A backtracking search hyper-heuristic for the distributed assembly flow-shop scheduling problem

Jian Lin\textsuperscript{a,}*, Zhou-Jing Wang\textsuperscript{a}, Xiaodong Li\textsuperscript{b}

\textsuperscript{a} School of Information, Zhejiang University of Finance & Economics, Hangzhou 310018, China
\textsuperscript{b} School of Science (Computer Science and IT), RMIT University, Melbourne, VIC 3001, Australia

\textbf{ARTICLE INFO}

Keywords:
Hyper-heuristic
Backtracking search algorithm
Distributed assembly
Flow-shop scheduling

\textbf{ABSTRACT}

Distributed assembly permutation flow-shop scheduling problem (DAPFSP) is recognized as an important class of problems in modern supply chains and manufacturing systems. In this paper, a backtracking search hyper-heuristic (BS-HH) algorithm is proposed to solve the DAPFSP. In the BS-HH scheme, ten simple and effective heuristic rules are designed to construct a set of low-level heuristics (LLHs), and the backtracking search algorithm is employed as the high-level strategy to manipulate the LLHs to operate on the solution space. Additionally, an efficient solution encoding and decoding scheme is proposed to generate a feasible schedule. The effectiveness of the BS-HH is evaluated on two typical benchmark sets and the computational results indicate the superiority of the proposed BS-HH scheme over the state-of-the-art algorithms.

1. Introduction

Production scheduling has been a very active research area because of its practical significance in decision-making of manufacturing systems [1–4]. As one of the most studied scheduling problems, the permutation flow-shop scheduling problem (PFSP) is an extensively investigated combinatorial optimization problem in manufacturing systems and industrial processes. The PFSP with the makespan criterion has been proven to be NP-hard when the number of machines is no less than three [5]. Following the pioneering work of Johnson [6], many approaches have been proposed to solve the PFSP [7–18]. A common assumption among these studies is that there is only a single production center, or factory, and all jobs in the permutation are assigned to the same factory. However, production systems with more than one production center (namely, a distributed manufacturing system) is more common in practice [19–23], since it can achieve higher product quality while reducing production distribution costs and management risks [24]. Scheduling in distributed systems is more challenging than in regular shop scheduling problems; in particular, job allocation to factories and job scheduling at each factory must be both considered when making decisions.

Recently, an extension of the regular PFSP called the distributed assembly permutation flow-shop scheduling problem (DAPFSP) was introduced by Hatami et al. [25], where a set of products and a set of factories are assembled in a single assembly factory with an assembly machine. Hatami et al. [25] also considered the minimization of makespan at the assembly factory and presented 14 heuristics based on constructive heuristics and variable neighborhood descent (VND). In [26], an estimation of distribution algorithm based memetic algorithm (EDAMA) was developed for solving the DAPFSP with the objective to minimize the maximum completion time. In our previous work [27], an effective hybrid biogeography-based optimization (HBBO) algorithm that integrates several novel heuristics is proposed to solve the DAPFSP.

A recent trend in search and optimization suggests that hyper-heuristic has emerged as an effective search methodology that controls other heuristics to provide near-optimal solutions for various problems [28,29]. Instead of searching directly in the solution space, hyper-heuristics operate on a set of low-level heuristics (LLHs), and attempt to find an optimal sequence of heuristics [30]. During the past few years, there is a growing literature in the field of hyper-heuristics [28]. In particular, meta-heuristics have been used to construct hyper-heuristic schemes, e.g., a particle swarm optimization based hyper-heuristic approach by Koulinas et al. [31], evolutionary hyper-heuristics by Sanz et al. [32] and Moreno et al. [33], a harmony search based hyper-heuristic by Anwar et al. [34], and a bacterial foraging based hyper-heuristic by Rajini and Chana [35]. However, to the best of our knowledge, there is no hyper-heuristic approach for solving the DAPFSP.

The motivation behind this paper is to propose a hyper-heuristic
based scheduling algorithm which would be applicable in solving the DAPFSP. The backtracking search optimization algorithm (BSA) [36] is a newly developed powerful evolutionary algorithm, which has been proved to be very promising when compared with other evolutionary algorithms (EAs) [36–40]. Especially, BSA is a dual-population algorithm that uses as well as the current historical populations, and also has a simple structure. This paper aims at employing an effective backtracking search hyper-heuristic (BS-HH) algorithm to solve the DAPFSP with the objective of minimizing the makespan value. In BS-HH, the BSA is used as the high-level hyper-heuristic strategy, which manages solution methods rather than solutions, and employs a set of designed LLHs. Experiments and comparisons are conducted on two sets of benchmarks provided in Hatami et al. [25] to verify the effectiveness of the proposed scheme.

The rest of the paper is organized as follows. In Section 2, the DAPFSP is briefly introduced. In Section 3, the BS-HH scheme is proposed for the DAPFSP. The computational results on benchmark instances together with comparison to some state-of-the-art algorithms are presented in Section 4. Finally, a conclusion is drawn in Section 5.

### 2. Distributed assembly permutation flow-shop scheduling problem

As illustrated in Fig. 1, DAPFSP [25,27] is a combination of the distributed PFSP and the assembly flow-shop scheduling problem, which consists of two stages: production and assembly, and can be generalized into three sub-problems: job scheduling, product scheduling and factory assignment. The notations used in the optimization model for the DAPFSP are presented in Table 1.

In the production stage, there are \( n \) jobs \( \{J_1, J_2, ..., J_n\} \) to be processed in \( F \) identical factories. All factories are capable of processing all jobs, and each factory can be considered as a PFSP with \( m \) machines \( \{M_1, M_2, ..., M_m\} \). Each job \( J_i \) requires a sequence of operations \( \{O_{i1}, O_{i2}, ..., O_{ih}\} \) to be processed one after another on \( m \) machines. In the assembly stage, there is an assembly factory with a single assembly machine \( M_A \) which assembles all jobs into \( H \) different products \( \{P_1, P_2, ..., P_H\} \). Each product \( P_k \) has \( N_k \) jobs, with these jobs first processed in the production stage before assembling into the product \( P_k \); hence \( \sum_{k=1}^{H} N_k = n \). In this paper, the maximum completion time (makespan) at the assembly factory is the objective to minimize.

Let \( \pi_k = [\pi_{k1}, \pi_{k2}, ..., \pi_{kh}] \) be the sequence of jobs in factory \( f(f = 1, ..., F) \) that belong to product \( P_k \), where \( \pi_{kj} (\pi_{kj} < N_k) \) is the total number of jobs in product \( P_k \) assigned to factory \( f \). \( C_{M_A} \) and \( C_{M} \) denote the completion time of product \( P_k \) on assembly machine \( M_A \) and the operation \( O_h \) on machine \( M_f \), respectively. For a schedule \( \Lambda \) of the DAPFSP, i.e., a set of sequences \( \{\pi_1', \pi_2', ..., \pi_F'\} \), the makespan \( C_{max}(\Lambda) \) is given by:

\[
C_{\pi_1'(1),1} = p_{\pi_1'(1),1} + \sum_{h=2}^{L} p_{\pi_1'(1),h}, \quad f = 1, 2, ..., F; \quad h = 1, 2, ..., H, \quad (1)
\]

\[
C_{\pi_{k-1}'(k),1} = C_{\pi_{k-1}'(k-1),1} + \sum_{h=2}^{L} p_{\pi_{k-1}'(k),h}, \quad f = 1, 2, ..., F; \quad k = 1, 2, ..., N_k; \quad h = 1, 2, ..., H, \quad (2)
\]

\[
C_{\pi_{k+1}'(k+1),1} = C_{\pi_{k+1}'(k-1),1} + \sum_{h=2}^{L} p_{\pi_{k+1}'(k+1),h}, \quad f = 1, 2, ..., F; \quad k = 1, 2, ..., m; \quad h = 1, 2, ..., H, \quad (3)
\]

\[
C_{\pi_{F}'(F),1} = \max\left\{C_{\pi_{F-1}'(F-1),1}, C_{\pi_{F}'(F-1),1}\right\}, \quad f = 1, 2, ..., F; \quad k = 2, ..., m; \quad j = 1, 2, ..., m; \quad h = 1, 2, ..., H. \quad (4)
\]

<table>
<thead>
<tr>
<th>Table 1</th>
<th>The notations used in the optimization model for the DAPFSP.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Indices</strong></td>
<td></td>
</tr>
<tr>
<td>( i )</td>
<td>Index for jobs where ( i = 1, ..., n )</td>
</tr>
<tr>
<td>( f )</td>
<td>Index for machines where ( f = 1, ..., m )</td>
</tr>
<tr>
<td>( h )</td>
<td>Index for products where ( h = 1, ..., H )</td>
</tr>
<tr>
<td>( k )</td>
<td>Index for factories where ( k = 1, 2, ..., F )</td>
</tr>
</tbody>
</table>

![Fig. 1. Illustration of the DAPFSP.](Image)
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