Applied Soft Computing 11 (2011) 4521–4529

Contents lists available at SciVerse ScienceDirect

Applied Soft Computing

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

An ant colony optimization algorithm for setup coordination in a two-stage production system

Ching-Jong Liao*, Yu-Lun Tsai, Chien-Wen Chao

Department of Industrial Management, National Taiwan University of Science and Technology, 43 Keelung Road, Section 4, Taipei 106, Taiwan

ARTICLE INFO

Article history:
Received 7 February 2009
Received in revised form 23 June 2011
Accepted 14 August 2011
Available online 22 August 2011

Keywords:
Scheduling
Ant colony optimization
Manufacturing system
Sequence-dependent setup time

ABSTRACT

This paper is concerned with the coordination of setup times in a two-stage production system. The problem is derived from a furniture plant, where there are two consecutive departments including cutting and painting departments. Items with the same levels of both attributes are grouped into a single batch in advance. A sequence-dependent setup time is required in a stage when a new batch has a different level of attribute from the previous one. The objective is to minimize the total setup time. In this paper, we first propose a simple dispatching rule called the Least Flexibility with Setups (LFS) rule. The LFS rule can yield a solution better than an existing genetic algorithm while using much less computation time. Using the LFS rule as both the initial solution method and the heuristic desirability, an Ant Colony Optimization (ACO) algorithm is developed to further improve the solution. Computational experiments show that the proposed ACO algorithm is quite effective in finding the near-optimal solution.

1. Introduction

Although the literature on supply chain management is extensive, the coordinated scheduling within supply chain models has received much less attention [1,2]. In this paper, we add to the limited literature on supply chain scheduling by considering a scheduling problem that arises in the setup coordination between two and three consecutive stages of a supply chain.

The setup coordination problem was first proposed by Agnetis et al. [3]. As depicted in Fig. 1, the problem arises from the manufacturing of a kitchen furniture plant [3], where there are two consecutive departments including cutting and painting departments. In this production system, each attribute has many different levels, e.g., the color attribute may have black, white, yellow, blue, and red different colors. Items with the same levels of both attributes are grouped into a single batch in advance. A setup occurs when a new batch has a level of an attribute different from the previous batch. Since no buffer is available between two consecutive departments, these departments have to follow the same batch sequence. Hence, we need only to determine a permutation of batches.

Agnetis et al. [3] considered the problem with two objectives. One is minimizing the total number of setups (MINSUM) in the two departments, and the other is minimizing the maximum number of setups (MINMAX) in either department. It was assumed that all the setup times are equal and sequence independent, i.e., unitary setup time. This implies that in the painting department the setup changed from blue to white has the same time as that changed from white to blue. Each of the two objectives has been proved to be NP-hard by Agnetis et al. [3].

In the literature, metaheuristics have been applied to many manufacturing systems [4]. In particular, several heuristics have been proposed for the problem. For the MINSUM problem, Agnetis et al. [3] and Meloni [5] proposed a constructive heuristic and a metaheuristic for the problem, respectively. Both solution approaches perform satisfactorily on a set of real cases and a large sample of experimental data. Detti et al. [6] transformed the MINSUM problem into an edge-domination problem related to the Hamiltonian completion number of line graphs. They developed an iterated local search (ILS) algorithm, which outperforms the heuristic of Agnetis et al. [3].

Considering both MINSUM and MINMAX simultaneously as a multi-objective problem, most studies applied metaheuristics to find a good approximation of the Pareto-optimal front. Mansouri [7] proposed a multi-objective genetic algorithm (MOGA), and shortly after Mansouri [8] presented a better multi-objective simulated annealing (MOSA). Also, Meloni et al. [9] compared three approaches based on different combinations of multi-objective evolutionary algorithms and local-search heuristics. Recently, Detti et al. [10] developed a graph-based ILS algorithm in which two greedy heuristics are used to obtain the initial solution. The computational results show that the ILS algorithm combining with the initial solution can obtain the effectiveness with respect to solution quality and computation time. With regard to applications of

* Corresponding author. Fax: +886 2 27376344.
E-mail address: cjl@im.ntust.edu.tw (C.-J. Liao).

1568-4946/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2011.08.014

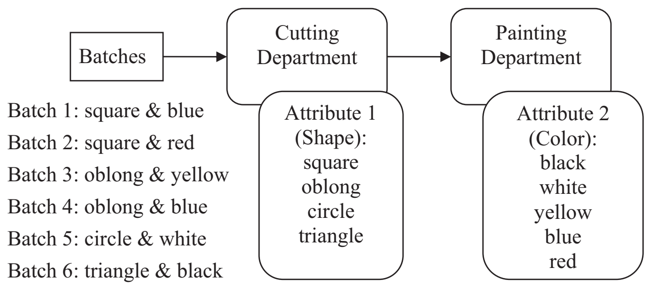

Fig. 1. The two-stage manufacturing system.

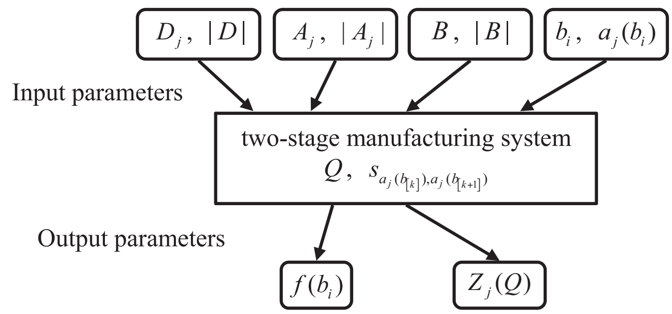

Fig. 2. Overview of input and output parameters.

ACO in supply chain, Liao and Chang [11] first used ACO in forecasting future demands and determining the optimal inventory policy values in a supply chain network.

In all the above mentioned papers, a unitary setup time is assumed. However, an explicit consideration of sequence-dependent setup time is usually required in many practical manufacturing environments, including the furniture manufacturing. As an illustration in the painting department, the setup time changed from blue to white is usually different from that changed from white to blue. Naso et al. [12] developed and compared three GAs for solving the problems related to three cost structures for setup operations, including the case of sequence-dependent setup times. In this paper, we address the MINSUM problem with sequence-dependent setup times and propose an Ant Colony Optimization (ACO) algorithm, which will be compared with the GA developed by Naso et al. [12].

The remainder of this paper is organized as follows. In Section 2, we give a formal definition of the problem. Section 3 describes the GA developed by Naso et al. [12]. Section 4 provides a detailed description of the proposed dispatching rule and ACO algorithm. Results of computational experiments to evaluate the performance of the proposed solution method are reported in Section 5. Finally, the concluding remarks and further works are given in Section 6.

2. Problem formulation

In this section, we give a formal description of the problem. The following notation is used throughout the paper:

Q	a sequence of batches
D_j	stage j, $j = 1, 2$
$\|D\|$	number of stages
A_j	set of all possible levels of attribute j
$\|A_j\|$	number of levels of attribute j
B	set of batches
$\|B\|$	number of batches
b_i	batch i, $i = 1, 2, \ldots, \|B\|$
$a_j(b_i)$	level of attribute j for b_i
$f(b_i)$	flexibility index of b_i (defined as the number of batches with a level of attribute same as b_i)
$b_{[k]}$	batch scheduled in the kth position, $k = 1, 2, \ldots, \|B\|$
$s_{a_j(b_{[k]}),a_j(b_{[k+1]})}$	setup time between batches k and $k+1$ in stage D_j
U	set of unscheduled batches
$Z_j(Q)$	total setup time in stage D_j in a sequence Q
$Z(\sigma)$	total setup time of partial sequence σ
$Z(\sigma, i)$	total setup time after batch b_i is appended to partial sequence σ

Example 1. We now elaborate the above parameters by referring to Fig. 1. The two-stage manufacturing system includes cutting and painting departments, so we have parameters D_1 and D_2 with $|D| = 2$. There are four shapes, $A_1 \in \{$square, oblong, circle, and triangle$\}$, in the cutting department and five colors, $A_2 \in \{$black, white, yellow, blue, and red$\}$, in the painting department. So we have $|A_1| = 4$ and $|A_2| = 5$. Six batches, $B \in \{b_1, b_2, b_3, b_4, b_5, b_6\}$, are processed in this manufacturing system so $|B| = 6$. $a_1(b_1)$ ($a_2(b_1)$) is the level of shape (color) for b_1, so $a_1(b_1)$ ($a_2(b_1)$) is square (blue). Similarly, the remaining levels of shape and color for each batch are shown in Fig. 1. Then, the flexibility index of each batch can be calculated. For instance, $f(b_1) = 2$ since $a_1(b_2)$ is square and $a_2(b_4)$ is blue, which are equal to $a_1(b_1)$ and $a_2(b_1)$, respectively. Now, suppose a sequence of batches $Q \in \{b_2, b_5, b_1, b_6, b_3, b_4\}$ is obtained. Then $b_{[1]} = b_2$, $b_{[2]} = b_5$, etc. $s_{a_1(b_{[1]}),a_1(b_{[2]})}$ ($s_{a_2(b_{[1]}),a_2(b_{[2]})}$) is the setup time between batches $b_{[1]}$ and $b_{[2]}$ in stage D_1(D_2), then Z_1(Z_2) can be calculated. Fig. 2 gives an overview of these input and output parameters. The remaining parameters, including U, $Z(\sigma)$, and $Z(\sigma, i)$, are used in the proposed algorithms in Section 4.

The two-stage coordination problem with sequence-dependent setup times consists in scheduling a set of batches B on the two stages D_1 and D_2. Because no buffer is available between the two consecutive stages, the sequence Q in which each stage processes all batches is identical on the two stages. Hence, we need only to determine a permutation of batches for the problem.

Each batch is characterized by two attributes a_1 and a_2. Let A_1 and A_2 denote the sets of all possible levels of a_1 and a_2, respectively. Let $b_{[k]}$ denote the batch scheduled in the kth position of a sequence. For two consecutive batches $b_{[k]}$ and $b_{[k+1]}$, a setup time $s_{a_j(b_{[k]}),a_j(b_{[k+1]})}$ is required in stage D_j if $a_j(b_{[k]}) \neq a_j(b_{[k+1]})$. Following Naso et al. [12], the sequence-dependent setup time for each couple of different levels of attribute a_j is generated from a discrete uniform distribution $U(1, |A_j|)$. Thus, there exists a square setup time matrix for each of the two attributes.

For a given sequence Q, we can calculate the setup times incurred in the two stages, denoted by $Z_1(Q)$ and $Z_2(Q)$, as follows:

$$Z_1(Q) = \sum_{k=1}^{|B|-1} s_{a_1(b_{[k]}),a_1(b_{[k+1]})} \tag{1}$$

$$Z_2(Q) = \sum_{k=1}^{|B|-1} s_{a_2(b_{[k]}),a_2(b_{[k+1]})} \tag{2}$$

The objective of the problem can then be expressed as

$$\text{Min} \quad Z = Z_1(Q) + Z_2(Q) \tag{3}$$

3. Genetic algorithm

In this section, we briefly introduce the GA developed by Naso et al. [12]. In particular, we present only the GA for the single-objective problem, including the MINSUM and MINMAX objectives.