Int. J. Production Economics 169 (2015) 1–10

Contents lists available at ScienceDirect

Int. J. Production Economics

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

Effective heuristic for makespan minimization in parallel batch machines with non-identical capacities

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

[a] Key Lab of Intelligent Computing and Signal Processing of Ministry of Education, Anhui University, Hefei, Anhui 230039, PR China
[b] School of Management, Hefei University of Technology, Hefei, Anhui 230009, PR China
[c] Department of Computer Science, New Jersey Institute of Technology, Newark, NJ 07102, USA

ARTICLE INFO

Article history:
Received 9 June 2014
Accepted 16 July 2015
Available online 28 July 2015

Keywords:
Parallel batch machines
Heuristic
Max–Min Ant System
NP-hard
Non-identical machine capacity
Makespan

ABSTRACT

We consider the problem of scheduling a set of n jobs with arbitrary job sizes on a set of m parallel batch machines with non-identical capacities; the objective is to minimize the makespan. The problem is known to be NP-hard. A heuristic based on the First-Fit-Decreasing (FFD) rule is presented as well as a meta-heuristic based on Max-Min Ant System (MMAS). The performances of the two heuristics are compared with a previously studied heuristic by computational experiments. The results show that both proposed algorithms outperform the previously studied heuristic. Moreover, the MMAS heuristic obtains better solutions compared with the FFD heuristic.

1. Introduction

As a new type of scheduling problems, scheduling on batch processing machines (BPMs) is widely encountered in the fields such as industrial manufacturing, cargo handling in port, ship-lock scheduling, and so on. It is originally abstracted from the burn-in operations in semiconductor fabrication, where different kinds of circuits are loaded onto boards and then placed into the ovens for burn-in test. In this abstraction, the circuits, the boards and the ovens can be regarded as the jobs, the batches and the BPMs, respectively. Because the processing times of the burn-in operation is generally much longer than other operations (e.g., 120 h versus 4–5 h) (Lee et al., 1992), the burn-in operation is often the bottleneck in the process of semiconductor production. Therefore, efficiently scheduling the burn-in operation is of great importance in the enhancement of productivity. Different from the machine of classical scheduling, a BPM can process several jobs in a batch at the same time. Moreover, the sizes of the jobs are generally non-identical in practice. There are two types of batch scheduling problems; i.e., s-batch and p-batch. In the case of s-batch, the jobs in a batch are processed in serial and the processing time of a batch is the sum of the processing times of all the jobs in that batch, while in the case of p-batch, the jobs of a batch are processed in parallel and the processing time of a batch is the longest processing time of the jobs in the batch. P-batch scheduling is more important than s-batch scheduling in semiconductor manufacturing (Mönch et al., 2011). Besides, p-batch scheduling is commonly encountered in many other modern manufacturing industries such as food, chemical and mineral processing, pharmaceutical and metalworking industries as well as environmental stress screening chamber fabrication (Xu et al., 2013).

In this paper, we consider p-batch scheduling on parallel BPMs where the machine capacities are non-identical. A set of jobs with non-identical job sizes have to be grouped into batches such that the total size of the jobs in the batch cannot exceed the capacity of the machine that processes it. The jobs have non-identical processing times and are assumed to be ready at time zero. The processing time of a batch is determined by the largest processing time of all the jobs in the batch (Mathirajan and Sivakumar, 2006). The batches are then scheduled on the machines to minimize the makespan. Once a batch is being processed, it cannot be interrupted and no job can be added into or removed from the batch. The problem of minimizing makespan on a single BPM with non-identical job sizes has been proved to be NP-hard (Uzsoy, 1994). Therefore, the problem we will study is also NP-hard.

The problem can be solved by solving two independent subproblems; i.e., grouping the jobs into batches and scheduling the batches on the parallel BPMs. We propose two different batching algorithms to group the jobs into batches; i.e., a heuristic based on the First-Fit-Decreasing (FFD) rule and a meta-heuristic based on the Max–Min Ant System (MMAS) algorithm. Then we apply the

* Corresponding author at: Department of Computer Science, New Jersey Institute of Technology, Newark, NJ 07102, USA.
Tel.: +1 9735963387; fax: +1 9735965777.
E-mail addresses: zhjia@mail.ustc.edu.cn (Z.-h. Jia), hfutlk@139.com (K. Li), leung@njit.edu (J.-T. Leung).

http://dx.doi.org/10.1016/j.ijpe.2015.07.021
0925-5273/© 2015 Elsevier B.V. All rights reserved.

Multifit (MF) heuristic (Coffman et al., 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 MMAS algorithm. Section 3 defines the studied problem. The proposed heuristic algorithm and its implementation is described in Section 4. Additionally, an example is given in Section 4. The MMAS-based batching algorithm is provided in Section 5. With elaborative experimental designs, the effectiveness of our algorithms are compared with a previously studied meta-heuristic in Section 6. Finally, we draw some concluding remarks in Section 7.

2. Literature review

In classical scheduling, scheduling jobs on machines with processing set restrictions have been extensively studied; see the survey by Leung and Li (2008). The type of processing set restrictions studied in this paper has been called "inclusive processing set restrictions". Ou et al. (2008) have given a heuristic with worst-case performance ratio of 4/3. Huo and Leung (2010) gave a faster algorithm and for a more general processing set restrictions with the same worst-case bound. Our paper differs from previous studies in that we are scheduling p-batch jobs rather than traditional jobs.

Recently, significant studies have been devoted to batch scheduling problems and MMAS algorithm. Related work 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 those studies that have commonalities in their assumptions with ours, especially those investigating the case of non-identical job sizes.

Research on the problem of p-batch scheduling began with the simplest model; i.e., scheduling on a single BPM with identical job size. The first study can be traced back to Ikura and Gimple (1986) who 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. Since then, much research has been done in this area. Uzsoy (1994) proved that minimizing the makespan on a single BPM with non-identical job sizes is strongly NP-hard; he provided several heuristics and a branch-and-bound algorithm. Dupont and Jolai Ghazvini (1998) gave two heuristics; i.e., the Best-Fit Longest Processing Time (BFLPT) and the Successive Knapsack (SK). BFLPT is based on the Best-Fit algorithm for the bin-packing problem while SK attempts to construct a schedule batch by batch, where the jobs are grouped to minimize the unoccupied space of the batches. Uzsoy and Yang (1997) presented several heuristics and a branch-and-bound algorithm to minimize the total weighted completion time. Considering a single BPM with job release times, Sung and Choung (2000) developed several better heuristics. To minimize the makespan on a single BPM, Dupont and Dhaenens-Flipo (2002) presented branch-and-bound method. Jolai (2005) provided a dynamic programming algorithm with polynomial time complexity for minimizing the number of tardy jobs for a fixed number of job families and limited machine capacity. Zhang et al. (2001) presented the first theoretical results in the worst-case ratios of the makespan minimization on a single BPM. Li et al. (2005) provided an approximation algorithm for the general problem with arbitrary release times and job sizes. Recently, meta-heuristics have been applied to solve the p-batch scheduling problems. Damodaran et al. (2006) presented a simulated annealing (SA) algorithm to minimize the makespan on a single BPM. Kashan et al. (2006a) provided two genetic algorithms (GA) for the same problem.

Since parallel machines are closer to the real-world production environments, some researchers further studied the p-batch problems on parallel machines. Lee et al. (1992) applied the Longest-Processing-Time (LPT) algorithm to the problem of identical and parallel BPMs with identical job sizes. Uzsoy (1995) proposed several algorithms to minimize the makespan, maximum lateness, and total weighted completion time for single and parallel identical BPMs with incompatible job families and dynamic job arrivals. Brucker et al. (1998) proved that scheduling on two identical and parallel BPMs with a common deadline, unit processing time and unit setup time is NP-hard; they then developed a dynamic programming algorithm for the problem. To minimize the total weighted tardiness for scheduling parallel BPMs with incompatible families, identical job size and arbitrary job weights, Mönch and Almeder (2009) provided two meta-heuristic algorithms; i.e., an Ant Colony System (ACS) and a Max–Min Ant System (MMAS), which are both improved variants of the Ant Colony Optimization (ACO) algorithms. Venkataramana and Srinivasa Raghavan (2010) gave an ACO-based algorithm by using the structural properties of the problem. Almeder and Mönch (2011) proposed an ACO algorithm and a Variable Neighborhood Search (VNS) approach hybridized with a decomposition heuristic and a local search scheme for the same problem. According to the experimental results, the VNS method shows better performance than the ACO and the GA approach in both the running time and the solution quality.

In practice, job sizes are generally non-identical and hence researchers began to study the scheduling problem with non-identical job sizes. Chang et al. (2004) presented a simulated annealing (SA) method to minimize the makespan for scheduling jobs with non-identical sizes on identical parallel BPMs. After being randomly sequenced, the jobs are grouped into batches, which are then assigned to the machines by the LPT rule to construct an initial solution. Then, SA is used to exploit a better neighboring solution. Comparing with the results of CPLEX, the SA method is found to gain better solutions with less running time. Damodaran and Chang (2008) provided heuristics to minimize the makespan on parallel BPMs. Two heuristics are used to batch the jobs, where the jobs are all first sorted in descending order of their processing times. In the First-Fit-Decreasing (FFD) heuristic, the jobs in the sequence are put, one by one, into the first batch with enough space to accommodate it, while in the Best-Fit-Decreasing (BFD) heuristic, the jobs are put, one by one, into the feasible batch with the smallest residual capacity. After all the batches are generated, either the LPT rule or the Multifit (MF) rule is used to schedule the batches on the BPMs. Hence, there are totally four heuristics for the studied problem; they are called FFD-LPT, FFD-MF, BFD-LPT and BFD-MF. The solutions of the four heuristics are compared with the results of the SA (Chang et al., 2004) and CPLEX. Meta-heuristics have also been applied to the parallel BPM problems. Kashan et al. (2008) provided a hybrid genetic heuristic (HGH) to minimize the makespan on parallel BPMs with arbitrary job sizes. According to the computational experiments, the HGH outperforms the SA method (Chang et al., 2004). Shao et al. (2008) applied the neural network (NN) approach to the problem; comparative experiments with the heuristics of Damodaran and Chang (2008) verified the better performance of the NN method. Chen et al. (2010) developed an ACO and a GA algorithm for solving the batch scheduling problem on parallel BPMs with dynamic job arrivals.

The above research assumes that the machines are identical; i.e., the machines have the same capacity. In practice, factories may add new machines that have a larger capacity than the older models. Jobs may have sizes larger than the capacity of the older models. Thus, it is important to study this scheduling problem with machines having different capacities and jobs may not fit in any machine of the older models. To the best of our knowledge, there are only three papers that study this problem. Xu and Bean (2007) proposed a GA based on