Computers & Industrial Engineering 56 (2009) 1393–1401

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

Bottleneck-based heuristics to minimize total tardiness for the flexible flow line with unrelated parallel machines ☆

Chun-Lung Chen [a], Chuen-Lung Chen [b,*]

[a] Department of Accounting Information, Takming University of Science and Technology, Taipei, Taiwan, ROC
[b] Department of Management Information Systems, National Chengchi University, 64, Sec. 2, Zhi-nan Road, Wenshan, Taipei 116, Taiwan, ROC

ARTICLE INFO

Article history:
Received 19 October 2006
Received in revised form 17 August 2008
Accepted 31 August 2008
Available online 6 September 2008

Keywords:
Bottleneck
Flexible flow line
Unrelated parallel machines
Dispatching rules
Tabu search algorithm
Total tardiness

ABSTRACT

This paper considers the flexible flow line problem with unrelated parallel machines at each stage and with a bottleneck stage on the line. The objective of the problem is to minimize the total tardiness. Two bottleneck-based heuristics with three machine selection rules are proposed to solve the problem. The heuristics first develop an indicator to identify a bottleneck stage in the flow line, and then separate the flow line into the upstream stages, the bottleneck stage, and the downstream stages. The upstream stages are the stages ahead of the bottleneck stage and the downstream stages are the stages behind the bottleneck stage. A new approach is developed to find the arrival times of the jobs at the bottleneck stage. Using the new approach, the bottleneck-based heuristics develop two decision rules to iteratively schedule the jobs at the bottleneck stage, the upstream stages, and the downstream stages. In order to evaluate the performance of the bottleneck-based heuristics, seven commonly used dispatching rules and a basic tabu search algorithm are investigated for comparison purposes. Seven experimental factors are used to design 128 production scenarios, and ten test problems are generated for each scenario. Computational results show that the bottleneck-based heuristics significantly outperform all the dispatching rules for the test problems. Although the effective performance of the bottleneck-based heuristics is inferior to the basic tabu search algorithm, the bottleneck-based heuristics are much more efficient than the tabu search algorithm. Also, a test of the effect of the experimental factors on the dispatching rules, the bottleneck-based heuristics, and the basic tabu search algorithm is performed, and some interesting insights are discovered.

1. Introduction

This paper considers the flexible flow line scheduling problem with unrelated parallel machines at each stage and with a bottleneck stage on the line. A typical flexible flow line (FFL) scheduling problem can be defined as follows: many jobs pass through multiple stages with one or more parallel machines at each stage, and there is unlimited intermediate storage between any two successive stages. The flow of jobs through the shop moves in one direction from the first stage to the last. The flexible flow line is also called a flexible flow shop (FFS), a hybrid flow shop (HFS), or a flow shop with multiple processors (FSMP). Fig. 1 illustrates the physical relationship between the machines and the stages.

Flexible flow lines occur in many different manufacturing environments, including printed circuit board (PCB) assembly (Jin, Ohno, Ito, & Elmaghraby, 2002; Sawik, 2002), PCB fabrication (Alisantoso, Khoo, & Jiang, 2003; Choi, Kim, & Lee, 2005; Lee, Kim, Kim, & Choi, 2003), multilayer ceramic capacitor (MLCC) manufacture (Yang, Kuo, & Chang, 2004) and leadframe manufacture (Lee, Kim, & Choi, 2004). In this paper we study the problem of scheduling a set of independent jobs in a flexible flow line with unrelated parallel machines. The processing time of a job at a stage with unrelated parallel machines is dependent on the machine assigned to the job at the stage. This means that a job may have different processing times at a stage in a flexible flow line. Unrelated parallel machines are also common in a real-world factory. For example, the shop may need to extend its capacity, so supplementary parallel machines can be added at certain stages. Load imbalance also leads to the supplementing of the machines at different stages. In addition, unrelated parallel machines will exist in stages due to the coexistence of new and old machines. Unrelated parallel machines scheduling can be found in the real-life manufacturing environments, such as the drilling operations of PCB fabrication (Hsieh, Chang, & Hsu, 2003; Yu, Shih, Pfund, Carlyle, & Fowler, 2002) and the dicing of semiconductor wafer manufacturing (Kim, Na, & Chen, 2003). Since these manufacturing systems usually include a large number of stages, they can be classified as flexible flow line with unrelated parallel machines problems. Ruiz and

☆ This manuscript was processed by Area Editor John W. Fowler.

* Corresponding author. Tel.: +886 2 2939 3091x81218; fax: +886 2 2939 3754.
E-mail address: charleschen@mail.takming.edu.tw (C.-L. Chen).

0360-8352/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cie.2008.08.016

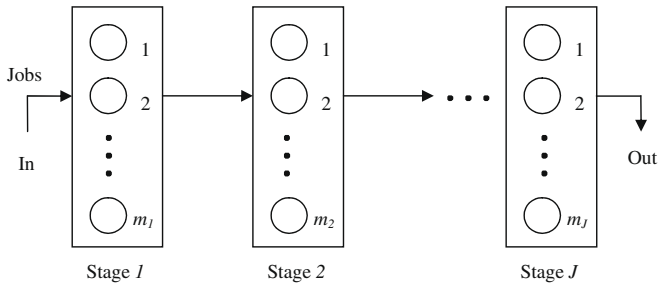

Fig. 1. An example of a flexible flow line.

Maroto (2006) also presented a ceramic tile manufacturing system as a flexible flow line with unrelated parallel machines problem.

For the last few decades, many researchers have studied FFL scheduling problems. As surveyed in Linn and Zhang (1999), most of the studies consider only two stages, identical machines, and throughput related measures, such as makespan and total flow time. Lately, there have been studies concerning multiple stages and due date related criteria. However, few of them have dealt with multiple stages involving unrelated parallel machines in stages. Azizoglu, Cakmak, and Kondakci (2001) developed a branch and bound algorithm to minimize the total flow time for a flexible flow shop problem with identical machines. This algorithm could find the optimal schedule and solve moderately sized problems in a reasonable amount of time. Lee et al. (2003) focused on a hybrid flow shop with identical machines. They developed several dispatching rules to solve the problem in order to minimize the total tardiness. Bertel and Billaut (2004) developed a genetic algorithm to solve a multiprocessor flow shop problem involving recirculation and considered the objective of minimizing the total number of weighted tardy jobs. They considered the problem with uniform parallel machines at each stage. Low (2005) developed a simulated annealing heuristic to solve flexible flow line with unrelated parallel-machine problem in a flow shop. They considered the objective of minimizing the total flow time. Ruiz and Maroto (2006) proposed a genetic algorithm to solve a hybrid flow shop with sequence dependent setup times and with minimum makespan as the objective.

The bottleneck phenomena occur frequently in many manufacturing systems. Goldratt and Cox (1992) stated the idea that the bottleneck resource governs the overall system's performance. Bottleneck management is a very important task on the shop floor and is really effective in production scheduling. Using bottleneck-based heuristics to solve the FFL problems has attracted many researchers. Adler et al. (1993) considered a practical scheduling problem for plants that produce multiple paper bags. The machine environment can be regarded as a flexible flow shop, and the machines at a stage may not all be identical. They developed an ad hoc bottleneck-based heuristic to solve the specific problem. Chen and Lee (1998) suggested a bottleneck-based group scheduling procedure to solve flow line cell scheduling problems. The procedure was based on the bottleneck machine and attempted to fully utilize the bottleneck machine and minimize makespan. Lee et al. (2004) developed a bottleneck-based heuristic to solve a multistage hybrid flow shop problem with identical parallel machines at each stage and with minimum total tardiness as the objective. The heuristic first focuses on the bottleneck stage, constructs the schedule of the bottleneck stage, and constructs schedules for other stages based on the schedule of the bottleneck stage. The heuristic uses the sum of processing times of a job at the upstream stages to be the arrival time of the job at the bottleneck stage. If the procedure results in an infeasible schedule, then the arrival times of the jobs at the bottleneck stages will be iteratively modified until a feasible schedule is obtained. They compared the performance of eight well-known dispatching rules and the bottleneck-based heuristic. The computational results showed that the heuristic dominated all the dispatching rules.

In this paper, we develop bottleneck-based heuristics to solve the flexible flow line problem with unrelated parallel machines and with minimum total tardiness as the objective. The heuristics first identify the bottleneck stage on the flow line. Then, the heuristics separate the flow line into the upstream stages, the bottleneck stage, and the downstream stages. A new approach is developed to find the arrival times of the jobs at the bottleneck stage. Using this new approach, the bottleneck-based heuristics develop two decision rules to iteratively schedule the jobs at the bottleneck stage, the upstream stages, and the downstream stages. The rest of this paper is organized as follows. Section 2 describes the problem considered in this paper. Section 3 presents the bottleneck-based heuristic. Section 4 describes and analyzes computational experiments. Finally, Section 5 summarizes the major findings of this paper and proposes some further research.

2. Description of the problem

The FFL manufacturing system considered in this paper assumes that there are J stages and include a bottleneck stage b. For convenience, we separate the J stages into three groups: the upstream stages (the stages ahead of the bottleneck stage), the bottleneck stage, and the downstream stages (the stages behind the bottleneck stage). There are m_j unrelated parallel machines in stage j and the number m_j may vary from stage to stage. There are N jobs to be processed and each job has the same routing and must visit all stages consecutively. The processing time of the operation of a job on a stage is dependent on the machine in the stage assigned to the job, and it is known in advance. A machine can process only one job at a time, and jobs cannot be preempted. There are unlimited buffers between stages, and there is no machine breakdown and no setup time required before jobs are processed on any machine. Also, we assume that all the jobs and the machines are available at time zero. The following notations are used to describe the FFL problem and the proposed bottleneck-based heuristics in the next section:

i	job index, $i = 1,2,3,\ldots,N$
j	stage index, $j = 1,2,3,\ldots,J$
k	machine index, $k = 1,2,3,\ldots,m_j$
m_j	number of unrelated parallel machines at stage j
M_{jk}	the machine index for machine k at stage j
b	bottleneck stage index, $b \in [1,2,3,\ldots,J]$
AV_{bk}	the available time for machine k at the bottleneck stage b
AR_{ib}	the arrival time of job i at the bottleneck stage b
RT_{ibk}	the ready time for job i on machine k at the bottleneck stage b (the larger value of the arrival time of job i and the available time of machine k at the bottleneck stage b)
C_{ij}	the completion time of job i at the last stage J
C_{ibk}	the completion time of job i on machine k at the bottleneck stage b
C_{ijk}	the completion time of job i at the last stage J while using machine k at the bottleneck stage
d_i	the due date of job i
d_{ibk}	the operational due date of job i using machine k at the bottleneck stage b
$\bar{p}_{ij}$	the average processing time of job i at stage j
p_{ibk}	the processing time of job i on machine k at the bottleneck stage b
$\bar{p}_j$	the average processing time of all the jobs in the queue of stage j
t	the decision time (current time) when the scheduling is made