

330

B385

1991:171 COPY 2

STX

A Fixed Interval Due-Date Scheduling Problem with Earliness and Due-Date Costs

The Library of the

MAR 20 1992

University of Illinois
at Urbana-Champaign

Dilip Chhajed

Department of Business Administration

BEBR

FACULTY WORKING PAPER NO. 91-0171


College of Commerce and Business Administration

University of Illinois at Urbana-Champaign

October 1991

A Fixed Interval Due-Date Scheduling Problem
with Earliness and Due-Date Costs

Dilip Chhajed
Department of Business Administration



Digitized by the Internet Archive
in 2011 with funding from
University of Illinois Urbana-Champaign

**A Fixed Interval Due-Date Scheduling Problem
with Earliness and Due-Date Costs**

Dilip Chhajed*

April 1991

Abstract

In this paper we consider a problem where N jobs are to be scheduled on a single machine and assigned to one of the two due-dates which are given at equal intervals. Tardy jobs are not allowed, thus the due-dates are deadlines. There is a linear due-date penalty and linear earliness penalty, and the sum of these penalties is to be minimized. This problem is shown to be NP-hard. One case of the problem (the unrestricted case) is shown to be solvable in linear time. For the restricted case, lower and upper bounds are developed. Computational experience is reported which suggests that the bounds are quite effective.

* University of Illinois, Department of Business Administration, 350 Commerce West,
Champaign, IL 61820.

1. Introduction

To motivate the problem in this paper, consider a shop with a single machine and N jobs available at time 0 that need to be assigned a due-date and sequenced on the machine. The time required by each job on the machine is known and deterministic. Finished jobs are supplied to the customer by a truck that is dispatched at a fixed time interval, for example, at the end of each week. Since all jobs finished during the week will be shipped at the end of the week, the due-date assigned to a job will be the end of the week in which the job is scheduled to be completed. Since customers prefer their jobs to be shipped as early as possible, there is a penalty for assigning jobs to a due-date that is proportional to the length of the due-date. A finished job incurs a holding cost until it is shipped. The shop has the policy of delivering on the promised due-date, thus tardy jobs are not allowed. The measure of customer service level is through due-date cost. The problem is to find an assignment of due-dates (the week in which the job will be completed) and sequence such that the sum of earliness penalty and due-date penalty is minimized, subject to no tardy job.

The problem of scheduling a given number of jobs to minimize the sum of earliness, tardiness, and due-date costs where the due-date cost of an order is proportional to its assigned due-date is well studied. Panwalkar, Smith, and Seidmann (1982) consider the case of single due-date and Chand and Chhajed (1990) consider the case of multiple due-dates. Models in which there is a single fixed due-date and a schedule of minimum total earliness and tardiness costs are also considered (Hall, Kubiak, and Sethi, 1989; Kanet, 1981; Bagchi, Sullivan, and Chang, 1986).

The interest in studying problems with earliness and tardiness costs (E/T Cost) supports the growing success and popularity of Just-In-Time (Baker and Scudder, 1990). Typically in a schedule obtained for a problem with E/T costs, some jobs will be finished on time, some will be early and others will be tardy. An early job incurs a penalty for each time unit it has to wait after completion until shipped to the customer, whereas a tardy job is

shipped as soon as it is completed. In the literature, the cost associated with shipping each tardy job separately is not considered. A comprehensive review of the literature on scheduling problems with E/T costs is given in Baker and Scudder (1990). Note that in an JIT environment a tardy job may force a customer to shut down operations, the cost of which may be very high.

In the problem we consider, the due-dates cannot be violated and so they are deadlines as considered in Ahmadi and Bagchi (1986) and Chand and Schneeberger (1988). Matsuo (1988) considers the problem with fixed shipping times (similar to ours) and minimizes the sum of overtime and tardiness costs.

Our focus in this paper is on a restricted version of the above problem where only two due-dates are considered. We concentrate on this restricted version because even this case is computationally difficult (Appendix A). However this simple case provides us insights and results which can be useful in solving the general case. This paper also provides a framework for developing a solution scheme for many other useful and interesting variations of the problem, some of these are outlined in the concluding section.

The rest of the paper is organized as follows. In section 2, we introduce the notation and formulate the problem for the two due-dates case. In section 3, we show that certain special case of the problem is easy to solve. In section 4, we provide a method to obtain lower and upper bounds. Our computational study, reported in section 5, indicates that these bounds are very effective. Concluding remarks are in section 6. In Appendix A, we show the complexity of the problem and proofs of several results are in Appendix B.

2. Preliminaries

The statement of the problem we consider is: Assign N jobs to one of two due-dates and schedule them such that there is no tardy job and the sum of due-date penalty and earliness penalty is minimum. The two due-dates are given and are at equal intervals. We assume that pre-emption is not allowed and only one job can be processed at a time. All

cost functions are assumed to be linear. We call this problem the *2-Due-Date Scheduling Problem (2DSP)*.

We now introduce the notation.

Notation:

N : number of jobs

t_i : processing time of job i

δ : due-date penalty per time unit (the earliness penalty per time unit is assumed to be one)

τ : time period (interval between two successive due-dates)

J_i : job i . Occasionally we will refer to a job by its index, i.e. job i

D_i : due-date of job i

C_i : Completion time of job i

T : sum of the processing times of all jobs = $\sum_{i=1}^N t_i$

$\langle a, b \rangle$: Set of integers $a, a+1, \dots, b$

In our development, due-date j mean that the due-date is $j\tau, j=1,2$. We assume that $T \leq 2\tau$ so that the problem is feasible. Jobs are indexed such that $t_1 \leq t_2 \leq \dots \leq t_N$.

This problem can be formulated as:

$$(P_1) \quad \min Z = \sum_{i=1}^N (D_i - C_i) + \delta \sum_{i=1}^N D_i$$

$$\text{ST.} \quad D_i \geq C_i$$

$$D_i \in \{\tau, 2\tau\}.$$

Note that there may be idle time before the starting time of the first jobs assigned to due-dates 1 and 2. Thus a permutation schedule may not be optimal. However, there will be no idle time between jobs assigned to the same due-date. This can be shown by using a simple interchange argument. Also the completion time of the last job assigned to the first due-date may not coincide with the due-date. It is optimal to schedule jobs assigned to the same due-date in non-decreasing order of their processing time. Figure 1 shows the nature

of different kinds of schedules that may occur. We will denote the optimal value of function Z by Z^* .

Given a schedule, the job at position i , denoted as $J_{[i]}$, refers to the job which is preceded by $(i-1)$ jobs. Since there may be idle time, the completion time of $J_{[i]}$ may not be equal to $\sum_{j=1}^i t_{[j]}$. Let n_1 denote the number of jobs (a decision variable) assigned to the first due-date and let $C_{[0]} = 0$. With these notation (P_1) can be rewritten as:

$$(P_1) \min Z = \sum_{i=1}^{n_1} (\tau - C_{[i]}) + \sum_{i=n_1+1}^N (2\tau - C_{[i]}) + (n_1 + 2(N-n_1))\tau\delta \quad (1)$$

$$\text{S. T.} \quad C_{[i]} \geq C_{[i-1]} + t_{[i]}, \quad i=1, \dots, N \quad (2)$$

$$C_{[n_1]} \leq \tau. \quad (3)$$

$$n_1 \in \langle 1, N \rangle \quad (4)$$

Any optimal solution to 2DSP will satisfy one of the following cases (Figure 1):

Case I: $C_{[n_1]} = \tau$

Case II: $C_{[n_1]} < \tau$.

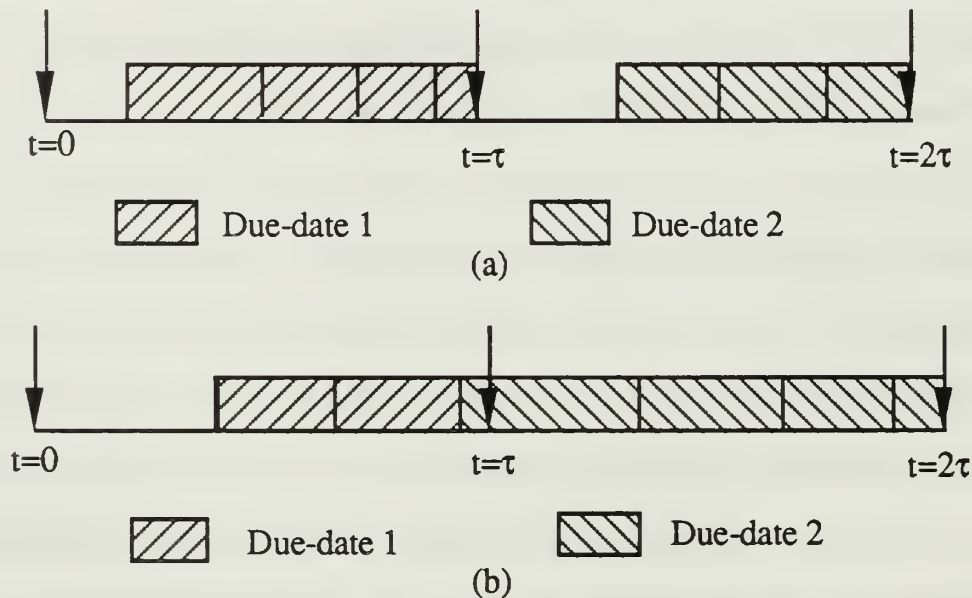
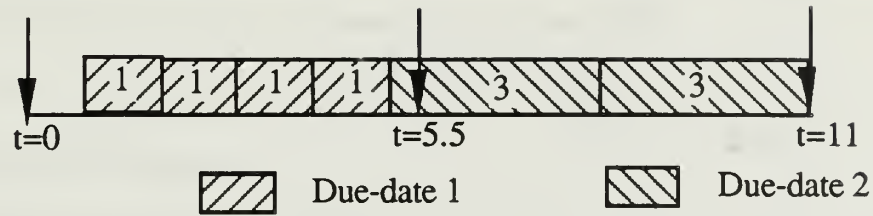


Figure 1. Two Kinds of Schedules

We now give an example which has an optimal solution corresponding to Case II.

Example: Let $N=6$, $\tau=5.5$, $t_i = \{1,1,1,1,3,3\}$, $\delta=1$.

The optimal solution for this case is:



which is a Case II solution.

The two cases are now analyzed separately.

Case I: $C_{[n_1]} = \tau$.

In this case, the following property is satisfied by all optimal solutions:

Property 1: If an optimal solution to (P_1) satisfies $C_{[n_1]} = \tau$, then idle time can be only before the first job that is processed in due-date 1 and the first job (position n_1+1) in due-date 2.

For this case $C_{[n_1]} = \tau$ and $C_{[N]} = 2\tau$ and the objective function of (P_1) can be rewritten as:

$$\begin{aligned}
 & \sum_{i=1}^{n_1} (\tau - C_{[i]}) + \sum_{i=n_1+1}^N (2\tau - C_{[i]}) + (2N-n_1)\tau\delta \\
 &= \sum_{i=1}^{n_1} (C_{[n_1]} - C_{[i]}) + \sum_{i=n_1+1}^N (C_{[N]} - C_{[i]}) + (2N-n_1)\tau\delta \\
 &= \sum_{i=1}^{n_1} \sum_{j=i+1}^{n_1} t_{[j]} + \sum_{i=n_1+1}^N \sum_{j=i+1}^N t_{[j]} + (2N-n_1)\tau\delta \\
 &= \sum_{i=1}^{n_1} (i-1) t_{[i]} + \sum_{i=n_1+1}^N (i-1 - n_1) t_{[i]} + (2N-n_1)\tau\delta \quad (5)
 \end{aligned}$$

Therefore (P_1) , with the additional constraint that $C_{[n_1]} = \tau$, can be formulated as,

$$(P_2) \min Z_1 = \sum_{i=1}^{n_1} (i-1) t_{[i]} + \sum_{i=n_1+1}^N (i-1 - n_1) t_{[i]} + (2N-n_1)\tau\delta$$

$$S. T. \quad C_{[1]} \geq t_{[1]}$$

$$\begin{aligned}
C_{[i]} &= C_{[i-1]} + t_{[i]}, \quad i=2, \dots, n_1 \\
C_{[n_1]} &\geq \tau + t_{[n_1]} \\
C_{[i]} &= C_{[i-1]} + t_{[i]}, \quad i=n_1+2, \dots, N \\
C_{[n_1]} &= \tau, \text{ and (4).}
\end{aligned}$$

We will refer to the number multiplied to the processing time of the job assigned to position i as the *positional weight* of position i . In (P_2) , for example, the positional weight of position i ($i \leq n_1$) is $(i-1)$.

Case II: $C_{[n_1]} < \tau$.

All optimal solutions satisfying this case satisfy the following property:

Property 2: If there is idle time, it will only be before the job scheduled in position 1.

The objective function of (P_1) can be simplified as:

$$\begin{aligned}
& \sum_{i=1}^{n_1} (\tau - C_{[i]}) + \sum_{i=n_1+1}^N (2\tau - C_{[i]}) + (2N-n_1)\tau\delta \\
&= \sum_{i=1}^{n_1} (C_{[N]} - C_{[i]}) + \sum_{i=n_1+1}^N (C_{[N]} - C_{[i]}) + (2N-n_1)\tau\delta - n_1\tau \\
&= \sum_{i=1}^N \sum_{j=i+1}^{n_1} t_{[j]} + (2N-n_1)\tau\delta - n_1\tau \\
&= \sum_{i=1}^N (i-1) t_{[i]} + (2N-n_1)\tau\delta - n_1\tau \tag{6}
\end{aligned}$$

The formulation (P_1) now takes the following form,

$$(P_3) \min Z_2 = \sum_{i=1}^N (i-1) t_{[i]} + (2N-n_1)\tau\delta - n_1\tau$$

$$\text{S. T.} \quad C_{[1]} \geq t_{[1]}$$

$$C_{[i]} = C_{[i-1]} + t_{[i]}, \quad i=2, \dots, N$$

$$C_{[n_1]} < \tau, \text{ and (4).}$$

In order to solve (P_1) , we could solve (P_2) and (P_3) . Then the optimal value of (P_1) will be given by

$$Z^* = \min \{Z_1^*, Z_2^*\}. \tag{7}$$

In Appendix A we show that (P_1) is NP-hard. If $T \leq \tau$, then all optimal solutions

will correspond to Case I. We call such problems as *unrestricted*. The unrestricted case can be easily solved in $O(N)$ time as we show in the next section.

3. The Unrestricted Case

When $T \leq \tau$, any number of jobs can be assigned to the first due-date without the sum of their processing times exceeding the length of the period. It is this observation which makes this problem easy to solve. With n_1 as the number of jobs assigned to the first due-date, let

$$w_i = \begin{cases} (i-1) & i=1 \dots n_1 \\ (i-1-n_1) & i= n_1+1 \dots N \end{cases} \quad (8)$$

be the positional weight of position i . If we fix the value of n_1 , the optimal solution to (5) can be obtained by sorting the positional weights in decreasing order and assigning job i to position k where k is such that w_k has rank i .

To find an optimal solution, we compute the cost Z^*_1 for different values of fixed n_1 and choose the value of n_1 , and the corresponding solution, which gives the minimum cost. The values of n_1 that need to be examined are $n_1 = m_0$ to N , where $m_0 = \min$ integer $\geq N/2$. Since sorting takes $O(N \log N)$ time and there are $O(N)$ possible values of n_1 , this suggests an $O(N^2 \log N)$ algorithm. However, as we show, this procedure can be reduced to $O(N)$ time.

We first note that $w_i < w_{i+1}$, $i \leq n_1$ and $w_i < w_{i+1}$, $i > n_1$. Thus, the w_i 's can be sorted in $O(N)$ time. In the first iteration, with $n_1 = m_0$, the solution can be obtained in $O(N)$ time. We now show that, given a solution to any value of $n_1 = n^o$, the solution for $n = n^o + 1$ can be obtained in constant time. To show this, consider an example with 10 jobs and processing times $t_1 \leq t_2 \leq \dots \leq t_{10}$. When $n_1 = 6$, the positional weights and the assignment of jobs is as follows:

	Due Date 1						Due Date 2			
Position	1	2	3	4	5	6	7	8	9	10

w_i	0	1	2	3	4	5	0	1	2	3
Job #	10	8	6	4	2	1	9	7	5	3

Thus, jobs {10,8,6,4,2,1} are assigned to due-date 1 and the remaining jobs are assigned to due-date 2. Note that jobs assigned to a due-date are processed in LPT order. When $n_1 = 7$, the new weights and the corresponding assignments will be:

	Due Date 1							Due Date 2		
Position	1	2	3	4	5	6	7	8	9	10
w_i	0	1	2	3	4	5	6	0	1	2
Job #	10	8	6	4	3	2	1	9	7	5

In going from a solution with $n_1=6$ to $n_1=7$, the smallest job (here job 3) assigned to the second due-date is reassigned to due-date 1. All the other jobs remain assigned to their earlier due-dates. The relative positions of some jobs in due-date 1 are increased by 1, and the relative position of all jobs assigned to due-date 2 increases by 1. Note that the cost resulting from these changes can be computed in constant time, once the optimal cost and the solution for $n_1=6$ is given. Thus, after the initial computation of the solution for $n_1=m_0$, each subsequent optimal cost to (5) as a function of n_1 can be computed in constant time resulting in a total complexity of $O(N)$.

4. Sharp Lower and Upper Bounds

In this section we provide a method to obtain lower and upper bounds to 2DSP when $2\tau \geq T > \tau$. Suppose we know that n_1 jobs are assigned to due-date 1 and the remaining $N-n_1$ jobs are assigned to due-date 2, i.e. constraint (4) is no longer present. Let the optimal objective function value with n_1 jobs assigned to due-date 1 be $Z^*(n_1)$. As the number of jobs assigned to the first due-date can vary from 1 to N (more on this later), the optimal value of (P_1) is given by, $Z^* = \min_{n_1 \in \{1, N\}} Z^*(n_1)$.

With n_1 jobs assigned to due-date 1, we find $Z^*(n_1)$ by considering the two cases

as before. We now analyze these two cases separately in order to obtain a lower bound to $Z^*(n_1)$.

Case I: $C_{[n_1]} = \tau$, n_1 fixed.

In order for the completion time of the last job assigned to due-date 1 (the job in position n_1) to be τ , it is necessary and sufficient to have the sum of the processing time of jobs assigned to each of the two due-dates to be less than or equal to τ . This is written as,

$$\sum_{i=1}^{n_1} t_{[i]} \leq \tau \text{ and} \quad (9)$$

$$\sum_{i=n_1+1}^N t_{[i]} \leq \tau. \quad (10)$$

As $\sum_{i=n_1}^N t_{[i]} = T - \sum_{i=1}^{n_1} t_{[i]}$, (10) becomes, $T - \sum_{i=1}^{n_1} t_{[i]} \leq \tau$ or $\sum_{i=1}^{n_1} t_{[i]} \geq T - \tau$. Thus, an equivalent

formulation of (P₂) with an additional requirement that n_1 is fixed is written as:

$$(P_4) \min Z_1(n_1) = \sum_{i=1}^{n_1} (i-1)t_{[i]} + \sum_{i=n_1+1}^N (i-1-n_1) t_{[i]} + (2N-n_1)\tau\delta$$

$$S. T. \quad \sum_{i=1}^{n_1} t_{[i]} \leq \tau \quad (11)$$

$$- \sum_{i=1}^{n_1} t_{[i]} \leq \tau - T. \quad (12)$$

We form a Lagrangian relaxation of (P₄) by multiplying constraints (11) and (12) by non-negative numbers λ_1 and λ_2 , respectively, and adding them to the objective function:

$$\begin{aligned} \min Z_1(n_1, \lambda_1, \lambda_2) &= \sum_{i=1}^{n_1} (\lambda_1 - \lambda_2 + i-1)t_{[i]} + \sum_{i=n_1+1}^N (i-1-n_1) t_{[i]} + (2N-n_1)\tau\delta - \lambda_1\tau + \lambda_2(T-\tau). \\ &= \sum_{i=1}^{n_1} (\lambda_1 - \lambda_2 + i-1)t_{[i]} + \sum_{i=1}^{N-n_1} (i-1) t_{[i+n_1]} + (2N-n_1)\tau\delta - \lambda_1\tau + \lambda_2(T-\tau). \end{aligned} \quad (13)$$

The choice of (λ_1, λ_2) that solves

$$\underline{Z}^*_1(n_1) = \max_{\lambda_1, \lambda_2} Z^*_1(n_1, \lambda_1, \lambda_2) \quad (14)$$

provides us with the best choice of the Lagrangian multipliers in the sense of providing the tightest lower bound. Thus,

$$\underline{Z}^*_1(n_1) = \max_{\lambda_1, \lambda_2} Z^*_1(n_1, \lambda_1, \lambda_2) \leq Z^*_1(n_1).$$

We note that for fixed values of n_1 , λ_1 and λ_2 , $Z^*_1(n_1, \lambda_1, \lambda_2)$ is obtained by arranging the sequence $\{(\lambda_1 - \lambda_2 + i - 1): i = 1 \dots n_1\} \cup \{(i - 1 - n_1): i = n_1 + 1 \dots N\}$ in decreasing order and multiplying the sequence with t_i 's.

Case II: $C_{[n_1]} < \tau$, n_1 fixed.

In order to have a schedule with the completion time of the last job assigned to the first due-date not coincide with τ , we must have,

$$\sum_{i=1}^{n_1} t_{[i]} < \tau \text{ and} \quad (15)$$

$$\sum_{i=n_1+1}^N t_{[i]} > \tau. \quad (16)$$

Constraint (15) restricts the total processing times of jobs assigned to due-date 1 to less than the period and constraint (16) is required because the total processing time of jobs assigned to due-date 2 must exceed τ . With $\sum_{i=n_1+1}^N t_{[i]} = T - \sum_{i=1}^{n_1} t_{[i]}$, (16) becomes $T - \sum_{i=1}^{n_1} t_{[i]} > \tau$ or $T - \tau > \sum_{i=1}^{n_1} t_{[i]}$. Since $T - \tau \leq \tau$, constraint (16) implies (15). (P₃) with n_1 fixed can

be rewritten as:

$$\begin{aligned} \min Z_2(n_1) &= \sum_{i=1}^N (i-1)t_{[i]} + (2N-n_1)\tau\delta - n_1\tau \\ \text{S. T.} \quad \sum_{i=1}^{n_1} t_{[i]} &< T - \tau. \end{aligned} \quad (17)$$

Again we form a Lagrangian relaxation by multiplying constraint (17) by λ and including it in the objective function to get,

$$\min Z_2(n_1, \lambda) = \sum_{i=1}^{n_1} (\lambda + i - 1)t_{[i]} + \sum_{i=n_1+1}^N (i-1)t_{[i]} + (2N-n_1)\tau\delta - n_1\tau - (T-\tau)\lambda. \quad (18)$$

We want a value of λ that solves the dual problem,

$$\underline{Z}^*_2(n_1) = \max_{\lambda} Z^*_2(n_1, \lambda) \leq Z^*_2(n_1). \quad (19)$$

Theorem 1 shows how a lower bound to 2DSP can be computed.

Theorem 1: The value of $\underline{Z}^ = \min (\min_{n_1} \{ \max_{\lambda_1, \lambda_2} Z^*_1(n_1, \lambda_1, \lambda_2) \}, \min_{n_1} \{ \max_{\lambda} Z^*_2(n_1, \lambda) \})$ is a valid lower bound to (2DSP).*

Proof: By (7), $Z^* = \min \{ Z^*_1, Z^*_2 \}$,

$$\begin{aligned} &= \min (\min_{n_1} \{ Z^*_1(n_1) \}, \min_{n_1} \{ Z^*_2(n_1) \}), \\ &\geq \min (\min_{n_1} \{ \underline{Z}^*_1(n_1) \}, \min_{n_1} \{ \underline{Z}^*_2(n_1) \}), \\ &= \min (\min_{n_1} \{ \max_{\lambda_1, \lambda_2} Z^*_1(n_1, \lambda_1, \lambda_2) \}, \min_{n_1} \{ \max_{\lambda} Z^*_2(n_1, \lambda) \}) \\ &= \underline{Z}^*. \llcorner \llcorner \end{aligned}$$

The dual problems (14) and (19) can be solved by any of the methods given in Fisher (1981). We now develop several results that reduce the computational time required to obtain \underline{Z}^* and present an algorithm that uses these results to obtain \underline{Z}^* .

Lemma 1: There exists an optimal solution to (14) with at least one of λ_1 and λ_2 equal to 0.

Proof: We prove this by contradiction. Suppose for fixed n_1 , we have multipliers $\lambda^*_1 > 0$ and $\lambda^*_2 > 0$ for which $Z^*_1(n_1, \lambda^*_1, \lambda^*_2)$ is an optimal solution to (14).

Case A: $\lambda^*_1 > \lambda^*_2$. If we use new multipliers $\lambda'_1 = \lambda^*_1 - \lambda^*_2$ and $\lambda'_2 = \lambda^*_2 - \lambda^*_2 = 0$, the relative ranking of the positional weights does not change and so the given sequence is still optimal with these new values of the multipliers. The new objective function value in (13) is,

$$\begin{aligned} Z^*_1(n_1, \lambda'_1, \lambda'_2) &= \sum_{i=1}^{n_1} (\lambda'_1 - \lambda'_2 + i - 1) t_{[i]} + \sum_{i=1}^{N-n_1} (i - 1) t_{[i+n_1]} + (2N - n_1) \tau \delta - \lambda'_1 \tau - \lambda'_2 (\tau - T) \\ &= Z^*_1(n_1, \lambda^*_1, \lambda^*_2) + \lambda^*_2 (2\tau - T). \end{aligned}$$

As $2\tau \geq T$, $Z^*_1(n_1, \lambda'_1, \lambda'_2) \geq Z^*_1(n_1, \lambda^*_1, \lambda^*_2)$ and thus we have found a solution which is at least as good as the given optimal solution to (14) and satisfies the Lemma.

Case B: $\lambda^*_1 < \lambda^*_2$. The proof of this case is similar to the proof of Case A. Now we set $\lambda'_1 = \lambda^*_1 - \lambda^*_1$ and $\lambda'_2 = \lambda^*_2 - \lambda^*_1$. «»

Lemma 2: It is sufficient to consider only integer values of the multipliers in (13) and (18).

Proof: Consider (13). Only one λ_i ($i=1,2$) is positive. Suppose $\lambda_1 > 0$ and $k \leq \lambda_1 \leq k+1$ for some integer k . For any value of λ_1 , $Z^*_1(n_1, \lambda_1, 0)$ is obtained by arranging the sequence $\{(\lambda_1 + i - 1): i=1 \dots n_1\} \cup \{(i - 1 - n_1): i=n_1+1 \dots N\}$ in decreasing order and multiplying it with t_i 's. If we vary λ_1 between k and $k+1$, the ordering of the sequence does not change and so the first two terms in (13) are linear functions of λ_1 . Thus (13), with $k \leq \lambda_1 \leq k+1$ has an optimal solution at an extreme point, i.e., $\lambda_1=k$ or $\lambda_1=k+1$.

The proof of $\lambda_2 > 0$ in (13) and $\lambda > 0$ in (18) is similar. «»

By the virtue of the above two Lemmas, we only have to consider integer values of the multipliers. Furthermore, while computing $Z^*_1(n_1, \lambda_1, \lambda_2)$, at most one of the two multipliers will be non-zero. We need the following definitions for the next Theorem.

Let,

$$m_0 = \text{minimum integer } \geq N/2,$$

$$m_1 = \max \{j: \sum_{i=1}^j t_i < \tau\},$$

$$m_2 = \min \{j: \sum_{i=1}^j t_{N-i+1} > T - \tau\},$$

$$m_3 = \max \{j: \sum_{i=1}^j t_i < T - \tau\}.$$

Note that in defining m_1 and m_3 we add the smallest jobs, whereas in defining m_2 we are adding the largest jobs. Theorem 2 provides values of n_1 and the multipliers $(\lambda_1, \lambda_2, \lambda)$ that are sufficient to evaluate \underline{Z}^* .

Theorem 2: The lower bound \underline{Z}^ in (20) can be computed by restricting the range of search*

to, a) For $Z_1(n_1, \lambda_1, \lambda_2)$:

$$a1) \max\{m_2, m_0\} \leq n_1 \leq m_1,$$

$$a2) \lambda_1 \in \langle 0, N - n_1 \rangle, \text{ and}$$

$$a3) \lambda_2 \in \langle 0, n_1 - 1 \rangle.$$

b) For $Z_2(n_1, \lambda)$:

$$a1) N - m_1 - 1 \leq n_1 \leq m_3 \text{ and}$$

$$a2) \lambda \in \langle 0, N - n_1 \rangle.$$

Proof: The proof of this theorem is given in the Appendix. «»

We now outline the solution procedure to obtain the lower bound \underline{Z}^* .

Algorithm DDLB

Step 1. Compute $\underline{Z}^*_{11}(n_1) = \max\{Z^*_{11}(n_1, \lambda_1, 0) : \lambda_1 \in \langle 0, N - n_1 \rangle\}$ for $n_1 \in \langle \max\{m_2, m_0\}, m_1 \rangle$.

Step 2. Compute $\underline{Z}^*_{12}(n_1) = \max\{Z^*_{12}(n_1, 0, \lambda_2) : \lambda_2 \in \langle 0, n_1 - 1 \rangle\}$ for $n_1 \in \langle \max\{m_2, m_0\}, m_1 \rangle$.

Step 3. Set $\underline{Z}^*_1(n_1) = \min\{Z^*_{11}(n_1), Z^*_{12}(n_1)\}$ for $n_1 \in \langle \max\{m_2, m_0\}, m_1 \rangle$.

Step 4. Compute $\underline{Z}^*_2(n_1) = \max\{Z^*_1(n_1, \lambda) : \lambda \in \langle 0, N - n_1 \rangle\}$ for $n_1 \in \langle N - m_1, m_3 \rangle$.

Step 5. Set $\underline{Z}^* = \min(\min_{n_1}\{Z^*_1(n_1)\}, \min_{n_1}\{Z^*_2(n_1)\})$.

To obtain the time complexity, we note that functions $Z^*_{11}(n_1, \lambda_1, 0)$, $Z^*_{12}(n_1, 0, \lambda_2)$, and $Z^*_1(n_1, \lambda)$ are piece-wise concave function in the multipliers (Nemhauser and Wolsey, 1988). Thus for a given value of n_1 , $\max\{Z^*_{11}(n_1, \lambda_1, 0) : \lambda_1 \in \langle 0, N - n_1 \rangle\}$ can be computed by performing a binary search over the feasible range of λ_1 . This will require $O(\log N)$ computations of $Z^*_{11}()$. Computation of $Z^*_1()$ requires finding the weights $\{w_i = \lambda_1 + i - 1 : i = 1 \dots n_1\} \cup \{v_i = i - 1 - n_1 : i = n_1 + 1 \dots N\}$ and sorting them. Since $w_i < w_{i+1}$ and $v_i < v_{i+1}$ finding a sorted list of these two can be done in $O(N)$ time and hence $Z^*_1()$ can be

computed in $O(N)$ time. Consequently, $\max_{\lambda_2} \{ Z^*_1(n_1, 0, \lambda_2) \}$ can be computed in $O(N \log N)$ time. A similar argument shows that $\max \{ Z^*_1(n_1, 0, \lambda_2) \}$ and $\max \{ Z^*_2(n_1, \lambda) \}$ can also be computed in $O(N \log N)$ time. Since $n_1 \leq N$, the total complexity is $O(N^2 \log N)$.

Upper Bound: While solving each problem in steps 1, 2, and 4, an upper bound is computed as follows. Given the assignment of jobs, if the assignment is feasible (the sum of the processing times of the n_1 jobs assigned to the first due-date is less than or equal to τ) then the cost of the schedule is computed using (5) or (6). Finally, the lowest cost feasible assignment gives an upper bound.

5. Empirical Analysis

To investigate the effectiveness of the proposed lower and upper bounds we coded the heuristic algorithm in the Pascal programming language on a personal computer (Macintosh IIcx). A 3^4 experimental design was used involving number of jobs (N); processing times of jobs; tightness of due-date; and the due-date penalty. The values of N were (20, 30, 40). The processing times (integer) of the jobs were generated from a uniform distribution between (1, t_{max}) with t_{max} being (10, 20, 30). The tightness of the due-date refers to the relative amount of idle time. We set the time period $\tau = \alpha \sum t_i / 2$ where α is a parameter with values (1.1, 1.3, 1.4). Here $\alpha = 1.1$ implies 10% idle time in the data. The earliness penalty was set to 1 and (0.1, .75, 1.25) were the values used for the due-date penalty per job per time period. Thus 81 different problems were studied. For each problem five replications were done. Lower bounds were computed using the algorithm DDLB and upper bounds were obtained as per the method outlined earlier.

Table 1 summarizes the average and the maximum values of the relative gap, measured as $100 * (\text{upper bound} - \text{lower bound}) / \text{lower bound}$. As Table 1 indicates, the lower and the upper bounds are very close. The maximum gap for the 405 problems we solved was 0.24%. For over 65% of the problems, optimal solution was found. The average time for the 40 job problems was approximately 14 seconds.

6. Conclusion

This paper introduces a new scheduling problem with due-date and earliness costs that has applications in JIT environments. The problem with two due-dates was shown to be NP-hard. A linear time algorithm for a special case and a lower bound for the general case were provided. Computational experience suggests that these bounds are quite tight.

There are several possible extensions, some of which are subject of our on-going research.

- (i) The extension of this methodology to more than two due-dates and the identification of special cases that are solvable in polynomial time.
- (ii) Variable delivery intervals and the possibility of dispatching additional trucks with extra cost or perhaps not making a trip in some time periods.
- (iii) Truck capacity constraints, job dependent earliness penalties, and different release times of jobs.

Appendix A

Problem Complexity

We show that 2DSP is NP-hard by reducing the NP-hard even-odd partitioning problem (Garey, Tarjan, and Wilfong, 1988) to 2DSP in polynomial time.

Even-Odd Partitioning Problem (EOPP): *Given $2n$ numbers, $x_1 < x_2, \dots, < x_{2n}$, does there exist a partition X_1 and X_2 of these numbers such that $\sum_{x_i \in X_1} x_i = \sum_{x_i \in X_2} x_i$ and exactly one of $\{x_{2i-1}, x_{2i}\}$ is in X_1 (and so in X_2) for $i=1, \dots, n$.*

Given an instance of EOPP, we define an instance of 2DSP with $2n$ jobs. The processing time of the jobs are, $t_i = x_i + \mu$, for $i = 1, \dots, 2n$ with $\mu = \sum_{i=1}^{2n} x_i$ and the time period $\tau = \frac{1}{2} \sum_{i=1}^{2n} t_i = (n + \frac{1}{2})\mu$. Observe that $t_1 < t_2, \dots, < t_{2n}$ and the total processing time of the $2n$ jobs is $\sum_{i=1}^{2n} t_i = 2\tau$. So, it is possible to complete all the jobs in the first two time periods and no idle time is possible. We set the earliness penalty to 1 and the due-date penalty $\delta = 2^\mu$. We consider the decision version of 2DSP, where we want to find a schedule with cost $\leq \Theta = 3n\tau 2^\mu + \sum_{i=1}^n (t_{2(n-i+1)} + t_{2(n-1)+1})$, if it exists. We show that solving this decision problem gives us a solution to the EOPP, thus showing that decision version of 2DSP is NP-complete.

Since the due-date penalty is large, there is an incentive to assign more number of jobs to the first due-date. However, it is not possible to complete more than n jobs in the first period since the sum of the processing times of the smallest $(n+1)$ jobs is $\sum_{i=1}^{n+1} t_i = (n+1)\mu + \sum_{i=1}^{n+1} x_i > \tau$. As $\sum_{i=1}^n t_i < \tau$, the optimal schedule will have n jobs assigned to each of two due-dates. We now analyze the two cases considered in section 2 with the value of n_1 fixed to n .

Case I: The completion time of the n^{th} job coincides with τ . The total penalty of such a schedule is given by (5), which in this case, is:

$$Z_1(n) = \sum_{i=1}^n (i-1)t_{[i]} + \sum_{i=n+1}^{2n} (i-1-n) t_{[i]} + 3n\tau\delta = \sum_{i=1}^n (i-1)(t_{[i]} + t_{[n+i]}) + 3n\tau\delta.$$

The optimization problem can be written as,

$$\min Z_1(n)$$

S.T. (2) and

$$C_{[n]} = \tau. \quad (A1)$$

Let $Z_1^*(n)$ the optimum solution value to the above problem and let $\underline{Z}_1^*(n)$ be the minimum value of $Z(c1)$ with constraint (A1) relaxed to $C_{[n]} \leq \tau$.

Case II: $C_{[n]} < \tau$. To compute the total cost of such a schedule consider (1),

$$\begin{aligned} & \sum_{i=1}^{2n} (\tau - C_{[i]}) + \sum_{i=n+1}^{2n} (2\tau - C_{[i]}) + (4n-n)\tau\delta \\ &= \sum_{i=1}^n (C_{[n]} - C_{[i]}) + \sum_{i=n+1}^{2n} (C_{[2n]} - C_{[i]}) + 3n\tau\delta + n(\tau - C_{[n]}) \\ &= \sum_{i=1}^n \sum_{j=i+1}^n t_{[j]} + \sum_{i=n+1}^{2n} \sum_{j=i+1}^{2n} t_{[j]} + 3n\tau\delta + n(\tau - C_{[n]}) \\ &= \sum_{i=1}^n (i-1) t_{[i]} + \sum_{i=n+1}^{2n} (i-1 - n) t_{[i]} + 3n\tau\delta + n(\tau - C_{[n]}) \\ &= \sum_{i=1}^n (i-1) (t_{[i]} + t_{[n+i]}) + n(\tau - C_{[n]}) + 3n = Z_2(n). \end{aligned} \quad (A2)$$

Thus, the optimization problem for Case II is:

$$\min Z_2(n)$$

S. T. (2) and

$$C_{[n]} < \tau. \quad (A3)$$

Let $\underline{Z}_2^*(n)$ be the minimum value of $Z_2(n)$ with the constraint (A3) relaxed to $C_{[n]} \leq \tau$.

We note that $\underline{Z}_1^*(n)$ minimizes the first term of (A2). Since the second term in (A2) is non-negative, $\underline{Z}_1^*(n) \leq \underline{Z}_2^*(n)$. We now show that a minimum cost solution to 2DSP with cost C^* is possible if and only if EOPP has a solution. Furthermore, this optimal solution to 2DSP will satisfy Case I. We note that $Z_1(n)$ is multiplication of two series of $2n$ numbers. One series is formed by the processing times of the $2n$ jobs and the other series consists of integers $\{0, \dots, n-1\}$ with each integer repeated twice. To minimize this

sum of products (without constraint (A1)), we have to arrange one series in increasing order and the other in decreasing order. This is equivalent to assigning jobs $\{1,2\}$ to position $\{n,2n\}$, jobs $\{3,4\}$ to positions $\{n-1,2n-1\}, \dots$, and jobs $\{2n-1,n\}$ to positions $\{1,n+1\}$. This gives $Z_1^*(n) = 3n\tau 2^\mu + \sum_{i=1}^n (t_{2(n-i+1)} + t_{2(n-1)+1}) = \Theta \leq \min \{Z_2^*(n), Z_1^*(n)\}$. If such an assignment also satisfies (A1) then the resulting schedule is optimal. (A1) is satisfied if and only if this assignment can be made such that $\sum_{i=1}^n t_{[i]} = \sum_{i=n+1}^{2n} t_{[i]} = \tau$. Since $t_{[i]} = a_{[i]} + \mu$, such an assignment is possible if and only if EOPP has a solution.

Appendix B

Proof of Theorem 2:

(a1) For Case I, it is clear that at least as many jobs are assigned to the first due-date as in the second. (Otherwise interchanging the due-dates of all jobs gives a better solution.) Thus n_1 should be at least equal to m_0 . We note that m_2 is the minimum number of jobs required to satisfy (12) and so n_1 cannot be smaller than m_2 . Therefore, n_1 has to be at least the maximum of m_2 and m_0 .

Since the sum of the smallest (m_1+1) jobs violates (11), n_1 cannot exceed m_1 .

(a2) By Lemma 1, if $\lambda_1 > 0$, $\lambda_2 = 0$. By Lemma 2 we restrict λ_1 to non-negative integers.

Consider (13) and let the positional weights be,

$$w_i = \lambda_1 + i - 1 \quad i=1, \dots, n_1$$

$$v_i = i - 1 \quad i=1, \dots, N - n_1.$$

Note that $w_1 < w_2 < \dots < w_{n_1}$ and $v_1 < v_2 < \dots < v_{N-n_1}$. If $\lambda_1 \geq N - n_1$, $v_{N-n_1} \leq w_1$ and thus the smallest n_1 jobs are assigned to the first due-date to get $Z_1^*(n_1, \lambda_1, 0)$. This optimal value is equal to,

$$\begin{aligned} Z_1^*(n_1, \lambda_1, 0) &= \sum_{i=1}^{n_1} w_i t_{n_1+1-i} + \sum_{i=1}^{N-n_1} v_i t_{N+1-i} + (2N-n_1)\tau\delta - \lambda_1\tau \\ &= \lambda_1 \left(\sum_{i=1}^{n_1} t_i - \tau \right) + \text{Constant} \end{aligned}$$

Since n_1 satisfies (a1), $\sum_{i=1}^{n_1} t_i \leq \tau$, and so $Z_1^*(n_1, \lambda_1, 0) \geq Z_1^*(n_1, \lambda_1+1, 0)$ for $\lambda_1 \geq N-n_1$.

Therefore $\max_{\lambda_1} Z_1^*(n_1, \lambda_1, 0) = \max_{\lambda_1 \leq N-n_1} Z_1^*(n_1, \lambda_1, 0)$.

(a3) Again by Lemma 2, when $\lambda_2 > 0$, it is sufficient to set $\lambda_1 = 0$ and by Lemma 2, integer values of λ_2 are sufficient. Consider (18) and let

$$\begin{aligned} w_i &= -\lambda_2 + i - 1 & i=1, \dots, n_1 \\ v_i &= i - 1 & i=n_1+1, \dots, N. \end{aligned}$$

Note that $w_1 < w_2 < \dots < w_{n_1}$ and $v_1 < v_2 < \dots < v_{N-n_1}$. If $\lambda_2 \geq n_1 - 1$, $v_1 \geq w_{n_1}$. Therefore, the largest n_1 jobs are assigned to the first due-date. This gives

$$\begin{aligned} Z_2^*(n_1, 0, \lambda_2) &= \sum_{i=1}^{n_1} w_i t_{N-i+1} + \sum_{i=n_1+1}^N v_{i-n_1} t_{N-i+1} + (2N-n_1)\tau\delta - \lambda_2(\tau-T) \\ &= \lambda_2(T - \tau - \sum_{i=1}^{n_1} t_{N-i+1}) + \text{Constant} \end{aligned}$$

As $\sum_{i=1}^{n_1} t_{N-i+1} > T - \tau$ when n_1 satisfies (a1), $Z_2^*(n_1, 0, \lambda_2) \geq Z_2^*(n_1, 0, \lambda_2+1)$ for $\lambda_2 \geq n_1-1$.

Therefore $\max_{\lambda_2} Z_2^*(n_1, 0, \lambda_2) = \max_{\lambda_2 \leq n_1-1} Z_2^*(n_1, 0, \lambda_2)$.

(b1) The argument for the upperbound on n_1 is similar to the corresponding case in (a1).

To show that $N-m_1-1 \leq n_1$, note that the $N-n_1$ jobs assigned to the second due-date must be such that the sum of their processing times exceed τ (to satisfy constraint (16)). m_1 is the maximum number of jobs which can be assigned to a due-date without their total processing time exceeding τ and so m_1+1 is the maximum number of jobs which can be assigned to the second due-date. Thus, the minimum number of jobs which can be assigned to the first due-date is $N-m_1-1$.

(b2) The proof of this is similar to (a2).

Table 1

		$\alpha=1.1$			$\alpha=1.3$			$\alpha=1.5$		
tmax		$\delta=0.1$	$\delta=.75$	$\delta=1.25$	$\delta=0.1$	$\delta=.75$	$\delta=1.25$	$\delta=0.1$	$\delta=.75$	$\delta=1.25$
N=20	10	0 0	.076 .24	.052 .16	0 0	0 0	0 0	0 0	0 0	0 0
	20	0 0	.13 .21	.084 .13	0 0	.032 .11	.02 .07	0 0	0 0	0 0
	30	.002 .011	.024 .12	.031 .07	.012 .03	0 0	.016 .08	.01 .05	0 0	.011 .038
N=30	10	0 0	.036 .18	.024 .12	0 0	.012 .06	0 0	0 0	.012 .06	.008 .04
	20	0 0	.072 .33	.048 .22	0 0	.044 .13	.029 .085	0 0	0 0	0 0
	30	0 0	.11 .19	.073 .12	0 0	.025 .04	.017 .03	0 0	.006 .03	.04 .2
N=40	10	0 0	0 0	0 0	0 0	.026 .07	.018 .05	0 0	.006 .03	.004 .02
	20	0 0	.049 .14	.035 .1	0 0	.016 .03	.01 .02	0 0	0 0	0 0
	30	0 0	.07 .122	.044 .08	0 0	.008 .02	.006 .01	0 0	0 0	0 0

$$\text{gap} = 100 * (\text{ub} - \text{lb}) / \text{lb}$$

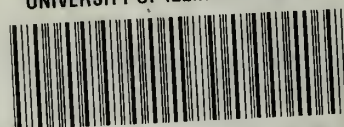
First Number: Average gap of 5 replications

Second Number: Maximum gap of 5 replications

References

- Bagchi, U., R.S. Sullivan, and Y.L. Chang, "Minimizing Mean absolute Deviation of Completion Times About a Common Due Date," *Naval Research Logistics Quarterly*, 1986, 33, 227-240.
- Baker, K.R. and G.D. Scudder, "Sequencing with Earliness and Tardiness Penalties: A Review," *Operations Research*, 1990,38,22-36.
- Chand, S. and D. Chhajed, "Determination of Optimal Due Dates and Sequence," Working Paper 90-1714, Department of Business Administration, University of Illinois, Champaign, IL.
- Garey, M.R., R.E. Tarjan, and G.T. Wilfong, "One-processor Scheduling with Earliness and Tardiness Penalties," *Mathematics of Operations Research*, 1988, 13, 330-348.
- Hall, N.G., W. Kubiak, and S.P. Sethi, "Earliness-Tardiness Scheduling Problems, II: Deviation of Completion Times About a Restrictive Common Due Date," 1989, to appear in *Operations Research*.
- Kanet, J.J., "Minimizing the Average Deviation of Job Completion Times About a Common Due Date," *Naval Research Logistics Quarterly*, 1981,28,643-651.
- Ahmadi, R. and U. Bagachi, "Single Machine Scheduling to Minimize Earliness Subject to Deadlines," Working Paper 86/86-4-17, Department of Management, University of Texas, Austin.
- Chand, S. and H. Schneeberger, "Single Machine Scheduling to Minimize Weighted Earliness Subject to no No Tardy Jobs, *Eor. J. Opnl. Res.*, 1984, 34, 221-230.
- Fisher, M.L., "The Lagrangian Relaxation Method for Solving Integer Programming Problems," *Management Science*, 1981, 27,1-18.
- Matsuo, H., "The Weighted Total Tardiness Problem with Fixed shipping Times and Overtime Utilization," *Operations Research*, 1988, 36, 293-307.
- Nemhauser, G.L. and L.A. Wolsey, *Integer and Combinatorial Optimization*, 1988, John Wiley & Sons.
- Panwalkar, S.S., M.L. Smith, and A. Seidmann, "Common Due Date Assignment to Minimize Total Penalty for the Once Machine Scheduling Problem," *Operations Research*, 1982,30, 391-399.

UNIVERSITY OF ILLINOIS-URBANA



3 0112 005706103