Minimum Total Processing Time Analysis for Moore-Algorithm

Wei Cai & Shanlin Li
Department of Mathematics, Taizhou University, Taizhou, Zhejiang, 317000, China

ABSTRACT: Moore (1968) gives an $O(n\log n)$ time algorithm to find an optimal schedule for the single-machine number of late jobs problem. Pinedo (2002) points out, without giving a detail proof, that the early job set generated by the Moore-Algorithm obtains not only the maximum number of jobs but also the smallest total processing time among all optimal schedules. In this paper, we give a new optimality proof of the Moore-Algorithm and show simultaneously that the statement of Pinedo is correct. This property may be applied to study of the hard real-time systems.

KEYWORD: Production scheduling; single machine deterministic sequencing; algorithm; real-time system

1 BACKGROUND

The single-machine problem of minimizing the number of late jobs can be described as follows. There is a set of n jobs, $J=\{1,\cdots,n\}$, to be processed on a single machine. Each job $j\in J$ has a processing time p_j, and a due date d_j $(p_j\le d_j)$. In a given schedule, for each job $j\in J$, we define C_j to be its completion time, and U_j to be 1 if $C_j>d_j$ and 0 otherwise. We say that in a given schedule job j is *early* if $U_j=0$ and *late* if $U_j=1$. The problem is to find a non-preemptive schedule that minimizes $\sum_{j=1}^{n}U_j$.

This problem and its several variants have been extensively studied in the production scheduling literature. Moore (1968) gives an $O(n\log n)$ algorithm for finding an optimal schedule for the problem. For ease of presentation, this algorithm is called Moore-Algorithm hereafter. Without loss of generality, we assume that the jobs are indexed such that $d_1\le\cdots\le d_n$. Furthermore, given a set of jobs S, we define some notations as follows.

$\pi(S)$: the schedule that jobs in S are se quenced according to the Earliest Due Date First rule.

$|S|$: the number of jobs in S.

$F(\pi(S))$: the completion time of the last job of the schedule $\pi(S)$.

S is *feasible* if all the jobs in the schedule $\pi(S)$ are early.

The Moore-algorithm is stated as follows.

Moore-Algorithm

Step 1. $E_0\leftarrow\phi$, $j\leftarrow 0$ and go to Step 2.

Step 2. If $j=n$, then go to Step 3. Otherwise, $j\leftarrow j+1$; compute E_j as follows and return to step 2.

$$E_j=\begin{cases} E_{j-1}\cup\{j\}, & \text{if } F(\pi(E_{j-1}\cup\{j\}))\le d_j \\ E_{j-1}\cup\{j\}\setminus\{l\} & \text{otherwise}\end{cases} \tag{1}$$

where l is a job in $E_{j-1}\cup\{j\}$ satisfying

$$p_l=\max\{p_i\mid i\in E_{j-1}\cup\{j\}\}. \tag{2}$$

Step 3. STOP.

The schedule $((\pi(E_n),J\setminus E_n)$ is the output of the Moore-Algorithm, where the sequence of jobs in $J\setminus E_n$ is an arbitrary one.

Example 1. Consider four jobs $J=\{1,2,3,4\}$ with the following parameters.

Table 1: Problem instance of four jobs.

job	*1*	*2*	*3*	*4*
p_j	5	4	3	2
d_j	6	10	13	13

The schedule generated by the Moore-Algorithm is $(\pi(E_4), J \setminus E_4) = (2,3,4,1)$ with $\sum_{j=1}^{4} U_j = 1$.

Note that there may exist many optimal schedules for the number of late jobs problem. For example, there exist four optimal schedules: (1,2,3,4), (1,2,4,3), (1,3,4,2), and (2,3,4,1) in the example 1. Then it seems to be an interesting issue what characteristics the schedule by the Moore-Algorithm has when compared with other optimal schedules. This idea is motivated in part by two facts. One is that Pinedo (2002) mentions in the optimality proof of Moore-Algorithm that E_j, for $j = 1, \cdots, n$, satisfies the following two conditions:

(i) E_j has the maximum number of jobs among all feasible subsets of $\{1, \cdots, n\}$.

(ii) Among all feasible subsets with $|E_j|$ jobs of $\{1, 2\cdot;$ „ the jobs in $\pi(E_j)$ have the smallest total processing time.

But a detailed proof is not given. Recently Pinedo (2008) provides a proof of condition (i). This seems that condition (ii) is valid but not easy to prove. Another fact is related to the problem of nonpreemptive scheduling of n tasks on a single processor of a hard real-time system. In a hard real-time system, the computer is required to support the execution of applications in which the timing constraints of the tasks are specified. The correctness of the system depends on the temporal correctness as well as the functional correctness of the tasks. Failure to satisfy the timing constraints can cause catastrophic effects. For each task, the system have the choices: either rejects it or accepts it. Once a task is accepted by the system, the system should be able to finish it under the timing constraint of the task. A task i can be characterized as a triple of (r_i, p_i, d_i), representing the ready time, the computation time, and the deadline of the task respectively. A task cannot be started before its ready time. Once started the task must use the processor for a consecutive period of p_i and be finished by its deadline. The problem is NP-complete (Garey and Johnson, 1979). Hwang and Cheng (2001) explore the problem and propose an optimal schedule such that the number of rejected tasks is minimized, and then the finish time is minimized for the accepted tasks. The complexity of their scheduling algorithm in the worst case remains exponential. Now we consider a special case of the problem where r_i is a constant. In the case, the problem reduces to finding a non-preemptive schedule that satisfies the conditions (i) and (ii), where the set of the late jobs is treated as the set of rejected tasks. If we can show that the schedule by Moore-Algorithm satisfies the conditions (i) and (ii), then the problem in the special case is polynomial solvable.

In the next section, we show that the schedule by the Moore-Algorithm satisfies conditions (i) and (ii).

2 A NEW ANALYSIS

We give the main result in this section. In the following theorem, a new condition (ii′) is presented instead of the condition (ii), it is a more concise description for the output of the Moore-Algorithm than (ii) and implies (ii). For any $j \in J$, some additional notations are defined as follows.

$\sigma(E_j)$: the schedule that the jobs in E_j are sequenced according to the Shortest Processing Time First rule.

e_j^*: the maximum number of jobs among all feasible subsets of $\{1, 2, \cdots, \mathrm{j}\}$.

$S_j^*(k)$: the feasible subset with the smallest total processing time among all feasible subsets with k jobs of $\{1, 2, \cdots, \mathrm{j}\}$, for $k = 1, 2, ..., |E_j|$.

Let $S_j = \sigma(E_j) = (h_1, h_2, ..., h_{|E_j|})$; $S_j(\mathrm{k}) = (h_1, h_2, ..., h_k)$ for $k = 1, 2, ..., |E_j|$. When $k = 0$, $S_j(k)$ is defined to be ϕ. For example, $S_4 = \sigma(E_4) = (4,3,2)$; $S_4(1) = \{4\}$, $S_4(2) = \{4,3\}$, $S_4(3) = \{4,3,2\}$; $F(\pi(S_4(1)))$, $= 2$, $F(\pi(S_4(2))) = 5$, $F(\pi(S_4(3))) = 9$ in Example 1.

Theorem 1. For all j and k, the subset E_j generated by the Moore-Algorithm satisfies the following two conditions:

(i) $E_j = e_j^*$.

(ii) $S_j(k) = S_j^*(k)$.

Proof: The proof consists of two steps. The first step shows that the job sets $E_1, E_2, ..., E_n$ by Moore-Algorithm are all feasible. We prove it by the induction. E_1 is obviously feasible. Assume that E_{j-1} is feasible. If $F(\pi(E_{j-1} \cup \{j\})) \le d_j$, then $E_j = E_{j-1} \cup \{j\}$, and $\pi(E_j) = (\pi(E_{j-1}, j))$. Therefore, E_j is feasible. If $F(\pi(E_{j-1} \cup \{j\})) > d_j$, then $E_j = E_{j-1} \cup \{j\} \setminus \{l\}$, and either $\pi(E_j) = \pi(E_{j-1})$ (where l=j) or $\pi(E_j) = (\pi(E_{j-1} \setminus \{l\}, j)$ (where $l \in E_{j-1}$) holds. E_j is feasible in the case $\pi(E_j) = \pi(E_{j-1})$ naturally. Note first that $E_{j-1} \setminus \{l\}$

is feasible since so is E_{j-1}. In addition, $F(\pi(E_{j-1}\cup\{j\}\setminus\{l\}))\leq \quad F(\pi(E_{j-1}))\leq d_{j-1}\leq d_j$ follows from (2). Thus, E_j is feasible in the case $\pi(E_j)=(\pi(E_{j-1}\setminus\{l\},j)$.

The second step shows the results (i) and (ii'). To prove this, assume that it is true for $j-1$ and $S_{j-1}=\sigma(E_{j-1})=(h_1,h_2,...,h_{|E_{j-1}|})$. By the definition of S_{j-1}, we have

$$p_{h_1}\leq p_{h_2}\leq\cdots\leq p_{h_{|E_{j-1}|}} \tag{3}$$

Next we show that it is also true for j by considering two cases.

Case (a): $F(\pi(E_{j-1}\cup\{j\}))\leq d_j$. By the Moore-Algorithm, $E_j=E_{j-1}\cup\{j\}$. Hence $|E_j|=|E_{j-1}|+1$. We first show result (i). By the induction assumption, $|E_{j-1}|=|e^*_{j-1}|$. Thus, $|E_j|=|e^*_{j-1}|+1$. This, along with the fact that $e^*_j\leq e^*_{j-1}+1$ and $e^*_j\geq|E_j|$, implies that $|E_j|=e^*_j=|e^*_{j-1}|+1$.

We now show result (ii'). Insert p_j into the string inequality (3) such that p_j satisfies

$$p_{h_1}\leq\cdots p_{h_u}\leq p_j\leq p_{h_{u+1}}\leq\cdots\leq p_{h_{|E_{j-1}|}} \tag{4}$$

So we obtain that $S_j=\sigma(E_j)=(h_1,...,h_u,j,h_{u+1},...,h_{|E_{j-1}|})$ and $S_j(k)=S_{j-1}(k)$, for $1\leq k\leq u$; $S_j(k)=S_{j-1}(k-1)\cup\{j\}$, for $u+1\leq k\leq|E_j|$. Clearly, both $\pi(S_{j-1}(k))$ and $\pi(S_{j-1}(k-1)\cup\{j\})$ are feasible since so is E_j, for $k=1,2,...,|E_{j-1}|$. Note the fact that $S^*_j(k)$ either contains the job j or does not. Furthermore, by the induction assumption, either $S^*_j(k)=S_{j-1}(k-1)\cup\{j\}$ or $S^*_j(k)=S_{j-1}(k)$ holds. Then by (4), we have that $F(\pi(S_{j-1}(k)))\leq F(\pi(S_{j-1}(k-1)\cup\{j\}))$ if $1\leq k\leq u$. Thus $S^*_j(k)=S_{j-1}(k)=S$ for $1\leq k\leq u$. For $u+1\leq k\leq|E_j|-1$, by (4), we have that $F(\pi(S_{j-1}(k)))\geq F(\pi(S_{j-1}(k-1)\cup\{j\}))$. Thus $S^*_j(k)=S_{j-1}(k-1)\cup\{j\}=S_j(k)$, for $u+1\leq k\leq|E_j|-1$. For $k=|E_j|=|E_{j-1}|+1$, since there does not exist a feasible subset with $|E_{j-1}|+1$ jobs of {1,2,…,*j*}, we have $S^*_j(|E_j|)=S_{j-1}(|E_{j-1}|)\cup\{j\}=S_j(E_j)$. These imply the result (ii').

Case (b): $F(\pi(E_{j-1}\cup\{j\}))>d_j$. By the Moore-Algorithm, $E_j=E_{j-1}\cup\{j\}\setminus\{l\}$. Hence $|E_j|=|E_{j-1}|$. We first show result (i). Suppose $e^*_j=e^*_{j-1}+1$. Then, there exists a feasible subset with $e^*_{j-1}+1$ jobs of $\{1,2,\cdots,\text{j}\}$, say $S^*_j(e^*_{j-1}+1)$ without loss of generality. Due to $|E_{j-1}|=|e^*_{j-1}|$, $S^*_j(e^*_{j-1}+1)$ must contain the job *j*, and the other e^*_{j-1} jobs must be from $j=1,2,\cdots,j-1$. Hence, the schedule $\pi(S^*_j(e^*_{j-1}+1)) \quad =(\pi(S^*_j(e^*_{j-1}+1)\setminus\{l\}),j)$, and the job j is an early job of the schedule. Then, $d_j\geq F(\pi(S^*_j(e^*_{j-1}+1)))=$ $F(\pi(S^*_j(e^*_{j-1}+1)\setminus\{j\}))+p_j\geq F(\pi(S^*_{j-1}(e^*_{j-1})))+p_j=$ $F(\pi(E_{j-1}\cup\{j\}$ follows from $E_{j-1}=S^*_{j-1}(e^*_{j-1})$. This contradicts with the given assumption $F(\pi(E_{j-1}+1)\cup\{j\}))>d_j$. Thus, $|E_j|=e^*_j=e^*_{j-1}$.

Next we show result (ii'). Inserting p_j into the string inequality (3), we obtain the string inequality (4). Let $p_l=\max\{p_j\mid j\in E_{j-1}\cup\{j\}\}$. There are three cases to consider as follows.

(b1) If $l=j$, then $E_j=E_{j-1}$, $S_j=\sigma(E_j)=(h_1,...h_u,j,h_{u+1},...,h_{|E_{j-1}|})$, and $S_j(k)=S_{j-1}(k)$ for $k=1,2,...|E_j|$. As in case (a), we have that either $S^*_j(k)=S_{j-1}(k-1)\cup\{j\}$ or $S^*_j(k)=S_{j-1}(k)$ holds for all *k*. By $p_j=\max\{p_i\mid i\in E_{j-1}\cup\{j\}\}$, $F(\pi(S_{j-1}(k)))\leq F(\pi(S_{j-1}(k-1)\cup\{j\}))$ for all k clearly. Thus $S^*_j(k)=S_{j-1}(k)=S_j(k)$ for all *k*.

(b2) If $l=h_{u_0}$ with $u_0\leq u$, then $E_j=E_{j-1}\cup\{j\}\setminus\{l\}$, $S_j=\sigma(E_j)=(h_1,...,h_{u_0-1},h_{u_0+1},...,h_u,h_{u+1}...,h_{|E_{j-1}|})$ and $S_j(k)=S_{j-1}(k)$ for $l\leq k\leq u_0-1$; $S_j(k)=S_{j-1}(k+1)\setminus\{h_{u_0}\}$ for $u_0\leq k\leq u-1$; $S_j(k)=S_{j-1}(k)\setminus\{h_{u_0}\}\cup\{j\}$ for $u\leq k\leq|E_j|$. Notes the fact that $p_{h_k}=p_j$ for $u\leq k\leq|E_j|$. We have that $F(\pi(S_j(k)))=F(\pi(S_{j-1}(k)))$ for $k=1,2,...,\ |E_j|$. By similar arguments as in case (b1), we have $S^*_j(k)=S_j(k)$ for all *k* that follows from $p_j=p_{u_0}$ $=\max\{p_i\mid i\in E_{j-2}\cup\{j\}\}$.

(b3) If $l=h_{u_0}$ with $u_0\geq u+1$, then $E_j=E_{j-1}\cup\{j\}\setminus\{l\}$, $S_j=\sigma(E_j)=(h_1,...,h_u,j,h_{u+1},...,h_{u_0-1},h_{u_0+1},...,h_{|E_{j-1}|})$, and $S_j(k)=S_{j-1}(k)$ for $l\leq k\leq u$; $S_j(k)=S_{j-1}(k-1)\cup\{j\}$ for $u+1\leq k\leq u_0$; $S_j(k)=S_{j-1}(k)\setminus\{h_{u_0}\}\cup\{j\}$ for $u_0+1\leq k\leq|E_j|$. Notes the fact that $p_{h_{u_0}}=p_k$ for $u_0\leq k\leq|E_j|$. We have that $F(\pi(S_j(k)))=F(\pi(S_{j-1}(k)\setminus\{h_{u_0}\}\cup\{j\}))$

for $u_0 + 1 \le k \le |E_j|$. Using the string inequality $p_{h_1} \le \cdots p_{h_u} \le p_j \le p_{h_{u+1}} \le \cdots \le p_{h_{|E_{j-1}|-1}}$, by similar arguments as in case (a), we have that $F(\pi(S_{j-1}(k))) = F(\pi(S_j^*(k)))$ for $l \le k \le u$ and $F(\pi(S_{j-1}(k))) = F(\pi(S_j^*(k)))$ for $u + 1 \le k \le |E_j| - 1$. This implies that $S_j^*(k) = S_j(k)$ for all k and completes the proof.

3 ACKNOWLEDGEMENT

This research was supported in part by the Zhejiang Natural Science Foundation of China Grant Y6110054.

REFERENCES

[1] Moore, J. M. 1968. An n job, one machine sequencing algorithm for minimizing the number of late jobs. *Management Science*, 15, 102-109.

[2] Pinedo, M. 2002. *Scheduling: Theory, Algorithms, and Systems, second edition.* New Jersey: Prentice Hall.

[3] Pinedo, M. 2008. Scheduling: *Theory, Algorithms, and Systems, third edition.* New York: Springer Science + Business Media.

[4] M.R. Garey and D.S. Johnson, 1979. *Computers and Intractability, a Guide to the Theory of NP-Completeness.* W.H. Freeman Company: San Francisco.

[5] Shyh-In Hwang and Sheng-Tzong Cheng, 2001. Combinatorial Optimization in Real-Time Scheduling: Theory and Algorithms. *Journal of Combinatorial Optimization.* 5, 345-375.