Particle swarm optimization with cocktail decoding method for hybrid flow shop scheduling problems with multiprocessor tasks

Fuh-Der Chou*

Department of Industrial Management, Ching Yun University, Jung-Li, Tao Yuan, Taiwan, ROC

**Abstract**

This paper addresses the problem of multiprocessor task-scheduling in a hybrid flow shop (HFS) problem to minimize the makespan. Due to the complex nature of an HFS problem, it is decomposed into the following two sequential decision problems: determining the job permutation in stage 1, followed by a decoding method to assign jobs into each machine in subsequent stages when designing a heuristic algorithm. The decoding method plays a pivotal role for improving the solution quality of any algorithm for the HFS problem. However, the majority of existing algorithms ignores the problem and is only concerned with the first decision problem. This study emphasizes the importance of the decoding method via a small test, and searches for a number of solid decoding methods that can be incorporated into the cocktail decoding method. Then, this study develops a particle swarm optimization (PSO) algorithm that can be combined with the cocktail decoding method. In the PSO, a variety of job sequences are generated using the PSO procedure in stage 1, and the cocktail decoding method is used to assign the jobs to machines in sequential stages. Moreover, a modified lower bound is introduced. Computational results show that the proposed lower bound is competitive, and with the help of the cocktail decoding method, the proposed PSO, and even the adoption of a standard PSO framework, significantly outperforms the majority of existing algorithms in terms of quality of solutions, especially for large problems.

© 2012 Elsevier B.V. All rights reserved.

1. Introduction

Hybrid flow shop (HFS) problems have recently attracted considerable attention from researchers. HFS is an extension model of the flow shop system that relaxes the assumption of the one-stage-one-machine pattern based on the entire issue that machines operate in parallel in at least one stage to prevent the whole system from being blocked (e.g., a breakdown) by a single machine in many industrial environments. Thus, the HFS problem has two basic characteristics as follows: (1) a set of \( n \) jobs is sequentially processed in a series of \( m \) stages; and (2) at least one of the stages has \( m_i \) machines in parallel. Rao (1970) conducted the early work on HFS scheduling, and variants of HFS problems with different constraints and criteria have been studied by many researchers since then. Gupta (1988) considered the two-stage special case with a single machine in the first stage and two identical machines in the second stage, and showed that the case was NP-hard. Brah and Hunsucker (1991) proposed a branch and bound (B&B) algorithm to minimize the makespan for HFS problems. Portmann et al. (1998) applied a genetic algorithm (GA) within a B&B algorithm to further improve the performance of the B&B algorithm. With the computational complexity of HFS problems, a number of heuristics and metaheuristic algorithms have also been developed to obtain good enough solutions in a short time for medium-to-large problems (Haouari and M’Hallah, 1997; Brah and Loo, 1999; Lin and Liao, 2003; Jin et al., 2006; Janiak et al., 2007; Alaykiran et al., 2007). The studies mentioned above focused on the model of the one-job-one-machine pattern, in which each job is processed on only one machine at a given time. Ruiz and Vazquez-Rodriguez (2010) conducted a comprehensive survey of relevant research on HFS problems from 1970 to 2009, clearly revealing that among the majority of HFS problems that were concerned with the one-job-one-machine model, only a few studies discussed multiprocessor tasks where each job is processed on a number of identical machines simultaneously at each stage.

This study focuses on multiprocessor task-scheduling in a HFS system. The problem can be encountered in a number of industrial environments, such as berth allocation of container terminals, real-time machine-vision systems, and work force management. In the considered problem, a given set of \( n \) jobs, each with \( k \) tasks, is processed sequentially from stage 1 to stage \( k \). Each stage consists of \( P_m(1=1, \ldots, k) \) identical parallel machines. Each job \( j \) is characterized by \( p_j \) and \( \text{size}_j \), indicating that the tasks of job \( j \) are processed by
size$_i$ parallel machines simultaneously for $p_k$ time period without interruption at stage $i$. Any machine can only process one task at a time. All jobs and machines are available at time zero, setup times are negligible, and preemption is not allowed. The problem aims to minimize the maximum completion time of all jobs, i.e., the makespan. Using the standard three-field notation, the problem can be described as follows: $F_k(P_{m_1}, \ldots, P_{m_k})/size_{ij}$ \(C_{\text{max}}\).

In literature, several heuristic algorithms have been developed to solve multiprocessor task HFS problems. Oğuz and Erkan (1997) and Oğuz et al. (2003) examined the multiprocessor task in a two-stage HFS problem motivated by a computer vision system. The two studies presented different constructive heuristic algorithms based on several dispatching rules wherein Oğuz and Erkan assumed a unit-processing time for all jobs. Ying and Lin (2009) developed the first simple constructive heuristic algorithm with a computational complexity of $O(n^2)$ to solve a general $F_k(P_{m_1}, \ldots, P_{m_k})/size_{ij}$ \(C_{\text{max}}\) problem. Karaman et al. (2010) developed a parallel greedy algorithm, which is an effective stochastic local search algorithm that is easy to implement and has great potential to provide good solutions, for the HFS problem with multiprocessor tasks. Their results are competitive compared with the results of Oğuz and Erkan (2005). Lahimer et al. (2011) developed the climbing depth-bounded adjacent discrepancy search (CDADS) based on a limited discrepancy search, which was effective for both small and large problems.

Recently, metaheuristic algorithms have been developed for multiprocessor task-scheduling in an HFS system, including simulated annealing (SA) by Wang et al. (2011), tabu search (TS) by Oğuz et al. (2004), GA (Sivrikaya-Serifoglu and Ulusoy, 2004; Oğuz and Erkan, 2005) particle swarm optimization (PSO) algorithm by Tseng and Liao (2008), and ant colony optimization (ACO) by Ying and Lin (2006). These studies decomposed the problem into two decision problems, namely, the determination of job permutation in stage 1 and the assignment of jobs to each machine in the subsequent stages. For two sub-problems, the metaheuristic algorithms mentioned above usually follow a two-phase structure in which different permutations of the jobs are generated in the first phase, and then a decoding method is used to assign the jobs to machines for the remaining stages according to the first come, first served (FCFS) manner (i.e., list scheduling). Despite the relative popularity of list scheduling for assigning the unscheduled jobs to machines in the subsequent stages decoding method. Wang et al. (2011) stated that the decoding method exhibits certain influences on the solution quality for the considered problem. Moreover, priority rules are popular for assigning the unscheduled jobs to machines in the scheduling area, and thus, a variety of popular priority rules are adopted as decoding methods, which are described as follows:

- **Shortest processing time (SPT)**, where the jobs are sorted by the ascending order of their processing times $p_k$ for stage $k$.
- **Longest processing time (LPT)**, where the jobs are sorted by the descending order of their processing times $p_k$ for stage $k$.
- **Shortest total remaining processing time (STRPT)**, where the jobs are sorted by the ascending order of their total processing times $\sum_{j=1}^{m} p_{ij}$ from stage $k$ to last stage $m$.
- **LTPRT** (largest total remaining processing time) where the jobs are sorted by the descending order of their total processing times $\sum_{j=1}^{m} p_{ij}$ from stage $k$ to last stage $m$.
- **Smallest processor requirement (SPR)**, where the jobs are sorted by the ascending order of their processor requirements $size_{ij}$ for stage $k$.
- **Largest processor requirement (LPR)**, where the jobs are sorted by the descending order of their processor requirements $size_{ij}$ for stage $k$.
- **Smallest occupied capacity (SOC)**, where the jobs are sorted by the ascending order of their processing time multiplied by processor requirement $p_k \times size_{ij}$ for stage $k$.
- **Largest occupied capacity (LOC)**, where the jobs are sorted by the descending order of their processing time multiplied by processor requirement $p_k \times size_{ij}$ for stage $k$.
- **Smallest total remaining occupied capacity (STROC)**, where the jobs are sorted by the descending order of a total of their processing time multiplied by processor requirement $\sum_{i=1}^{m} p_{ij} \times size_{ij}$ from stage $k$ to the last stage $m$.
- **Largest total remaining occupied capacity (LTROC)**, where the jobs are sorted by the descending order of a total of their processing time multiplied by processor requirement $\sum_{i=1}^{m} p_{ij} \times size_{ij}$ from stage $k$ to the last stage $m$.

2. **Cocktail decoding method**

As mentioned above, the performance of a heuristic algorithm in solving the HFS problem with multiprocessor tasks is influenced by two factors, namely, the determination of job sequence at stage 1 and the assignment of jobs to each machine for the subsequent stages. Jouglet et al. (2009) named the method to assign unscheduled jobs on each machine for the subsequent stages decoding method. Wang et al. (2011) stated that the decoding method exhibits certain influences on the solution quality for the considered problem. Moreover, priority rules are popular for assigning the unscheduled jobs to machines in the scheduling area, and thus, a variety of popular priority rules are adopted as decoding methods, which are described as follows:

- **Shortest processing time (SPT)**, where the jobs are sorted by the ascending order of their processing times $p_k$ for stage $k$.
- **Longest processing time (LPT)**, where the jobs are sorted by the descending order of their processing times $p_k$ for stage $k$.
- **Shortest total remaining processing time (STRPT)**, where the jobs are sorted by the ascending order of their total processing times $\sum_{j=1}^{m} p_{ij}$ from stage $k$ to last stage $m$.
- **LTPRT** (largest total remaining processing time) where the jobs are sorted by the descending order of their total processing times $\sum_{j=1}^{m} p_{ij}$ from stage $k$ to last stage $m$.
- **Smallest processor requirement (SPR)**, where the jobs are sorted by the ascending order of their processor requirements $size_{ij}$ for stage $k$.
- **Largest processor requirement (LPR)**, where the jobs are sorted by the descending order of their processor requirements $size_{ij}$ for stage $k$.
- **Smallest occupied capacity (SOC)**, where the jobs are sorted by the ascending order of their processing time multiplied by processor requirement $p_k \times size_{ij}$ for stage $k$.
- **Largest occupied capacity (LOC)**, where the jobs are sorted by the descending order of their processing time multiplied by processor requirement $p_k \times size_{ij}$ for stage $k$.
- **Smallest total remaining occupied capacity (STROC)**, where the jobs are sorted by the descending order of a total of their processing time multiplied by processor requirement $\sum_{i=1}^{m} p_{ij} \times size_{ij}$ from stage $k$ to the last stage $m$.
- **Largest total remaining occupied capacity (LTROC)**, where the jobs are sorted by the descending order of a total of their
دریافت فوری
متن کامل مقاله
امکان دانلود نسخه تمام متن مقالات انگلیسی
امکان دانلود نسخه ترجمه شده مقالات
پذیرش سفارش ترجمه تخصصی
امکان جستجو در آرشیو جامعی از صدها موضوع و هزاران مقاله
امکان دانلود رایگان ۲ صفحه اول هر مقاله
امکان پرداخت اینترنتی با کلیه کارت های عضو شتاب
دانلود فوری مقاله پس از پرداخت آنلاین
پشتیبانی کامل خرید با بهره مندی از سیستم هوشمند رهگیری سفارشات