Computers & Industrial Engineering 125 (2018) 423–433

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

# Memetic social spider optimization algorithm for scheduling two-stage assembly flowshop in a distributed environment

Guanghui Zhang[a,b], Keyi Xing[a,1,*]

[a] State Key Laboratory for Manufacturing Systems Engineering, and Systems Engineering Institute, Xi'an Jiaotong University, Xi'an 710049, PR China
[b] School of Sciences, Guizhou Institute of Technology, Guiyang 550003, PR China

ARTICLE INFO

*Keywords:*
Assembly flowshop
Distributed scheduling
Memetic algorithm
Separate setup time
Social spider optimization

ABSTRACT

This paper studies the distributed two-stage assembly flowshop problem with separate setup times, which is a generalisation for the regular two-stage assembly flowshop problem in the distributed manufacturing environment. The optimization objective is to find a suitable job schedule such that the criterion of total completion time is minimized. To deal with such a problem, we propose a novel memetic algorithm (MA) based on a recently developed social spider optimization (SSO). To the best of our knowledge, it is the first effort to explore the SSO-based MA (MSSO) and to apply SSO in the field of combinational optimization. In the proposed MSSO algorithm, we first modify the original version of SSO to adapt to the distributed problems, and then integrate two improvement techniques, problem-special local search and self-adaptive restart strategy, within MA framework. In the numerical experiment, the parameters used in MSSO are calibrated and suitable parameter values are suggested based on the Taguchi method. Experimental results and comparisons with the existing algorithms validate the effectiveness and efficiency of the proposed MSSO for addressing the considered problem. In addition, the effect of problem scale parameters on MSSO and the effectiveness of the proposed improvement techniques are also investigated and demonstrated.

## 1. Introduction

The two-stage assembly flowshop scheduling problem (TSAFSP), as a generalisation of the regular flowshop problem, is more practical and has become one of the most investigated production scheduling problems over the past few decades in engineering and operational research due to both the theoretical significance and wide application in practice (Allahverdi & Al-Anzi, 2006a, 2006b; Kazemi, Mazdeh, & Rostami, 2017; Lee, Cheng, & Lin, 1993; Lin & Liao, 2003; Mozdgir, Fatemi Ghomi, Jolai, & Navaei, 2013; Navaei, Fatemi Ghomi, Jolai, Shiraqai, & Hidaji, 2013; Potts, Sevast'janov, Strusevich, van Wassenhove, & Zwaneveld, 1995).

In such an assembly-type flowshop, there is a finite set $N$ of $n$ jobs to be scheduled. Each job is comprised of $m$ different components that are first processed separately on $m$ machines disposed in parallel, and then are assembled into the required product by employing a single assembly machine. The optimization objective is to find a suitable processing sequence or schedule of the $n$ jobs on the considered machines such that optimizing a certain scheduling objective that is formulated in terms of the processing and assembly time metrics.

The two-stage assembly flowshop problems always make a common hypothesis: there is only one production factory in a manufacturing campaign. It suggests that all production objects have to be processed and then assembled in a single manufacturing system. Nevertheless, with the advent of just-in-time fabricating and mass production era, modern enterprises are more and more being dragged into the market in which the cutthroat competition is everywhere. To enhance the competitiveness, the reform that transforms the traditional single-centre production pattern to more flexible distributed manufacturing has become a successful way to make enterprises bigger and stronger in short time (Behnamian & Fatemi Ghomi, 2016; Moon, Kim, & Hur, 2002).

In the distributed manufacturing system, total production tasks are accomplished by means of multiple independent production units, which enable enterprises harvest many potential advantages like higher product quality, better corporate reputation and lower production period and cost (Kahn, Castellion, & Griffin, 2004). Compared with the single-factory scheduling problem, the scheduling in distributed settings is more complicated since a schedule has to address two interrelated decisions simultaneously: designating an appropriate factory for

* Corresponding author.
*E-mail addresses:* zhangguanghui@stu.xjtu.edu.cn (G. Zhang), kyxing@mail.xjtu.edu.cn (K. Xing).
[1] Postal address: The Systems Engineering Institute, Xi'an Jiaotong University, No. 28, Xianning West Road, Xi'an City, Shaanxi 710049, PR China.

https://doi.org/10.1016/j.cie.2018.09.007
Received 20 March 2018; Received in revised form 27 July 2018; Accepted 5 September 2018
Available online 07 September 2018
0360-8352/ 

each job and determining the processing order of jobs at each factory.

The distributed scheduling problems (DSPs) have already been widely focused on although explored just recently. We review the limited existing literature and find that almost all of the studies are devoted to the scheduling of typical production systems in the distributed manufacturing environment, such as flowshop (Fernandez-Viagas & Framinan, 2015; Gao & Chen, 2011; Gao, Chen, & Deng, 2013; Gao, Chen, & Liu, 2012; Lin, Ying, & Huang, 2013; Naderi & Ruiz, 2010, 2014; Wang, Wang, Liu, & Xu, 2013; Xu, Wang, Wang, & Liu, 2014; Zhang, Xing, & Cao, 2018), jobshop (De Giovanni & Pezzella, 2010; Hsu, Kao, Ho, & Lai, 2016; Jia, Fuh, Nee, & Zhang, 2007; Jia, Nee, Fuh, & Zhang, 2003; Naderi & Azab, 2014, 2015; Ziaee, 2014), and flexible manufacturing system (Chan, Chung, & Chan, 2006; Chan, Chung, Chan, Finke, & Tiwari, 2006). However, very little work involves the DSP for the TSAFSP, referred to as distributed TSAFSP or DTSAFSP in short. After the pioneer contributions of Xiong, Xing, Wang, Lei, and Han (2014) and Xiong and Xing (2014), actually just one paper (Deng, Wang, Wang, & Zheng, 2016) is published for the DTSAFSP with makespan minimization. It is evident that there exists broad headroom for research to provide more efficient and diversified solution schemes for such a hard and practical scheduling problem.

Based on a newly developed social spider optimization (SSO) (Cuevas, Cienfuegos, Zaldívar, & Pérez-Cisneros, 2013), this paper proposes a memetic algorithm (MA), referred to as memetic social spider optimization (MSSO) algorithm, for the DTSAFSP with the objective of minimizing total completion time. To the best of our knowledge, it is the first attempt to apply the SSO in the combinational optimization, and it is also the first exploration for the SSO-based MA or MSSO. In the proposed MSSO, we first improve the original SSO for tackling the scheduling problem under consideration. Then, the MSSO is developed by integrating some advanced improvement strategies such as problem-special local search and self-adaptive restart strategy within MA framework. In this way, the global exploration and local exploitation of MSSO are expected to balance effectively. The extensive experiments and comparisons with the state-of-the-art algorithms validate the excellent performance of MSSO.

The remainder of the paper is arranged as follows. Section 2 reviews the relevant literature. Section 3 describes the details of DTSAFSP. Section 4 improves the social spider optimization algorithm. In Section 6, several problem-dependent techniques are presented to develop overall MSSO algorithm. Numerical experiments are carried out in Section 7, and finally conclusions and further research directions are summarized in Section 8.

## 2. Literature review

The first study on the TSAFSP was accomplished independently by Lee et al. (1993) and Potts et al. (1995), where they proved that the problem was NP-hard when focusing on two or more parallel processing machines. After these two pioneering works, there have been many literatures on the TSAFSP and its variants devoted to different optimization criteria. In the solving procedures, researchers (Fattahi, Hosseini, Jolai, & Tavakkoli-Moghaddam, 2014; Haouari & Daouas, 1999; Hariri & Potts, 1997; Lee, 2018; Lee et al., 1993; Sung & Juhn, 2009; Sung & Kim, 2008; Tozkapan, Kırca, & Chung, 2003) established the exact methods, mainly branch and bound algorithm, to obtain the problem's optimal schedule. However, most exact algorithms are time-consuming when solving the TSAFSP with practical scale. Therefore, more and more researchers are turning to the exploration on heuristics, metaheuristics or their hybrids for a suboptimal schedule with reasonable time.

Allahverdi and Al-Anzi considered the TSAFSP with different objective functions like makespan (Allahverdi & Al-Anzi, 2006a), maximum lateness (Allahverdi & Al-Anzi, 2006b), total completion time (Allahverdi & Al-Anzi, 2009), and a combination of makespan and mean completion time (Allahverdi & Al-Anzi, 2008). In these papers, the authors developed multiple metaheuristics: the tabu search, simulated annealing (SA), differential evolution (DE), particle swarm optimization (PSO), and ant colony optimization. To address the same problem in literature (Allahverdi & Al-Anzi, 2008), Torabzadeh and Zandieh (2010) proposed a cloud theory-based SA, Shokrollahpour, Zandieh, and Dorri (2011) presented an imperialist competitive algorithm (ICA), and Tian, Liu, Yuan, and Wang (2013) established a discrete PSO. Both Mozdgir et al. (2013) and Navaei et al. (2013) considered to extend the TSAFSP to more than one assembly machines. They proposed two hybrid algorithms based on variable neighborhood search (VNS) and SA, respectively. Recently, aiming to minimize the objective of total tardiness for the TSAFSP with setup times, Allahverdi, Aydilek, and Aydilek (2016) proposed six new algorithms by adapting the SA, genetic algorithm (GA), and insertion algorithm. Kazemi et al. (2017) addressed the TSAFSP with an additional batched delivery system. In order to optimize the sum of tardiness plus delivery costs, they developed a hybrid algorithm that is a combination of ICA and some dominance relations. Jung, Woo, and Kim (2017) studied the TSAFSP by a GA, which processed the products with dynamic component-sizes.

The study on the DTSAFSP is very scant up to present and mainly focuses on the heuristic method. To optimize both makespan and mean completion time, Xiong and Xing (2014) presented a VNS and a hybrid GA combined with reduced VNS to be designed for local exploitation. To minimize the total completion time, Xiong et al. (2014) proposed three hybrid metaheuristics by integrating VNS, GA and discrete DE with a simple local search. Deng et al (2016) considered the makespan minimization criterion and presented a competitive memetic algorithm (CMA). It is noted that the partial operators in CMA is designed based on the concept of critical factory existing only for makespan. Therefore, it cannot handle the DTSAFSP with other objective functions, such as total completion time considered in this paper.

The memetic algorithms (MA) have received more and more attention due to the successful applications in continuous and discrete optimization domains (Chen, Ong, Lim, & Tan, 2011; Ong, Lim, & Chen, 2010). MA is actually a hybrid metaheuristic approach that aims to enhance the performance of evolutionary algorithms by introducing several problem-dependent local improvement procedures. To a large degree, the success of the MA depends on the rational design of hybridized components, which are expected to focus on the global and local search simultaneously. The social spider optimization (SSO) algorithm was recently developed and reported high performance when solving the continuous optimization problems. Its development is inspired by the cooperative behaviours of social spider colony. As general metaheuristics, the SSO stresses more on global exploration while its local exploitation ability is relatively poor. As a result, in this paper a novel SSO-based MA is proposed and expected to efficiently deal with the DTSAFSP.

## 3. Illustration of DTSAFSP

The illustration of the DTSAFSP is as follows: a set $J = \{J_1, J_2,\ldots, J_n\}$ of $n$ jobs are processed and assembled in a set $F = \{F_1, F_2,\ldots, F_f\}$ of $f$ identical factories. A job can be assigned to any one of the $f$ factories. Once assigned, it is prohibited for the transfer to other factories. The machine setting at each factory consists of two stages: first set $M = \{M_1, M_2,\ldots, M_m\}$ of $m$ processing machines disposed in parallel and then a single assembly machine $M_A$. The completion of job $J_j$, $j = 1, 2,\ldots, n$, requires $m + 1$ operations $O = (\{O_{j,1}, O_{j,2},\ldots, O_{j,m}\}, O_{j,A})$, where $O_{j,i}$, $i = 1, 2,\ldots, m$, is performed on $M_i$ with time $P_{j,i}$ whereas $O_{j,A}$ on $M_A$ with time $P_{j,A}$. Before starting each operation, separate setup time is considered and denoted as $S_{j,i}$ for $O_{j,i}$ and $S_{j,A}$ for $O_{j,A}$. All factories, machines and jobs can be available at time zero. At a time, each machine operates at most one job and each job is processed on at most one machine. The preemption is not permitted. The job order on each machine is same. The buffers between two machine stages have infinite