Journal of the Operations Research
Society of Japan
Vol. 24, No. 1, March 1981

ONE MACHINE SCHEDULING PROBLEM WITH DUAL CRITERIA

Shigeji Miyazaki
University of Osaka Prefecture

(Received January 12, 1980; Revised August 28, 1980)

Abstract In this paper we deal with a one machine scheduling problem to minimize the mean weighted flow-time subject to the constraint that the job tardiness is not greater than a specified value. An algorithm to obtain a pairwise local optimum schedule for this problem has been presented by Smith [4]. We give a necessary condition under which a pairwise local optimum schedule should not coincide with the global optimum one, and give an improved schedule for this case. On the basis of the analysis, an efficient algorithm is developed to obtain global optimum schedules for more cases than Smith's algorithm. Computational experiment is performed to show the quality of the solution, the computational time, and core memory size required for the algorithm in comparison with the previous three algorithms: Smith's, Burns', and DP.

1. Introduction

This paper deals with a one machine scheduling problem to minimize the mean weighted flow-time subject to the constraint that the job tardiness is not greater than a specified value T^* where all the jobs are available on time zero. The majority of papers about scheduling seem to have devoted for an optimal schedule under a single performance measure selected from various kinds. In many practical situations, however, managements reasonably conceive a desire to improve another performance measure after obtaining an aspiration level of the main measure. These problems are so called dual criteria scheduling problems.

The problem treated in this paper was first considered by Smith [4] in the case that T^* equals 0. He presented an algorithm to obtain a solution in which all adjacent pairs of jobs were scheduled optimally. This schedule shall be said a pairwise local optimum schedule. Heck & Roberts [3] extend Smith's results to the case that T^* equals the maximum job tardiness sequenced by EDD rule which can minimize the maximum job tardiness.

They considered that their algorithms could produce the global optimum

schedule for any problem. Emmons [2] and Burns [1], later, gave counter examples independently for which Smith's algorithm could not produce the global optimum schedule. Unfortunately, neither Emmons has performed the further investigation on the data structure of the counter examples, nor has Burns.

This paper presents the necessary condition under which the pairwise local optimum schedule can not coincide with the global optimum. On the basis of the analysis, an efficient algorithm is developed to obtain an improved schedule induced from the pairwise local optimum schedule. The efficiency of the proposed algorithm is tested by the numerical experiment in connection with the following three points: the quality of the solution, the computational times, and the core memory size needed.

2. Problem Formulation

The notation throughout this study is mainly defined as follows:

$\bar{F}_w$: mean weighted flow-time
n : the number of jobs to be processed
J_i : the ith job in an arbitrary sequence
w_i : the weight (importance) of J_i
F_i : flow-time of J_i
$F_i(S)$: flow-time of J_i under schedule S
p_i : processing time of J_i
d_i : due-date of J_i
T^* : the limit of job tardiness $(T^* \geq \min_S T_{max}(S))$
$T_{max}(S)$: maximum job tardiness under schedule S

Using this notation, we can formulate the scheduling problem considered as:

Problem 1.

$$\text{min.} \quad \bar{F}_w = \frac{1}{n} \sum_{i=1}^{n} w_i F_i$$

$$\text{sub. to} \quad \sum_{j=1}^{i} p_j - d_i \leq T^*, \quad i = 1, 2, \cdots, n,$$

$$\text{where} \quad T^* \geq \min_S T_{max}(S) = \text{maximum tardiness by EDD rule.}$$

Substitution of $d_i' = d_i + T^*$ for d_i makes it possible to reduce the Problem 1 to

Problem 2.

(2.1) $$\text{min.} \quad \bar{F}_w = \frac{1}{n} \sum_{i=1}^{n} w_i F_i$$

$$\text{(2.2)} \qquad \text{sub. to} \quad \sum_{j=1}^{i} p_j - d_i \leqq 0 \ , \ i = 1,2,\cdots,n,$$

where d_i' is rewritten by d_i again. Consequently it is sufficient to consider only the Problem 2 (which is equivalent to Smith's problem) for the representative of the more general Problem 1.

3. Pairwise Local Optimum Schedule

Two schedules S and S' are called to be pairwise adjacent if one can be formed from the other by a single interchange of pairwise adjacent jobs referring to Fig. 1. A schedule is feasible if and only if it satisfies (2.2).

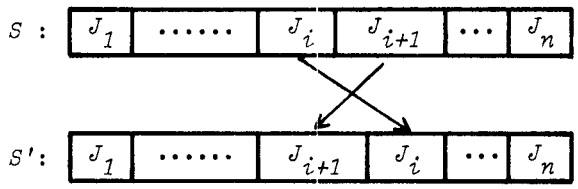


Fig. 1 Pairwise Adjacent Schedule.

Definition 1. A schedule is a pairwise local optimum for (2.1) subject to (2.2) if its $\bar{F}_w$ is less than or equal to the mean weighted flow-time of all feasible pairwise adjacent schedules of the schedule.

Definition 2. A schedule is a global optimum for (2.1) subject to (2.2) if its $\bar{F}_w$ is less than or equal to the mean weighted flow-time of all feasible schedules.

The pairwise local optimum schedule can be efficiently produced by the following algorithm presented by Smith:

Algorithm 1 (Smith).

[Step 1] Place job J_k last of n jobs where

$$\text{(3.1)} \qquad \frac{p_k}{w_k} = \max_{i \in G} \frac{p_i}{w_i} \ , \quad G = \{ j \mid \sum_{l=1}^{n} p_l - d_j \leqq 0 \} .$$

[Step 2] Eliminate J_k from n jobs and reduce n by 1. Return to Step 1 until all jobs have been scheduled by this method.

Smith [4] considered that this algorithm could produce a global optimum schedule for any problem. Emmons [2] and Burns [1], later, showed that Algorithm 1 was not a global optimum solution technique. We shall present the following theorem to give a sufficient condition under which the pairwise local optimum coincides with the global optimum.

Theorem 1. A pairwise local optimum schedule S_n composed of n jobs is to be a global optimum, if

$$F_{i+1} \leqq d_i , \qquad \text{for all } i = 1,2,\cdots,n-1. \tag{3.2}$$

Proof: If inequalities (3.2) are satisfied, $(i,i+1) \in G$ holds at each completion time F_{i+1} $(i=1,2,\cdots,n-1)$ in schedule S_n. Therefore, we have, from eq. (3.1)

$$p_i / w_i \leqq p_{i+1} / w_{i+1}, \qquad \text{for all } i=1,2,\cdots,n-1. \tag{3.3}$$

The literature [4] shows that the schedule which satisfies ineq. (3.3) yields the least of the mean weighted flow-times among all feasible schedules. Hence S_n inevitably becomes a global optimum if it satisfies ineq. (3.2).

(Q. E. D.)

From Thorem 1, we can see that if a local optimum is different from a global optimum, at least one job J_i should not satisfy ineq. (3.2). In this case the due-date d_i must exist in the region:

$$F_i \leqq d_i < F_{i+1}. \tag{3.4}$$

Therefore,we can lead the following theorem which gives the necessary condition under which a pairwise local optimum schedule should not coincide with a global optimum, and gives an improved schedule for this case.

Theorem 2. If

$$F_i \leqq d_i < F_{i+1} \tag{3.5}$$

holds for at least one job J_i, $i=1,2,\cdots,n-1$, in a pairwise local optimum schedule, we may construct an improved schedule $S_n\dagger$ such that:

$$\bar{F}_w(S_n\dagger) < \bar{F}_w(S_n), \tag{3.6}$$

and

$$T_{max}(S_n\dagger) = 0. \tag{3.7}$$

Where schedule $S_n\dagger$ can be formed by the following two steps: removing the last job of S_n to the first position and sequencing the remaining jobs with Algorithm 1.

Proof: Suppose that ineq. (3.2) holds for all the jobs except for only one job J_ε with which ineq. (3.5) is satisfied. Under this condition schedules S_n, S_n', and $S_n\dagger$ are illustrated in Fig. 2, where schedule S_n' can be formed from S_n by the two steps: removing the last job J_n to the first position and postponing the remaining jobs by p_n.

The difference between the total weighted flow-time of S_n and $S_n\dagger$ is given by:

$$\begin{aligned}(3.8)\qquad \sum_{i=1}^{n} w_i F_i(S_n) - \sum_{i=1}^{n} w_i F_i(S_n\dagger) &= \{ w_n (p_1 + p_2 + \cdots + p_{\varepsilon-1} + p_\varepsilon) \\ &- (w_1 + w_2 + \cdots + w_{\varepsilon-1} + w_\varepsilon) p_n \} + \{ w_n (p_{\varepsilon+1} + \cdots + p_{n-1}) \\ &- (w_{\varepsilon+1} + \cdots + w_{n-1}) p_n \} - X_w \ .\end{aligned}$$

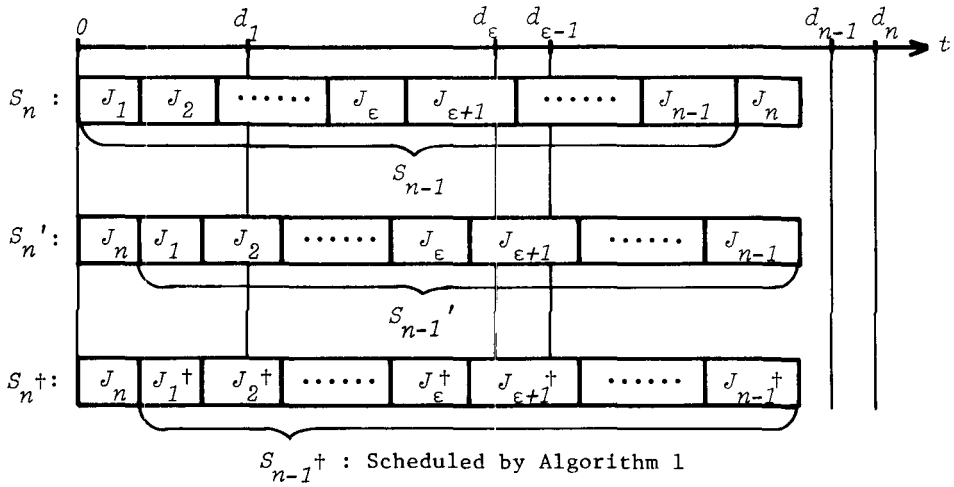
Where X_w is defined by:

$$X_w = \sum_{i=1}^{n-1} w_i F_i(S_{n-1}\dagger) - \sum_{i=1}^{n-1} w_i F_i(S_{n-1}') \geq 0 \ .$$

From the assumption that ineq. (3.2) is satisfied by all the jobs except J_ε, we have

$$p_i / w_i \leq p_{i+1} / w_{i+1} \ , \quad \text{for } i = 1,2,\cdots,\varepsilon-1,\varepsilon+1,\cdots,n-1.$$

So that the second brace of eq. (3.8) should be less than or equal to zero. Nevertheless the first brace may take a positive or negative value.



$S_{n-1}\dagger$: Scheduled by Algorithm 1

Fig. 2 Schedules S_n, S_n', and $S_n\dagger$.

Hence there may exist the case:

$$(3.9)\qquad w_n\,(\,p_1+p_2+\cdots+p_{\varepsilon-1}+p_\varepsilon\,)-(\,w_1+w_2+\cdots+w_{\varepsilon-1}+w_\varepsilon\,)\,p_n$$
$$>(\,w_{\varepsilon+1}+\cdots+w_{n-1}\,)\,p_n-w_n\,(\,p_{\varepsilon+1}+\cdots+p_{n-1}\,)+X_w\geq 0\ .$$

In this case we have

$$(3.10)\qquad \sum_{i=1}^{n} w_i F_i(S_n)-\sum_{i=1}^{n} w_i F_i(S_n\dagger)>0\ ,$$

which is equivalent to ineq. (3.6).

In the case there exists more than one job which satisfies ineq. (3.5), the discussion above can be legitimated by considering J_ε as the last of the jobs which satisfy ineq. (3.5).

If

$$(3.11)\qquad \max_{J_i\in S_{n-1}} L_i(EDD)\leq -p_n,$$

we have

$$T_{max}(S_n\dagger)=0.$$

Where $L_i(EDD)$ is the lateness of J_i scheduled by EDD rule, and here $L_i(EDD)$ takes non-positive value for any J_i because the lateness of any J_i in schedule S_{n-1} is also non-positive. So that there can exist the case (3.11).

The above argument shows that there may exist $S_n\dagger$ which satisfies ineq. (3.6) and eq. (3.7), if (3.5) holds for at least one job belonging to S_n.

(Q. E. D.)

4. Development of the Algorithm

On the basis of the Theorem 2, we can propose an algorithm (Algorithm 2) for improving the pairwise local optimum schedule when it does not coincide with the global optimum schedule. The algorithm (described in detail later) is constructed so as to search a partial schedule S_k which contains at least one job satisfying ineq.(3.5), and to produce the improved schedule $S_k\dagger$. For the preparation we shall define the terminology used in the algorithm (referring to Fig. 3).

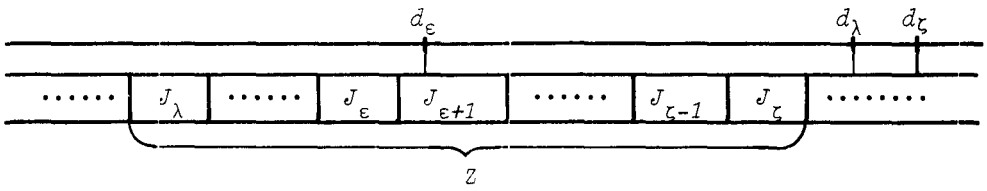


Fig. 3 Jobs J_λ, J_ε, J_ζ and Partial Schedule Z.

4.1. Terminology

J_ε : pivot job such that $J_\varepsilon \in \{J_i \mid F_i \leq d_i < F_{i+1}\}$

Z : a part of a pairwise local optimum schedule which contains at least one pivot job and at least two due-dates which are greater than the completion time of Z

J_λ : the first job of Z

J_ζ : the last job of Z

Z' : a partial schedule formed by the two steps: removing the last job J_ζ to the first place of Z and postponing the other jobs by time p_ζ

$Z\dagger$: a partial schedule formed by the two steps: removing the last job J_ζ to the first place of Z and rearranging the other jobs by Algorithm 1

$\Delta 1$: the difference of mean weighted flow-time between Z' and Z,

$$\Delta 1 = \bar{F}_w(Z') - \bar{F}_w(Z)$$

Δ : the difference of mean weighted flow-time between $Z\dagger$ and Z,

$$\Delta = \bar{F}_w(Z\dagger) - \bar{F}_w(Z)$$

J_s : the starting job for searching J_ε

Expansion of Z : the operation of adding $J_{\lambda-1}$ such that

$$p_{\lambda-1} / w_{\lambda-1} > p_\zeta / w_\zeta$$

to the partial schedule Z and renewing the first job J_λ by $J_{\lambda-1}$

Adjacent pairwise interchange method : the method of improving the performance measure by interchanging an adjacent pair of jobs in the schedule

4.2. Algorithm 2

[Step 0] Produce a pairwise local optimum schedule by Algorithm 1. Set $J_s = J_1$, and proceed.

[Step 1] Find the first pivot job J_ε between J_s and J_{n-1}. Set $J_\zeta = J_{\varepsilon+1}$, then go to Step 2. If no pivot job J_ε exists, terminate the algorithm. The

present schedule is the solution.

[Step 2] Detect the smallest partial schedule Z, and expand Z as possible. Go to Step 3. In case no Z exists, go to Step 7.

[Step 3] Calculate $\Delta 1$ for the partial schedule Z. If $\Delta 1 \geqq 0$, go to Step 6. Otherwise go to Step 4.

[Step 4] Produce the partial schedule Z†, and calculate Δ. If $\Delta < 0$, adopt Z† then go to Step 5. If $\Delta \geqq 0$ or if it is impossible to produce Z†, adopt the partial schedule Z. Then go to Step 6.

[Step 5] Remove the first job in Z† to the later position as possible by adjacent pairwise interchange method. Let J_s = the first job of the finally obtained schedule (Z†') in this step, and return to Step 1.

[Step 6] Renew the first job J_λ by the direct predecessor of J_λ. Then detect new Z and return to Step 3. If no new J_λ exists, go to Step 7.

[Step 7] Renew the last job J_ζ by the direct successor of J_ζ, then return to Step 2. If there exists no new J_ζ, set $J_s = J_{\varepsilon+1}$ and return to Step 1.

4.3. Numerical example

We illustrate the application of the proposed algorithm on a 5-job problem of which data are provided in Table 1.

[Step 0] Algorithm 1 (Smith) yields the schedule J_5-J_1-J_4-J_2-J_3 in which $\bar{F}_w$ = 707.8 as illustrated in Fig. 4.

[Step 1] $J_\varepsilon = J_1$, $J_\zeta = J_4$, then go to Step 2.

[Step 2] $Z = J_5$-J_1-J_4. Go to Step 3.

[Step 3] $Z' = J_4$-J_5-J_1, and $\Delta 1$ = 1202 - 1408 = -206 < 0. Hence go to Step 4.

[Step 4] Δ = 1305 - 1408 = -103 < 0, since $Z\dagger = J_4$-J_1-J_5. Therefore, adopt Z† and go to Step 5.

[Step 5] $Z\dagger' = J_4$-J_1-J_5. Then set $J_s = J_4$ and return to Step 1.

Table 1 Example Problem.

J_i	J_1	J_2	J_3	J_4	J_5
p_i	44	47	63	26	13
d_i	77	186	265	250	271
w_i	9	6	7	10	5
p_i/w_i	4.9	7.8	9.0	2.6	2.6

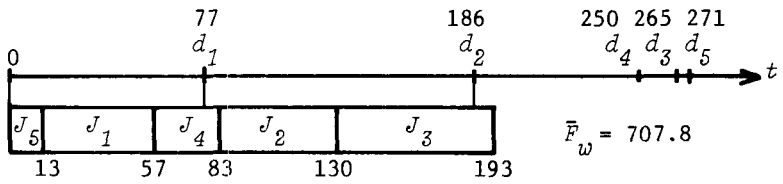


Fig. 4 Pairwise Local Optimum Schedule.

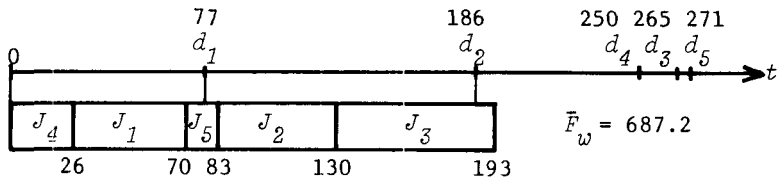


Fig. 5 Improved Schedule (Global Optimum).

[Step 1] $J_\varepsilon = J_1$ and $J_\zeta = J_5$. Proceed to Step 2.

[Step 2]
. } $\Delta \geqq 0$ for any partial schedule Z. Then $J_s = J_2$, and return to Step 1.
[Step 7]

[Step 1] $J_\varepsilon = J_2$, $J_\zeta = J_3$, then go to Step 2.

[Step 2]
. } $\Delta \geqq 0$ for any partial schedule Z. Then $J_s = J_3$, and return to Step 1.
[Step 7]

[Step 1] There exists no pivot job J_ε, so that terminate the algorithm. We obtain J_4-J_1-J_5-J_2-J_3 in which $\bar{F}_w = 687.2$ as shown in Fig. 5. This schedule is the global optimum schedule.

5. Computational Experiance

In order to verify the efficiency of the proposed Algorithm (Algorithm 2), we prepare various kinds of example problems produced by the following procedure. The job numbers of example problems are set to be 8, 10, 12, and 15. The due-dates are assigned from the uniform distribution of which mean and range are provided by, respectively:

(5.1) $$\mu_d = n\,(1 - Q)\,\mu_p,$$

and

(5.2) $$b - a = R\,n\,\mu_p.$$

Where

Q : the coefficient of the tightness of due-dates ($Q \leqq 1$),

R : the coefficient of the range of due-dates ($R \geqq 0$),

μ_d : the mean of due-dates,

n : the number of jobs to be processed,

μ_p : the mean of processing times,

b : the maximum value of due-dates,

a : the minimum value of due-dates.

Due-dates become more tightly as Q closes to 1, and the variation of due-dates becomes larger as R increases. For the investigation of the effect of both the tightness and the variation of due-dates, we make four types of due-dates shown in Table 2. The weights are assigned from two kinds of uniform distributions of which ranges are from 1 to 10, and 1 to 40. The processing times for each problem are assigned from the common uniform distribution between 1 and 99. Ten different problems are randomly created for each combination of the three factors: the type of due-dates, the number of jobs, and the range of weights. The total number of combinations amounts 32.

These example problems (totally 320 problems) were solved through the three algorithms: Algorithm 1 (Smith), Burns' Algorithm, and Algorithm 2 (proposed), respectively. The numbers of problems of which solutions do not coin-

Table 2 The Types of Due-Dates.

	Tightness of Due-Dates	Range of Due-Dates
Type 1	Slack ($Q=0.3$)	Large ($R=0.8$)
Type 2	Slack ($Q=0.3$)	Small ($R=0.2$)
Type 3	Tight ($Q=0.6$)	Large ($R=0.8$)
Type 4	Tight ($Q=0.6$)	Small ($R=0.2$)

Table 3 The Number of Unoptimal Solutions through Algorithm 1 (Smith).

Range of Weights	Number of Jobs	Number of Problems	Types of Due-Dates				Total
			1	2	3	4	
1 - 10	8	40	2	0	1	0	3
	10	40	0	1	1	1	3
	12	40	0	1	4	1	6
	15	40	4	0	3	0	7
	Total	160	6	2	9	2	19
1 - 40	8	40	2	1	2	0	5
	10	40	2	1	2	1	6
	12	40	0	0	1	0	1
	15	40	2	0	2	3	7
	Total	160	6	2	7	4	19

cide with the optimal ones are shown in Table 3, 4, and 5, in turn. The optimal schedules were obtained through DP.

Algorithm 1 can not produce the optimal schedule for 11.9 % of the problems under the both ranges of weights. Burns' Algorithm can improve these percentages to 5 % and 7.5 %, respectively. The Algorithm 2 fails to obtain optima for only 1 and 2 example problems under each range of weights, respectively.

Table 4 The Number of Unoptimal Solutions through Burns' Algorithm.

Range of Weights	Number of Jobs	Number of Problems	Types of Due-Dates				Total
			1	2	3	4	
1 - 10	8	40	1	0	0	0	1
	10	40	0	0	1	0	1
	12	40	0	0	2	0	2
	15	40	3	0	1	0	4
	Total	160	4	0	4	0	8
1 - 40	8	40	1	1	0	0	2
	10	40	2	1	2	0	5
	12	40	0	0	0	0	0
	15	40	1	0	2	2	5
	Total	160	4	2	4	2	12

Table 5 The Number of Unoptimal Solutions through Algorithm 2 (Proposed).

Range of Weights	Number of Jobs	Number of Problems	Types of Due-Dates				Total
			1	2	3	4	
1 - 10	8	40	1	0	0	0	1
	10	40	0	0	0	0	0
	12	40	0	0	0	0	0
	15	40	0	0	0	0	0
	Total	160	1	0	0	0	1
1 - 40	8	40	1	0	0	0	1
	10	40	0	1	0	0	1
	12	40	0	0	0	0	0
	15	40	0	0	0	0	0
	Total	160	1	1	0	0	2

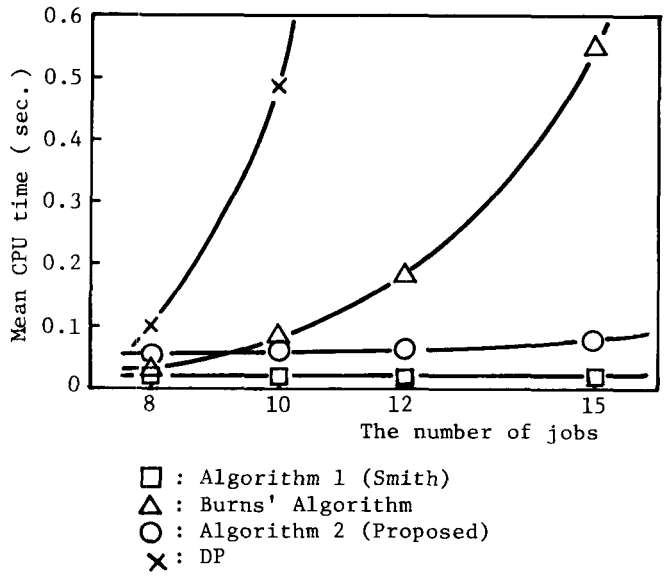


Fig. 6 Mean Solution (CPU) Time for Each Algorithm.

The number of unoptimal solutions through Algorithm 1 under $R = 0.2$ (Type 2 and 4) is less than under $R = 0.8$ (Type 1 and 3). This effect is caused by the reason that the condition of Theorem 1 can be more easily satisfied under the smaller variation of due-dates. Such is the case by Burns' too. These facts show us that the efficiency of these two algorithms under the larger variation of due-dates will be less than under the smaller variation. On the contrary, the number of unoptimal solutions through Algorithm 2 seems to be independent of the characteristics of the problems tested.

Algorithm 1 and Burns' can produce the optimal solution for none of the three example problems for which Algorithm 2 generates unoptima. Algorithm 1 can produce the optimum for only one problem for which Burns' can not produce the optimum. The mean weighted flow-times of the unoptimal solutions through Algorithm 1, Burns', and Algorithm 2 are greater than the optimal solutions by 1.8 percent, 1.6 percent, and 0.5 percent, on an average, respectively.

Time complexity of the each algorithm can be considered as follows. The structure of Algorithm 1 shows us easily that the solution time of it should be proportional to the number of jobs. The order of the solution time by Burns' is:

$$\sum_{i=1}^{n} i(n+1-i) = o(n^3). \tag{5.3}$$

The worst case of the time complexity of Algorithm 2 is:

$$\sum_{i=1}^{n-2} i(n-1-i)\delta(i) < o(n^3). \tag{5.4}$$

Where $\delta(i)$ is a function defined by:

$$\begin{aligned} \delta(i) &= 1 \qquad J_i \in \{J_\varepsilon\}, \\ \delta(i) &= 0 \qquad J_i \notin \{J_\varepsilon\}, \end{aligned} \tag{5.5}$$

and $i(n-1-i)$ is the time of maximum possible calculations for detecting Z when J_ε exists in ith position in the schedule.

The time complexity of Algorithm 2 depends upon the number of jobs belonging to the set $\{J_\varepsilon\}$. In order to verify the actual solution time for these three algorithms together with DP, the mean solution times required to solve the example problems by ACOS-77 system 600 are shown in Fig. 6. The actual solution time of Algorithm 2 for these randomly generated problems is much less than the worst case of eq. (5.4). The advantage of Algorithm 2 to Burns' becomes larger as the number of jobs increases. DP increases the solution time exponentially, as a result the mean time for 12- and 15-job problems are 2.25 sec and 22.5 sec, in turn.

6. Conclusions

Computational experience shows that the proposed Algorithm 2 fails to obtain the global optimum for only the three among totally three hundred and twenty example problems. Example problems are provided with different types of due-dates, different job numbers, and different ranges of weights. The unoptimal schedules through Algorithm 2 are one twelfth of those through Algorithm 1 (Smith) and one seventh of those through Burns'.

The solution time of Algorithm 2 hardly increases as the number of jobs increases. This behaviour is similar to Algorithm 1. The advantage of Algorithm 2 to Burns' in solution time becomes larger as the job number increases. The memory size needed by Algorithm 2 is $o(n)$. For example, 1000-job problem can be solved with 24K core memory. On the contrary the solution time of DP algorithm grows exponentially and DP has a severe restriction on the memory size. For instance 80K is required to solve 15-job problem.

These results show that Algorithm 2 can be regarded as a more practical solution technique than the previous algorithms in connection with the three factors: the quality of the solution, the computational time, and the core memory storage needed.

Acknowledgement

The author wishes to express his sincere thanks to Associate Professor Noriyuki Nishiyama for his helpful advice and encouragement.

References

[1] Burns, R. N.: Scheduling to Minimize the Weighted Sum of Completion Times with Secondary Criteria. *Naval Research Logistics Quarterly*, Vol. 23, No. 1(1976), 125-129.

[2] Emmons, H.: A Note on a Scheduling Problem with Dual Criteria. *Naval Research Logistics Quarterly*, Vol. 22, No. 3(1975), 615-616.

[3] Heck, H. and Roberts, S.: A Note on the Extension of a Result on Scheduling with Secondary Criteria. *Naval Research Logistics Quarterly*, Vol. 19, No. 2(1972), 403-405.

[4] Smith, W. E.: Various Optimizers for Single-Stage Production. *Naval Research Logistics Quarterly*, Vol. 3, No. 1(1956), 59-66.

Shigeji MIYAZAKI: Department of Industrial Engineering, College of Engineering, University of Osaka Prefecture, MozuUme-Machi, Sakai, Osaka, 591 Japan.