Computers & Operations Research 39 (2012) 3316–3324

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

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

Dynamic routing strategies for JIT production in hybrid flow shops

Wei Weng*, Xin Wei, Shigeru Fujimura

Graduate School of Information, Production and Systems, Waseda University, 2-7 Hibikino, Wakamatsu-ku, Kitakyushu, Fukuoka 808-0135, Japan

ARTICLE INFO

Available online 27 April 2012

Keywords:
Dynamic routing
Hybrid flow shop
Dispatching rules
Just-in-time production

ABSTRACT

In production processes, just-in-time (JIT) completion of jobs helps reduce both the inventory and late delivery of finished products. Previous research which aims to achieve JIT job completion mainly worked on static scheduling problems, in which all jobs are available from time zero or the available time of each job is known beforehand. In contrast, dynamic scheduling problems which involve continual arrival of new jobs are not much researched and dispatching rules remain the most frequently used method for such problems. However, dispatching rules are not high-performing for the JIT objective. This study proposes several routing strategies which can help dispatching rules realize JIT completion for jobs arriving dynamically in hybrid flow shops. The routing strategies are based on distributed computing which makes realtime forecast of completion times of unfinished jobs. The advantages include short computing time, quick response and robustness against disturbance. Computer simulations show that the performance of dispatching rules combined with the proposed routing strategies is significantly higher than that of dispatching rules only and that of dispatching rules combined with the previous routing methods.

1. Introduction

In the manufacturing industry, production processes are usually conducted in shops which are linked as a production chain. For each shop along the chain, temporary inventory of finished products will be produced if jobs are completed earlier than the time when they can be delivered to the successor shop, and late delivery of finished products will be caused if jobs are completed later than the time mentioned above. This indicates that only when each job is completed on time can the inventory and late delivery of finished products be minimized simultaneously. The same applies to the flow of products between the parties along a supply chain, as JIT job completion can lead to zero inventory of finished products for the manufacturer as well as timely delivery of products to the downstream customer.

To achieve JIT completion for a job, a number of factors should be taken into account, including the due date (i.e. the time that a job should be finished), earliness and tardiness. Perhaps due to this reason, most existing research for the JIT objective work on problems with relatively simple conditions such as the single machine common due date problem [13] and identical parallel machine problem [20,1,19]. Only a small number of studies have tried to solve more complex problems such as the hybrid flow shop (HFS) problem [6,10].

The HFS problem is one of the typically complex problems which are widely seen in industries [16]. It is the same as the flow shop problem except that there are multiple machines available at each stage. As a job must go through all the stages in sequence and be processed by one of the machines at each stage, there exist a lot of different routes for a job to go through an HFS. Because of this, the HFS problem is proven to be strongly NP-hard and even solving a small-sized problem may require considerable time [17].

Therefore, researchers have been trying to develop efficient models which can find the optimal or near-optimal solutions for the HFS problem within reasonable time. Commonly used methodology includes exact methods such as branch and bound [12] and mixed integer programming [4,10] as well as heuristic methods such as genetic algorithm and simulated annealing [4,5,9].

These efforts have surely upgraded the searching efficiency and helped reduce the difficulty in dealing with large-scale HFS problems. However, there are still limitations in applying the methods to daily production, in that these methods are developed based on the assumption that all jobs are available from time zero or the available time of each job is known beforehand. In other words, these are static scheduling methods which plan a given number of received jobs. For jobs which are unknown beforehand or arrive dynamically at the HFS production shop, rescheduling must be made, which may require long computing time and, in the case that previously received jobs have already started undergoing processing, may reduce the quality of solutions.

According to these limitations of static scheduling and the complex nature of HFS, this paper presents some routing strategies

* Corresponding author.
E-mail address: wengwei@toki.waseda.jp (W. Weng).

0305-0548/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cor.2012.04.022

which use dynamic scheduling to solve the HFS problem for JIT objective. The reason to choose dynamic scheduling lies in that dynamic scheduling has the advantage of being much less restricted by the size and complexity of a problem than static scheduling. Instead of seeking for the optimal solution for a given number of received jobs, dynamic scheduling focuses more on the provision of quick solutions to jobs which are received continually over time.

Existing research on dynamic HFS scheduling is still small in number and dispatching rules and their combinations remain the most widely used method [7,18,2]. Since most dispatching rules are designed for the objective of minimizing the makespan, mean flow time or tardiness [14,3], and almost no dispatching rule considers both the due date of a job and the speed of a machine, it is difficult to use these rules to deliver good JIT performance.

To make up for this void, this study proposes routing strategies which can help dispatching rules realize the function of JIT job completion in HFS's. The strategies work based on realtime distributed computing, which requires only small amount of calculation to make a routing decision. Such autonomous control based methods are believed to be suitable for dynamic HFS problems [18]. By using the strategies, a job can be delivered to the most suitable machine at each stage for processing and finally be completed at a time close to its due date.

As the proposed strategies preserve all the advantages of dynamic scheduling, the available time of a job is not required to be known before the job becomes available, and no rescheduling is necessary after new jobs are received. No matter how large the HFS environment is, no matter how many jobs are involved, and whether there exists dynamic arrival of new jobs or not, the proposed strategies is able to work unaffected.

This paper is organized as follows: Section 2 gives the problem statement; Section 3 presents the proposed routing strategies; Section 4 gives the detailed computer simulations and discussions; and Section 5 concludes the paper with some suggestions on future topics.

2. Problem statement

Notation

C_j	completion time of job j
D_j	due date of job j
E_j	earliness of job j
J	number of jobs received dynamically
M_s	set of stage-s machines
P	number of predefined product types
S	number of stages
T_j	tardiness of job j

We consider an HFS production system (Fig. 1) which is consisted of S stages ($s=1, 2, \ldots, S$), with $|M_s|$ machines ($m=1, 2, \ldots, |M_s|$) which are different in processing speed at stage s. Each machine has a buffer where jobs to be processed by it wait in a queue.

Jobs arrive dynamically over time and each job must be made into one of the predefined P types of products ($p=1, 2, \ldots, P$). A job must go through the S stages in sequence for processing so as to become a final product. At each stage, a job can be processed by any one of the machines, and the processing time differs depending on the machine. For jobs of the same product type, their processing times are identical on the same machine. Waiting jobs at each machine are processed in the order of a dispatching rule, and one dispatching rule is used throughout the system. No delivery time exists between the stages and a job is considered to arrive at the next stage as soon as it is completed by the predecessor stage.

Each job has a due date D_j, which is the time when the job should be completed as the product. D_j must be longer than the total processing time of job j excluding the waiting time, and is given at the time when job j arrives in the HFS system. If job j is completed before D_j, its earliness is expressed by $E_j=\max(0, D_j-C_j)$, in which C_j represents the completion time of job j; otherwise, its tardiness is expressed by $T_j=\max(C_j-D_j, 0)$. The maximum function is used in E_j and T_j, accordingly an early job produces zero tardiness; and a tardy job, zero earliness.

The objective of this study is to minimize the total earliness and tardiness of each job. For evaluation, the average earliness and tardiness of all the jobs that are received dynamically is used

$$\text{Minimize} \sum_{j=1}^{J} (E_j+T_j) \cdot \frac{1}{J}. \tag{1}$$

Besides, some assumptions are made in this study.

Assumptions:

(1) One machine can only process one job at a time.
(2) The processing of each job is non-preemptive.
(3) The buffer of each machine is unlimited.

3. Proposed routing strategies

Notation

$DM_{j,s}$	destination machine of job j at stage s
$FC_{j,s,m}$	forecast completion time of job j if machine m processes the job at stage s
$L_{m,T}$	time before machine m finishes the job in which it is processing at time T
$PT_{j,s,m}$	processing time of job j on machine m of stage s

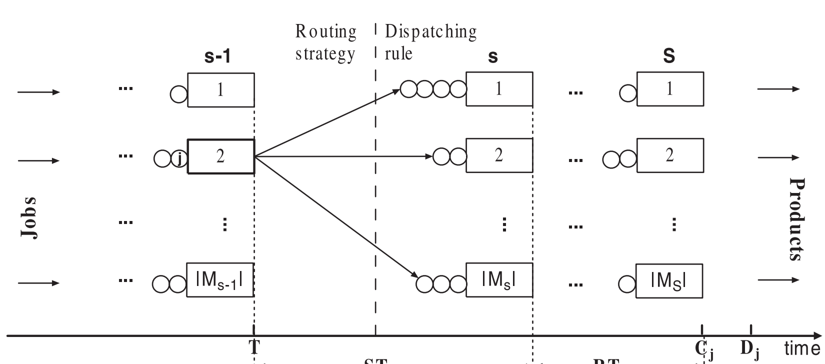

Fig. 1. The production environment of HFS and illustration of notation of time.