Applied Soft Computing xxx (2016) xxx–xxx

Contents lists available at ScienceDirect

Applied Soft Computing

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

Distance matrix based heuristics to minimize makespan of parallel batch processing machines with arbitrary job sizes and release times

Shengchao Zhou [a,*], Huaping Chen [b], Xueping Li [c]

[a] School of Logistics, Central South University of Forestry and Technology, Changsha, Hunan 410004, China
[b] School of Management, University of Science and Technology of China, Hefei, Anhui 230026, China
[c] Department of Industrial and Systems Engineering, University of Tennessee, Knoxville, TN 37996, USA

ARTICLE INFO

Article history:
Received 5 May 2014
Received in revised form 7 May 2016
Accepted 5 October 2016
Available online xxx

Keywords:
Scheduling
Batch processing machines
Makespan
Heuristics
Distance matrix

ABSTRACT

Batch scheduling is prevalent in many industries such as burn-in operations in semiconductor manufacturing and heat treatment operations in metalworking. In this paper, we consider the problem of minimizing makespan on parallel batch processing machines in the presence of non-identical job sizes and arbitrary release times. Since the problem under study is strongly NP-hard (non-deterministic polynomial-time hard), we develop a number of efficient heuristics. In the proposed heuristics, several distance matrix based approaches are first developed to form batches and a scheduling technique is devised to assign the batches to parallel machines. The performance of the proposed heuristics is evaluated by comparing their results to a set of existing heuristic and meta-heuristic algorithms in the literature. The computational experiments show that the proposed heuristics are competitive with respect to solution quality. Moreover, the computational costs of the proposed heuristics are very little.

1. Introduction

Machine scheduling is one of many important optimization problems in manufacturing industry [1–3]. This paper deals with the problem of scheduling a set of jobs on a group of identical parallel batch processing machines. Each job is characterized by a job size, a processing time, and a release time. A batch processing machine is one which can simultaneously process several jobs in a batch as long as the total size of all the jobs in the batch does not exceed the capacity of the machine. All jobs in a batch begin and complete processing simultaneously. The processing time and ready time of a batch are given by the longest processing time and latest release time among all jobs in the batch, respectively. The goal is to find a schedule for the jobs so that the makespan, defined as the completion time of the last batch leaving the production system, is minimized.

This research is motivated by burn-in operations for final testing in semiconductor manufacturing. In burn-in operations, integrated circuits are put in an oven in batches and heated for a prolonged period of time to detect any early failures. Compared to other testing operations, the processing times in burn-in operations are generally longer (see Lee et al. [4] for the detailed process). Thus the burn-in operation is usually a bottleneck in the final testing operation. Therefore, effective scheduling of these operations is important and can greatly improve the machine utilization and production rate. By minimizing the makespan, the utilization of the machines can be improved. Hence, the makespan performance measure is selected as the scheduling objective in this study.

Scheduling on batch processing machines involves two interdependent decisions: forming batches and scheduling the batches formed. In determining the jobs of each batch, most heuristics consider only a single factor, e.g. job processing times or release times, to determine a job sequence, and then put successive jobs into a batch within the limit of machine capacity. Both job processing times and release times have important effects on solutions in dynamic scheduling problems. We argue that a heuristic considering them simultaneously will be more efficient. In practice, job processing times and release times are usually contradictory when making decisions to form batches. This is indeed a challenge for solving batch scheduling problems. It is rare in the literature to simultaneously consider the combined effects of these two factors when grouping jobs into batches. The purpose of this paper is to propose a number of heuristics from a new perspective. The heuristics attempt to make a trade-off between job release times

* Corresponding author.
E-mail addresses: zhousc@mail.ustc.edu.cn (S. Zhou), hpchen@ustc.edu.cn (H. Chen), Xueping.Li@utk.edu (X. Li).

http://dx.doi.org/10.1016/j.asoc.2016.10.008
1568-4946/

and processing times and hence can take both factors into account to form batches. Details are given in Section 4.

The reminder of the paper is organized as follows. In the next section, a review of literature pertinent to batch processing machine scheduling problems is presented. Assumptions and mathematical formulations are presented in Section 3. Section 4 provides a number of different heuristics. In Section 5, experimental results are presented to examine the effectiveness of the proposed approaches. A summary and discussion of future research directions concludes this paper in Section 6.

2. Literature review

The problem relevant to scheduling on batch processing machines has attracted a plethora of research. Ikura and Gimple [5] provided an efficient algorithm to find a due date feasible schedule which minimizes makespan under the assumption that release times and due dates are agreeable and all jobs have identical processing times. Lee et al. [4] studied the problem where the processing times of jobs are the same and release times and due dates are agreeable. They developed heuristic algorithms for minimizing the number of tardy jobs and maximum tardiness. Chandru et al. [6] provided an exact solution procedure to minimize the total completion time for the single machine problem. They also developed heuristic algorithms for both single and parallel machine problems. Li and Lee [7] proved that minimizing the maximum tardiness and minimizing the number of tardy jobs are strongly NP-hard even when release times and due times of the jobs are agreeable. They addressed the problems of minimizing maximum tardiness and minimizing the number of tardy jobs with dynamic programming algorithms. Brucker et al. [8] addressed the problem of scheduling jobs on a batching machine to minimize regular scheduling criteria with dynamic programming algorithms. Lee and Uzsoy [9] considered the problem of minimizing makespan on a single batching machine in the presence of dynamic job arrivals and identical job sizes. They proposed a number of efficient algorithms for this problem. Sung and Choung [10] analyzed the dynamic problem with different job release times and exploited a branch-and-bound algorithm and several heuristics. Sung et al. [11] discussed job families and arbitrary arrival times and proposed a dynamic programming algorithm of polynomial time complexity. Wang and Uzsoy [12] considered the problem of minimizing maximum lateness on a batch processing machine in the presence of dynamic job arrivals and developed a genetic algorithm (GA). Deng et al. [13] presented a polynomial time approximation scheme for minimizing total completion time on a batching machine with release times. Jolai [14] considered the problem of scheduling a batch processing machine with incompatible job families to minimize the number of tardy jobs. They showed that this problem is NP-hard and developed a dynamic programming algorithm with polynomial-time complexity for a fixed number of families. Behnamian et al. [15] addressed a scheduling problem in a three-machine flowshop environment with a batch processing machine using GA.

All of the above studies concentrated on the problem with identical job sizes. Uzsoy [16] considered the problems of minimizing makespan and minimizing total completion time with non-identical job sizes. Both problems were proved as NP-hard and several heuristics were proposed. Dupont and Jolai Ghazvini [17] proposed two heuristics, namely SKP (successive knapsack problem) and BFLPT (best-fit longest processing time), which achieve better makespan than FFLPT (first-fit longest processing time) presented by Uzsoy [16]. Zhang et al. [18] proposed an approximation algorithm with the worst-case ratio 7/4 and proved that FFLPT has a worst case ratio no greater than 2. Dupont and Dhaenens-Flipo [19] presented a branch and bound algorithm to optimize makespan and Melouk et al. [20] introduced a simulated annealing (SA) approach. Kashan et al. [21] developed two different GAs for scheduling jobs with non-identical sizes on a batch processing machine. Chou et al. [22] examined the dynamic batch scheduling problem with non-identical job sizes and proposed a hybrid GA. Kashan and Karimi [23] considered the scheduling problem of a batch processing machine with incompatible job families and the performance measure of minimizing total weighted completion time. An ant colony framework in two versions was proposed for solving this case. Xu et al. [24] considered the same problem as in Chou et al. [22] and introduced an ant colony optimization (ACO) algorithm and Zhou et al. [25] also developed a number of constructive heuristics. Jia and Leung [26] studied the problem of minimizing makespan on a batching machine and presented a max-min ant system algorithm.

Most studies aforementioned focused on single batch processing machine problems. Chang and Chen [27] introduced a GA with dominance properties for parallel machine scheduling problems with setup times. Li et al. [28] considered the parallel machine scheduling problem to minimize makespan with controllable processing times and proposed a SA algorithm. Chang et al. [29] developed a SA approach for minimizing makespan of parallel batch processing machines under the assumption of zero job release times. Kashan and Karimi [23] and Damodaran et al. [30] proposed two different GAs to minimize makespan objective, respectively, which both outperformed the SA proposed by Chang et al. [29]. Cheng et al. [31] dealt with the problem of scheduling parallel batching machines with arbitrary job sizes using an ACO. Li et al. [32] studied scheduling unrelated parallel batch processing machines to minimize makespan and provided several heuristics.

The above research assumes that job release times are equal, while in reality, jobs may be released dynamically. Chung et al. [33] studied the problem of minimizing makespan on parallel batch processing machines with dynamic job release times. They proposed a mathematical model and two modified delay heuristics (H1 and H2). Damodaran et al. [34] considered the same problem as in Chung et al. [33] and presented a greedy randomized adaptive search procedure (GRASP). Chen et al. [35] introduced two metaheuristics (GA and ACO) to solve the problem and Damodaran and Velez-Gallego [36] also proposed a SA. The proposed research falls into the case of parallel machines with dynamic job release times and we will compare our proposed methods with the state-of-the-art.

3. Problem definition

The problem can be defined as follows:

1. There are n independent jobs to be processed in the job set J. The processing time, release time and size of job j are denoted by p_j, r_j and s_j, respectively.
2. There are m identical parallel batch processing machines in the machine set M and each machine has a capacity C. B is the set of batches. The sum of the job sizes in batch b cannot exceed C.
3. Once the processing of a batch is started, it cannot be interrupted and other jobs cannot be introduced into the machine until the processing is completed. The processing time P^{bl} of the bth batch processed on machine l is equal to the longest processing time of the jobs in the batch. Similarly, the ready time R^{bl} of the bth batch processed on machine l is equal to the latest release time of the jobs in the batch. The start time of the bth batch processed on machine l is denoted by S^{bl}.
4. The maximum completion time of machine l is denoted by T_l. Our objective is to minimize the makespan C_{max}, i.e., the longest completion time of machines.