Computers & Industrial Engineering 75 (2014) 116–120

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

Algorithms for scheduling incompatible job families on single batching machine with limited capacity ☆

Bayi Cheng *, Junfeng Cai, Shanlin Yang, Xiaoxuan Hu

School of Management, Hefei University of Technology, Hefei 230009, PR China
Key Laboratory of Process Optimization and Intelligent Decision-making, Ministry of Education, Hefei University of Technology, Hefei 230009, PR China

ARTICLE INFO

Article history:
Received 3 May 2012
Received in revised form 21 February 2013
Accepted 20 June 2014
Available online 30 June 2014

Keywords:
Scheduling
Single batching machine
Incompatible job families
Arbitrary job sizes
Approximation algorithm

ABSTRACT

Motivated by applications in food processing and semiconductor manufacturing industries, we consider the scheduling problem of a batching machine with jobs of multiple families. The machine has a limited capacity to accommodate jobs. The jobs are in arbitrary sizes and multiple families. Jobs from different families cannot be processed in a batch. We show the problems of minimizing makespan and total batch completion time are both NP-hard in the strong sense. We present a mixed integer programming model for the problems. Then we propose two polynomial time heuristics based on longest processing time first rule and first fit rule. For the special case where a larger job also has a longer processing time, the heuristic for minimizing makespan is optimal. For the general case, we show the performance guarantee of the methods for the two objectives respectively.

1. Introduction

The scheduling problem of batching machines has many applications in the real industries such as semiconductor manufacturing, food processing and metal working. In a typical scheduling of batching machines, the jobs are processed in batches and the machines have limited capacity to accommodate jobs. In contrast to the classical scheduling, this type of scheduling is more complex for the jobs should first be assigned into batches and then scheduled on machines.

The early research concentrated on the problems where the batching machines are supposed to process a fixed number of jobs. Lee, Uzsoy, and Martin-Vega (1992) presented the first paper on the topic. They showed that the problem of minimizing makespan is NP-hard in the strong sense. Then the scheduling of batching machines with agreeable release times and due dates was proposed from the semiconductor manufacturing systems (Li & Lee, 1997). Ji and Cheng (2010) and Li, Ng, Cheng, and Yuan (2011) considered the batching machine with deteriorating jobs and release dates and provide polynomial time algorithms. The reviews of the related research on batching machines were made in Potts and Kovalyov (2000) and Méndez, Cerdá, Grossmann, Harjunkoski, and Fahl (2006).

The recent literature has been focusing on the scheduling problem of batching machines with jobs of arbitrary sizes, which is an extension of the above scheduling. In this type of scheduling, the machines have a given capacity and the jobs have arbitrary sizes which are determined by the customers. Uzsoy (1994) introduced the single-machine problem and proposed several heuristics for minimizing makespan and total completion time. The performances of the heuristics were then analyzed by Zhang, Cai, Lee, and Wang (2001). Other heuristics has been seen in Li, Li, Wang, and Liu (2005) and Kashan, Karimi, and Ghomi (2009). Besides, Dupont and Flipo (2002) provided a branch and bound method for the problem. On the other hand, meta-heuristics has also been introduced to solve the scheduling problem. Sevaux and Peres (2003) applied genetic algorithm to minimize the weighted number of late jobs. Then the performance of genetic algorithm was improved by modifying the coding and decoding methods in Koh, Koo, Kim, and Hur (2005), Damodaran, Manjeshwar, and Srihari (2006) and Kashan, Karimi, and Jenabi (2008). Other meta-heuristics are also applied such as simulated annealing (Melouk, Demodaran, & Chang, 2004), branch and price algorithm (Parsa, Karimi, & Kashan, 2010) and ant colony optimization (Cheng, Li, & Chen, 2010).

In practice the scheduling on a single batching machine where the jobs are divided into multiple incompatible families are often encountered. A job family is a subset of jobs which can be processed together in a batch, i.e., jobs from different families cannot be assigned together in a batch for they have different requirements on processing conditions. In practice the customers have

☆ This manuscript was processed by Area Editor T.C. Edwin Cheng.
* Corresponding author at: School of Management, Hefei University of Technology, Hefei 230009, PR China. Tel./fax: +86 55162919325.

http://dx.doi.org/10.1016/j.cie.2014.06.014
0360-8352/© 2014 Elsevier Ltd. All rights reserved.

various demands on the product and this is a challenge for companies to optimize the production. The methods for scheduling job families have been an interesting direction for researchers. Exact algorithms for the problem were studied in Yuan, Shang, and Feng (2005), Fu, Tian, and Yuan (2009) and Liu, Ng and Cheng (2010). Meta-heuristics were studied in Malve and Uzsou (2007), Chiang, Cheng, and Fu (2010) and Tian, Cheng, Ng, and Yuan (2011). However, no algorithm with a performance guarantee has been proposed for the single batching machine with jobs of arbitrary sizes and incompatible families. In this paper, we provide a polynomial time algorithm with a performance guarantee for the problem. The remainder of the paper is organized as follows. In Section 2, we present a mathematical model for the problem under investigation and discuss the computational complexity. In Section 3, we present optimality properties and provide our heuristic algorithms. In Section 4, we analyze the time complexity of the heuristics and their performance guarantees for minimizing makespan and total batch completion time. Finally in Section 5, we conclude this paper and give directions for future work.

2. Mathematical model and computational complexity

The problem under investigation can be described as follows. There are n jobs to be processed on a single machine and the job set is $J = \{1, \ldots, n\}$. The jobs belong to m families and the set of families is $F = \{F_1, \ldots, F_m\}$ where $|F_i| = n_i$. Job j has a size s_j and a processing time t_j, both of which are determined by the customers. The jobs are processed in batches given that the total size of jobs in each batch cannot exceed the machine capacity C. P represents the problem of minimizing makespan and π represents a feasible solution. Q represents the problem of minimizing total batch completion time and μ represents a feasible solution. Then π is a set of batches $\{b_{ig}: i = 1, \ldots, m; g = 1, \ldots, G_i\}$, where G_i is the number of batches containing jobs form F_i. The processing time of b_{ig} is T_{ig} and $G = \sum_{i=1}^{m} G_i$. Since at most one batch can be processed on the machines simultaneously, in a feasible solution the batches can be ordered as B_g ($g = 1, \ldots, G$) according to the processing. So μ is a permutation of batches $B_1, \ldots, B_G$.

We set 0–1 variables as decision parameters. The variables are defined as:

$$\alpha_{ig} = \begin{cases} 1 & \text{the g}-\text{th batch is created for} F_i \\ 0 & \text{otherwise} \end{cases}; \quad \beta_{jig} = \begin{cases} 1 & \text{if } j \in b_{ig} \\ 0 & \text{otherwise} \end{cases}.$$

Then the model of minimizing makespan for P is as following.

$$\text{Minimize } C_{\max} = \sum_{i=1}^{m} \sum_{g=1}^{G_i} T_{ig} \tag{1}$$

Subject to

$$\beta_{jig} \leqslant \alpha_{ig} \qquad i = 1, \ldots, m; \; j = 1, \ldots, n \tag{2}$$

$$n_i = \sum_{j=1}^{n} \beta_{jig} \qquad i = 1, \ldots, m \tag{3}$$

$$\sum_{g=1}^{G_i} \beta_{jig} = 1 \qquad j = 1, \ldots, n \tag{4}$$

$$\sum_{i=1}^{m} \sum_{g=1}^{G_i} \beta_{jig} = 1 \qquad j = 1, \ldots, n \tag{5}$$

$$\sum_{j=1}^{n} \beta_{jig} s_j \leqslant C \qquad i = 1, \ldots, m \tag{6}$$

$$T_{ig} = \max\{t_j : \beta_{jig} = 1\} \qquad i = 1, \ldots, m \tag{7}$$

$$\alpha_{ig}, \beta_{jig} \in \{0, 1\} \qquad \forall i, j, g. \tag{8}$$

Objective (1) implies that the makespan is the sum of the processing times of all the batches. On the other hand, the objective function of total batch completion time is:

$$\sum C_i = \sum_{g=1}^{G} \sum_{l=1}^{g} T_l \tag{9}$$

Constraint (2) ensures that a job can be assigned in a batch only when the batch is created. Eq. (3) implies the value of β_{jig} determines the number of batches created for family F_i. Eqs. (4) and (5) ensures that a job should be assigned in only one batch and the other jobs in this batch should belong to the same family. Constraint (6) ensures that the total size of job in each batch should not exceed the machine capacity. Eq. (7) implies that the processing of a batch is non-preemptive and constraint (8) is the definition of the 0–1 variables. Clearly the problem is to decide the value of α_{ig} and β_{jig}.

Theorem 1. *P and Q are both NP-hard in the strong sense.*

Proof. First we analyze the computational complexity of P. Let P_1 represent a special case of P where $m = 1$ and t_i ($i = 1, \ldots, n$) is a constant t_0. Then the processing time of any batch is a constant and any two jobs can be combined together in the same batch. By contrast, consider the famous bin-packing problem where the item list is $\{1, \ldots, n\}$ and the size of item j is s_j/C and the objective is to assign the items into the minimum number of bins. Denote the bin-packing problem as P_2. We show there is a solution for P_1 if and only if there is a solution for P_2.

If π is an optimal solution for P_1, i.e.,

$$C_{\max} = \sum_{i=1}^{m} \sum_{g=1}^{G_i} T_{ig} = \sum_{i=1}^{m} G_i t_0 = G t_0.$$

This implies that G is minimized since t_0 is a constant. Now in P_1, assign all the items using the same method as π, and we see G is also the number of bins. On the other hand, if G is the feasible number of bins in P_2, similarly we get that the method of assigning the items is also the optimal schedule for P_1. Therefore, P_1 is equivalent to P_2. Since P_1 is a special case of P and the bin-packing problem is *NP-hard* in the strong sense, P is NP-hard in the strong sense.

Q is to minimize $\sum C_i = \sum_{g=1}^{G} \sum_{l=1}^{g} T_l$ and here we also consider a special case Q_1 where $m = 1$ and $t_i = t_0$ ($i = 1, \ldots, n$). Note that in this case,

$$\sum C_i = \sum_{g=1}^{G} \sum_{l=1}^{g} T_l = \sum_{g=1}^{G} g t_0 = \frac{G(G-1)}{2} t_0.$$

Q_1 is just to find an optimal G, which is shown to be NP-hard in the ordinary sense as above. Therefore, Q_1 is NP-hard in the strong sense and consequently Q is NP-hard in the strong sense. This completes the proof of Theorem 1. □

3. Proposed heuristics

3.1. Implementation of heuristics

First we discuss the properties of optimal solutions π^* and μ^*. For the sake of simplicity, in the following content we use U and V to denote $C_{\max}$ and $\sum C_i$.

Theorem 2. *In π^* and μ^*, if B_x^* and B_y^* are two batches assigned with jobs from the same family, then $\sum_{j \in B_x^*} s_j + \sum_{j \in B_y^*} s_j > C$.*

Proof (*By contradiction*). Suppose in an optimal solution π^*, there are two batches B_x^* and B_y^* such that $\sum_{j \in B_x^*} s_j + \sum_{j \in B_y^*} s_j \leqslant C$. Then the