JOURNAL OF MATHEMATICAL ANALYSIS AND APPLICATIONS 196, 181–195 (1995)

An LPT-Bound for a Parallel Multiprocessor Scheduling Problem

FARID HARCHE AND SRIDHAR SESHADRI

Department of Statistics & Operations Research, Leonard N. Stern School of Business, New York University, New York, New York 10012

Submitted by Laurence A. Baxter

Received June 10, 1994

This paper is concerned with the problem of assigning n jobs with known processing times to m machines to minimize makespan. Each machine has a fixed capacity expressed as the maximum number of jobs that can be assigned to it. We investigate the worst-case behavior of the longest processing time heuristic for this problem. For the case of two machines with equal capacity, it is shown that the worst case error bound is $\frac{7}{6}$. © 1995 Academic Press, Inc.

1. INTRODUCTION

We consider a parallel multi-processor scheduling problem, denoted CMP, defined as follows. There is a set of n independent jobs, each with a known processing time, that must be processed on m machines, each with a fixed capacity. The problem is to allocate jobs to machines subject to the following assumptions:

(i) a job can be processed on any machine,

(ii) each job must be processed on one and only one machine,

(iii) the number of jobs assigned to each machine may not exceed its capacity,

(iv) no job preemption is allowed,

(v) the processing time of a job is independent of the machine.

The scheduling objective is to minimize the makespan, the time required for the completion of all jobs. The importance of this problem is reflected by the wide range of its applications which include assembly of printed

0022-247X/95 $12.00

circuit boards [10, 1, 2], design of flexible manufacturing systems [9, 12], configuration of multi-aisle automated storage and retrieval systems [13], and the optimal assembly of systems [3]. In all these applications, the machine capacity is often expressed in terms of the maximum number of jobs that can be processed instead of the total processing time available on a machine. For instance, in the context of the assembly of printed circuit boards, the problem under consideration can be described as follows. A set of components is to be inserted on printed circuit boards using a flexible manufacturing system (FMS). The FMS contains a number of robotic insertion machines which insert the components. The components have to be staged on a machine before they can be inserted into circuit boards, and there is a limit on the number of components which can be staged on any machine, called the staging capacity. The insertion of each component has an associated processing time. The problem is then to make an assignment of components to machines so that the maximum total processing time assigned to a machine is minimized while satisfying the staging capacity. Unfortunately, the CMP problem is well known to be NP-hard. Therefore, previous work on this problem has typically been devoted to developing heuristic approaches [4, 14, 10]. Recently, some efforts have been made to obtain a deeper understanding of the mathematical structure underlying the CMP [8].

By far the largest body of research work has been done on the special case of the CMP problem when all machines have unlimited capacity. Much of the research work has centered on developing bounds on the worst-case behavior of heuristic algorithms. Worst-case analyses of several heuristics can be found in [5–7, 11]. Among these heuristics, particular attention has been paid to the longest processing time (LPT) heuristic. The LPT heuristic begins by sorting the n jobs in order of nonincreasing processing time. It then sequentially assigns the next job to the machine for which the current total processing time is least. A basic result due to Graham [7] states that the LPT heuristic gives a worst case ratio of $(\frac{4}{3} - 1/3m)$. Surprisingly, the worst case behavior of the LPT heuristic for the CMP is still an open problem. The purpose of this study is to bridge this gap.

2. Worst-Case Analysis

In this section, we prove a key result concerning the worst-case performance of the LPT heuristic in the case when $m = 2$ and the machines have identical capacity. We first introduce some notation.

Let $I = \{1, ..., 2l + 2\}$ denote the set of jobs to be processed. For each $i \in I$, w_i denotes the processing time of job i. Assume that $m = 2$ and that machine capacity is set equal to $l + 1$. It is henceforth assumed that the

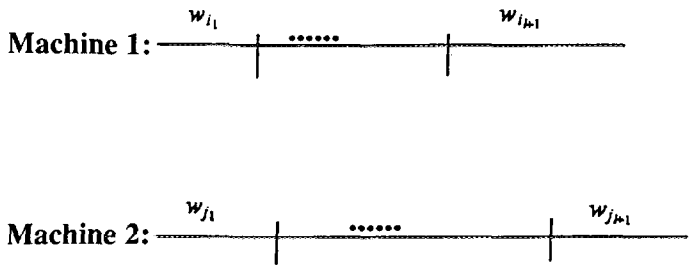

FIGURE 1

jobs are labeled such that $w_1 \geq w_2 \geq \cdots \geq w_{2l+2}$. Let $W = \{w_1, w_2, ..., w_{2l+2}\}$. Denote by $I_1 = \{i_1, i_2, ..., i_{l+1}\}$ and $I_2 = \{j_1, j_2, ..., j_{l+1}\}$ the sets of jobs assigned by the LPT rule to machine 1 and machine 2, respectively.

For a given set W and machine capacity $(= l + 1)$, let $v(2l + 2, W, \text{CAP})$ be the value of the makespan obtained by the LPT rule, and $\bar{v}(2l + 2, W, \text{CAP})$ be the optimal value of the makespan. Similarly, let $v(2l + 2, W, \text{UNCAP})(\bar{v}(2l + 2, W, \text{UNCAP}))$ be the value of makespan obtained by the LPT (optimal) rule when all machines have unlimited capacity.

Now we are ready to state the main result of the paper.

THEOREM 1. *For the two-machine case and any set W of job processing times and any integer l, we have*

$$\sup_{l,W} \frac{v(2l + 2, W, \text{CAP})}{\bar{v}(2l + 2, W, \text{CAP})} \leq \frac{7}{6}. \tag{1}$$

Proof. By contradiction. Suppose that the theorem is false. Then there exists an example with the smallest value of l that does not satisfy inequality (1). Consider an LPT assignment of jobs to machines for this example. We may assume without loss of generality that the last unscheduled job is allocated to machine 2, i.e., $j_{l+1} = 2l + 2$. We consider two cases.

Case 1. $\sum_{k=1}^{l} w_{j_k} \leq \sum_{k=1}^{l+1} w_{i_k}$.

Note that job i_{l+1} must have started processing on machine 1 before job j_{l+1} was assigned to machine 2; see Fig. 1. Therefore the capacity constraint did not play a role in the LPT assignment. It follows that $v(2l + 2, W, \text{UNCAP}) = v(2l + 2, W, \text{CAP})$. Moreover, it is not difficult to see that the inequality $\bar{v}(2l + 2, W, \text{CAP}) \geq \bar{v}(2l + 2, W, \text{UNCAP})$ always holds. Now, Graham [7] has shown that $v(2l + 2, W, \text{UNCAP})/\bar{v}(2l + 2, W, \text{UNCAP}) \leq \frac{7}{6}$. We then obtain

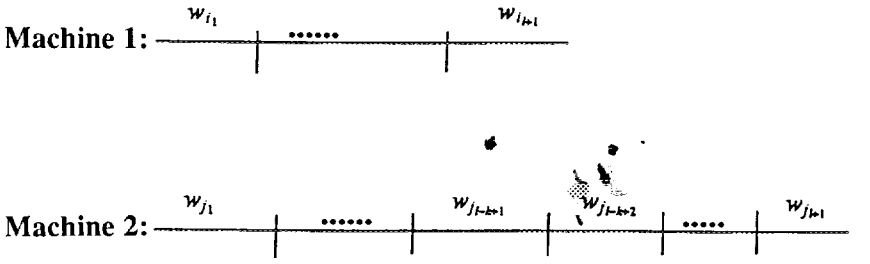

FIGURE 2

$$\frac{7}{6} \geq \frac{v(2l+2, W, \text{UNCAP})}{\bar{v}(2l+2, W, \text{UNCAP})} \geq \frac{v(2l+2,, W, \text{CAP})}{\bar{v}(2l+2, W, \text{CAP})},$$

implying that the theorem holds.

Case 2. $\sum_{s=1}^{l} w_{j_s} > \sum_{s=1}^{l+1} w_{i_s}$.

Let $k > 0$. Suppose that job i_{l+1} starts processing on machine 1 when job j_{l-k+1} was being processed in machine 2; this situation is depicted in Fig. 2. Consider any pair of jobs (i_{l+1}, j_r) where $r \in \{l - k + 2, l - k + 3, ..., l + 1\}$. Assume that in the optimal assignment this pair is allocated to two different machines. Then removing this pair of jobs will give an example having a smaller value of l for which inequality (1) does not hold. To see this let us assume that the pair (i_{l+1}, j_r) is scheduled on different machines in the optimal assignment. Then the removal of this pair from the set of jobs I will reduce the optimal makespan by at least w_{j_r} and will decrease the makespan obtained by the LPT rule by w_{j_r}. Hence

$$\frac{7}{6} < \frac{v(2l+2, W, \text{CAP})}{\bar{v}(2l+2, W, \text{CAP})} > \frac{v(2l+2, W, \text{CAP}) - w_{j_r}}{\bar{v}(2l+2, W, \text{CAP}) - w_{j_r}},$$

$$\text{for } r = l - k + 2, ..., l + 1.$$

It follows that all jobs i_{l+1} and j_r, $l - k + 2 \leq r \leq l + 1$, must have been scheduled on the same machine in the optimal assignment. Now we may assume that the jobs just described have equal processing times. If this is not the case, then set

$$w_{i_{l+1}} = w_{j_r} = \frac{w_{i_{l+1}} + \sum_{s=l-k+2}^{l+1} w_{j_s}}{k+1}, \qquad \text{where } r = l - k + 2, ..., l + 1.$$

As a result, the value of the makespan obtained by LPT may increase

whereas the value of the optimal makespan may either decrease or remain unchanged. Note that

$$\frac{w_{i_{l+1}} + \sum_{s=l-k+2}^{l+1} w_{j_s}}{k+1} > 0. \tag{2}$$

For if not, i.e., if the equality holds in (2), it may be verified that the capacity constraint did not play a role in the LPT assignment. Then, as in the proof of Case 1, we have a contradiction. So assume without loss of generality that $w_{i_{l+1}} = w_{j_r} = 1$, for $r = l - k + 2, \ldots, l + 1$.

Denote $\zeta = \sum_{s=1}^{l+1} w_{j_s} - \sum_{s=1}^{l+1} w_{i_s}$.
Note that $\tilde{v}(2l + 2, W, \text{CAP}) \geq (\sum_{s=1}^{l+1} w_{j_s} + \sum_{s=1}^{l+1} w_{i_s})/2$.

Now, in view of our assumption that

$$\frac{7}{6}\tilde{v}(2l + 2, W, \text{CAP}) < v(2l + 2, W, \text{CAP})$$

it follows that

$$v(2l + 2, W, \text{CAP}) \geq \frac{7}{6}\tilde{v}(2l + 2, W, \text{CAP}) \geq \left(\frac{7}{6}\right)\left(\sum_{s=1}^{l+1} w_{j_s} + \sum_{s=1}^{l+1} w_{i_s}\right)/2$$

and since

$$v(2l + 2, W, \text{CAP}) = \sum_{s=1}^{l+1} w_{j_s} = \left(\sum_{s=1}^{l+1} w_{j_s} + \sum_{s=1}^{l+1} w_{i_s}\right)/2 + \frac{\zeta}{2}$$

we therefore have

$$\zeta \geq \left(\sum_{s=1}^{l+1} w_{j_s} + \sum_{s=1}^{l+1} w_{i_s}\right)/6. \tag{3}$$

In what follows, we need to distinguish between two subcases.

Subcase 2.1. $j_{l-k+1} < i_{l-k+1}$.

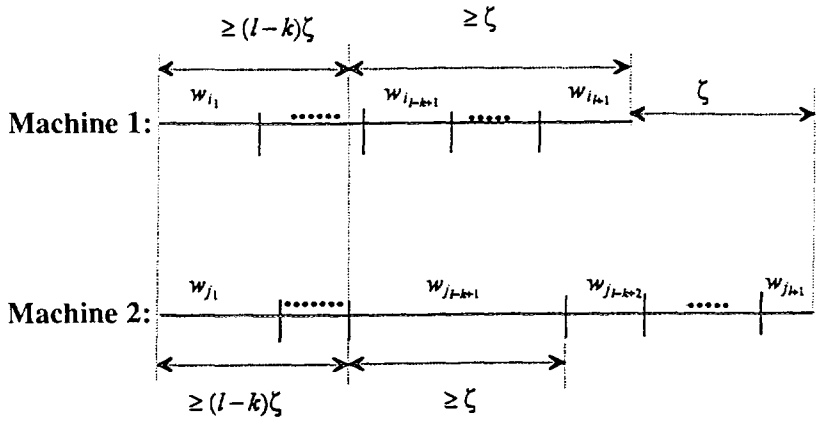

FIGURE 3

(i) First assume $\sum_{s=1}^{l-k+1} w_{j_s} \leq \sum_{s=1}^{l+1} w_{i_s}$; see Fig. 3. This implies that

$$\zeta \leq \sum_{s=l-k+2}^{l+1} w_{j_s} \leq \sum_{s=l-k+1}^{l} w_{i_s} \leq w_{j_{l-k+1}},$$

and so

$$\sum_{s=1}^{2l+2} w_s \geq 3\zeta + 2(l-k)\zeta.$$

By virtue of (3), and our assumption that (1) is not satisfied, we deduce that $l - k \leq 1$. When $l - k = 0$, then j_1 has to be equal to 1 and the value of the makespan generated by LPT is equal to $w_1 + k$ (recall that we set the smallest $k + 1$ processing times equal to 1). But since the optimal value of the makespan cannot be smaller than $w_1 + k$, $l - k \neq 0$. Now if $l - k = 1$ then $j_1 = 2$ (because $i_{l-k+1} = i_2 > j_2$) and $j_{l-k+1} = j_2 = 3$. It follows at once that the value of the makespan generated by LPT is equal to $w_2 + w_3 + k$. But in the optimal assignment, we must have one pair of jobs (1, 2) or (1, 3) or (2, 3) allocated to the same machine. This implies that the value of the optimal makespan could not be smaller than $w_2 + w_3 + k$, leading to a contradiction. Consequently, we can write $\sum_{s=1}^{l-k+1} w_{j_s} = \sum_{s=1}^{l+1} w_{i_s} + \delta$, where $\delta > 0$.

(ii) Now assume that

$$k + \sum_{s=l-k+1}^{l+1} w_{i_s} \geq \zeta. \tag{4}$$

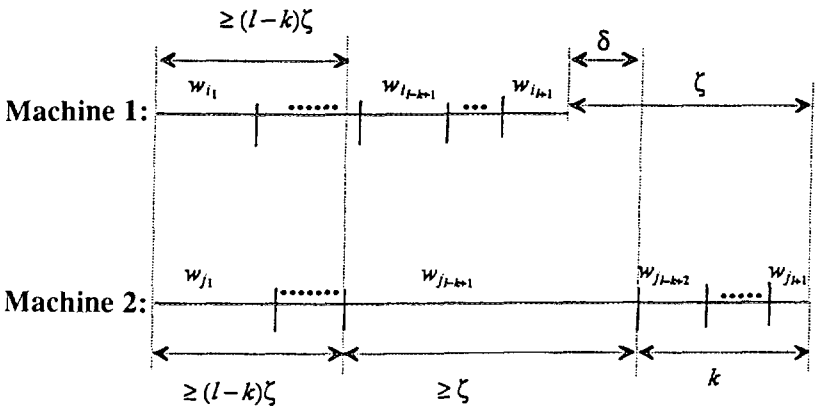

FIGURE 4

By reference to Fig. 4, we obtain

$$w_{j_{l-k+1}} \geq \delta + \sum_{s=l-k+1}^{l+1} w_{i_s} \geq \delta + k = \zeta \tag{5}$$

and considering (4) we obtain that

$$\sum_{s=1}^{2l+2} w_s \geq 2\zeta + 2(l-k)\zeta.$$

Similar to (i) above, it can be deduced that $(l - k) > 1$. But then

$$\sum_{s=1}^{2l+2} w_s \geq 6\zeta,$$

a contradiction. Thus $k + \sum_{s=l-k+1}^{l+1} w_{i_s} < \zeta$.

Next we wish to show that $l - k \geq k + 1$. Inequality (5) still holds. It might be useful to the reader to refer to Fig. 4. Using the inequality (5) one can also show that

$$\sum_{s=1}^{2l+2} w_{i_s} \geq \zeta + 2(l-k)\zeta. \tag{6}$$

The same argument as that used in the beginning of Case 2 can be used to show that in the optimal assignment all jobs i_r, for $r = l - k + 1, \ldots, l + 1$, and j_p, for $p = l - k + 2, \ldots, l + 1$, must be allocated to the same machine, say machine 2. (Observe that if jobs i_r and j_p were allocated to

different machines in the optimal schedule, then their removal can at most reduce the value of the LPT makespan by w_{j_p}.) Since these $2k + 1$ jobs have the smallest processing times among the set of all jobs, there must be at least one additional job allocated to machine 2 in the optimal assignment. Therefore, $l + 1 \geq (2k + 1) + 1$, showing that $l - k \geq k + 1$. Next we consider two subcases according to whether $k = 1$ or $k > 1$.

If $k > 1$ then $l - k > 2$. Therefore, combining inequalities (3) and (6) leads to a contradiction.

Now consider the case when $k = 1$. For $l - k > 2$ we obtain a contradiction as before. Suppose now that $l - k = 2$. Then this reduces to an example having $2l + 2 = 8$ jobs. Moreover, by the nature of the LPT rule, we must have $j_1 = 1$ and $j_2 = 4$ or $j_1 = 2$ and $j_2 = 3$, $j_{l-k+1} = j_3 = 5$, and $j_{l+1} = j_4 = 8$. As before assume that, in the optimal assignment, the $2k + 1 = 3$ smallest jobs 6, 7, and 8 are allocated to the same machine, say machine 2. Suppose that $j_1 = 1$ and $j_2 = 4$ implying $w_1 + w_4 \leq w_2 + w_3$. Then it can be checked that the value of the makespan (obtained by LPT) $v(8, W, \text{CAP}) = w_1 + w_4 + w_5 + w_8$. If in the optimal assignment job 1 is allocated to machine 2 (together with jobs 6, 7, and 8), we then have the optimal makespan

$$\tilde{v}(8, W, \text{CAP}) \geq w_2 + w_3 + w_4 + w_5 \geq w_1 + w_4 + w_4 + w_5 \geq v(8, W, \text{CAP}),$$

a contradiction. Similar arguments can be used to obtain a contradiction when jobs 2, 6, 7, and 8, or jobs 3, 6, 7, and 8, or jobs 4, 6, 7, and 8 are allocated to machine 2 in the optimal assignment.

Assume now that $j_1 = 2$ and $j_2 = 3$. In this case, $v(8, W, \text{CAP}) = w_2 + w_3 + w_5 + w_8$. If job 1 is allocated to machine 2 in the optimal assignment, we then have

$$\tilde{v}(8, W, \text{CAP}) \geq w_2 + w_3 + w_4 + w_5 \geq v(8, W, \text{CAP}),$$

a contradiction. The cases where jobs 2, 3, 4, respectively, one assigned to machine 2 in the optimal assignment (together with jobs 6, 7, and 8) can be analyzed in a similar fashion to obtain a contradiction. This concludes the analysis of Case 2.1.

Case 2.2. $j_{l-k+1} > i_{l-k+1}$.

Let $x \geq 1$, and assume that $j_{l-k+1} > i_{l-k+x}$ but $j_{l-k+1} < k_{l-k+x+1}$ as depicted in Fig. 5 above. Exactly the same argument as in the previous cases allows us to suppose, without loss of generality, that $l - k > 0$ and $w_{i_{l-k+x+1}} = w_{i_{l-k+x+2}} = \cdots = w_{i_{l+1}} = w_{j_{l-k+2}} = \cdots = w_{j_{l+1}} = 1$. Denoting $\delta = \sum_{s=1}^{l-k+1} w_{j_s} - \sum_{s=1}^{l-k+x} w_{i_s}$, we then have $\sum_{s=l-k+1}^{l-k+x} w_{i_s} + \delta - w_{j_{l-k+1}} \geq x - 1 + \delta = \zeta$.

Considering that job j_{l-k} was under process when job i_{l-k+1} started its

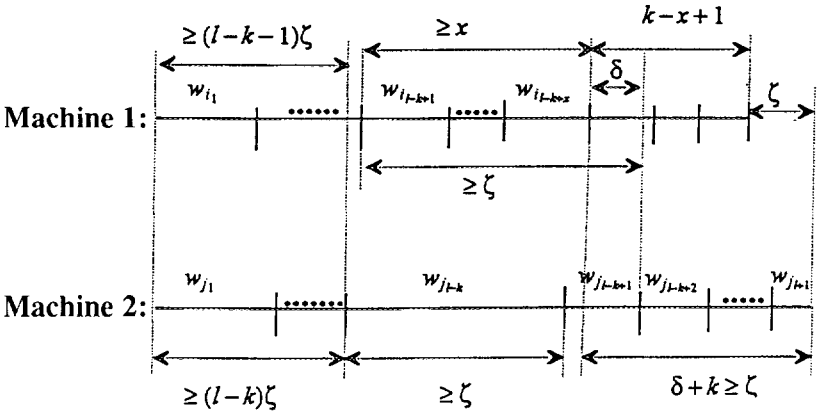

FIGURE 5

processing and that δ is the elapsed time between the completion of jobs i_{l-k+x} and j_{l-k+1} as indicated in Fig. 5, we then have

$$w_{j_{l-k}} \geq \sum_{s=l-k+1}^{l-k+x} w_{i_s} + \delta - w_{j_{l-k+1}} \geq x - 1 + \delta = \zeta.$$

By reference to Fig. 5, it then follows that $\sum_{s=1}^{2l+2} w_s \geq 2(l-k)\,\zeta + 2\zeta$.

Next we claim that $l - k > 1$. Note that $l - k = 0$ is not possible. So, assume that $l - k > 0$. It is sufficient to show that the case where $l - k = 1$ leads to a contradiction. If $l - k = 1$, then $j_1 = 1$. By the same reasoning as in the preceding cases we may assume that the $2k - x + 1$ smallest jobs $(2(l-k) + x + 2, 2(l-k) + x + 3, \ldots, 2l + 2)$ are allocated to the same machine, say machine 1, in the optimal assignment. Also, there must be at least one more job together with these $2k - x + 1$ jobs assigned to machine 1 in the optimal assignment. It must then be true that $2k - x + 2 \leq l + 1$ or $k - x \leq l - k - 1$. Since $l - k = 1$ we then have $k \leq x$. But by assumption $k - x + 1 > 0$. This follows from the fact that if $k - x + 1 = 0$ then the job that was under process in machine 2 when job i_{i+1} started processing in machine 1 will not be job j_{l-k+1}. Hence $x = k$. Before proceeding to the rest of the proof, it might be useful to describe the partial assignments thus far. Specifically, the sets of jobs $\{2, 3, \ldots, k + 2, k + 4\}$ and $\{1, k + 3, k + 5, \ldots, 2k + 4\}$ are assigned by the LPT rule to machines 1 and 2, respectively. However, in the optimal assignment, the sets of jobs $\{2, 3, \ldots, k + 2, k + 3\}$ and $\{1, k + 4, \ldots, 2k + 4\}$ are allocated to machines 1 and 2, respectively.

Let $b > 0$. First note that we may assume, without loss of generality, that $w_2 = \cdots = w_{k+2} = b$, as all these jobs are assigned to machine 1 in

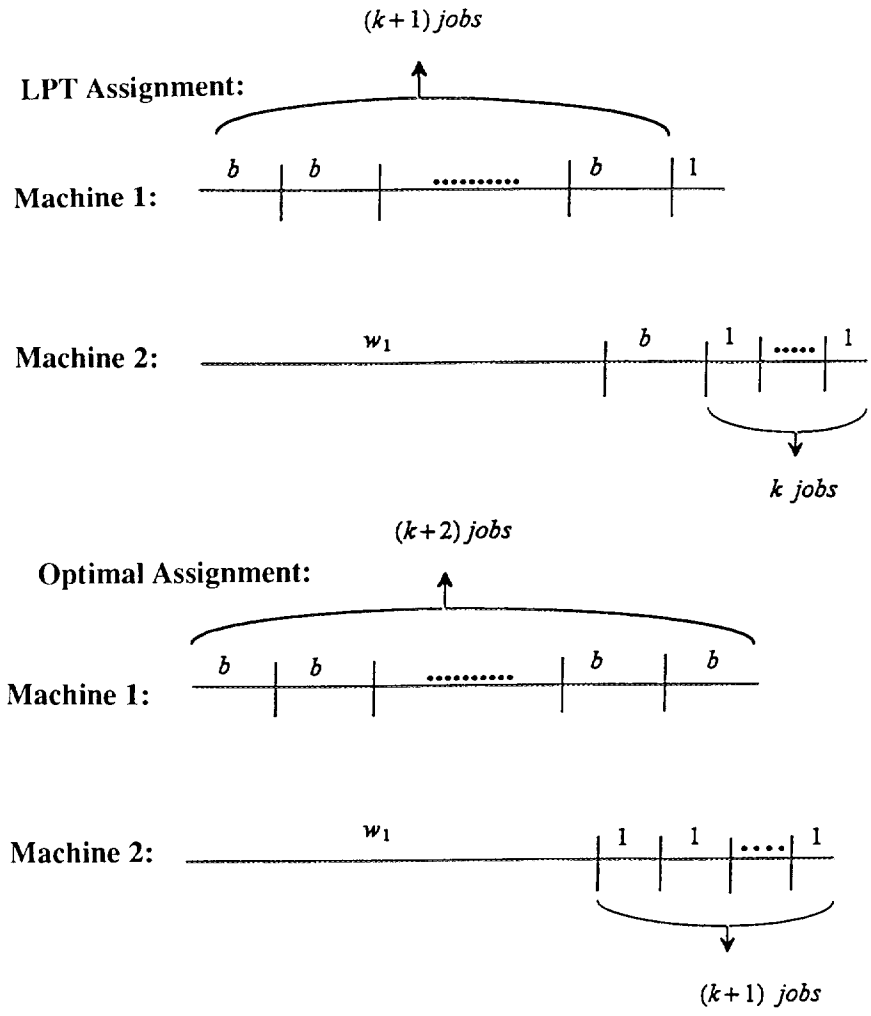

FIG. 6. Assignments of jobs in Case 2.2.

the optimal assignment. Moreover, if $w_{k+3} = b^1 < b$, then let $\delta = b - b^1$, and set $w_1 = w_1 - (k + 1)\,\delta$ and $w_2 = \cdots = w_{k+2} = b - \delta$. This will not affect the LPT assignment. However, the value of the optimal assignment will decrease by at least $(k + 1)\delta$. We may therefore assume without loss of generality that $w_2 = w_3 = \cdots = w_{k+3} = b$, as this does not affect the optimal value of the makespan and may only increase the value of the makespan generated by LPT. This is depicted in Fig. 6. In what follows,

we seek to show that it can be assumed, without loss of generality, that $w_1 = (k + 1)b$. Consider now the following rescaling procedure.

1. Compute $\varepsilon = ((k + 2)/(k + 1))((k + 1)b - w_1)$. Go to 2. ($\varepsilon$ has to non-negative because job $(k + 2)$ is in progress when job $(k + 3)$ is assigned to machine 2 in the LPT schedule.)

2. If ε is equal to zero, stop. Else go to 3.

3. Compute $\varepsilon_1 = w_{k+4} + w_{k+5} + \cdots + w_{2k+4} - \varepsilon = k + 1 - \varepsilon$. If $\varepsilon_1 \leq 0$, set $w_{k+4} = w_{k+5} = \cdots = w_{2k+4} = 0$ and $w_3 = w_4 = \cdots w_{k+3} = w_2 - 1$ and stop. Else go to 4.

4. Set $w_{k+4} = w_{k+5} = \cdots = w_{2k+4} = 1 - \varepsilon/(k + 1)$ and $w_3 = w_4 = \cdots = w_k+3 = w_2 = b - (1/(k + 2))\varepsilon$. Rescale the length of all jobs till $w_{2k+4} = 1$. Stop.

The above procedure will terminate at step 2 or 3 or 4. First, we need to show that the strict inequality

$$7\bar{v}(2l + 2, W, \text{CAP}) < 6v(2l + 2, W, \text{CAP}) \tag{7}$$

is satisfied after performing the rescaling procedure. This is easily seen if the procedure stops at step 2 with $\varepsilon = 0$ as shown in Fig. 7. Suppose now that the procedure stops at step 3, as indicated in Fig. 8. First note that $\sum_{s=1}^{l} w_{i_s} = (k + 1)\, b - k$. We need to check that for the LPT assignment shown in Fig. 8 we have $\sum_{s=1}^{l} w_{i_s} > w_1$. Now

$$\begin{aligned}(k + 1)\, b - k - w_1 &= \frac{k + 1}{k + 2}\varepsilon - k \\ &> \frac{k + 1}{k + 2}(\varepsilon - k - 1) \\ &\geq 0.\end{aligned}$$

The last inequality follows from the fact that in step 3 $\varepsilon_1 = k + 1 - \varepsilon \leq 0$. The effect of the modifications in Step 3 is to reduce the value of the LPT makespan as well as the optimal makespan by $k + 1$. Therefore, inequality (7) is satisfied.

Assume next that the procedure stops at step 4, and consider the values of the makespan produced before rescaling all jobs in step 4, as shown in Fig. 9. It can be verified that the value of the makespan generated by LPT may be reduced by $((k/(k + 1) + 1/(k + 2))\varepsilon$. But, in the optimal assignment, the sum of the processing times of jobs allocated to machines 1 and 2 will both reduce by ε. Therefore, inequality (6) is satisfied.

Next, we shall show that applying the rescaling procedure will lead to

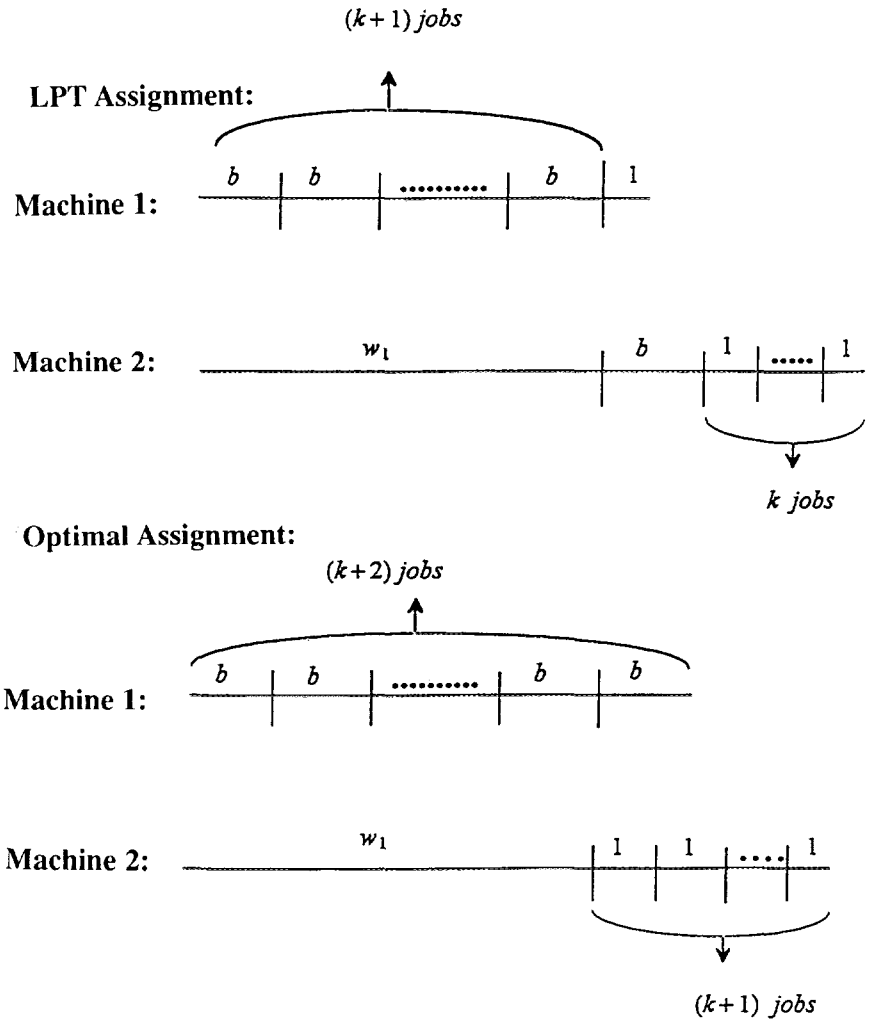

FIG. 7. Assignments of jobs in Case 2.2, $\varepsilon = 0$.

contradictions. In the case where the procedure terminates at step 3, it follows that the capacity constraint did not play a role in the LPT assignment. Therefore, the same argument as in the beginning of Case 1 can be used to derive a contradiction. As for the case where the procedure terminates at either step 2 or step 4, let $w_3 = w_4 = \cdots = w_{k+3} = w_2 = b$. Then $w_1 = (k + 1)b$. Moreover, by virtue of inequality (6), we have that $w_1 + b + k > 7(w_1 + k + 1)/6$, and $w_1 + b + k > 7((k + 2)b)/6$. It follows from the above that

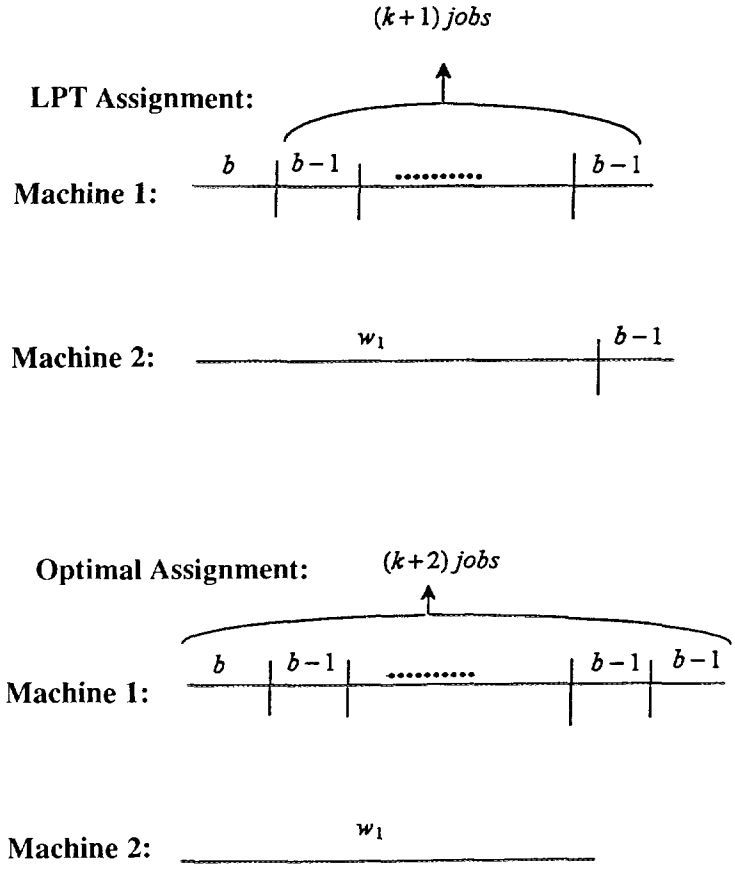

FIG. 8. Assignments of jobs in Case 2.2. Procedure stops at Step 3.

$$(6(k+2) - 7(k+1))b > k + 7$$

and

$$k > \frac{(k+2)b}{6},$$

implying

$$7k^2 - 21k + 14 < 0. \tag{8}$$

Since inequality (7) can never hold for all values of $k = 1, 2, \ldots$, we have a contradiction. This concludes the analysis of Case 2.2, and thus of Case 2.

Since both Cases 1 and 2 lead to contradictions, inequality (1) must hold.

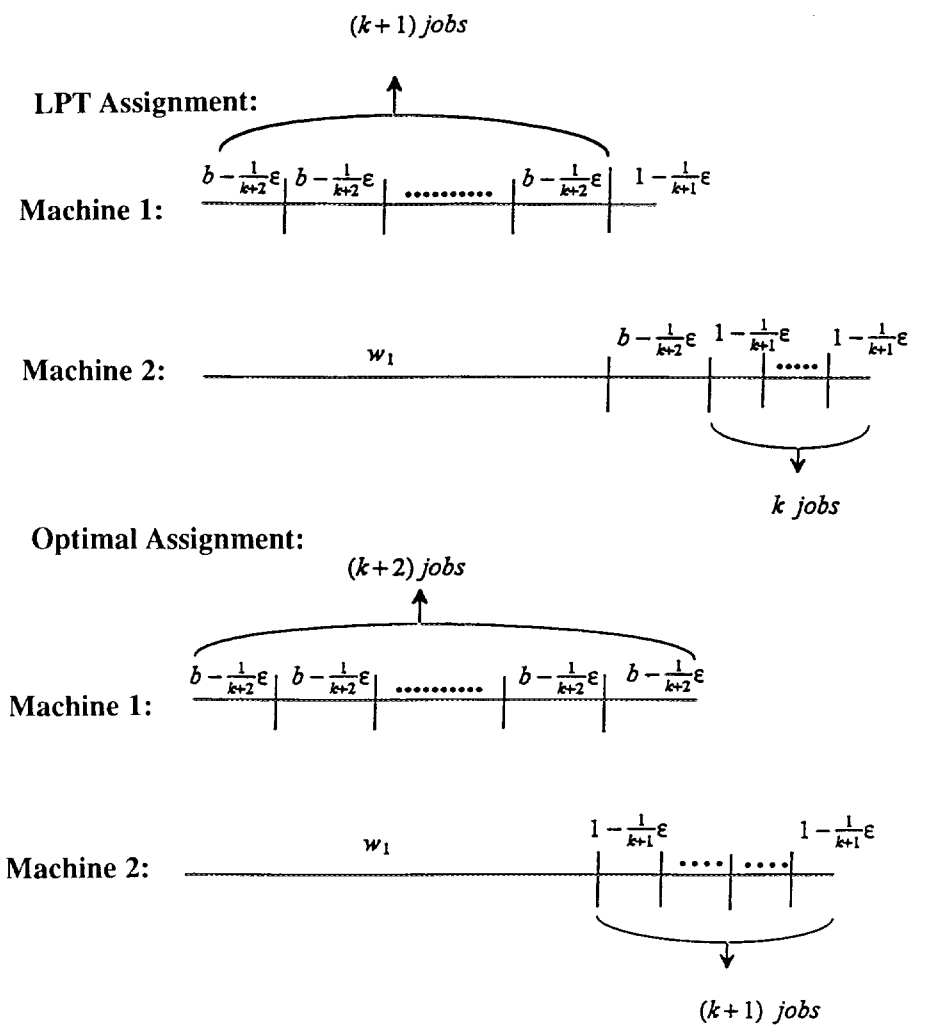

FIG. 9. Assignments of jobs in Case 2.2, Step 4.

ACKNOWLEDGMENTS

The authors thank the Associate Editor, Laurence A. Baxter, and the referee for their many helpful comments.

REFERENCES

1. J. C. AMMONS, C. B. LOFGREN, AND L. F. MCGINNIS, A large scale machine loading problem in flexible assembly, *Ann. Oper. Res.* **3** (1985), 319–322.

2. M. O. Ball and M. J. Magazine, Sequencing of insertions in printed circuit board assembly, *Oper. Res.* **36** (1986), 192–201.
3. L. A. Baxter and F. Harche, On the optimal assembly of series-parallel systems, *Oper. Res. Lett.* **11** (1992), 153–157.
4. L. A. Baxter, F. Harche, and C. A. Tovey, A simple heuristic to minimize makespan on parallel processors with unequal capacities, working paper, Stern School of Business, New York University, New York, 1993.
5. E. G. Coffman, G. S. Lueker, and A. H. G. Rinnooy Kan, Asymptotic methods in the probabilistic analysis of sequencing and packing heuristics, *Management Sci.* **34** (1988), 266–290.
6. R. L. Graham, Bounds for certain multiprocessing anomalies, *Bell System Technol. J.* **45** (1966), 563–581.
7. R. L. Graham, Bounds on multiprocessing timing anomalies, *SIAM J. Appl. Math.* **17** (1969), 263–269.
8. F. Harche, Analysis of a capacitated parallel multi-processor scheduling problem, working paper, Stern School of Business, New York University, New York, 1993.
9. C. B. Lofgren, Machine configuration in flexible assembly systems, *in* "Proceedings of the Materials Handling Focus'85, Atlanta, GA, Sept. 1985."
10. C. B. Lofgren and C. A. Tovey, Performance bounds on machine configuration in flexible manufacturing systems, unpublished manuscript, 1985.
11. R. Loulou, Tight bounds and probabilistic analysis of two heuristics for parallel processor scheduling, *Math. Oper. Res.* **9** (1984), 142–150.
12. K. E. Stecke, Design, planning, scheduling, and control problems of flexible manufacturing systems, *Ann. Oper. Res.* **3** (1985), 51–60.
13. J. A. Tompkins and J. A. White, "Facilities Planning," Wiley, New York, 1984.
14. L. H. Tsai, Asymptotic analysis of an algorithm for balanced parallel processor scheduling, *SIAM J. Comput.* **21** (1992), 59–64.