Available online at www.sciencedirect.com
ScienceDirect
European Journal of Operational Research 186 (2008) 496–503
EUROPEAN JOURNAL OF OPERATIONAL RESEARCH
www.elsevier.com/locate/ejor

Discrete Optimization

Minimizing total tardiness on a single machine with unequal release dates

Ling-Huey Su *, Chung-Jung Chen

Department of Industrial Engineering, Chung-Yuan Christian University, Chung-Li, Taiwan, ROC

Received 27 April 2004; accepted 5 July 2006
Available online 14 February 2007

Abstract

This study addresses the problem of minimizing total tardiness on a single machine with unequal release dates. Dominance properties established in previous literatures and herein are adopted to develop branch and bound and heuristic procedures. Computational experiments were conducted to evaluate the approaches. The results revealed that the branch and bound algorithm is efficient in solving hard problems and easy problems that involve up to 50 and 500 jobs, respectively. The computational effectiveness of the heuristic is also reported.

Keywords: Single machine; Total tardiness; Unequal release dates; Branch and bound algorithm; Heuristic algorithm

1. Introduction

This work considers the scheduling of n independent jobs on a single machine to minimize total tardiness with unequal release dates. Each job i has a release date r_i, a processing time p_i and a due date d_i. A job's tardiness T_i is defined as $T_i = \max\{0, C_i - d_i\}$, where C_i is the job's completion time. According to the tardiness criterion, no benefit is gained from completing jobs early and a delay leads to a proportional penalty. The objective is to minimize the penalty over all jobs $\sum_{i=1}^{n} T_i$.

The total tardiness problem on a single machine has received considerable attention and has been shown to be NP-complete. Many priority rules have been developed to solve the static problem. Emmons [9] derived rules for establishing dominance among elements within an n-job set. According to that study, he showed that the shortest processing time (SPT) sequence is optimal if all jobs have positive tardiness and the earliest due date (EDD) sequence is optimal if no more than one job has positive tardiness. These rules have been used to restrict the search and the optimal solution was found through a branch and bound algorithm. Rinnooy Kan et al. [15] established four theorems for the general non-decreasing cost function. Lawler [10] developed a pseudo-polynomial algorithm for solving the problem under the assumption that the weighting of jobs is agreeable with the processing time. Potts and Van Wassenhove [12] utilized

* Corresponding author.
E-mail address: linghuey@cycu.edu.tw (L.-H. Su).

0377-2217/$ - see front matter
doi:10.1016/j.ejor.2006.07.051

Lawer's decomposition theorem in combination with Schrage and Baker's dynamic programming to develop an efficient algorithm that can solve problems up to 100 jobs. Baker and Bertrand [5] compared priority rules such as EDD, SPT, CON (common due date), SLK (equal slack), TWK (total work time) and MDD (modified due date) and found that MDD outperformed all of the others. Sen et al. [16] established certain precedence relations among jobs using Emmons' results and described an implicit enumeration scheme which requires $O(n^2)$ computer storage. Using Lagrangian relaxation, Potts and Van Wassenhove [13] obtained a lower bound and used it in a branch and bound algorithm. Rachamadugu [14] identified a condition for characterizing adjacent jobs in an optimal sequence for the weighted tardiness problem and indicated that the MDD rule is a special case of this condition. He also identified a set of circumstances under which the first job in an optimal sequence can be determined without fully solving the problem. Abdul-Razaq et al. [1] surveyed algorithms on tardiness in a single machine. Panwalkar et al. [11] presented a heuristic algorithm which yielded better results than the heuristics proposed by Wilkerson–Irwin (W–I), Holsenback–Russell (H–R), and also better than the API heuristics. Alidaee and Gopalan [2] showed that the heuristic algorithm presented by Panwalkar et al. [11] is only one of many for implementing the MDD rule.

The total tardiness problem with unequal job release dates is much more difficult to solve than those with equal job release dates. Chu and Portman [8] showed that the problem can be simplified by using corrected due dates: if $r_j + p_j > d_j$ then d_j takes the value $r_j + p_j$. They identified a sufficient condition for local optimality. On the basis of this condition, they define a new dominant subset of schedules and proposed several new dominant approximate algorithms. Chu [7] also proposed some dominance properties, and provided a polynomially computed lower bound for this problem. Based on the results, he constructed a branch and bound algorithm to solve the problem, which was tested on hard problems that involved 30 jobs and on relatively easy problems with up to 230 jobs. Akturk and Ozdemir [4] presented new dominance rules by considering the time-dependent orderings between each pair of jobs. Their dominance rule developed provides a sufficient condition for local optimality. According to their results, they showed that if any sequence violated the dominance rule, then switching the violating jobs could reduce, or at least maintain the total weighted tardiness. They introduced an algorithm based on the dominance rule and compared it with many heuristics. Baptiste et al. [6] presented a branch-and-bound to minimize total tardiness with arbitrary release dates. They introduced new lower bound and generalize some well-known dominance properties. Their computational results showed that their procedure can solved problem as large as 500 jobs with some 60 jobs instances remaining open.

2. Dominance properties

Two propositions, on which the exact algorithms are based, are first proposed. The current decision point is denoted by t and $A(t)$ is defined as the set of available unscheduled jobs at time t; $B(t)$ as the set of unavailable and unscheduled jobs at time t, and $U(t)$ as the set of unscheduled jobs at time t. Therefore $U(t) = A(t) \cup B(t), C_i(\pi)$ the completion time of the ith position job in schedule π.

Proposition 1. *Number the unscheduled jobs in ascending order of r_i, breaking ties in favor of the job with the smallest processing time. A set of jobs $\{i, i+1, \ldots, k\}$ form a block F if*

(1) $C_{i-1}(\pi) \leqslant r_i$,
(2) $C_{j-1}(\pi) > r_j, \quad j \in \{i+1, i+2, \ldots, k\}$,
(3) $C_k(\pi) \leqslant r_{k+1}$.

Then, considering the job in F in the first unscheduled position in an optimal sequence suffices.

Proof. Assume that job $j \in F$ and job $i \notin F$, since $C_j < r_i$, therefore scheduling job j in the first unscheduled position does not increase tardiness in a given sequence. □

Proposition 2. *If one of the following conditions hold;* (a) $i, j \in A(t)$ *and* $d_j \geqslant \max(t, \max_{k\in U(t)} r_k) + \sum_{k\in U(t)} p_k$, (b) $i \in A(t), \quad j \in B(t), \quad$ *and* $d_j \geqslant \max(t, \max_{k\in U(t)} r_k) + \sum_{k\in U(t)} p_k$, (c) $i, j \in$ B(t),{r}_{i}< $r_j, \quad$ *and* $\quad d_j \geqslant \max(t, \max_{k\in U(t)} r_k) + \sum_{k\in U(t)} p_k$, *then job i precedes job j.*

Proof. Proof by the pairwise interchange method is easy.

Tables 1 and 2 summarize various dominance properties developed in the literatures as well as the aforementioned properties. Table 1 presents the