ORSA Journal on Computing
Vol. 2, No. 3, Summer 1990

0899-1499/90/0203-0219 $01.25

Optimal Preemptive Scheduling of Two Unrelated Processors

TEOFILO GONZALEZ *Department of Computer Science, University of California, Santa Barbara, CA 93106*
ARPANET: teo@ucsb.edu

EUGENE L. LAWLER *Department of Electrical Engineering and Computer Sciences, and the Electronics Research Laboratory, University of California, Berkeley, CA 94720*
ARPANET: lawler@ernie.berkeley.edu

SARTAJ SAHNI *Department of Computer Science, University of Minnesota, Minneapolis, MN 55455*
ARPANET: sahni@umn-cs.cs.umn.edu

(Received: September 1989; final revision received: December 1989; accepted: February 1990)

The problem of constructing makespan-optimal preemptive schedules for n independent jobs on m unrelated parallel processors is discussed. For the case of two processors, we present a linear time algorithm to construct optimal schedules. The schedules generated by our algorithm have at most two preemptions.

In this paper we study the problem of preemptively scheduling n independent jobs on m unrelated parallel processors, with the objective of minimizing "makespan," i.e., the completion time of the last job to be finished. For the case of two processors, we present a linear time algorithm to construct optimal schedules. Our algorithm constructs optimal schedules with no more than two preemptions. This algorithm can be generalized to construct optimal schedules in $O(n^k)$ time when the number of processors is bounded by some fixed constant k. For arbitrary values of m, our problem can be formulated as a linear programming problem and thus it can be solved by standard simplex algorithms or by other algorithms that solve linear programming problems.[9, 10]

As part of the general problem formulation, we assume that there are m *processors*, indexed $i = 1, 2, \ldots, m$ and n *jobs*, indexed $j = 1, 2, \ldots, n$. A processor can work on only one job at a time, and a job can be worked on by only one processor at a time. The processing of a job may be interrupted at any time and resumed at a later time, by the same processor or a different processor. There is no cost and no time loss associated with such an interruption or "preemption."

The input data for a problem instance consists of integers n and m, and mn positive numbers $p_{i,j}$, where $p_{i,j}$ represents the total processing time required to complete job j, if the job is worked on exclusively by processor i. More generally, if processor i works on job j for a total time $x_{i,j}$, then it is necessary that $\sum_{i=1}^{m} (x_{i,j}/p_{i,j}) = 1$, in order for the job to be completed.

We assume no particular relation between the $p_{i,j}$ values. That is, the processors are *unrelated*. This is in contrast to the following two more specialized cases. If for all i, j, and k, $p_{i,j} = p_{k,j}$, then the processors are *identical*. If each $p_{i,j}$ can be expressed in the form $p_{i,j} = p_j/s_i$ where s_i (a "speed factor") and p_j are parameters associated with machine i and job j, then the machines are said to be *uniform*. For identical processors, a simple $O(n)$ algorithm, due to McNaughton,[7] yields an optimal schedule with no more than $m - 1$ preemptions. Gonzalez and Sahni[5] have obtained a more complex $O(n + m \log m)$ algorithm for the case of uniform processors, and have shown that no more than $2(m - 1)$ preemptions are required in an optimal solution.

For a given feasible schedule, the last point in time at which job j is processed is its *completion time* C_j. We wish to find a feasible schedule for which *makespan* or maximum completion time, $C_{\max} = \mathbf{max}_j \{C_j\}$, is minimized. We assume that all jobs are available for processing at time $t = 0$. Consider any feasible schedule of n jobs on m unrelated processors, where with respect to this schedule, $x_{i,j}$ is the total amount of time that processor i works on job j. It is evident that the values of $C_{\max}$ and $x_{i,j}$ for the schedule constitute a feasible

Subject classification Production/Scheduling, analysis of algorithms.
Other key words Deterministic scheduling, linear time algorithms, minimize makespan, unrelated processors.

solution to the following linear programming problem:

$$\text{minimize} \quad C_{\max}$$

subject to

$$\begin{aligned} &\sum_{i=1}^{m} \frac{x_{i,j}}{p_{i,j}} = 1, \qquad j = 1, 2, \ldots, n \\ &\sum_{i=1}^{m} x_{i,j} \leqslant C_{\max}, \quad j = 1, 2, \ldots, n \\ &\sum_{j=1}^{n} x_{i,j} \leqslant C_{\max}, \quad i = 1, 2, \ldots, m \\ &x_{i,j} \geqslant 0. \end{aligned} \tag{1}$$

The converse is also true. That is, for any feasible solution to the linear programming problem, there is a feasible schedule with the same values of $x_{i,j}$ and $C_{\max}$, as shown in [6]. Such a schedule can be obtained through the preemptive "open shop" problem.[3,4]

The bulk of this paper is concerned with finding an efficient procedure for solving the linear programming problem (1) for the case when $m = 2$ (Sections 1 and 2). In Section 3 we briefly discuss the case when $m > 2$.

1. Optimal Solutions to the Two-Processor Problem

Let us restrict our attention to the case $m = 2$, with the objective of obtaining a characterization of optimal solutions to the linear programming problem (1). For simplicity, let us assume that the jobs have been ordered in such a way that $p_{1,1}/p_{2,1} \leqslant p_{1,2}/p_{2,2} \leqslant \ldots \leqslant p_{1,n}/p_{2,n}$.

Let us call a column t of a feasible solution $X = (x_{i,j})$ and the corresponding job t tight if $x_{1,t} + x_{2,t} = C_{\max}$. If $n > 2$, as we shall hereafter assume, there can be at most one tight column. Hence, an optimal solution to (1) is of one of the following three types.

Type-1. There is a tight column t for which either

$$\frac{p_{1,t}}{p_{2,t}} \leqslant 1, \quad p_{1,t} \geqslant \sum_{j \neq t} p_{2,j} \tag{2}$$

or

$$\frac{p_{1,t}}{p_{2,t}} \geqslant 1, \quad p_{2,t} \geqslant \sum_{j \neq t} p_{1,j} \tag{3}$$

Type-2. There is a tight column t for which neither (2) nor (3) holds.

Type-3. There are no tight columns.

A schedule corresponding to a type-i optimal solution to (1) is called a *type-i schedule.* Let us now examine in more detail and establish some important properties of the three different types of optimal solutions to (1).

Type-1 Solutions and Schedules

A type-1 schedule (corresponding to a type-1 optimal solution to (1)) for a problem instance that satisfies (2) has no preemptions and it is of the form shown in Figure 1. Similarly, for a problem instance that satisfies (3), the type-1 schedule has no preemptions and it is of the form shown in Figure 2.

Type-2 Solutions and Schedules

Type-2 schedules are of the form shown in Figure 3. The shaded portions of the schedule are completely occupied with the processing of jobs other than t.

We assert that there exists a job s such that

$$\begin{aligned} &x_{2,j} = 0, \quad \text{for all } j < s, j \neq t \\ &\text{and} \\ &x_{1,j} = 0, \quad \text{for all } j > s, j \neq t. \end{aligned} \tag{4}$$

In other words, there exists an optimal schedule, with at most two preemptions, of the form given in Figure 4. This assertion can be established by the following argument: If there exists an optimal schedule with jobs j and k, $j < k$ and $j, k \neq t$; $x_{2,j} > 0$; and $x_{1,k} > 0$; then it is possible to increase the values of $x_{1,j}$ and $x_{2,k}$ and to decrease $x_{2,j}$ and $x_{1,k}$ while maintaining optimality. A finite sequence of such rearrangements yields an optimal schedule satisfying (4).

We also claim that if $p_{1,t}/p_{2,t} \leqslant 1$ then there is an optimal solution with $s < t$, otherwise $s > t$. Let us prove first the case when $p_{1,t}/p_{2,t} \leqslant 1$. The proof is by contradiction. Suppose that $p_{1,t}/p_{2,t} \leqslant 1$ and $s > t$. Then it is possible to increase the values of $x_{1,t}$ and $x_{2,s}$ and to decrease $x_{2,t}$ and $x_{1,s}$ and maintain or improve the length of the schedule. After a finite sequence of such rearrangements we know that there is a better schedule

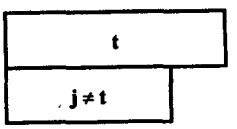

Figure 1. Schedule type-1 for a problem instance that satisfies (2).

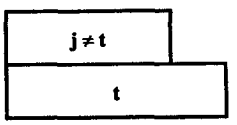

Figure 2. Schedule type-1 for a problem instance that satisfies (3).

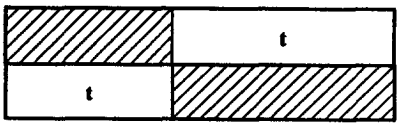

Figure 3. Type-2 schedule.

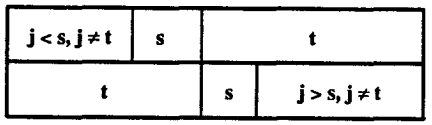

Figure 4. Type-2 schedule.

(contradicting the optimality of the previous one) or there is an optimal schedule with $s < t$. It is important to note that the above interchange argument will not work when $p_{1,t}/p_{2,t} < 1$ and $s < t$. The reason is that the length of the schedule will increase because the total execution time requirement for job t increases. Also, when $p_{1,t}/p_{2,t} = 1$ and $s < t$ the interchange argument will not reduce the length of the schedule. We omit the proof for the case when $p_{1,t}/p_{2,t} \geq 1$ and $s > t$ since it is similar to the one above. This concludes the proof of the claim.

Type-3 Solutions and Schedules

We assert that in this case there exists a job s satisfying (4) and that there is an optimal schedule of the form shown in Figure 5. As in the case of type-2 optimal schedules, the existence of such an optimal schedule can be verified by a sequence of rearrangements.

The question that arises at this point is: How can the value of t be determined? The answer to this question is simple and it is based on the following lemma.

Lemma 1. *If there is a type-1 or type-2 optimal solution to* (1), *the tight job t is uniquely determined by the inequalities*

$$\sum_{j<t} p_{1,j} \leq \sum_{j\geq t} p_{2,j},$$

and (5)

$$\sum_{j\leq t} p_{1,j} \geq \sum_{j>t} p_{2,j}.$$

And, moreover, if there is type-3 optimal solution, the inequalities (5) *are satisfied with job s in the role of t.*

Proof. The proof is partitioned into three parts depending on the type of optimal solution to (1).

Case 1. If there is a type-1 optimal solution to (1) then t is given by (5). There are two cases depending on whether (2) or (3) hold.

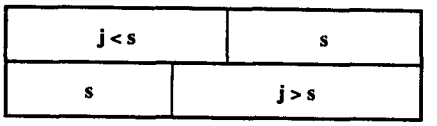

Figure 5. Type-3 schedule.

Subcase 1.1. $p_{1,t}/p_{2,t} \leq 1$ and $p_{1,t} \geq \sum_{j\neq t} p_{2,j}$.

Since the $p_{i,j}$'s are positive real numbers, we know that $\sum_{j\neq t} p_{2,j} \geq \sum_{j<t} p_{2,j}$ and since $p_{1,j}/p_{2,j} \leq 1$ for $j \leq t$, we know that $\sum_{j<t} p_{2,j} \geq \sum_{j<t} p_{1,j}$. Clearly, $\sum_{j\geq t} p_{2,j} \geq p_{2,t}$ and since $p_{1,t}/p_{2,t} \leq 1$, we know that $p_{2,t} \geq p_{1,t}$. If it were the case that $\sum_{j<t} p_{1,j} > \sum_{j\geq t} p_{2,j}$, then combining the above inequalities we know that $\sum_{j\neq t} p_{2,j} > p_{1,t}$ which contradicts the condition of this case. Therefore, it must be that $\sum_{j<t} p_{1,j} \leq \sum_{j\geq t} p_{2,j}$.

Since the $p_{i,j}$'s are positive real numbers, we know that $\sum_{j\leq t} p_{1,j} \geq p_{1,t}$. This inequality together with the condition of this case ($p_{1,t} \geq \sum_{j\neq t} p_{2,j}$) and the fact that $\sum_{j\neq t} p_{2,j} \geq \sum_{j>t} p_{2,j}$, we know that $\sum_{j\leq t} p_{1,j} \geq \sum_{j>t} p_{2,j}$. Therefore (5) holds when there is a type-1 optimal solution to (1) and inequality (2) holds.

Subcase 1.2. $p_{1,t}/p_{2,t} \geq 1$ and $p_{2,t} \geq \sum_{j\neq t} p_{1,j}$. The proof is omitted since it is similar to the one for Subcase 1.1. This completes the proof of case 1.

Case 2. If there is a type-2 optimal solution to (1) then t is given by (5). There are two subcases depending on whether $p_{1,t}/p_{2,t} \leq 1$ or not.

Subcase 2.1. $p_{1,t}/p_{2,t} \leq 1$. Since $p_{1,j}/p_{2,j} \leq 1$ for all $j < t$, $\sum_{j<t} p_{1,j} < \sum_{j<t} x_{1,j} + \sum_{j<t} x_{2,j} \leq C_{\max}$. Since $C_{\max} = x_{1,t} + x_{2,t}$, and $p_{1,t}/p_{2,t} \leq 1$, it follows that $C_{\max} \leq p_{2,t}$. Since the $p_{i,j}$'s are positive real values, we know that $p_{2,t} \leq \sum_{j\geq t} p_{2,j}$. Combining the above inequalities, $\sum_{j<t} p_{1,j} < p_{2,t} \leq \sum_{j\geq t} p_{2,j}$. From Figure 4 and the fact that $s < t$ we know that

$$\sum_{j\leq t} p_{1,j} \geq p_{1,t} \geq x_{1,t} \sum_{j>s} p_{2,j} \geq \sum_{j>t} p_{2,j}.$$

Hence, (5) holds when there is a type-2 optimal solution to (1) and $p_{1,t}/p_{2,t} \leq 1$.

Subcase 2.2. $p_{1,t}/p_{2,t} \geq 1$. The proof for this case is omitted since it is similar to the one of Subcase 2.1.

Case 3. If there is a type-3 optimal solution to (1) then s is given by t in (5). The proof for this case is omitted since it is similar to the one of Case 2.

This completes the proof of the lemma. ■

Our strategy is to first find the unique job t satisfying inequalities (5). Having found such a job t, it is easy to check if there is a type-1 solution to (1) and, if so, to construct a corresponding type-1 optimal schedule. Optimality is guaranteed by Lemma 1. So let us

suppose that there is no type-1 optimal solution to (1). In this case, we have the following two equations:

$$\sum_{j<t} p_{1,j} + x_{1,t} = \sum_{j>t} p_{2,j} + x_{2,t},$$

and

$$\frac{x_{1,t}}{p_{1,t}} + \frac{x_{2,t}}{p_{2,t}} = 1.$$

These two equations have the following solution.

$$x_{1,t} = \left[\frac{p_{1,t}}{p_{1,t} + p_{2,t}}\right]\left[\sum_{j\geqslant t} p_{2,j} - \sum_{j<t} p_{1,j}\right]$$

and (6)

$$x_{2,t} = \left[\frac{p_{2,t}}{p_{1,t} + p_{2,t}}\right]\left[\sum_{j\leqslant t} p_{1,j} - \sum_{j>t} p_{2,j}\right].$$

Note that necessarily $x_{1,t}, x_{2,t} \geqslant 0$, since the bracketed expressions are nonnegative by (5). If

$$x_{1,t} < \sum_{j>t} p_{2,j}$$

and (7)

$$x_{2,t} < \sum_{j<t} p_{1,j},$$

then there is a type-3 solution with $s = t$. The type-3 schedule is shown in Figure 5. Optimality follows from Lemma 1. If there are no type-1 and type-3 optimal solutions to (1) then there is a type-2 optimal solution to (1). For this case we know that the appropriate choice of t is given by (5). But, what is the appropriate choice for s? Equations 8 and 9 define a value for s. In Lemma 2 we show that this is an appropriate choice for s.

If there is a type-2 optimal solution to (1) and there are no type-1 and type-3 optimal solutions to (1), then s is defined as follows.

(i) If $p_{1,t}/p_{2,t} \leqslant 1$, column s ($s < t$) is given by the smallest integer that satisfies the following two inequalities

$$\sum_{j<s,j\neq t} p_{1,j}/p_{2,t} + \sum_{j\geqslant s,j\neq t} p_{2,j}/p_{1,t} > 1,$$

and (8)

$$\sum_{j\leqslant s,j\neq t} p_{1,j}/p_{2,t} + \sum_{j>s,j\neq t} p_{2,j}/p_{1,t} \leqslant 1.$$

(ii) If $p_{1,t}/p_{2,t} > 1$, column s ($s > t$) is given by the largest integer that satisfies the following two inequalities

$$\sum_{j\leqslant s,j\neq t} p_{1,j}/p_{2,t} + \sum_{j>s,j\neq t} p_{2,j}/p_{1,t} > 1,$$

and (9)

$$\sum_{j<s,j\neq t} p_{1,j}/p_{2,t} + \sum_{j\geqslant s,j\neq t} p_{2,j}/p_{1,t} \leqslant 1.$$

Let us now show that such a value for s always exists. Since the proof for (ii) is similar to the one for (i), we only prove (i). Clearly, $p_{1,t}/p_{2,t} \leqslant 1$. Since there is no type-1 solution to (1), we know that $p_{1,t} < \sum_{j\neq t} p_{2,t}$. Therefore when $s = 1$,

$$\sum_{j<s,j\neq t} p_{1,j}/p_{2,t} + \sum_{j\geqslant s,j\neq t} p_{2,j}/p_{1,t} > 1.$$

Since there is no type-3 optimal solution to (1), we know that $x_{1,t} > \sum_{j>t} p_{2,j}$ and $x_{2,t} > \sum_{j<t} p_{1,t}$. Substituting these bounds in the equation $x_{1,t}/p_{1,t} + x_{2,t}/p_{2,t} = 1$, we know that for $s = t - 1$,

$$\sum_{j\leqslant s,j\neq t} p_{1,j}/p_{2,t} + \sum_{j>s,j\neq t} p_{2,j}/p_{1,t} < 1.$$

Therefore there exists an s ($s < t$) that satisfies (8). This completes the proof of (i) and the claim.

The following lemma shows that the value of s given by (8) and (9) is optimal.

Lemma 2. *If there are no type-1 and type-3 optimal solutions to* (1), *then there is a type-2 optimal solution to* (1) *with t given by* (5) *and s given by either* (8) *or* (9).

Proof. Since there are no optimal solutions of type-1 and type-3, it is simple to see that there is a type-2 optimal solution to (1) whose corresponding schedule is of the form given in Figure 4. This schedule is defined over variables $x_{i,j}$ and $C_{\max}$, with $C_{\max} = x_{1,t} + x_{2,t} = \sum_{j<s,j\neq t} p_{1,j} + x_{1,s} + x_{2,s} + \sum_{j>s,j\neq t} p_{2,j}$. From Lemma 1 we know that the appropriate choice for t is given by (5). Let us prove that the appropriate choice of s is given by either (8) or (9).

We only prove that if $p_{1,t}/p_{2,t} \leqslant 1$ then s is given by (8), since the proof for the other case is similar. Suppose that the above schedule is not optimal. Let $x'_{i,j}$ and $C'_{\max}$ ($C'_{\max} < C_{\max}$) be an optimal schecule of the same form with job s' and t being the jobs scheduled with preemptions. The schedule is given in Figure 6. If $p_{1,t}/p_{2,t} = 1$, we know that $C_{\max} = x_{1,t} + x_{2,t} = x'_{1,t} + x'_{2,t} = C'_{\max}$. A contradiction. So it must be that $p_{1,t}/p_{2,t} < 1$. Let us now consider the case $s' < s$. Since $s < t$, we know that $p_{1,s'}/p_{2,s'} \leqslant p_{1,s}/p_{2,s} \leqslant p_{1,t}/p_{2,t} < 1$. Therefore it must be that $C_{\max} = \sum_{j<s,j\neq t} p_{1,j} + x_{1,s} + x_{2,s} + \sum_{j>s,j\neq t} p_{2,j} < \sum_{j<s',j\neq t} p_{1,j} + x_{1,s'} + x_{2,s'} + \sum_{j>s',j\neq t} p_{2,j} = C'_{\max}$. A contradiction. So, it must be that $s' > s$. However, for this case we know that

$$\sum_{j<s',j\neq t} p_{1,j} + x'_{1,s'} > \sum_{j<s,j\neq t} p_{1,j} + x_{1,s}.$$

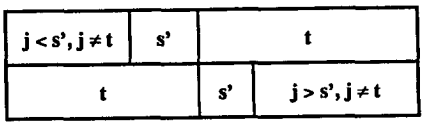

Figure 6. Schedule for $x'_{i,j}$.

Therefore it must be that $x'_{2,t} > x_{2,t}$. But since $p_{1,t}/p_{2,t} < 1$, $C_{\max} = x_{1,t} + x_{2,t} < x'_{1,t} + x'_{2,t} = C'_{\max}$. A contradiction. So, it must be that an optimal solution is given by the choice of s in (8). This completes the proof of the lemma. ■

Once we have determined t and s we solve the following linear programming problem. Since this linear programming problem has at most 4 variables and 6 constraints, it is fair to assume that it can be solved in constant time.

$$\text{minimize} \quad C_{\max}$$

subject to

$$\begin{aligned}
&\frac{x_{1,j}}{p_{1,j}} + \frac{x_{2,j}}{p_{2,j}} = 1, \qquad j = s, t \\
&x_{1,j} + x_{2,j} \leqslant C_{\max}, \quad j = s, t \\
&x_{1,s} + x_{1,t} + \sum_{j<s, j\neq t} p_{1,j} \leqslant C_{\max}, \qquad (10) \\
&x_{2,s} + x_{2,t} + \sum_{j>s, j\neq t} p_{2,j} \leqslant C_{\max}, \\
&x_{1,s},\ x_{2,s},\ x_{1,t},\ x_{2,t} \geqslant 0.
\end{aligned}$$

2. An $O(n)$ Algorithm for Two Processors

To summarize, our procedure is as follows:

(i) Find the unique job t satisfying inequalities (5).
(ii) If t satisfies either (2) or (3), then there is a type-1 solution and an optimal schedule with no preemptions has been found.
(iii) If the solution to (6) satisfies inequalities (7), then there is a type-3 solution and an optimal schedule with at most one preemption has been found.
(iv) Find the job s satisfying inequalities (8) or (9). An optimal schedule with at most two preemptions can be found from the solution to the linear programming problem given by (10).

If the ratios are initially sorted, it is simple to see that all steps can be carried out in $O(n)$ time. When the input is not initially sorted all parts of this procedure can be implemented to run in $O(n)$ time except possibly (i), finding the job t satisfying inequalities (5), and (iv), finding the job s satisfying inequalities (8) or (9). We propose to refine this approach so as to reduce the overall time required to solve the two-processor problem to $O(n)$. We first observe that it is not necessary to sort the $p_{1,j}/p_{2,j}$ values in order to identify job t by conditions (5). Instead, we propose to employ as a subprocedure an algorithm which can find the median of n numbers in $O(n)$ time.[2,8] First compute all the ratios $p_{1,j}/p_{2,j}$, $j = 1, 2, \ldots, n$, which can, of course be done in $O(n)$ time. Next find $p_{1,k}/p_{2,k}$, the median of the n values $p_{1,1}/p_{2,1}$, $p_{1,2}/p_{2,2}$, $\ldots$, $p_{1,n}/p_{2,n}$ and with respect to k let

$$J = \left\{ j \;\middle|\; \left(\frac{p_{1,j}}{p_{2,j}} < \frac{p_{1,k}}{p_{2,k}}\right) \text{ or } \left(\frac{p_{1,j}}{p_{2,j}} = \frac{p_{1,k}}{p_{2,k}} \text{ and } j < k\right) \right\},$$

and

$$J' = \{j \mid 1 \leqslant j \leqslant n\} - J.$$

Recalling conditions (5), it is clear that if both

$$\sum_{j\in J} p_{1,j} < \sum_{j\in J'} p_{2,j}$$

and

$$\sum_{j\in J} p_{1,j} + p_{1,k} \geqslant \sum_{j\in J'} p_{2,j} - p_{2,k},$$

then $t = k$, and job t has been found in $O(n)$ time. If the first inequality in (5) does not hold, it is known that $t \in J$. If the first inequality in (5) holds, but the second inequality in (5) does not, it is known that t is contained in J' (i.e., the complement of J). In either of these latter two cases, the search for job t is narrowed to a set of $n/2$ possibilities. The procedure is then iterated until $p_{1,k}/p_{2,k}$ and corresponding set J are found for which conditions (5) are satisfied. At this point job t has been found.

In the worst case, the median-finding algorithm is applied successively to sets of size n, $n/2$, $n/4$, $\ldots$, which require time $\leqslant cn + cn/2 + cn/4 + \ldots$, or $O(n)$. Since the sums appearing in (5) are known from the previous iteration, the sums required for testing (5) can be found at each iteration in time proportional to the application of the median-finding algorithm at that iteration. It follows that job t can be found in $O(n)$ time overall. It is simple to show that finding job s can also be carried out in $O(n)$ time by following a procedure similar to the one used to find t.

It is possibly worth noting the similarity between some of the techniques applied here and the methods used by Adolphson and Thomas[1] for obtaining a linear time algorithm for the $2 \times n$ transportation problem. Lemma 1 here is comparable to Theorem 1 in [1].

The authors have been able to generalize the preceding results to the case of $m > 2$ processors so as to obtain an algorithm with $O(n^m)$ running time. This generalization is fairly involved and difficult to describe. Moreover, it is unclear that it has any practical value, in competition with a straightforward solution of the LP by other means.

ACKNOWLEDGMENT

This research was supported in part by the National Science Foundation under grants MCS76-21024 and MCS76-17605.

REFERENCES

1. D.L. ADOLPHSON and G.N. THOMAS, 1977. A Linear Time Algorithm for a $2 \times n$ Transportation Problem, *SIAM Journal on Computing 6*, 481–486.
2. M. BLUM, R.W. FLOYD, V. PRATT, R.L. RIVEST and R.E. TARJAN, 1973. Time Bounds for Selection, *Computer and Systems Science 7*, 448–461.
3. T. GONZALEZ and S. SAHNI, 1976. Open Shop Scheduling to Minimize Finish Time, *Journal of the ACM 23:4*, 665–679.
4. T. GONZALEZ, 1979. A Note on Open Shop Preemptive Schedules, *IEEE Transactions on Computers C-28:10*, 782–786.
5. T. GONZALEZ and S. SAHNI, 1978. Preemptive Schedules of Uniform Processor Systems, *Journal of the ACM 25:1*, 92–101.
6. E.L. LAWLER and J. LABETOULLE, 1978. On Preemptive Scheduling of Unrelated Parallel Processors by Linear Programming, *Journal of the ACM 25:4*, 612–619.
7. R. MCNAUGHTON, 1959. Scheduling with Deadlines and Loss Functions, *Management Science 6*, 1–12.
8. A. SCHONHAGE, M. PATERSON and N. PIPPENGER. Finding the Median, University of Warwick Technical Report 6, Coventry, England.
9. L.G. KHACHIYAN, 1979. A Polynomial Algorithm in Linear Programming, *Doklady Akademiia Nauk SSSR 244:S*, 1093–1096, translated in *Soviet Mathematics Doklady 20:1*, 191–194.
10. N. KARMARKAR, 1984. A New Polynomial-Time Algorithm for Linear Programming, *Combinatorica 4*, 373–395.