Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

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

Parallel machine scheduling with splitting jobs by a hybrid differential evolution algorithm

Wan-Liang Wang [a], Hai-Yan Wang [b,*], Yan-Wei Zhao [c], Li-Ping Zhang [c], Xin-Li Xu [a]

[a] College of computer science and technology, Zhejiang University of Technology, Hangzhou 310014, PR China
[b] Mechanical & Electrical Engineering College, Jiaxing University, Jiaxing 314000, PR China
[c] Key Laboratory of Special Purpose Equipment and Advanced Processing Technology of Ministry of Education, Zhejiang University of Technology, Hangzhou 310014, PR China

ARTICLE INFO

Available online 22 December 2012

Keywords:
Parallel machine scheduling
Job splitting
Differential evolution
Local search

ABSTRACT

The problem of parallel machine scheduling for minimizing the makespan is an open scheduling problem with extensive practical relevance. It has been proved to be non-deterministic polynomial hard. Considering a job's batch size greater than one in the real manufacturing environment, this paper investigates into the parallel machine scheduling with splitting jobs. Differential evolution is employed as a solution approach due to its distinctive feature, and a new crossover method and a new mutation method are brought forward in the global search procedure, according to the job splitting constraint. A specific local search method is further designed to gain a better performance, based on the analytical result from the single product problem. Numerical experiments on the performance of the proposed hybrid DE on parallel machine scheduling problems with splitting jobs covering identical and unrelated machine kinds and a realistic problem are performed, and the results indicate that the algorithm is feasible and efficient.

1. Introduction

As an important branch of machine scheduling problem, parallel machine scheduling (PMS) problem is a typical scheduling problem with extensive practical relevance [1,2]. It focuses on the problem of scheduling N jobs on M parallel machines. When the processing time of each job is the same on all those M machines, the problem is said to be identical PMS problem. Besides, there are two other types of the problem: uniform PMS problem and unrelated PMS problem.

The solution approach to the PMS problem mainly includes heuristic algorithm [3] and intelligent optimization method (i.e., tabu search, genetic algorithm, etc.). And the genetic algorithm has been widely studied to solve the problem. Fu [4], Liu and Wu [5] applied genetic algorithm to the identical PMS problem with the objective of minimizing the makespan, and the result shows that genetic algorithm performs better than heuristic algorithm. Cheng and Gen [6], Liu and Wu [7], and Zhao et al. [8] adopt genetic algorithm to the identical PMS problem with earliness and tardiness penalties. These researches focus on identical PMS problem. While in a real manufacturing environment, unrelated PMS problem is more widespread. Liu et al. [9], Yin et al. [10], Tavakkoli-Moghaddam et al. [11], Vallada and Ruiz [12], Liu and Wang [13] studied unrelated PMS problem, and genetic algorithm and particle swarm optimization were applied to the problem, respectively. Huang and Guo [14] put forward a hybrid genetic algorithm for the multi-objective unrelated PMS problem.

A job cannot be split in the traditional PMS problem studied in the above researches, while under the batch production mode in a real manufacturing environment, a job may consist of a batch of identical units in the production scheduling problem. And by splitting the original job and distributing to different machines, a faster completion can be obtained. The problem is called the PMS problem with splitting jobs [15]. There are very few researches on this kind of problem at present.

Xing and Zhang [15] presented a heuristic to solve the identical PMS problem with splitting jobs to minimize the makespan. Serafini [16] studied job splitting on the identical parallel machines in a textile industry to minimize the maximum weighted tardiness. Kim et al. [17] proposed a two-phase heuristic algorithm for the identical PMS problem with splitting jobs to minimize total tardiness: In the first phase, an initial sequence is constructed through an existing heuristic method in Ref. [18]; In the second phase, each job is split and distributed until the completion time cannot be reduced any more. Shim and Kim [19] put forward a branch and bound algorithm for the identical PMS problem with splitting jobs based on the objective of minimizing total tardiness, using a two-phase heuristic algorithm in Ref. [17] to get an initial

* Corresponding author. Tel.: +86 573 8364 7307; fax: +86 573 8364 2257.
E-mail address: wanghy@mail.zjxu.edu.cn (H.-Y. Wang).

0305-0548/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cor.2012.12.007

upper bound. Logendran and Subur [20] investigated an unrelated PMS problem with splitting jobs where jobs are split beforehand, and a tabu-search-based heuristic algorithm is developed for minimizing the total weighted tardiness. Although these existing researches either limit to the problem with identical parallel machines or with job splitting predetermined, their achievements still provide an important basis for our further study.

This paper studies the PMS problem with splitting jobs, covering identical and unrelated machine kinds. Considering the constraint that the total number of units contained in a job should remain the same before and after job splitting, differential evolution (DE) [21–22] is adopted as a solution approach, based on our finding that the sum of values of genes in an individual remains the same before and after mutation [23–24]. In spite of DE's successful application in production scheduling [25–28], this is the first time that DE is applied to the PMS problem, taking full advantage of its distinctive feature.

2. Problem description and model formulation

There are N jobs in set $JB=\{J_i|1\le i\le N\}$ to be processed on M parallel machines, denoted by set $EC=\{E_l|1\le l\le M\}$. Each job J_i consists of a batch of identical units, and BS_i denotes the number of units in the job (the original size of the job). Let T_{il} be the unit processing time of job J_i on machine E_l, where $i=1,2,\ldots,N$ and $l=1,2,\ldots,M$. The machines are said to be identical when $T_{i1}=T_{i2}=\cdots=T_{iM}$ for each job J_i. While in a more general case, machines are unrelated, meaning that the unit processing time of a job does not have to be the same on all those M machines.

Traditional PMS problems are under the hypothesis that each job can be processed on any one of the M machines. However, in a real manufacturing environment, not all of the M machines are capable of processing each job. Let $EQ_i=\{EQ_{ij}|1\le j\le EB_i\}$ be the set of available machines for job J_i. Apparently, $EQ_i\subseteq EC$ and $EB_i\le M$, where $i=1,2,\cdots,N$.

The objective of the problem is to minimize the makespan. A job is allowed to be split and distributed to different machines to obtain a faster completion. In our problem, the completion time of a machine depends on the number of units from each job distributed to the machine rather than the processing sequence. Thereby, the number of units from J_i distributed to machine E_l, denoted by BN_{il}, are to be determined, $i=1,2,\ldots,N$, $l=1,2,\ldots,M$. The problem is formulated as follows.

$$min\,Z=C_{max} \tag{1}$$

where C_{max} stands for the maximum completion time

$$C_{max}=\max_{l=1}^{M}\sum_{i=1}^{N}(BN_{il}\times T_{il}) \tag{2}$$

$$\sum_{l=1}^{M}BN_{il}=BS_i,\ BN_{il}\text{ is an integer} \tag{3}$$

Eq. (1) specifies the objective to minimize the makespan, defined by the maximum completion time among all machines. Eq. (2) provides the calculation method for the makespan. Eq. (3) requires the total number of units from a job distributed to the M machines should remain the same as the original size of the job. We call it the job splitting constraint.

3. Analysis for the problem with $N=1$

Consider a simple problem where $N=1$, $M=2$, the original size $BS_1=10$, and the unit processing time $T_{11}=T_{12}=2$. It is obvious that when the job is split equally and distributed to the two machines (that is, $BN_{11}=BN_{12}=5$), the optimal makespan can be obtained. In that case, machine E_1 and machine E_2 complete at the same time, and $C_{max}=10$.

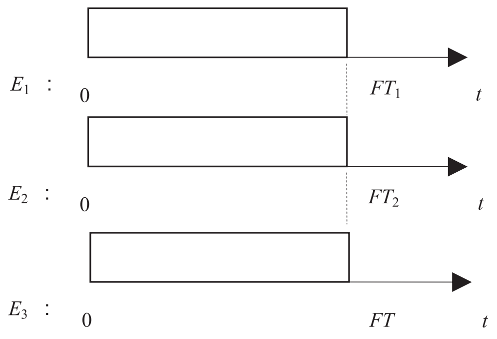

Fig. 1. Optimal schedule for the problem with $N=1$.

In fact, we can conclude that for the $N=1$ problem, when the job is split and distributed in such a way that makes each machine completes at the same time as shown in Fig. 1, the optimal makespan can be obtained. However, considering the job splitting constraint and the constraint that BN_{il} should be integer numbers, machines may not always complete at the same time. For example, when $N=1$, $M=3$, $BS_1=10$ and $T_{11}=T_{12}=T_{13}=2$, the optimal schedule occurs when we distribute 4 units to one machine and 3 units to each of the other two machines. In this case, the three machines complete at time 6, 6 and 8, respectively. And the optimal makespan is $C_{max}=8$.

On the basis of the above analysis, a job splitting and distributing method for the $N=1$ problem with makespan criterion is developed in the following steps.

Step 1. BN_{il} ($l=1,2,\ldots,M$) are calculated according to Eq. (4), where [] means to get the nearest integer.

$$BN_{il}=\left[\frac{1/T_{il}}{\sum_{l'=1}^{M}\left(1/T_{il'}\right)}\times BS_i\right],\ i=1, \tag{4}$$

In Eq. (4), the job is split and distributed to machines in proportion to their unit processing speeds. However, there may be some units left due to the rounding function in Eq. (4), which means there may exist $\sum_{l=1}^{M}BN_{il}<BS_i$.

Step 2. Execute step 4 if $\left(BS_i-\sum_{l=1}^{M}BN_{il}\right)=0$, meaning that all the units in the job are distributed. Otherwise, execute step 3 if $\left(BS_i-\sum_{l=1}^{M}BN_{il}\right)\ge 1$.

Step 3. Denote FT_l as the completion time of machine E_l. And we can get that $FT_l=(BN_{il}+1)\times T_{il}$ $(l=1,2,\cdots,M)$ when we distribute one more unit to each machine. Execute $BN_{il}=BN_{il}+1$ for the former $\left(BS_i-\sum_{l=1}^{M}BN_{il}\right)$ machines in the ascending order of their completion time.

Step 4. So far, all the units in the job have been distributed. Recalculate the completion time for each machine by carrying out $FT_l=BN_{il}\times T_{il}$, where $i=1$ and $l=1,2,\ldots,M$. And the maximum completion time $C_{max}=\max_{l=1}^{M}\{FT_l\}$.

4. New hybrid differential evolution

For the PMS problem with splitting jobs when $N>1$, each job J_i is to be split and distributed to machines in EQ_i, where