Available online at www.sciencedirect.com
ScienceDirect
Computers & Industrial Engineering 54 (2008) 1–11
computers & industrial engineering
www.elsevier.com/locate/dsw

A tabu search algorithm for the flowshop scheduling problem with changing neighborhoods

Burak Ekşioğlu [a,*], Sandra Duni Ekşioğlu [a], Pramod Jain [b]

[a] Department of Industrial and Systems Engineering, Mississippi State University, P.O. Box 9542, Mississippi State, MS 39762, USA
[b] Sabre Airline Solutions, 3150 Sabre Drive, Southlake, TX 76092, USA

Received 3 May 2006; received in revised form 16 January 2007; accepted 11 April 2007
Available online 13 April 2007

Abstract

Flowshop scheduling deals with the sequencing of a set of jobs that visit a set of machines in the same order. A tabu search procedure is proposed for the flowshop scheduling problem with the makespan minimization criterion. Different from other tabu search procedures, the neighborhood of a solution is generated using a combination of three different exchange mechanisms. This has resulted in a well-diversified search procedure. The performance of the algorithm is tested using Taillard's benchmark problems. The results are compared to recently developed neuro-tabu search and ant colony heuristics. The computational results indicate the effectiveness of the proposed approach.

Keywords: Flowshop; Scheduling; Tabu search; Heuristics; Neighborhood

1. Introduction

The classical flowshop problem with the makespan minimization criterion has always attracted the attention of researchers because of its applications in practice. For example, a typical ship builder builds a number of different ship models, but the individual parts that go into each ship follow similar processes. A shipyard can be thought of as a collection of several flowshops. In particular, almost all parts, i.e. the big metal blocks, go through a panel shop in a ship yard where they are cut or welded together. Panel shops are typically treated as flowshops. The flowshop problem is easy to describe and formulate, yet computationally it is rather challenging. Therefore, this problem has inspired the development of a number of solution procedures.

The flowshop scheduling problem refers to the sequencing of a set of n jobs (tasks or items) to be processed on a set of m machines (or processors) in the same order. That is, the jobs are first processed on machine 1, then on machine 2, and so on until machine m. The objective is to find a sequence of the jobs to be processed on the machines so that the makespan is minimized. This problem is known to be *NP*-complete in the strong sense when $m \geqslant 3$ (Garey, Johnson, & Sethi, 1976). Johnson (1954) presents a simple algorithm that optimally

* Corresponding author. Tel.: +1 662 325 7625; fax: +1 662 325 7618.
E-mail address: beksioglu@ise.msstate.edu (B. Ekşioğlu).

0360-8352/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cie.2007.04.004

solves the flowshop scheduling problem in polynomial time in the special case when $m = 2$. The complexity of the flowshop scheduling problem has encouraged the development of various heuristic methods that provide feasible permutations of the n jobs. Exact methods are simply not practical even for instances with few jobs and few machines. Reisman, Kumar, and Motwani (1994) and Ruiz and Maroto (2005) provide comprehensive reviews of flowshop scheduling problems.

The heuristic approaches for the flowshop problem are grouped into *constructive heuristics* and *improvement heuristics* (Ruiz & Maroto, 2005). Constructive heuristics build a feasible schedule for a given set of n jobs. For example, Johnson's algorithm (Johnson, 1954) is a constructive algorithm. Most of the constructive heuristics have extended the Johnson's algorithm for the m-machine flowshop problem (Campbell, Dudek, & Smith, 1970; Koulamas, 1998; Sarin & Lefoka, 1993). The NEH algorithm developed by Nawaz, Enscore, and Ham (1983) is regarded as the best constructive heuristic for the classical flowshop scheduling problem. On the other hand, the improvement heuristics start with a feasible schedule and try to improve the problem's objective considering problem-specific knowledge (Dannenbring, 1977; Ho & Chang, 1991; Suliman, 2000). Metaheuristics such as simulated annealing (Osman & Potts, 1989; Ogbu & Smith, 1991), tabu search (Taillard, 1990), genetic algorithms (Chen, Vempati, & Aljaber, 1995; Reeves, 1995), etc. have been used to generate feasible sequences for the flowshop problem.

We propose a tabu search approach for the flowshop scheduling problem. Tabu search has shown to be successful in solving the flowshop problem with makespan minimization criterion (Nowicki & Smutnicki, 1996; Ben-Daya & Al-Fawzan. M., 1998; Grabowski & Wodecki, 2004; Solimanpur, Vrat, & Shankar, 2004; Taillard, 1990; Widmer & Hertz, 1989). The concept behind tabu search is very simple. The algorithm starts with an initial solution. The neighborhood of this solution is searched to identify a new neighbor to which to move. The search procedure explores the solution space beyond local optimality by allowing moves to neighbors that have worse makespans. Key elements of the search path are selectively remembered (using a tabu list), and strategic choices are made to guide the search out of local optima and into diverse regions of the solution space. The algorithm ends when some stopping criterion is satisfied. Despite the simplicity of this approach – defining the neighborhood, the move strategies, the forbidden moves, the tabu list length, etc. – remains an art. Different implementations of these elements result in different tabu search algorithms. The major difference between our algorithm, called 3XTS, and other tabu search procedures proposed for the flowshop problem is the neighborhood definition. The most common neighborhood types found in the literature are the ones created by adjacent exchange, random exchange, and insertion. The 3XTS algorithm uses a combination of three different exchange mechanisms that are described in Section 3.2.

The proposed tabu search algorithm is tested using the problem set provided by Taillard (1993). The results are compared to the results of the EXTS algorithm proposed by Solimanpur et al. (2004) and the results of an ant colony system proposed by Ying and Liao (2004).

2. Problem formulation

The flowshop scheduling problem with makespan criterion refers to the sequencing of a set of n jobs ($J = \{1,\ldots,n\}$) to be processed on a set of m machines ($I = \{1,\ldots,m\}$). Without loss of generality, assume that the jobs are processed in the order of indices of machines. The time it takes to process job j on machine i is p_{ji}. A sequencing of jobs can be represented by a permutation $\pi = (\pi(1),\ldots,\pi(n))$, where $\pi(a)$ is the ath element of permutation π. Let Π denote the set of all such permutations. The objective is to find a permutation $\pi^* \in \Pi$ that minimizes the makespan (the completion time of the last job on the last machine):

$$C_{\max}(\pi^*) = \min_{\pi\in\Pi} C_{\max}(\pi),$$

where $C_{\max}(\pi)$ is the time required to complete the processing of all jobs on all machines following the sequence given by permutation π. $C_{\max}(\pi)$ can be calculated using the following equation (Grabowski, 1979, 1982):

$$C_{\max}(\pi) = \max_{1\leqslant t_1\leqslant\ldots\leqslant t_{m-1}\leqslant n} \sum_{a=1}^{t_1} p_{\pi(a)1} + \sum_{a=t_1}^{t_2} p_{\pi(a)2} + \cdots + \sum_{a=t_{m-1}}^{n} p_{\pi(a)m} \tag{1}$$

The sequence of integers $t = (t_1, t_2, \ldots, t_{m-1})$ defines a path from node $(1,1)$ to (m,n) in a grid graph as the one presented in Fig. 1. We introduce the following equivalent makespan formulation: