Computers & Operations Research 40 (2013) 1224–1233

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

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

Scheduling jobs on a single batch processing machine with incompatible job families and weighted number of tardy jobs objective

Stéphane Dauzère-Pérès [a], Lars Mönch [b,*]

[a] Department of Manufacturing Sciences and Logistics, Centre Microélectronique de Provence – Site Georges Charpak, Ecole des Mines de Saint-Etienne, F-13541 Gardanne, France

[b] Department of Mathematics and Computer Science, University of Hagen, D-58097 Hagen, Germany

ARTICLE INFO

Available online 31 December 2012

Keywords:
Scheduling
Batching
Mixed integer linear programming
Genetic algorithms

ABSTRACT

In this paper, we minimize the weighted and unweighted number of tardy jobs on a single batch processing machine with incompatible job families. We propose two different mixed integer linear programming (MILP) formulations based on positional variables. The second formulation does not contain a big-*M* coefficient. Two iterative schemes are discussed that are able to provide tighter linear programming bounds by reducing the number of positional variables. Furthermore, we also suggest a random key genetic algorithm (RKGA) to solve this scheduling problem. Results of computational experiments are shown. The second MILP formulation is more efficient with respect to lower bounds, while the first formulation provides better upper bounds. The iterative scheme is effective for the weighted case. The RKGA is able to find high-quality solutions in a reasonable amount of time.

1. Introduction

This research is motivated by a scheduling problem found in the diffusion area in semiconductor wafer fabrication facilities (wafer fabs). In the diffusion area, there are diffusion furnaces which can process several jobs simultaneously. Due to the different chemical nature of the processes, only jobs of the same job family can be processed together. A batch is defined as a group of jobs that can be processed at the same time on the same machine [5]. The solution of a single-machine batch scheduling problem requires that batches are formed and sequenced. Carefully choosing which jobs to batch together and in which order these batches should be sequenced is of high importance as the processing times in the diffusion furnaces can take up to 10 h, versus 1 h in other wafer fab processes. Therefore, a proper scheduling of the machines dedicated to the diffusion process steps has a great impact on the performance of the entire wafer fab (cf. [16–18]).

In this paper, the performance measures are the number of tardy jobs $\sum_j U_j$ and the weighted number of tardy jobs $\sum_j w_j U_j$, where w_j is the weight of job j and U_j is equal to 1 when the job is tardy, and 0 otherwise. These measures can be seen as indicators for the quality of a solution or as major or minor objectives in multi-objective optimization (cf. [14,2] for bi-criteria scheduling problems including the number of tardy jobs objective). It is interesting to note that the corresponding unweighted scheduling problem for a single non-batching machine can be solved in polynomial time by Moore's algorithm [19]. However, the corresponding single-machine batch scheduling problem is NP-hard even for the unweighted cases [15]. We start by looking at a single machine scheduling problem for the sake of simplicity, but we believe that an extension of this research to the case of unrelated parallel machines with dedications which is more real-world like is possible. However, carrying out all the details is part of future research.

In this paper, we propose two different MILP formulations. It turns out that the first formulation is outperformed by the second one with respect to lower bounds because it includes a big-*M* coefficient and the second formulation does not. We propose two iterative schemes that reduce the number of positional decision variables. We develop also a RKGA as the dynamic programming (DP) formulation suggested by Jolai [13] cannot be used when there is more than one family because of computing time and memory requirements. It turns out that the proposed RKGA provides solutions with a similar quality as the MILP formulations in less time for both the unweighted and weighted cases.

The paper is organized as follows. In Section 2, we describe the problem and discuss related research. Then we introduce the two different MILP models in Section 3, and describe two iterative schemes to improve the resolution of the models with a standard

* Corresponding author.
E-mail addresses: Dauzere-Peres@emse.fr (S. Dauzère-Pérès), Lars.Moench@fernuni-hagen.de (L. Mönch).

0305-0548/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cor.2012.12.012

solver. In Section 4, we propose the RKGA. Finally, the design of experiments and the computational results are presented in Section 5.

2. Problem description and discussion of related work

2.1. Problem setting and analysis

We consider n jobs waiting to be scheduled on a single machine. The important process restriction is the batching of the jobs with f incompatible families. The performance measure is the weighted number of tardy jobs, defined as $\sum w_j U_j$, where w_j denotes the weight of job j and

$$U_j = \begin{cases} 1 & \text{if } C_j - d_j > 0 \text{ for job } j, \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

Here, C_j and d_j denote the completion time and the due date of job j, respectively. The assumptions of this research are as follows:

1. Jobs of the same family s, $s=1,\ldots,f$, have the same processing time p_s. Only jobs of the same family can be batched together.
2. All jobs are available at time zero.
3. Once a batch processing machine is started, it cannot be interrupted. No preemption of the processing of a batch is allowed.

The relation $\sum_{s=1}^{f} n_s = n$ holds, where n_s denotes the number of jobs of family s. Let $\mathcal{J}(s)$ denote the set of jobs in family s, and $s(j)$ the family of j. B is the capacity of the batch processing machine, i.e. at most B jobs can be processed simultaneously. We call B the maximum batch size. The researched problem can be written as

$$1|p\text{-}batch, incompatible|\sum w_j U_j \quad (2)$$

using the $\alpha|\beta|\gamma$ notation from the scheduling literature. Note that, in the unweighted case, we consider the criterion $\sum U_j$, i.e. $w_j \equiv 1$, $j=1,\ldots,n$.

Let us consider the following example. There are four jobs {1,2,3,4} in family 1 and two jobs {5,6} in family 2. The maximum batch size B is two. The processing time p_1 of the jobs of family 1 is 4, and the processing time p_2 of the jobs of family 2 is 3. The due dates of the six jobs are 3, 4, 4, 10, 5 and 20. A feasible schedule can be obtained by forming the batches $b_1=\{1,2\}$, $b_2=\{3,4\}$, and $b_3=\{5,6\}$ and processing them in the sequence $b_1-b_2-b_3$. We obtain $\sum_{j=1}^{6} U_j = 2$ for this schedule.

Now we provide some useful properties of solutions of Problem (2). The following property is shown by Uzsoy [25] for single static batch processing machine scheduling problems with incompatible job families and a regular performance measure.

Property 1. *There is an optimal schedule where all batches, except possibly the last scheduled batch of each job family, are fully loaded.*

Furthermore, the next two properties are shown in [15].

Property 2. *There is an optimal schedule such that all on-time jobs are processed before the tardy jobs.*

Property 3. *There is an optimal schedule in which all on-time jobs of one family are arranged in earliest due date (EDD) order, i.e. when batch b_i is processed before batch $b_{i'}$ then there do not exist two jobs $j \in b_i$ and $j' \in b_{i'}$ such that $d_j > d_{j'}$.*

The following property is also helpful.

Property 4. *For any pair of jobs i and j in the same family such that $d_j > d_i$, there is an optimal schedule for the unweighted version of Problem (2) in which j cannot be tardy if i is on time.*

Proof. Assume that there is an optimal schedule S that contains two jobs i and j from the same family such that $d_j > d_i$, j is tardy, and i is on time. It is clear that, because $d_j > d_i$, these two jobs cannot belong to the same batch. Therefore, let $\tilde{C}_j = C_j$ denote the completion time of the batch that contains j and $\tilde{C}_i = C_i$ the completion timeof the batch that contains i. Because $s(i)=s(j)$, the two jobs can be swapped across the two batches without changing the completion times of the batches. Let S' be the resulting schedule. We obtain

$$C_j - d_i > C_j - d_j > 0 \quad (3)$$

because j is tardy in S. Hence, i is tardy in S'. On the other hand, we obtain

$$C_i - d_j < C_i - d_i \le 0, \quad (4)$$

i.e. j is on time in S'. Therefore, the number of tardy jobs remains the same when such a job swap is performed. □

Property 4 implies, similarly to Property 3, that there is an optimal schedule such that the jobs of a given family that are on time have the largest due dates, i.e. they are the last ones if the jobs of the family were ranked in EDD order.

Using a complexity result for $1\|\sum w_j U_j$, it is shown in [13] that the unweighted version of problem (2) is NP-hard with respect to id-encoding. The complexity of this problem remained open under the standard encoding scheme. However, it was later shown in [15] that the unweighted version of Problem (2) is strongly NP-hard by a reduction from the 3-Partition problem. Hence, we typically have to look for efficient heuristics to solve large-sized problem instances. However, we will show in this paper that we can come up with exact methods that can solve relatively large instances quite fast.

2.2. Related literature

The single-machine scheduling problem $1\|\sum U_j$ can be solved optimally by Moore's algorithm in $O(n \log n)$ time [19]. Efficient heuristics for the scheduling problem $1\|\sum w_j U_j$ are proposed in [20]. This scheduling problem is NP-hard. However, because of the batch processing machine, it seems that there is no direct extension of these algorithms known. Single-machine scheduling problems with $\sum_j U_j$ and $\sum_j w_j U_j$ measures, i.e. for $1|r_j|\sum U_j$ and $1|r_j|\sum w_j U_j$, are considered, for example in [7–9]. Here, we denote by r_j the ready time of job j. Branch and bound techniques and Mathematical Programming based heuristics are suggested. Iterative schemes to compute tight lower bounds are proposed in [7]. The concept of a master sequence is introduced in [9] and is used to come up with efficient Lagrangean relaxation schemes. This idea is later extended to solve the problem $Pm|r_j|\sum w_j U_j$ in [10]. Here we denote by Pm identical parallel machines.

Another branch and bound technique that uses a surrogate relaxation resulting in a multi-choice knapsack problem to find appropriate bounds is suggested in [12]. DP approaches and heuristics are proposed for $1|prec, s_{ij}|\sum U_j$ in [24], where we denote by $prec$ precedence constraints that occur within the subproblem formulation step of the shifting bottleneck heuristic and by s_{ij} sequence-dependent setup times.

Several metaheuristic approaches are discussed in the literature for scheduling problems that involve the measures $\sum_j U_j$ and $\sum_j w_j U_j$. GAs and local search techniques for a single-machine scheduling problem with $\sum_j w_j U_j$ objective are presented in [21]. A variable neighborhood search (VNS) hybridized with threshold accepting is suggested for $Pm|r_j|\sum w_j U_j$ in [22]. Different local search heuristics for a s-batching problem with sequence-dependent setup times and the $\sum_j w_j U_j$ objective are discussed in [6].