Computers Ops Res. Vol. 18, No. 3, pp. 317-322, 1991
Printed in Great Britain. All rights reserved

0305-0548/91 $3.00 + 0.00
Copyright © 1991 Pergamon Press plc

SINGLE MACHINE SCHEDULING WITH NONLINEAR COST FUNCTIONS

BAHRAM ALIDAEE*
Mathematics and Physical Sciences Department, West Texas State University, Canyon, TX 79016, U.S.A.

(Received December 1989; in revised form July 1990)

Scope and Purpose—The problem we are concerned about here deals with sequencing of a given set of jobs with preemption on a single machine so as to minimize total cost, where associated with each job is a processing time and a nondecreasing cost function depending on the completion time of the job. The problem in general is known to be very difficult to solve; there is no algorithm which can solve the problem in reasonable time. The purpose of this study is to give some heuristic solution procedure to solve the problem. Running time of the algorithm is $O(n^2)$ and it is easy to implement. The algorithm proposed here is empirically compared with heuristic algorithm by Fisher and Krieger [1]. Computational experience shows that the proposed algorithm gives better results.

Abstract—The problem of scheduling n jobs with a single machine where each job has a differentiable cost function and the total cost being minimized is studied. A heuristic solution procedure is proposed. The algorithm is also applicable when the cost functions are piecewise differentiable. Running time of the algorithm is $O(n^2)$. The effectiveness of the algorithm is evaluated and is shown that in many cases the algorithm picks an optimal order. The algorithm proposed here empirically is compared with linearized algorithm by Fisher and Krieger [1].

INTRODUCTION

We consider the problem which can be defined as follows:

n = number of the jobs to be processed on a single machine, all jobs are available at the same time
s_i = processing time of job i for $1 \leqslant i \leqslant n$
$S = ([1], \ldots, [i], \ldots, [n])$ represents an ordering (sequence) of the jobs where $[i]$ means the ith job to be processed in S
$f_i(x_i)$ = cost function of job i for $1 \leqslant i \leqslant n$ where $x_i = s_1 + \cdots + s_i$. It is assumed here that the cost functions are nondecreasing and differentiable.

The objective is to find an ordering S such that the following function Z is minimized,

$$Z(S) = \sum_{i=1}^{n} f_{[i]}(x_{[i]}) \qquad \text{where } x_{[i]} = s_{[1]} + \cdots + s_{[i]}. \tag{1}$$

Various special cases corresponding to specific cost functions have been studied. The problem has been solved efficiently only for the case of linear cost functions, i.e. $f_i(x) = a_i(x - d_i)$ where d_i is the due date of job i. Letting $d_i = 0$ for all i gives a weighted flowtime criterion, also included are the special cases of average lateness and average flowtime where $a_i = 1/n$ for all i. Smith [2] proved that all these problems are solved by processing the jobs in order of decreasing a_i/s_i ratio.

For an arbitrary non-linear non-decreasing penalty function of completion times, the problem is clearly NP-complete, since a special case where $f_i(x) = a_i \max(0, x_i - d_i)$ for all i is known to be NP-complete [3, 4]. Therefore some heuristic technique to find an optimal or near optimal solution is desirable. Research on minimizing the general non-linear problem has primarily concentrated on techniques to find an optimal solution. The solution methodology used by these researchers has been either dynamic programming [3–9] or branch and bound [10–15]. In all of the published

*Bahram Alidaee is Assistant Professor at West Texas State University. He holds the B.Sc. in Business Administration from the University of Tehran, Iran, the M.B.A. in Industrial Management from the University of North Texas and the Ph.D. in Mathematical Sciences from the University of Texas at Arlington. His principal research interest is in scheduling and sequencing, game theory and matroid theory. He has published in the journals of *Theory and Decisions*, *Applied Mathematics Letters*, *Journal of the Operational Research Society* and *The Logistic and Transportation Review*.

research, there are only three heuristic solution procedures [16, 19], one is presented by Wilkerson–Irwin [17] (W–I) which utilizes an Adjacent Pairwise Interchange (API) methodology incorporating the dominance properties developed by Emmons [10]. API algorithms choose a sequence of jobs as a basis, for example $(1, 2, \ldots, n)$, and consider every pair of adjacent jobs and switch them if the switch brings down the total cost [17, 18]. This method gives a locally optimal solution. One of the difficulties of the API algorithm is, which sequence is to be chosen as starting base. Fry *et al.* [19] recently developed an heuristic solution procedure based on the W–I method for the mean tardiness problem where $a_i = 1$ for all i, they used three different sequences as the starting basis. Another heuristic procedure for mean tardiness when $a_i = 1$ for all i, is given by Baker and Bertrant [20]. The algorithm is efficient and easy to implement. This algorithm chooses a job at each iteration based on the smallest due-date or smallest completion time of a job whichever is the minimum. Another algorithm for a general case is the linearized algorithm by Fisher and Krieger (FK) [1]. FK theoretically analyzed an heuristic solution based on the ratio rule of Smith. They considered maximization of the sum of non-increasing concave profit functions. The linearized approximation of the profit function is proved to always obtain at least 2/3 of the optimal profit. This bound works only on maximizing the sum of profit functions not on the minimizing function which we are considering here. Of course their algorithm can be implemented on the minimizing problem. We here propose a heuristic solution procedure for solving the general non-linear non-decreasing penalty function for each job. The algorithm can be implemented to run in $O(n^2)$ time under assumption that each f_i can be evaluated in unit time for any value of the argument. Computational experience shows in many cases that the algorithm picks an optimal order. The algorithm is empirically compared with linearized algorithm by FK.

SOME SCHEDULING CRITERION

Let $S = ([1], \ldots, [i], [i+1], \ldots, [n])$ be a sequence of the jobs and let $S' = ([1], \ldots, [i+1], [i], \ldots, [n])$ be obtained from S by switching the position of the ith and $(i+1)$th jobs. Now we have

$$Z(S') - Z(S) = \{f_{[i]}(t + s_{[i]} + s_{[i+1]}) - f_{[i]}(t + s_{[i]})\} - \{f_{[i+1]}(t + s_{[i+1]} + s_{[i]}) - f_{[i+1]}(t + s_{[i+1]})\} \quad (2)$$

where $t = s_{[1]} + \cdots + s_{[i-1]}$.

By mean value theorem of calculus there exist real numbers $z_{[i]}$ and $z_{[i+1]}$ in the intervals $(t + s_{[i]}, t + s_{[i]} + s_{[i+1]})$ and $(t + s_{[i+1]}, t + s_{[i+1]} + s_{[i]})$, respectively, such that equation (2) can be written as,

$$Z(S) - Z(S') = s_{[i+1]} f'_{[i]}(z_{[i]}) - s_{[i]} f'_{[i+1]}(z_{[i+1]})$$

then we have, $Z(S') \geqslant Z(S)$ iff $f'_{[i]}(z_{[i]})/s_{[i]} \geqslant f'_{[i+1]}(z_{[i+1]})/s_{[i+1]}$. This is given in the following theorem.

Theorem

Let S and S' be two sequences of the jobs as defined before then

$$Z(S') \geqslant Z(S) \quad \text{iff } f'_{[i]}(z_{[i]})/s_{[i]} \geqslant f'_{[i+1]}(z_{[i+1]})/s_{[i+1]}$$

for some $z_{[i]}$ and $z_{[i+1]}$ in $(t + s_{[i]}, t + s_{[i]} + s_{[i+1]})$ and $(t + s_{[i+1]}, t + s_{[i+1]} + s_{[i]})$, respectively, and $t = s_{[1]} + \cdots + s_{[i-1]}$.

Example 1. If the cost function f_i is linear $\forall\, 1 \leqslant i \leqslant n$, i.e. $f_i = a_i x_i + b_i$ then a sequence $S = ([1], \ldots, [n])$ will minimize Z iff $a_{[1]}/s_{[1]} \geqslant \cdots \geqslant a_{[n]}/s_{[n]}$ where $a_{[i]} = f'_{[i]}\ \forall\, i$.

A geometric interpretation of the above theorem is given in Fig. 1. If the slope of the tangent line $1_{[i]}$ [Fig. 1(a)] divided by $s_{[i]}$ is larger than slope of the tangent line $1_{[i+1]}$ [Fig. 1(b)] divided by $s_{[i+1]}$ then $Z(S) < Z(S')$.

Based on the above theorem we give the following algorithm with running time $O(n^2)$ to solve the problem. Suppose jobs $1, \ldots, i-1$ already are chosen by the algorithm. The algorithm chooses the ith job from all remaining jobs where $f'_{[i]}(z)/s_{[i]}$ is largest and $z = t + s_i + (1/N)\, \Sigma\, s_k$ and N is number of the set of remaining jobs. In this formula summation is taken over all remaining jobs. We give the algorithm in the following form:

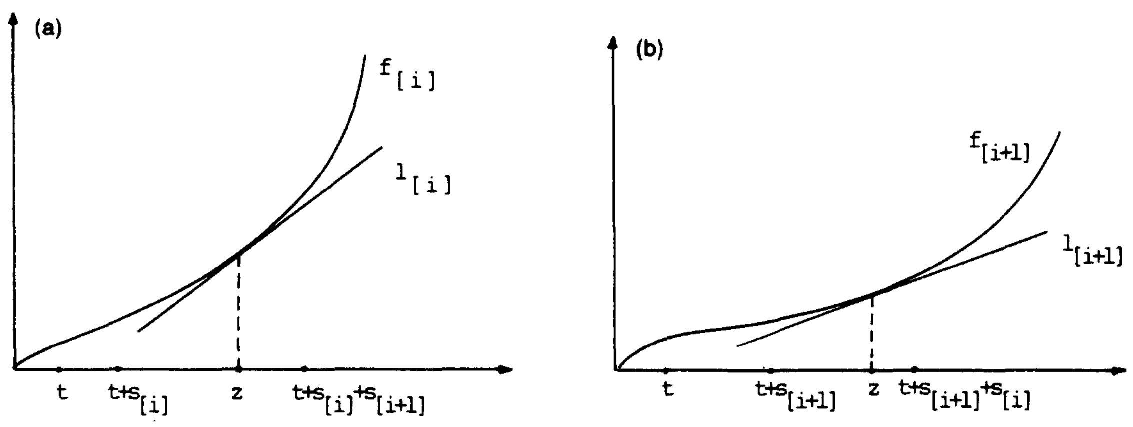

Fig. 1

Dynamic heuristic algorithm:

```
Step (1) Initialize $t = 0$, $U = \{1, \ldots, n\}$ and $U' = \emptyset$
Step (2) (a) Choose $i \in U$ such that
             $f_i'(z)/s_i = \max_{j \in U} f_j'(z)/s_j$
         where
             $z = t + s_j + (1/n) \sum_{j \in U} s_j.$
         If there are ties choose the least of index
     (b) Set, $t = t + s_i$, $U = U - \{i\}$, $U' = U' + \{i\}$ and $n = n - 1$.
Step (3) If $U = \emptyset$ then stop and the sequence chosen by the algorithm is $U'$ otherwise go to Step 2.
```

Note 1: In Step 2(a) of the algorithm if there are ties, depending on the particular problem being solved (specifically depending on cost functions of the jobs) the breaking decision may be different. For example in the weighted total tardiness (i.e. $f_i(x) = a_i \max\{0, x - d_i\}$ for all i), we define $f_i'(x_i)$ to be equal to zero if $x_i \leqslant d_i$ and equal to a_i if $x_i > d_i$ then we can use the dynamic algorithm and we may break the ties with the job with the shortest processing time or earliest due date or smallest ratio of a_i/s_i. In our computational testing the latter rule worked better. This is because at the beginning of iterations of the algorithm, f_i' is equal to zero for many different jobs, and this rule takes into account weight a_i, processing time s_i and implicitly due-date d_i.

Figure 2 gives a geometric interpretation of the algorithm at each iteration. At the ith iteration the ith job is chosen from the set of remaining jobs such that the slope of the tangent line l_i divided by s_i is largest. The linearized algorithm by FK at iteration i chooses the ith job from the remaining jobs such that the slope of line L_i (Fig. 3) divided by s_i is largest where T and t in Fig. 3 are equal to $\sum_{k=1}^{n} s_k$ and $\sum_{k=1}^{i-1} s_k$, respectively.

Example 2. Consider a 9-job total tardiness problem, i.e. $f_i(x_i) = a_i \max\{0, x_i - d_i\}$ for $i = 1, \ldots, n$. Let processing times s_i and due-dates d_i be as follows

s_i:	0.767	0.560	0.684	0.906	0.750	0.667	0.667	0.263	0.244
a_i:	0.106	0.128	0.256	0.391	0.375	0.500	0.699	0.581	0.916
d_i:	4.388	0.231	0.468	0.904	5.168	0.711	5.344	0.030	1.484

The dynamic algorithm chooses the sequence (8, 6, 9, 4, 3, 2, 1, 5, 7) with $Z = 1.80334$ which is optimal. The linearized algorithm chooses (9, 8, 6, 4, 3, 2, 5, 7, 1) with $Z = 2.07138$ which is 14.8% larger than the optimal value.

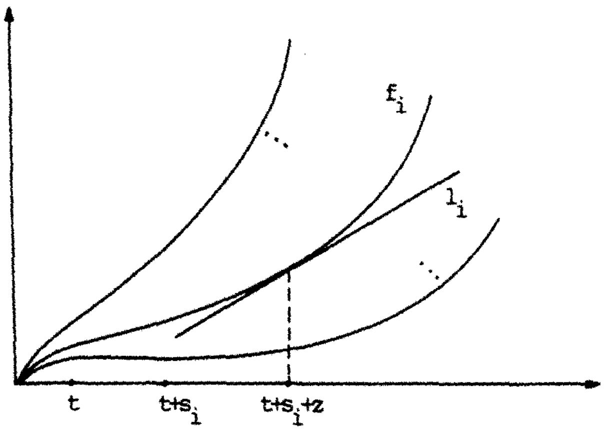

Fig. 2

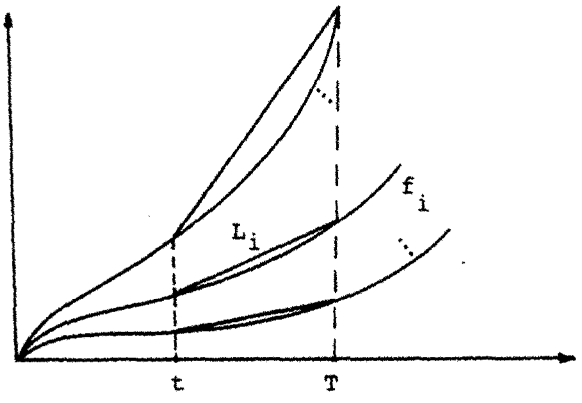

Fig. 3

Table 1. Comparative evaluation of the algorithms with optimal for $f_i(x_i) = \exp(a_i x_i)$

	Dynamic algorithm				Linearized algorithm			
n	#OPT	MIN	AVR	MAX	#OPT	MIN	AVR	MAX
6	19	0.0	0.0021	0.0368	14	0.0	0.0032	0.0068
7	15	0.0	0.0035	0.0076	13	0.0	0.0037	0.0075
8	14	0.0	0.0042	0.0138	14	0.0	0.0053	0.0138
9	14	0.0	0.0044	0.0106	10	0.0	0.0030	0.0196

COMPUTATIONAL EXPERIENCE

To test the effectiveness of the proposed algorithm in finding optimal or near optimal schedules we used the dynamic and linearized algorithms to solve two different sets of problems (1) $f_i(x_i) = \exp(a_i x_i)\ \forall\, 1 \leq i \leq n$, (2) $f_i(x_i) = a_i \max\{0, x_i - d_i\}\ \forall\, i$. The values of a_i, s_i were generated from a uniform distribution in the range (0, 1], and the values of d_i were generated in the range of (0, MS] where $\mathrm{MS} = s_i + \cdots + s_n$. We solved problems varying in size from 6 to 9 jobs. A set of 20 problems was chosen from each size. For each problem the effectiveness of the kth heuristic algorithm q_k^{D} (D stands for dynamic) and q_k^{L} (L stands for linearized) is defined as:

$$q_k^{\mathrm{D}} = (q_k^{\mathrm{D}} - p_0)/p_0 \qquad \text{and} \qquad q_k^{\mathrm{L}} = (q_k^{\mathrm{L}} - p_0)/p_0$$

where q_k^{D}, q_k^{L} and p_0 are kth dynamic heuristic, kth linearized heuristic and optimal solution, respectively. Tables 1 and 2 depict the number of optimal solutions (#OPT), the minimum (MIN), average (AVR) and maximum (MAX) values of q for each problem size and each algorithm.

We also used the heuristic algorithm to solve problems varying in size, 20, 30, 40, 50, 60, 70

Table 2. Comparative evaluation of the algorithms with optimal for $f_i = a_i \max\{0, x_i - d_i\}$

	Dynamic algorithm				Linearized algorithm			
n	#OPT	MIN	AVR	MAX	#OPT	MIN	AVR	MAX
6	3	0.0	0.3570	0.8650	2	0.0	0.4110	1.303
7	2	0.0	0.4210	1.272	2	0.0	0.5600	2.103
8	2	0.0	0.2180	1.002	2	0.0	0.3790	1.252
9	3	0.0	0.3610	1.740	2	0.0	0.4470	1.746

Table 3. Comparative evaluation of the algorithms for $f_i(x_i) = \exp(a_i x_i)$

n	Z^L/Z^D
20	1.0284
30	1.0243
40	1.0086
50	1.0200
60	1.0276
70	1.0178
80	1.0259

Table 4. Comparative evaluation of the algorithms for $f_i(x_i) = a_i \max\{0, x_i - d_i\}$

n	Z^L/Z^D
20	1.211
30	1.254
40	2.236
50	2.125
60	2.600
70	2.734
80	2.760

and 80 jobs. A set of 5 problems from each size was solved. For each set of problems the average of the total costs of 5 problems for dynamic and linearized algorithms as Z^D, and Z^L, respectively, is found and the ratio of Z^L/Z^D is given in Tables 3 and 4.

Considering Tables 1 and 3 it is clear that the dynamic algorithm gives a better result for exponential cost functions. The dynamic algorithm picked an optimal order more than 77% of the time while the linearized algorithm picked an optimal sequence only 63% of the time.

In the case of weighted tardiness problems the algorithm provides a practical solution (sequence) of which further improvement is possible by applying the API rule to this sequence. As can be seen from Table 4, for a large number of jobs the dynamic algorithm gives a better result than the linearized algorithm. On average the total cost obtained by the linearized algorithm was 2.13 times larger than the total cost obtained by the dynamic algorithm.

CONCLUSION

In this study we considered single machine scheduling with nonlinear cost function for each job, to minimize total cost. In most scheduling problems, and in particular in minimizing total cost, single machine scheduling with non-linear cost functions are NP-complete, therefore heuristic procedures are desirable. There has not been enough attempt to find heuristic solution procedures for minimizing total cost problems in literature. We hope this study will be a step towards more study to find approximate solutions. We gave a geometric interpretation of the algorithm, and the algorithm is compared with the linearized heuristic algorithm by Fisher and Krieger.

Acknowledgement—I would like to thank the two anonymous referees for their comments which greatly helped the presentation of this paper.

REFERENCES

1. M. L. Fisher and M. A. Krieger, Analysis of a linearization heuristic for single-machine scheduling to minimize profit. *Mathl Program.* **28**, 218–225 (1984).
2. W. E. Smith, Various optimizers for single-stage production. *Nav. Res. Logist. Q.* 3, 59–66 (1956).
3. E. L. Lawler, A 'pseudopolynomial' algorithm for sequencing jobs to minimize total tardiness. *Ann. disc. Math.* **1**, 331–342 (1977).
4. J. K. Lenstra, A. H. G. Rinnooy Kan and P. Brucker, Complexity of machine scheduling problems. *Ann. disc. Math.* **1**, 343–362 (1977).
5. K. R. Baker and L. Schrage, Finding an optimal sequence by dynamic programming: an extension to precedence-related tasks. *Ops Res.* **26**, 111–119 (1978).
6. M. Held and N. Karp, A dynamic programming approach to sequencing problems. *J. Soc. ind. appl. Math.* **10**, 196–209 (1962).
7. C. N. Potts and L. N. Van Wassenhove, A decomposition algorithm for the single machine total tardiness problem. *Ops Res. Lett.* **1**, 177–181 (1982).

8. J. Shwimer, On the *n*-job, one-machine, sequence-independent scheduling problem with tardiness penalties: a branch-bound solution. *Mgmt Sci.* **18**, B301–B313 (1972).
9. A. H. G. Rinnooy Kan, B. J. Lageweg and J. K. Lenstra, Minimizing total costs in one-machine scheduling. *Ops Res.* **23**, 908–927 (1975).
10. H. Emmons, One machine sequencing to minimize certain functions of job tardiness. *Ops Res.* **17**, 701–715
11. M. L. Fisher, A dual algorithm for the one-machine scheduling problem. *Mathl Program.* **11**, 229–251 (1976).
12. A. Schild and I. J. Freedman, Scheduling tasks with linear loss functions. *Mgmt Sci.* **7**, 280–285 (1961).
13. V. Srivasan, A hybrid algorithm for the one-machine sequencing problem to minimize total tardiness. *Nav. Res. Logist. Q.* **18**, 317–327 (1971).
14. W. Townsend, The single machine problem with quadratic penalty function of completion times: a branch-and-bound solution. *Mgmt Sci.* **24**, 530–534 (1978).
15. P. C. Bagga and K. R. Kalra, A node elimination procedure for Townsend's algorithm for solving the single machine quadratic penalty function scheduling problem. *Mgmt Sci.* **26**, 633–636 (1980).
16. S. K. Gupta and J. Kyparisis, Single machine scheduling research. *OMEGA Int. J. Mgmt Sci.* **15**, 207–227 (1987).
17. J. Wilkerson and J. D. Irwin, An improved algorithm for scheduling independent tasks. *AIIE Trans.* **3**, 333–342 (1971).
18. J. Evans, Structural analysis of local search heuristics in combinatorial optimiztion. *Computers Ops Res.* **14**, 465–477 (1987).
19. T. D. Fry, L. Vicens, K. Macleod and S. Fernandez, A heuristic solution procedure to minimize T on a single machine. *J. Opl Res. Soc.* **40**, 293–297 (1989).
20. K. R. Baker and J. W. M. Bertrand, A dynamic priority rule for scheduling against due-dates. *J. Ops Mgmt* **8**, 37–42 (1982).