European Journal of Operational Research xxx (2014) xxx–xxx

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Discrete Optimization

A meta-heuristic to minimize makespan for parallel batch machines with arbitrary job sizes

Zhao-hong Jia[a], Joseph Y.-T. Leung[b,*]

[a] *Key Lab of Intelligent Computing and Signal Processing of Ministry of Education, Anhui University, Hefei, Anhui 230039, PR China*
[b] *Department of Computer Science, New Jersey Institute of Technology, Newark, NJ 07102, United States*

ARTICLE INFO

Article history:
Received 6 September 2013
Accepted 25 July 2014
Available online xxxx

Keywords:
Meta-heuristic
Parallel batch machines
Max–min ant system
NP-hard

ABSTRACT

We consider the problem of scheduling a set of *n* jobs with arbitrary job sizes on a set of *m* identical and parallel batch machines so as to minimize the makespan. Motivated by the computational complexity of the problem, we propose a meta-heuristic based on the max–min ant system method. Computational experiments are performed with randomly generated test data. The results show that our algorithm outperforms several of the previously studied algorithms.

1. Introduction

With the popularity of intelligent electronics in daily life, the semiconductor industry has become one of the fastest growing and most competitive industries in the world. To enhance the competitiveness, the semiconductor manufacturers try their best to improve the efficiency of the manufacturing processes. Besides the production control methods based on dispatching rules, scheduling is crucial for the competitiveness of semiconductor manufacturing. The manufacturing of integrated circuits is a key business in electronics industry, where batch processing is a common procedure. The process of batching contributes to avoiding setups and facilitates the handling of materials. A batch is defined as a group of jobs that have to be processed jointly (Brucker et al., 1998). A batch scheduling problem consists of grouping the jobs on each machine into batches and scheduling these batches either in serial (named s-batching) or in parallel (named p-batching). Moreover, p-batching is more important than s-batching in semiconductor manufacturing (Mönch, Fowler, Dauzere-Peres, Mason, & Rose, 2011).

Batching problems can be found mainly in the oxidation/deposition/diffusion area of wafer fabs as well as the burn-in operation in the final testing (Xu, Chen, & Li, 2013). Batch processing facilities are generally bottlenecks in the manufacturing systems because of their high utilization and the lengthy processing time of the operations. The burn-in oven in the final testing stage of semiconductor manufacturing is an example of batch machines, where the integrated circuits are tested for defects by subjecting them to thermal stress for an extended period of time. Since the capacity of the burn-in oven is limited and the processing time of the burn-in operations are generally much longer than those of the other testing operations, the burn-in operation becomes a bottleneck in the final testing operation. Thus, effective scheduling of the burn-in operations is of great concern to the manufacturing performance and productivity. Besides, p-batch scheduling is commonly encountered in many other modern manufacturing industries such as the chemical, food and mineral processing, pharmaceutical and metalworking industries as well as with wafer and environmental stress screening chamber fabrication (Xu et al., 2013).

The problem under study in this paper is the p-batch scheduling on parallel and identical batch-processing-machines (BPMs). Each machine has a fixed capacity *C*. A set of jobs with non-identical job sizes have to be grouped into batches, where the total size of all the jobs in a batch cannot exceed *C*. The jobs have non-identical processing times. The processing time of a batch is given by the longest processing time of all the jobs contained in the batch (Mathirajan & Sivakumar, 2006). The batches are scheduled on the machines so as to minimize the makespan; i.e., the time taken to finish all the batches. Once a batch is being processed, it cannot be interrupted and no job can be added into the batch. A small makespan implies a high utilization of the machines (Kashan, Karimi, & Jenabi, 2008). Improving the utilization of a bottleneck station is beneficial to elevate the throughput rate of the system.

The problem can be solved by dealing with two independent subproblems; i.e., grouping the jobs into batches and scheduling

* Corresponding author. Tel.: +1 9735963387; fax: +1 9735965777.
E-mail address: leung@oak.njit.edu (J.Y.-T. Leung).

http://dx.doi.org/10.1016/j.ejor.2014.07.039
0377-2217/© 2014 Elsevier B.V. All rights reserved.

the batches on the parallel BPMs. Since minimizing the makespan for a set of equal-processing-time jobs on a single BPM is equivalent to the one-dimensional bin-packing problem which is known to be strongly NP-hard (Garey & Johnson, 1979), we propose a meta-heuristic to group the jobs into batches. Then we apply the Multifit (MF) heuristic (Coffman, Garey, & Johnson, 1978) to schedule the batches on the machines.

The rest of the paper is organized as follows. In Section 2, we review related work on BPM scheduling problems as well as the max–min ant system (MMAS) algorithm. Section 3 defines the problem under study. The proposed algorithm and its implementation is described in Section 4. With elaborative experimental designs, the effectiveness of our algorithm is compared with several previously studied heuristics in Section 5. Finally, we draw some concluding remarks in Section 6.

2. Related work

In recent years, considerable research have been devoted to scheduling problems related to the BPM problems and the MMAS algorithms. Literature germane to the problem of scheduling on the BPMs and the MMAS algorithm will be presented in the next two subsections.

2.1. BPM problem

Since the problem studied in this paper is p-batch scheduling, we focus on reviewing the studies that have commonalities in their assumptions with ours, especially those investigating the case of arbitrary job sizes.

Researchers started studying the problem of batch scheduling from the simplest model; i.e., scheduling on a single BPM. And more of the early works on BPM focused on the model with identical job size. Ikura and Gimple (1986) were probably the first who studied the BPM problem. They proposed an $O(n^2)$ algorithm to minimize the makespan on a single BPM with identical job processing time, unit job size, and dynamic job arrivals. Uzsoy (1994) later proved that both makespan minimization and total completion time minimization on a single BPM with non-identical job sizes are strongly NP-hard. He gave several heuristics and a branch-and-bound algorithm. Dupont and Jolai Ghazvini (1998) presented another two heuristics, BFLPT (Best-Fit Longest Processing Time) and SKP (Successive Knapsack Problem). The first heuristic, BFLPT, is based on the Best-Fit algorithm, developed for the bin-packing problem. The second one attempts to construct a schedule batch per batch, where the jobs are grouped to maximize the occupied space of the batches. Uzsoy and Yang (1997) provided several heuristics and a branch-and-bound algorithm to minimize total weighted completion time. Sung and Choung (2000) considered a single BPM with job release times and presented several better heuristics. Dupont and Dhaenens-Flipo (2002) developed a branch-and-bound procedure for minimizing the makespan on a single BPM. Jolai (2005) developed a dynamic programming algorithm with polynomial time complexity to minimize the number of tardy jobs for a fixed number of job families and limited machine capacity.

Zhang, Cai, Lee, and Wong (2001) provided the first theoretical results in the worst-case ratios of minimizing the makespan on a single BPM. Li, Li, Wang, and Liu (2005) presented an approximation algorithm for the general problem with arbitrary release times and job sizes.

Some researchers resort to meta-heuristic approaches. Kashan, Karimi, and Jolai (2006) proposed two genetic algorithms (GA) to minimize the makespan on a single BPM. Damodaran, Manjeshwar, and Srihari (2006) applied a simulated annealing (SA) method for the same problem.

Recently, studies on batch scheduling problems have been expanded from a single BPM to parallel machines, because parallel machines are closer to real-world production systems. Lee, Uzsoy, and Martin-Vega (1992) considered makespan minimization on parallel BPMs with unit job size, and examined the worst-case error bound of an arbitrary list scheduling algorithm. They applied the Longest-Processing-Time (LPT) algorithm to the case of identical and parallel BPMs. Uzsoy (1995) studied the scheduling problems with incompatible job families and dynamic job arrivals for both the single and parallel BPMs. He solved the parallel BPM problem by reducing it to the problem with parallel, unit-capacity machines. Brucker et al. (1998) proved that the problem of scheduling on two identical and parallel BPM with a common deadline, unit processing time as well as unit set-up time is NP-hard. They proposed a dynamic programming algorithm to solve the problem. To minimize the total weighted tardiness for scheduling the parallel BPMs with incompatible families, identical job sizes and arbitrary job weights, Mönch and Almeder (2009) proposed an ant colony system (ACS) that includes an efficient local search technique based on jobs swapping across different batches of the same family, and a MMAS which provides basically the same solution quality as that of the ACS approach. Venkataramana and Srinivasa Raghavan (2010) presented ant colony optimization (ACO)-based algorithms by using the structural properties of the problem. Almeder and Mönch (2011) provided an ACO and a variable neighborhood search (VNS) approach hybridized with a decomposition heuristic and a local search scheme to solve the same problem. The computational experiments showed that the VNS approach outperformed the ACO and a GA approach with respect to time and solution quality.

In order to be closer to the real world applications, the case of non-identical job sizes is considered. Chang, Damodaran, and Melouk (2004) studied the problem of scheduling jobs with non-identical job sizes on identical and parallel BPMs. They proposed a simulated annealing (SA) method to minimize the makespan. After the jobs are randomly sequenced, they are clustered to form batches. Then the batches are assigned to the machines by the LPT rule to generate the initial solution. Simulated annealing is used to locally search a better neighboring solution. They compared the results of the SA method with those of the CPLEX solver and concluded that the SA method obtained better solutions with less running time.

Damodaran and Chang (2008) proposed heuristics to solve the parallel BPM scheduling problem, which consists of two independent subproblems; i.e., grouping the jobs into batches and scheduling these batches to minimize the makespan. Two heuristics are applied to grouping the jobs. In both heuristics, the jobs are first sorted in descending order of their processing times. According to the First-Fit-Decreasing (FFD) heuristic, the jobs in the list are put, one by one, into the first batch with enough space to accommodate it, while according to the Best-Fit-Decreasing (BFD) heuristic, the jobs are put, one by one, into the feasible batch with the smallest residual capacity. After the jobs have been grouped into batches, they used either the LPT rule or the Multifit (MF) rule to schedule the batches on the BPMs. In all, there are four heuristics for the parallel BPM problem, by combining the two heuristics for batching and the two heuristics for scheduling. These heuristics are called FFD-LPT, FFD-MF, BFD-LPT and BFD-MF. Finally, the solutions obtained from these four heuristics are compared with the results of the SA (Chang et al., 2004) method and the CPLEX solver.

Some researchers have tried to solve the parallel BPM problems by meta-heuristics. Kashan et al. (2008) developed a hybrid genetic heuristic (HGH) to minimize the makespan on parallel BPMs with