Two-Machine Flowshop Batching and Scheduling

B.M.T. LIN bmtlin@mail.nctu.edu.tw
Department of Information and Finance Management, National Chiao Tung University, Hsinchu, Taiwan 545

T.C.E. CHENG LGTcheng@inet.polyu.edu.hk
Department of Logistics, The Hong Kong Polytechnic University, Kowloon, Hong Kong

Abstract. We consider in this paper a two-machine flowshop scheduling problem in which the first machine processes jobs individually while the second machine processes jobs in batches. The forming of each batch on the second machine incurs a constant setup time. The objective is to minimize the makespan. This problem was previously shown to be NP-hard in the ordinary sense. In this paper, we first present a strong NP-hardness result of the problem. We also identify a polynomially solvable case with either anticipatory or non-anticipatory setups. We then establish a property that an optimal solution for the special case is a lower bound for the general problem. To obtain near-optimal solutions for the general problem, we devise some heuristics. The lower bound is used to evaluate the quality of the heuristic solutions. Results of computational experiments reveal that the heuristics produce solutions with small error ratios. They also suggest that the lower bound is close to the optimal solution.

Keywords: production scheduling, flowshop, batch processing, makespan, strong NP-hardness, lower bound, heuristics

1. Introduction

Flowshop scheduling, initiated by Johnson (1954), is one of the most extensively studied topics in scheduling research. In a recent paper, Cheng and Wang (1998) consider batch scheduling problems in a two-machine flowshop which comprises a discrete processor and a batch processor. There is a set of jobs simultaneously available for processing in the flowshop. All jobs are first processed by the discrete processor, which processes one job at a time. The batch processor processes the jobs, transferred from the discrete processor, in batches. The forming of each batch on the batch processor incurs a constant setup time. The processing time of a batch on the batch processor is the sum of the constant setup time and the processing times of all jobs belonging to it. All jobs in a batch have a common completion time which is equal to the completion time of the last job of the batch. The objective is to batch, as well as schedule, the jobs so as to minimize the makespan.

This scheduling problem arises from the manufacturing of custom-built very large-scale integrated circuits by flexible manufacturing cells organized into a two-stage flowline. In the first stage, chips of various types are picked and placed on a circuit board according to its individual technical specifications by a pick and insertion machine –
the discrete processor. Each circuit board is unique and represents a discrete job. Upon completion of this operation, the circuit board is loaded onto a pallet to be transferred to the second stage for further processing. Circuit boards loaded on the same pallet correspond to a batch. Pallets are installed and removed by a robot before and after processing on the second machine. The fixed time incurred in removing a previous pallet and installing a new one is the constant setup time. In the second stage, each pallet will have its circuit boards soldered and tested one at a time by an integrated soldering and testing equipment – the batch processor. This is a highly sophisticated and expensive piece of equipment and so operations on it are performed in batches to minimize idle time caused by frequent setups. Thus, the batch processing time is equal to the sum of the setup time and the individual soldering and testing time for each circuit board loaded on the same pallet. The objective of the scheduling problem is to determine the optimal sequence and batching compositions so as to minimize the makespan.

In the literature, most of the previous research on batch scheduling problems is related to the single machine case to minimize the total flow time. An O(n) algorithm is first given by Naddef and Santos (1988) for the case where all jobs have the same processing time. Coffman, Nozari and Yannakakis (1989) later devise an improved O(\sqrt{n}) algorithm. Albers and Brucker (1993) identify many NP-hard problems in which precedence constraints between jobs are imposed. They also show that many problems can be transformed into the shortest path problem, which is solvable by an algorithm with a time complexity linear in the number of vertices visited. In (Cheng, Kovalyov and Lin, 1997), the problem of minimizing the sum of total weighted job earliness and mean batch delivery penalties is investigated. On m parallel machines, Cheng et al. (1996) present an O(nm^2) dynamic programming algorithm to determine the minimum total completion time.

Considering the two-machine flowshop environment, Ahmadi et al. (1992) study a class of batching and scheduling problems. In their model, a batch can accommodate all of its jobs simultaneously and the batch processing time is constant and independent of the batch size. A study similar to the problem of interest in this paper is conducted by Cheng, Lin and Toker (2000). In the two-machine flowshop environment they consider, both machines process the jobs in batches. After processing on the first machine, a batch is transferred to the second machine with the batch composition preserved. They prove that the general problem is strongly NP-hard and propose polynomial algorithms for some special cases. Glass, Potts and Strusevich (2001) consider a similar problem where setups on the second machine are anticipatory. The reader is referred to (Cheng, Gupta and Wang, 2000; Allahverdi, Gupta and Aldowaisan, 1999; Potts and Kovalyov, 2000) for comprehensive surveys on batch scheduling in flowshop environments.

The rest of this paper is organized as follows. In section 2, we introduce the notation that will be used throughout this paper. In section 3, we present the strong NP-hardness result by a reduction from 3-PARTITION. We identify special cases that are polynomially solvable in section 4. Besides, we also establish an interesting property concerning a lower bound for the general problem. In section 5, four heuristic methods are given to generate sub-optimal solutions. A series of computational experiments is