1, J_2, \ldots, J_{n-1}, J_n\}$. We define $K_r(\sigma)$ as