Int. J. Production Economics 141 (2013) 14–23

Contents lists available at SciVerse ScienceDirect

Int. J. Production Economics

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

A modified artificial bee colony algorithm for order acceptance in two-machine flow shops

Xiuli Wang [a,*], Xingzi Xie [a], T.C.E. Cheng [b]

[a] School of Economics and Management, Nanjing University of Science and Technology, Nanjing 210094, People's Republic of China
[b] Department of Logistics and Maritime Studies, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

ARTICLE INFO

Article history:
Received 13 October 2011
Accepted 2 June 2012
Available online 13 June 2012

Keywords:
Scheduling
Order acceptance
Artificial bee colony algorithm
Revenue

ABSTRACT

We consider a two-stage make-to-order production system characterized by limited production capacity and tight order due dates. We want to make joint decisions on order acceptance and scheduling to maximize the total net revenue. The problem is computationally intractable. In view of the fact that artificial bee colony algorithm has been shown to be an effective evolutionary algorithm to handle combinatorial optimization problems, we first conduct a pilot study of applying the basic artificial bee colony algorithm to treat our problem. Based on the results of the pilot study and the problem characteristics, we develop a modified artificial bee colony algorithm. The experimental results show that the modified artificial bee colony algorithm is able to generate good solutions for large-scale problem instances.

1. Introduction

In many industries product requirements for customers are customized and unique. As a result, firms in such industries often adopt the make-to-order approach to production. Given tight delivery requirements and limits on production capacity, both order acceptance decision and production scheduling decision need to be taken into account. Selecting the right orders to accept depends on the strategic direction of the firm and many other considerations. From a problem-oriented perspective, order acceptance should go along with a careful analysis of capacity utilization so as to maximize the profit to the firm.

In this paper we consider the problem in a two-stage production environment. Each order has distinct product characteristics and is thus described as a job with different processing times in stages 1 and 2. The model is motivated by many industries where the process to produce products typically comprises two consecutive stages, e.g., a processing stage followed by a testing stage. An example is a manufacturer of equipment products that produces large special-purpose pressure vessels. Each order typically includes only one equipment product with distinct characteristics in terms of material, size, and shape, technological process standards, pressure performance index, and so on. It is common that processing a product is time-consuming at one stage but not at the other, i.e., the manufacturing bottleneck stage is not static but depends on all the processed orders. So scheduling is an important issue. On the other hand, any delay in delivering an order beyond its due date may incur a penalty cost to the firm. Operating in such an environment, the firm faces the problem of order acceptance and scheduling in a two-machine flowshop to maximize the total net revenue.

The research on taking order acceptance decisions and scheduling decisions into account at the same time has received increasing attention in recent years. The study results mainly consider order acceptance and production in a single machine environment with various settings. Slotnick and Morton (1996) and Ghosh (1997) are regarded as pioneers in studying the order acceptance and scheduling problem. They consider the order acceptance and scheduling decisions at the same time so as to maximize the total revenue. Lewis and Slotnick (2002) extend the problem to multiple periods for the case where rejecting an order of a customer will lead to the loss of all the future orders from that customer. In recent years, research on this topic is further extended to studying problems with different objectives and in various settings. In terms of the solution approaches used to tackle the problem, Slotnick and Morton (2007), Oğuz et al. (2010), and Nobibon and Leus (2011) develop myopic heuristics and exact approaches such as branch-and-bound algorithm, dynamic programming, mixed integer linear programming formulation, and so on. However, the exact algorithms only can solve the small-scale problem since these problems are NP-hard. Recently, Rom and Slotnick (2009) and Cesaret et al. (2012) develop meta-heuristics that apply the techniques of computational intelligence to tackle the problem. The former team

* Corresponding author. Tel./fax: +86 025 84261056.
E-mail address: wangdu0816@163.com (X. Wang).

0925-5273/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.ijpe.2012.06.003

proposes genetic algorithms and the latter team designs a tabu search algorithm. Both teams show that their meta-heuristics outperform the myopic heuristics and can obtain optimal or near-optimal solutions for large-scale problem instances. More details on this topic can be found in the recent reviews by Keskinocak and Tayur (2004), and Slotnick (2011).

A special case of our problem where all the orders are accepted and processed is the two-machine flowshop scheduling problem to minimize the total weighted tardiness. This case is NP-hard in strong sense (see, e.g., Pinedo (2002)). For the case of the total tardiness problem in m-machine flowshops, Onwubolu and Mutingi (1999) propose a genetic algorithm. For two-machine flowshops, Schaller (2005) propose a branch-and-bound algorithm to solve small-size problem instances optimally. For the case of the total weighted tardiness problem in m-machine flowshops, Parthasarathy and Rajendran (1998), and Rajendran and Ziegler (2003) propose simulated annealing algorithms and improving heuristics, respectively. Detail research on the (weighted) tardiness problem in flowshops can be found in Vallada et al. (2008).

As the best of our knowledge, only Rom and Slotnick (2009) and Cesaret et al. (2012) propose meta-heuristics to solve the order acceptance and scheduling problem. We notice that the artificial bee colony (ABC) algorithm is a fairly new meta-heuristic proposed by Karaboga (2005), which is based on simulating the foraging behavior of honeybee swarms. Using some classic benchmark functions, Karaboga and Basturk (2007, 2008, and 2009) compare the performance of the ABC algorithm with that of other population-based algorithms such as differential evolution, particle swarm optimization, and evolutionary algorithm, and so on. Their research results demonstrate that the ABC algorithm is comparable to other population-based algorithms and the ABC algorithm on average shows good performance. Furthermore, Gao and Liu (2012) propose a modified ABC algorithm and show that it is superior to the basic ABC algorithm for 28 tested mathematical benchmark functions. Since its invention in 2005, the ABC algorithm has been applied to deal with practical combinatorial optimization problems (see, e.g., Singh (2009), Kang et al. (2009), and Samrat et al. (2010)). Szeto et al. (2011) provide an enhanced ABC algorithm to treat the capacitated vehicle routing problem (CVRP). They show that the algorithm performs better than some of the meta-heuristics (see, e.g., Toth and Vigo (2003), Berger and Barkoui (2003), and Ai and Kachitvichyanukul (2009)).

Since the problem under study is evidently NP-hard, only small-size instances can be optimally solved within a reasonable time. In view of the good performance of the ABC algorithm and its enhanced version in handling difficult combinatorial optimization problems such as the classical CVRP, we design variants of the ABC algorithm to treat the problem under study.

The paper is organized as follows. In Section 2, we give a formal description of the problem under study. In Section 3, we apply the basic ABC algorithm to solve our problem. In Section 4, we propose a modified ABC algorithm based on investigating the problem structure and optimal properties. In Section 5, we show the experimental results of the proposed ABC algorithms. Section 6 concludes our study with a summary.

2. Problem description

We formally describe the problem under study as follows: a pool of the potential orders, denoted by the set $N=\{1,2,\ldots,n\}$, is available for processing at time zero. Each order requires to be processed first on machine 1 and then on machine 2. The processing times of order i on machines 1 and 2 are a_i and b_i, respectively. Each machine can only process one order at a time and any order can begin processing on machine 2 only after finishing its processing on machine 1. Associated with order i are its revenue r_i, due date d_i, and weight w_i that represents its unit time delay penalty beyond d_i in delivery to the customer. The decisions are to determine the orders to accept for processing and how to schedule the accepted orders. The objective is to maximize the sum of the revenue of each accepted order minus its weighted tardiness, i.e., the total net revenue. Let $x_i \in \{0,1\}$ be a decision variable. If order i is accepted for processing, then $x_i=1$; otherwise $x_i=0$. Let C_i be the completion time of the accepted order i on machine 2. The objective is expressed as $\max \sum_{i=1}^{n} x_i(r_i - w_i \max\{0, C_i - d_i\})$.

3. Artificial bee colony algorithm

ABC algorithm belongs to the category of evolutionary algorithms that is inspired by the intelligent behavior of honeybees in finding nectar sources around their hives. In an ABC algorithm, the problem solutions are represented as food sources. The employed bees and onlookers exploit new food sources from the current ones. In the exploiting process, they communicate information on nectar quality between themselves by performing waggle dances. When a food source is abandoned by an employed bee, the employed bee becomes a scout and starts to explore randomly a new food source in the vicinity of the hive. This class of meta-heuristics has only started to be applied to solve various combinatorial optimization problems recently. To the best of our knowledge, there is no research on applying ABC algorithm to tackle problems that involve scheduling and other operational decisions.

In this section we first discuss how to apply the basic components of the ABC algorithm to treat our problem. We then combine these components to develop the basic ABC algorithm for treating the problem under study. It is evident that there is an optimal solution for the problem in which the accepted orders are processed in the same sequence on both machines. Thus, in the following we only search for solutions in which the processing of the accepted orders follows a permutation schedule.

3.1. Solution representation

We represent a solution by a vector in which the kth entry is the order in the kth position of a sequence. If order i does not appear in the vector, it is not accepted. Thus, a vector of size no more than n represents both order acceptance and two-machine sequencing decisions at the same time. For example, for the problem with the set of orders $\{1,2,\ldots,10\}$, a vector $v=(9, 2, 7, 6, 4, 10, 3)$ represents a problem solution in which orders 1, 5, and 8 are rejected, and the other orders are accepted and processed on both machines in the same sequence (9, 2, 7, 6, 4, 10, 3).

3.2. Initial solutions

For the problem with n orders, we first randomly generate n numbers from a uniform distribution on the interval [0, 1]. We then sort these n numbers in non-decreasing order. We record a list $S=[k_1,k_2,\ldots,k_n]$ in which the jth position is the k_jth generated number. For example, when $n=4$, we successively and randomly generate the numbers as 0.11, 0.75, 0.23 and 0.39. Then we create the list $S=[1, 3, 4, 2]$ by arranging the jobs in ascending order of their associated randomly generated number. According to list S, we generate a solution vector v as follows:

1. Let $\pi=\Phi$ (empty set) and $h=1$.
2. For $\ell=1$ to n, do