Division of Research
Graduate School of Business Administration
The University of Michigan

April 1985

SINGLE MACHINE SCHEDULING TO MINIMIZE WEIGHTED EARLINESS SUBJECT TO NO TARDY JOB

Working Paper No. 423

Suresh Chand
Purdue University
and
Hans Schneeberger
The University of Michigan

ABSTRACT

This paper considers the Single-Machine Scheduling Problem where the objective is to minimize the total weighted earliness subject to no tardy jobs. Such problems arise in "just-in-time" kind of production environments where customers accept delivery of their jobs exactly at the specified due dates -- not before and not after. (On the shop floor, a work station receiving partially completed parts from other work stations is a customer.) If a job is completed earlier than its due date then the shop holds the job and incurs holding cost for earliness. Jobs are not permitted to be tardy. Both exact and approximate methods are developed to solve this problem.

1. Introduction

This paper considers a Single-Machine Scheduling problem that a production shop is likely to face in a "just-in-time" kind of production environment. We assume that the due dates for the jobs are known and that the customers accept delivery of their jobs exactly at the specified due dates. The shop incurs a holding cost for early jobs; jobs are not allowed to be tardy. An ideal solution for such a scheduling problem would be to complete the jobs exactly at their due dates. However, if the shop capacity is tight and the due dates for several jobs are clustered together, then it may not be possible to schedule the jobs to finish exactly at their due dates. As a result, the shop may have to schedule some jobs early and incur a holding cost for earliness. Thus, the scheduling problem under consideration requires minimizing the total holding cost for early jobs (or the total weighted earliness) subject to no tardy jobs.

Let P_i, U_i, d_i, and C_i denote the processing time, weight, due date, and completion time, respectively, for job i. Let N be the number of jobs. The problem of finding the optimal values of $C_1, C_2, \ldots, C_N$ can be formulated as follows:

$$\text{Minimize} \quad \sum_{i=1}^{N} (d_i - C_i)U_i \tag{1}$$

$$\text{subject to} \quad C_i - d_i \leq 0 \quad \text{for } i=1,2,\ldots,N. \tag{2}$$

We will call this problem the Weighted Earliness Problem (WE-Problem). The problem with the added constraint that no machine idle time is allowed, that is:

$$C_i = \sum_{j=1}^{N} P_j$$

for i=1,2,...,N, will be referred to as the Constrained Weighted Earliness Problem (CWE-Problem). In both these problems and the rest of the paper, we assume that jobs are not allowed to be preempted.

Note that the scheduling criteria introduced above are not regular measures of performance (see Page 13 of Baker [1] for a definition of regular measures). Most scheduling research to date has considered only the problems with regular measures. For example, Baker [1] claims that nearly all important scheduling criteria use regular measures of performance. Thus, this paper identifies and analyzes an important class of scheduling problems which does not use regular measures.

In the next section, we analyze the CWE-Problem and give several results. Section 3 analyzes the WE-Problem and gives a heuristic algorithm to solve the problem. Section 4 gives a dynamic programming algorithm for solving the WE-Problem. Section 5 consists of computational results. The paper closes in Section 6 by giving a brief summary and conclusions. Several theorems and proofs are given in the appendix.

2. Analysis of the CWE-Problem

In this section, we prove several important results for the CWE-Problem. We first establish that the CWE-Problem is equivalent to a well researched single-machine scheduling problem where the objective is to minimize the total weighted completion time subject to no tardy jobs. (We will call this problem

the Constrained Weighted Completion Time Problem or the CWCT-Problem.) Most of the results for the CWE-Problem are developed by using the known results for the CWCT-Problem.

The CWCT-Problem is stated below:

$$\text{Minimize} \quad \sum_{i=1}^{N} A_i C_i$$

subject to $C_i - d_i \leq 0$ for i=1,2,...,N, where A_i is the known weight for job i. The reader is referred to Burns [2], Emmons [4], Heck and Roberts [5], Lenstra et al. [6], Miyazaki [7], Schneeberger [8], Shanthikumar and Buzacott [9], and Smith [10] for several results and algorithms for the CWCT-Problem.

Equivalent CWCT-Problems

A given CWE-Problem will be considered equivalent to a CWCT-Problem if these differ from each other only by a constant term in the objective function. The following result proved in Chand and Schneeberger [3] suggests a transformation process to find equivalent CWCT-Problems for a given CWE-Problem. (We repeat the proof here for the sake of completeness.)

THEOREM 1: The CWE-Problem with weights U_i, i=1,2,...,N, is equivalent to the CWCT-Problem with weights $A_i = qP_i - U_i$, i=1,2,....,N, for any value of q.

Proof: Let $P_{[i]}$ denote the processing time of the job in position i. Then $C_{[i]}$ can be expressed as:

$$C_{[i]} = \sum_{j=1}^{i} P_{[j]}.$$

The objective function for the CWCT-Problem can be written as:

$$V = \sum_{i=1}^{N} (qP_i - U_i)C_i$$

$$= \sum_{i=1}^{N} U_i(-C_i) + q \sum_{i=1}^{N} P_i C_i$$

$$= \sum_{i=1}^{N} U_i(d_i - C_i) + q \sum_{i=1}^{N} P_i C_i - \sum_{i=1}^{N} U_i d_i$$

= The objective function for the CWE-Problem

$$+ q \sum_{i=1}^{N} P_i C_i = \sum_{i=1}^{N} U_i d_i$$

Note that

$$q \sum_{i=1}^{N} P_i C_i = q \sum_{i=1}^{N} \sum_{j=1}^{i} P_{[i]} P_{[j]}$$

$$= q \sum_{i=1}^{N} \sum_{j=1}^{i} P_i P_j$$

= constant

and

$$\sum_{i=1}^{N} U_i d_i = \text{constant.}$$

Thus, the objective functions for the CWE- and the CWCT-Problems differ only by a constant. This completes the proof.

Later on, we extend the analysis for the CWCT-Problem in Chand and Schneeberger [3] to show that the CWE-Problem is NP-hard. As in [3], we also

identify some special cases of the CWE-Problem that can be solved optimally in polynomial time. Before that, we make some general observations about solving the CWE-Problem.

Procedures for Solving the CWE-Problem

Several exact and approximate procedures have been reported for solving the CWCT-Problem [2,4,6,7,8,9,10]. Most of these procedures require $A_i \geq 0$ for all i=1,2,...,N. These procedures can be extended to solve the CWE-Problem by transforming it into an equivalent CWCT-Problem with q large enough such that $A_i = qP_i - U_i \geq 0$ for i=1,2,...,N.

Many procedures for solving the CWCT-Problem require testing the $P_k/A_k \geq P_j/A_j$ - condition. The following lemma gives an efficient procedure to test this condition for the CWE-Problem.

Lemma 1: With $A_i = qP_i - U_i$, $A_i \geq 0$ for i=1,2,...,n, the condition $P_k/A_k \geq P_j/A_j$ is equivalent to $P_k/U_k \leq P_j/U_j$.

Proof for this lemma is straightforward, so it has been omitted. Using this result, we extend the Smith-heuristic [10] for the CWCT-Problem to solve the CWE-Problem. In the rest of the paper, we define $R_i = P_i/U_i$.

Modified Smith-heuristic for the CWE-Problem

(a) Arrange the jobs in the order of increasing due dates. It is assumed that all jobs are on time in this schedule; if not, then the problem does not have a feasible solution.

(b) Find all the jobs with their due dates greater than or equal to the total processing time for all jobs yet to be scheduled; these jobs are feasible candidates for the last position in the sequence.

(c) Find the job with the smallest R_i-value among all the feasible jobs and schedule it last.

(d) Reduce the set of jobs yet to be scheduled by one, by removing the job just scheduled in (c). Go to (b) until all jobs have been scheduled by this method.

As in Burns [2], it can be shown that this algorithm gives a locally pairwise optimal schedule. The computational time is bounded by O(NlogN).

Complexity of the CWE-Problem

Schneeberger [8] showed that the CWCT-Problem with $A_i = qP_i - U_i$, $q > 0$, $U_i > 0$, for $i=1,2,\ldots,N$, is NP-hard even for the special case when $U_1 = U_2 = \ldots = U_N$. Since the CWCT-Problem with $A_i = qP_i - U_i$, $i=1,2,\ldots,N$, is equivalent to the CWE-Problem with weights $U_1, U_2, \ldots, U_N$, the CWE-Problem is also NP-hard. And, it remains NP-hard even when $U_1 = U_2 = \ldots = U_N$.

As in Chand and Schneeberger [3], we identify several cases of the CWE-Problem that can be solved optimally by using the modified Smith-heuristic. We will refer to these as the easy problems. In the description of these problems, jobs are numbered such that $P_1 \leq P_2 \leq \ldots \leq P_N$.

Easy Problem 1: The weights for different jobs are an increasing convex function of their processing times and

$$\frac{P_2 - P_1}{U_2 - U_1} \leq \frac{P_N}{U_N}.$$

Easy Problem 2: The weights for different jobs are an increasing concave function of their processing times and

$$R_N \geq R_i \ , \ i=1,3,\ldots,N-1.$$

Easy Problem 3:

$$d_i \geq \sum_{j=1}^{N} P_j, \ i=1,2,\ldots,N.$$

Easy Problem 4: For every pair of jobs i and j if

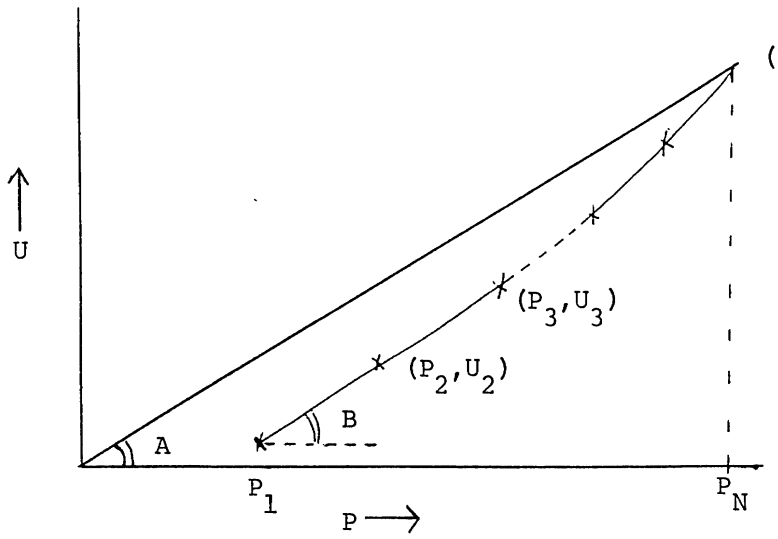
$$R_i \geq R_j \text{ then } d_i \leq d_j.$$

See Figures 1 and 2 for a pictorial description of easy problems 1 and 2; proofs for these follow from Theorem 2 in the Appendix. Note that easy problems 1 and 2 imply that the CWE-Problem with $P_1 = P_2 = \ldots = P_N$ is also an easy problem. For easy problems 3 and 4, the sequence found by the modified Smith-heuristic satisfies the conditions of Lemma 2 (in the Appendix) and therefore it is optimal.

3. Analysis and a Heuristic Algorithm for the WE-Problem

We first argue that the WE-Problem is NP-hard. Note that any CWE-Problem can be transformed into a WE-Problem simply by replacing the due dates exceeding $\sum_{j=1}^{N} P_j$ by $\sum_{j=1}^{N} P_j$. Since the CWE-Problem is NP-hard, the WE-Problem is also NP-hard.

We now propose an extension of the Smith-heuristic to solve the WE-Problem. Later, we will identify cases of the WE-Problem for which this



Slope B $\geq$ Slope A

Figure 1: Easy CWE-Problem with Convex Cost

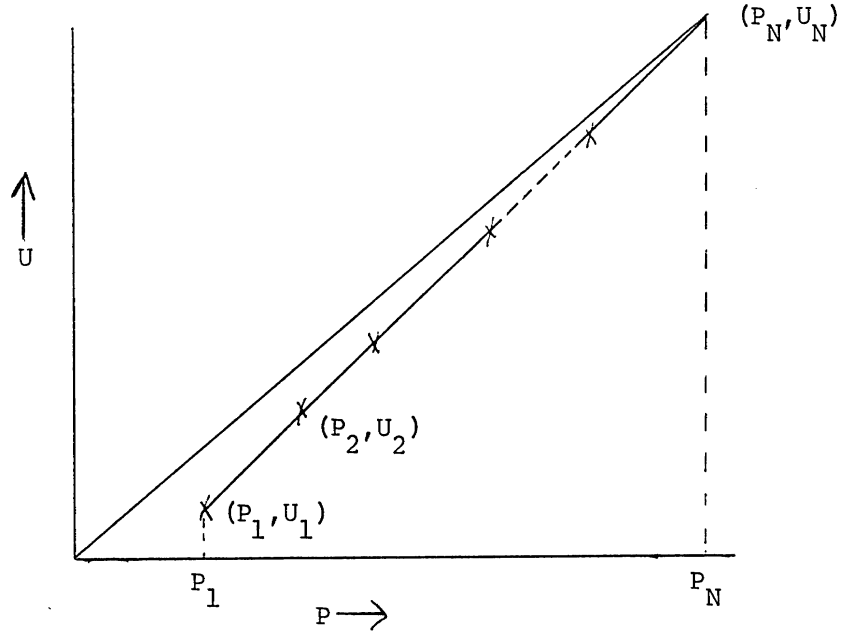


Figure 2: Easy CWE-Problem with Concave Cost

heuristic is optimal. In the description of this algorithm and the rest of the paper, we let $[1,N] = \{1,2,\ldots,N\}$. Recall that $R_i = P_i/U_i$.

Modified Smith-heuristic (MSH) for the WE-Problem:

The algorithm starts scheduling the jobs from the last position. Let S denote the set of jobs scheduled and $YS = [1,N] - S$ the set of jobs yet to be scheduled.

Step 1: (Initialization)

$S = \{\phi\}$, $YS = [1,N]$, $T = \max_{i\in[1,N]} \{d_i\}$, and $\ell = N$.

Step 2: (Finding job j to be scheduled in position ℓ with $C_j = T$)

Find job j such that

$$R_j = \min_{i \mid d_i \geq T,\ i\in YS} \{R_i\}$$

Step 3: $C_j = T$, $S = SU\{j\}$, $YS = YS - \{j\}$, $\ell = \ell - 1$, and

$$T = \min \{C_j - P_j, \max_{i\in YS} \{d_i\}\}.$$

Step 4: Stop if $\ell = 0$, otherwise go to Step 2.

Theorem 3 in the Appendix gives sufficient conditions for the optimality of a sequence found by this heuristic for the WE-Problem. It is straightforward to see that the following three problems satisfy the condition in Theorem 3 and therefore these are easy WE-Problems.

Case 1: $R_1 = R_2 = \ldots = R_N$.

Case 2: For any pair of jobs i and j if $R_i \geq R_j$ then $d_i \leq d_j$.

Case 3: $d_1 = d_2 = \ldots = d_N$.

In the next section, we present a dynamic programming algorithm for the WE-Problem.

4. Dynamic Programming Algorithm for the WE-Problem

We develop a Dynamic Goal Programming (DGP) algorithm for the WE-Problem. The algorithm is similar to the Bicriteria-DP algorithm except that we are not concerned with obtaining the efficient frontier. Rather, we seek the solution that minimizes the first criteria subject to a certain minimum level of achievement on the second criteria. We will refer to these as the objective function and the goal constraint, respectively. The objective is to minimize the total weighted earliness, that is:

$$\text{Minimize} \quad \sum_{i=1}^{N} (d_i - C_i)U_i$$

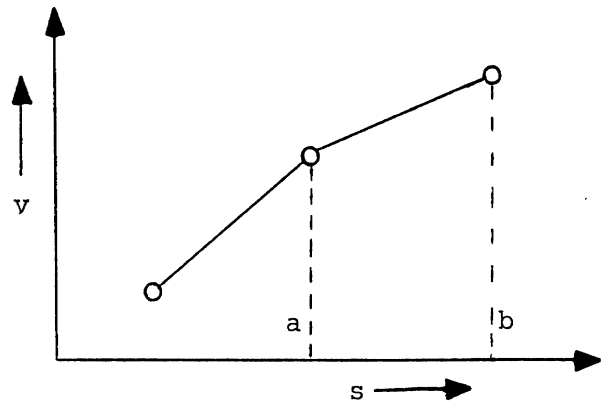
and the goal constraint enforces the time zero feasibility, that is,

$$\min_{i \in [1,N]} (C_i - P_i) \geq 0.$$

In addition, we have the regular constraint (2) requiring no tardy jobs.

The DGP algorithm starts by scheduling the jobs from the last position. Let S denote the set of scheduled jobs. Because of the possibility of machine idle time in the optimal solution, several sequences of jobs in S can be candidates for optimality. The starting time of the first job in S will

depend on the machine idle time. For this reason, our DP formulation requires storing all points on the efficient frontier for state S instead of just one function value for it. The two dimensions that define the efficient frontier are v and s, where v corresponds to the objective function value and s corresponds to the achievement level of the goal constraint or the start time of the first job. We set $v = \infty$ if $s < 0$. The efficient frontier for a state S may look as follows:



Efficient Frontier

Note that the points which are not on the efficient frontier for S cannot be optimal for the entire problem. Thus, in the DGP formulation, we only have to store the points on the efficient frontier.

The DGP algorithm starts with $S = \{\phi\}$ and the corresponding $v = 0$ and $s = \max_{i\in[1,N]} \{d_i\}$. We add jobs to S by using the reaching operation described below and find efficient frontiers for larger and larger S sets. The algorithm stops when we have the efficient frontier for $S = \{1,2,\ldots,N\}$.

Reaching Operation in DGP: Let S^j denote the sequence of jobs in S for the j^{th} data point on the efficient frontier for S. $v(S^j)$ and $s(S^j)$ denote the objective function value for the jobs in S and the start time of the first job, respectively, for S^j. Let $K(S^j)$ denote the jobs which are candidates for scheduling as immediate predecessors to the jobs in S^j

in the optimal solution to the entire problem. A procedure to find $K(S^j)$ is proposed below:

Let $YS = \{1,2,\ldots,N\} - S$,

$$t = \min \{s(S^j), \max_{i \in YS} \{d_i\}\},$$

$$L = \{\ell \mid d_\ell \geq t, \ell \in YS\}, \text{ and}$$

$$\Delta = \min_{i \in L} \{P_i\}$$

then $$K(S^j) = \{k \mid d_k \geq t - \Delta, k \in YS\}. \quad (3)$$

For the case where $t = s(S^j)$, Δ is used to allow for machine idle time immediately before the first job in S^j. In this case, it is easy to see that, in the optimal sequence for the entire problem, this machine idle time cannot exceed Δ. For the case where $t < s(S^j)$, Δ is used to allow for the machine idle time in excess of $s(S^j) - t$.

Consider reaching from the j^{th} data point of the efficient frontier of state (S-k) to state S. This reaching operation is performed only if $k \in K(S-k)^j$. Let sequence $S^o = (k,(S-k)^j)$, then $v(S^o)$ and $s(S^o)$ can be computed as follows:

$$s(S^o) = \min \{s((S-k)^j) - P_k, d_k - P_k\} \quad (4)$$

$$v(s^o) = V((s-k)^j) + (d_k - P_k - s(S^o))\, U_k. \quad (5)$$

After performing all the possible reaching operations to state S, we have several sequences of the jobs in S with the corresponding v ( )- and s( )-values. The efficient frontier for S can be found by dropping all the dominated sequences and retaining only the non dominated ones. (Sequence S^b dominates sequence S^a if $v(S^a) \geq v(S^b)$ and $s(S^a) \leq s(S^b)$. Thus, S^a and S^b belong to the efficient frontier only if either $v(S^a) \geq v(S^b)$ and $s(S^a) \geq s(S^b)$ or $v(S^a) \leq v(S^b)$ and $s(S^a) \leq s(S^b)$.)

We now give steps of the DGP algorithm. In this, we let n denote the stage, defined as the number of scheduled jobs.

Step 1: (Initialization)

$n = 0$, $S = \{\phi\}$, S^1 = no jobs, $v(S^1) = 0$, $s(S^1) = \max_{i \in [1,N]} \{d_i\}$

Step 2: (Reaching)

Find $K(S^j)$ for every data point S^j on the efficient frontier for state S and for every State S at stage n (using (3)).

Step 3: (Efficient Frontiers for States at Stage n+1)

The states for consideration at stage (n+1) are found by combining the elements of $K(S^j)$ with S for every data point S^j on the efficient frontier for S and every state S at stage n. Complete the reaching operation using (4) and (5), retain the nondominated points and find efficient frontiers for all states at stage (n+1).

Set $n = n+1$.

Step 4: Go to Step 2 if $n < N$, stop otherwise. Find the minimum objective function value and the corresponding sequence from the efficient frontier for the set of all jobs.

Example: Consider the following 5-job WE-Problem:

i	P_i	U_i	d_i
1	2	6	11
2	4	2	7
3	2	3	12
4	5	2	17
5	2	4	18

Table 1 summarizes the steps of the solution procedure. Note that the points on the efficient frontier for set S in column (7) are found after completing all the reaching operations to S.

The optimal sequence is 2-3-1-4-5 with the objective function value equal to 11. It is interesting to see that each state has only one point on the efficient frontier.

5. Computational Results

The objective of the computational study in this Section is to analyze the relative performance of the approximate and exact methods for several different scenarios. The best scenarios are those for which the problem data satisfy any of the conditions of cases 1, 2, and 3; that is, where the problem data is such that the modified Smith-heuristic will give an optimal sequence.

The experiment is designed such that the problems generated satisfy cases 1, 2, and 3 on one extreme and then violate these to different degrees. To

Table 1: DGP Solution

Stage n	State S	Data Point from which S is reached	Sequence of Jobs in S S^o	$v(S^o)$	$s(S^o)$	S^o on the efficient frontier?	$K(S^o)$ for points on the efficient frontier
0	{0}			0	18	Yes	4,5
1	{4}	-	4	0	12	Yes	1,3,5
	{5}	-	5	0	16	Yes	3,4
2	{1,4}	4	1-4	0	9	Yes	3,5
	{3,4}	4	3-4	0	10	Yes	1,5
	{4,5}	4	5-4	24	10	No	-
		5	4-5	2	11	Yes	1,3
	{5,3}	5	3-5	0	10	Yes	1,2,4
3	{1,3,4}	1-4	3-1-4	9	7	No	-
		3-4	1-3-4	6	8	Yes	2,5
	{1,4,5}	1-4	5-1-4	36	7	No	-
		4-5	1-4-5	2	9	Yes	3
	{3,4,5}	3-4	5-3-4	32	8	No	-
		4-5	3-4-5	5	9	Yes	1
		3-5	4-3-5	14	5	No	-
	{1,3,5}	3-5	1-3-5	6	8	Yes	2,4
	{2,3,5}	3-5	2-3-5	0	5	Yes	1,4
4	{1,2,3,4}	1-3-4	2-1-3-4	9	5	Yes	5
	{1,3,4,5}	1-3-4	5-1-3-4	46	6	No	-
		1-4-5	3-1-4-5	11	7	Yes	2
		3-4-5	1-3-4-5	17	7	No	-
		1-3-5	4-1-3-5	24	3	No	-
	{1,2,3,5}	1-3-5	2-1-3-5	6	3	Yes	4
		2-3-5	1-2-3-5	36	3	No	-
	{2,3,4,5}	2-3-5	4-2-3-5	24	0	Yes	1
5	{1,2,3,4,5}	2-1-3-4	5-2-1-3-4	51	3	No	-
		3-1-4-5	2-3-1-4-5	11	3	Yes	-
		2-1-3-5	4-2-1-3-5	∞	<0	No	-
		4-2-3-5	1-4-2-3-5	∞	<0	No	-

keep the computational work within reasonable limits, we kept the processing time constant for all jobs ($P_i = 5$). The weights and due dates were generated by using three control parameters (α, β, and D) as described below. These parameters are allowed to assume the following values:

α = 0.0, 0.25, 0.50, 0.75, 1.0

β = 0.0, 0.2, 0.4, 0.6, 0.8, 1.0

D = 0, 25, 50, 75, 85, 95, 100

α : Control parameter for the variance in the P_i/U_i - ratios. This parameter is used to control the variance in the P_i/U_i ratios. The variance increases with increasing α. The lower limit ($\alpha = 0$) implies that the ratio P_i/U_i is constant for all jobs. This implies the best scenario (case 1) and hence the modified Smith-heuristic will give an optimal sequence for $\alpha = 0$.

β : Control parameter for the correlation of P_i/U_i and d_i. This parameter is used to control the correlation between P_i/U_i-ratios and the due dates. If β is at its lower limit ($\beta = 0$) then $R_i < R_j \rightarrow d_i > d_j$ which implies the best scenario (case 2) and hence the modified Smith-heuristic will give an optimal sequence.

D : Control parameter for the due dates

The parameter D is used to control the due dates; a small value of D implies that the due dates are uniformally distributed over certain intervals of time and a large value of D means the due dates are clustered towards the end of the scheduling horizon. The upper limit

for D will give identical due dates for all jobs (case 3) and hence an optimal sequence for the modified Smith-heuristic.

The due dates and weights were generated as follows. Note that N denotes the number of jobs.

i) $d_i = 5N + U(D, 100)$,

where U(a,b) denotes the uniform distribution between a and b. The term 5N has been added to guarantee feasibility. It is easy to see that all due dates are identical (equal to 5N + 100) for D = 100. For D = 0, the due dates are uniformally distributed between 5N and 5N + 100.

ii) $$\frac{P_i}{U_i} = 1 + \alpha \left\{ \begin{array}{lll} -1 & \text{if} & x \leq 1.0 \\ 1 & \text{if} & x < 1.0 \end{array} \right\} \frac{d_i}{100 + 5N},$$

$x = U(\beta, 1 + \beta)$.

By using formula ii), we can generate weights (U_i's) while controlling the variance of P_i/U_i-ratios and also the correlation between P_i/U_i and d_i. For $\alpha = 0$, $P_i/U_i = 1$ for all jobs; the variance of P_i/U_i - ratios increases with increasing α. For $\beta = 0$, $x \leq 1$, and the correlation between P_i/U_i and d_i will be negative. Note that because $d_i \leq 5N + 100$, $P_i/U_i \geq 0$ for all jobs.

We used N = 10 jobs in this experiment. For each combination of parameter values, we generated 10 problems and solved each problem by using the DGP and MSH procedures. In total we used 5 x 6 x 7 x 10 = 2100 test problems. The results for $\alpha = 0$, $\beta = 0$ and D = 100 are not reported because the heuristic is optimal for all these cases. To our surprise, we found that the heuristic is optimal for $\beta = 1$ also; $\beta = 1$ implies that R_i is a linearly increasing function of d_i. (We do not yet have a proof for this empirical observation.) The results for $\beta = 1$ also are not being reported.

Table 2

Percentage Increase in the Average Objective Function value for MSH

D	α	$\beta = .2$	$\beta = .4$	$\beta = .6$	$\beta = .8$
0	.25	0.0	0.7	0.0	0.0
	.50	0.0	2.1	0.0	0.0
	.75	0.0	3.5	0.0	0.0
	1.00	0.0	5.1	0.6	0.0
25	.25	0.0	0.0	0.0	0.0
	.50	0.0	0.0	0.0	0.0
	.75	0.1	0.1	1.0	0.7
	1.00	3.7	3.8	7.6	3.9
50	.25	0.0	0.2	0.0	0.0
	.50	0.0	4.2	1.3	0.0
	.75	2.2	11.8	4.4	4.6
	1.00	19.2	24.9	11.3	13.8
75	.25	0.0	0.0	0.0	0.0
	.50	1.5	0.7	0.5	0.0
	.75	9.2	6.5	2.4	0.0
	1.00	43.8	51.2	6.7	1.7
85	.25	0.0	0.0	0.0	0.0
	.50	0.0	2.5	0.0	0.0
	.75	1.3	6.3	3.6	4.3
	1.00	21.7	54.5	26.9	18.2
95	.25	0.0	1.1	0.4	1.0
	.50	0.0	5.5	4.4	5.2
	.75	0.5	15.3	11.6	11.2
	1.00	60.6	134.2	42.2	28.8

Table 3

Summary of Percentage Increase in the Average Objective Function Value for Different Parameters

TABLE 3A: % Increase for Different α- values

α	0.00	0.25	0.50	0.75	1.00
% Increase	0.00	0.14	0.93	3.23	20.90

TABLE 3B: % Increase for Different β -values

β	0.00	0.20	0.40	0.60	0.80	1.00
% Increase	0.00	6.82	13.92	5.20	3.89	0.00

TABLE 3C: % Increase for Different D-values

D	0	25	50	75	85	95	1000
% Increase	0.75	1.30	6.11	7.76	8.70	20.12	0.00

Table 4

CPU-Time in Seconds for DGP[1]

D	α	β = .2	β = .4	β = .6	β = .8
0	.25	0.059	0.062	0.065	0.063
	.50	0.059	0.062	0.063	0.063
	.75	0.059	0.062	0.063	0.064
	1.00	0.058	0.062	0.063	0.063
25	.25	0.111	0.123	0.122	0.129
	.50	0.126	0.138	0.142	0.148
	.75	0.114	0.122	0.131	0.132
	1.00	0.132	0.144	0.151	0.152
50	.25	0.735	0.786	0.771	0.780
	.50	0.678	0.739	0.726	0.741
	.75	0.790	0.832	0.860	0.890
	1.00	0.782	1.020	1.170	1.179
75	.25	4.238	4.416	4.211	4.271
	.50	5.310	5.437	5.541	5.263
	.75	5.815	6.049	6.323	5.548
	1.00	6.202	7.041	7.433	6.626
85	.25	6.547	6.772	6.405	5.873
	.50	6.946	7.235	6.938	6.545
	.75	7.747	8.244	8.153	7.670
	1.00	8.157	11.317	10.127	9.190
95	.25	6.243	7.366	8.029	8.630
	.50	6.810	8.857	9.366	9.517
	.75	8.855	9.247	9.470	9.595
	1.00	11.900	10.540	10.052	9.937

[1]The CPU-time for MSH was between 0.005 and 0.006 seconds for all problems solved.

Table 2 gives the percentage increase in average objective function value (average computed over 10 problems) for the MSH proceudre over the DGP algorithm. Table 3 summarizes the results in Table 2 for different parameters. Table 4 reports the CPU run times for these two procedures.

Summary and Anaysis of the Computational Results:

Table 3A shows that the performance of the Smith-heuristic deteriorates with increasing α, or, in other words, with the increasing variance of R_i. This heuristic does well for problems with low variance of R_i. This result is as expected.

Table 3B shows that the heuristic performs well for extreme values β, i.e., for the cases when there is strong linear relationship between Ri and di values. For β values in the mid-range (around .5), the relationship between R_i and d_i is weak and the heuristic performs poorly. While we have proved in the paper that the heuristic is optimal for $\beta = 0$, we are not yet able to prove the same for $\beta = 1$.

Table 3C shows that the heuristic gives very good results if either the due dates for all jobs are identical or if these are uniformly distributed over a large interval. The heuristic performs poorly if the due dates are clustered together within a small interval. This result points to a weakness of the heuristic: it does not look ahead while scheduling a job; that is, it does not consider the effect of scheduling a job on the earliness of other jobs. It is easy to see that the machine idle time for the heuristic between the beginning of the first job and the completion of the last job is lower than (or equal to) the machine idle time for the optimal solution. The results in Table 3C imply that the heuristic, while attempting to reduce this idle time, has a tendency to schedule a less costly job (job with small u) at the expense of increased earliness for costly jobs.

With respect to CPU times, the heuristic is very fast. The DGP algorithm is inherently slow. Based on these results, we do not recommend using the DGP algorithm for solving large problems (N > 15) in practice.

Overall, we conclude that the heuristic does well when the due dates are relatively uniformly distributed over a wide range, R_i and d_i have a strong linear relationship, and Ri has a low variance. There is a need to develop better heuristics for the cases when these conditions are not met.

6. Conclusions:

This paper introduced a new single machine scheduling problem where the objective is to minimize the weighted earliness subject to no tardy jobs. This problem is relevant for manufacturing companies using "just-in-time" or "pull" type approaches. It should be remarked that most problems addressed in the scheduling literature so far relate to "push" type approaches.

Both optimal and heuristic procedures are suggested to solve this problem. The optimal procedure is developed to provide a benchmark to study the performance of the heuristics; we do not recommend using of this procedure to solve large problems in practice because of its excessive computational time. Situations where the heuristic performs well are identified. Indeed, there is a need to develop better heuristics for several realistic situations. We expect the heuristic and the computational results in this paper to provide a good starting point for developing better heuristics.

This paper discusses single level problems although most real problems in manufacturing are multi-level in nature. One possible way to solve a multi-level problem would be to decompose it into several single level problems by

using the Lagrangean relaxation approach. The results of this paper could then be used to solve the single level problems. However, much work is required to develop this approach.

References

1. Baker, K. R., Introduction to Sequencing and Scheduling, John Wiley and Sons, Inc., 1974.

2. Burns, R. N., "Scheduling to Minimize the Weighted Sum of Completion Times with Secondary Criteria," Naval Research and Logistics Quarterly, Vol. 23, No. 1 (1976), pp. 725- 729.

3. Chand, S. and H. Schneeberger, "A Note on Smith's Schedule to Minimize the Weighted Sum of Completion Times with Maximum Allowable Tardiness," Working Paper, Purdue University.

4. Emmons, H., "Note on a Scheduling Problem with Dual Criteria," Naval Research and Logistics Quarterly, Vol. 22, No. 4 (1975), pp. 615-616.

5. Heck, H. and S. Roberts, "A Note on the Extension of a Result on Scheduling with Secondary Criteria," Naval Research and Logistics Quarterly, Vol. 19, No. 3 (1972), pp. 403-405.

6. Lenstra, J.K., A.H.G. Rinnooy Kan, and P. Brucker, "Complexity of Machine Scheduling Problems," Annals of Discrete Mathematics, 1 (1979), pp. 343-362.

7. Miyazaki, Shigeji, "One Machine Scheduling Problem with Dual Criteria," Journal of the Operations Research Society of Japan, Vol. 24, No. 1 (1981), pp. 37-50.

8. Schneeberger, H.M., "Job Shop Scheduling in Pull Type Production Environments," Ph.D. Thesis, Krannert Graduate School of Management, Purdue University, West Lafayette, Indiana, May 1984.

9. Shanthikumar, J.G. and J.A. Buzacott, "On the Use of Decomposition Approaches in a Single Machine Scheduling Problem," Journal of the Operations Research Society of Japan, Vol. 25, No. 1 (1982), pp. 29-47.

10. Smith, W.E., "Various Optimizers for Single Stage Production," Naval Research Logistics Quarterly, Vol. 3, No. 1 (1956), pp. 56-66.

Appendix

Recall that [i] denotes the index of the job in position i and that $R_i = P_i/U_i$.

THEOREM 2: Let the jobs be labelled such that $P_1 \leq P_2 \leq \ldots \leq P_N$. Any CWE-Problem with weights $U_i \geq 0$ for all i=1,2,...,N, can be solved optimally using the modified Smith-Heuristic if the following two conditions are met:

$$\text{a)} \quad \frac{P_N}{U_N} \leq \frac{P_{N-1}}{U_{N-1}} \leq \ldots \leq \frac{P_1}{U_1}$$

$$\text{b)} \quad \max_{i\in[1,N-1]} \frac{P_{i+1} - P_i}{U_{i+1} - U_i} \leq \frac{P_N}{U_N}$$

Proof: We need the following result from Schneeberger [8]: any CWCT-Problem with weights $A_i \geq 0$ for all i=1,2,...,N, can be solved optimally using the Smith-heuristic if $P_i \geq P_j$ implies $A_i \leq A_j$ for all pairs of jobs i and j.

Consider the equivalent CWCT-Problem with $A_i = qP_i - U_i$, q sufficiently large such that $A_i \geq 0$ for all i=1,2,...,N. We show that there exists a value of q such that $A_1 \geq A_2 \geq \ldots \geq A_N \geq 0$ under conditions (a) and (b).

Let $\frac{1}{q} = \frac{P_N}{U_N}$. Then from a) we have

$$\frac{1}{q} \leq \frac{P_i}{U_i} \qquad \text{for i=1,2,...,N,}$$

or $qP_i - U_i \geq 0$ for i=1,2,...,N,

or $A_i \geq 0$ for i=1,2,...,N.

From (b) we have

$$\frac{1}{q} \geq \frac{P_{i+1}-P_i}{U_{i+1}-U_i} \quad \text{for } i=1,2,\ldots,N-1,$$

or $qP_i - U_i \geq qP_{i+1} - U_{i+1}$,

or $A_i \geq A_{i+1}$ for i=1,2,...,N-1,

or $A_1 \geq A_2 \geq \ldots \geq A_N$.

Thus, the Smith-heuristic is optimal for the equivalent CWCT-Problem. Because the sequence found by the modified Smith-heuristic for the CWE-Problem is the same as the one found by the Smith-heuristic for the equivalent CWCT-Problem, the modified Smith-heuristic is optimal for the CWE-Problem under conditions (a) and (b). This completes the proof.

Lemma 2: If the sequence found by the modified Smith-heuristic for the CWE-Problem has $R_{[1]} \geq R_{[2]} \geq \ldots \geq R_{[N]}$ then the sequence is optimal.

Proof for this is trivial, so we skip it.

Lemma 3: The modified Smith-heuristic for the WE-Problem is optimal if

$$d_1 = d_2 = \ldots d_N = d \geq \sum_{j=1}^{N} P_j .$$

Proof: Let $t_I = d - \sum_{j=1}^{N} P_j$; t_I denotes the total amount of the machine-idle time. In order to minimize the total weighted earliness, with identical due dates, it is easy to see that the entire idle time of t_I periods should be scheduled in the beginning in an optimal solution. The first job starts at the end of period t_I and all jobs are scheduled from t_I to d without any additional machine idle time. Scheduling the jobs from t_I to d is like solving a CWE-Problem for which the modified Smith-heuristic is optimal. This completes the proof.

THEOREM 3: Consider the following sequence found by the modified Smith-heuristic for the WE-Problem. The sequence has several production blocks as shown in the figure below:

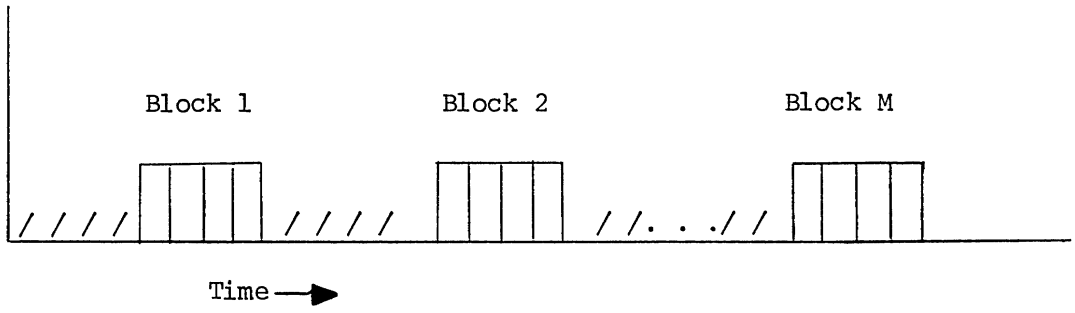


Figure 3: Example for Theorem 3

The sequence is optimal if the R-values for jobs within each production block are in a decreasing order.

Proof: If the sequence has only one production block (with $R_{[1]} \geq R_{[2]} \geq \ldots$ for the jobs in the block) then, from Lemma 3, it is optimal for the relaxed WE-Problem with the due date for every job increased to $\max_{i\varepsilon[1,N]} \{d_i\}$. Clearly, it is optimal for the WE-Problem also.

If the sequence has n production blocks, then consider a relaxed WE-Problem with n identical machines, all jobs in a production block to be processed on one machine, and one machine assigned to the jobs in one production block only. Using the Smith-heuristic for every machine, it is easy to see that each machine has one production block and the resulting schedule is optimal for the relaxed problem. The start and finish times for the jobs in the optimal schedule for the relaxed problem are the same as those in the schedule in Figure 3, implying that the sequence in Figure 3 is optimal for the WE-Problem.

THEOREM 4: Let $d = \max_{i\varepsilon[1,N]} \{d_i\}$. If $C_1^*, C_2^*, \ldots, C_N^*$ is optimal for the WE-Problem then $\underline{C}_i^* = d_{max} - C_i^*$, i=1,2,...,N, is optimal for the weighted flow time problem with nonsimultaneous job arrivals; the arrival time for job i, $r_i = d_{max} - d_i$, and weight for job i equal to U_i.

Proof: Using $r_i = d_{max} - d_i$ and $\underline{C}_i = d_{max} - C_i$ in the WE-Problem leads to

$$\min \sum_{i=1}^{N} (\underline{C}_i - r_i)U_i$$

$$\text{s.t. } \underline{C}_i - r_i \geq 0, \quad i=1,2,\ldots,N,$$

which is the same as the weighted flow time problem with non-simultaneous job arrivals. If C_i^*, $i \in [1,N]$, is optimal for the WE-Problem, then $\underline{C}_i^* = d_{max} - C_i^*$ is optimal for the corresponding weighted flow time problem.