SIAM J. COMPUT.
Vol. 9, No. 1, February 1980

0097-5397/80/09001-0007 $01.00/0

BOUNDS FOR LIST SCHEDULES ON UNIFORM PROCESSORS*

YOOKUN CHO† AND SARTAJ SAHNI‡

Abstract. Bounds are derived for the worst case performance of list schedules relative to minimum finish time schedules for uniform processor systems. The tasks to be scheduled are assumed to be independent and only nonpreemptive schedules are considered.

Key words. list schedules, nonpreemptive schedules, uniform processors, independent tasks

1. Introduction. A uniform processor system consists of m, $m \geqq 1$, processors $P_1, P_2, \cdots, P_m$. Associated with each processor is a speed s_i, $s_i \geqq 1$. In one unit of time P_i can carry out s_i units of processing. Without loss of generality, we may assume $s_i \leqq s_{i+1}$, $1 \leqq i < m$ and $s_1 = 1$. We are given n independent tasks that are to be processed. Task i requires t_i units of processing (t_i is the *task time* of task i). If task i is assigned to P_j then t_i/s_j time units are needed to finish this task. A *nonpreemptive schedule* is an assignment of tasks to processors such that each task is assigned to exactly one processor. For each processor the order in which tasks are to be processed is also specified. If T_i is the set of tasks assigned to P_i then the *finish time* of P_i is $(\sum_{j \in T_i} t_j)/s_i$. The finish time of the schedule is the time at which all processors $\sum_{j \in T_i} t_j$ have finished processing.

For the case when $s_i = 1$, $1 \leqq i \leqq m$ the processor system defined above is known as a system of *identical processors*. It is well known that finding minimum finish time nonpreemptive schedules for identical processors with $m \geqq 2$ is *NP*-hard (see e.g. [7]). Several heuristics to obtain "near optimal" schedules for identical processors have been studied. Graham [4] has studied the performance of LPT schedules. In an LPT schedule tasks are assigned to processors in nonincreasing order of task times. Whenever a task is to be assigned, it is assigned to that processor on which it will finish earliest. Ties are broken arbitrarily and tasks are processed in the order assigned. Let $\hat{f}$ be the finish time of an LPT schedule for any given task set. Let f^* be the finish time of an optimal schedule. Graham [4] has shown that

$$\hat{f}/f^* \leqq 4/3 - 1/(3m).$$

Another heuristic studied by Graham is the list schedule. This scheduling rule differs from the LPT rule only in the order in which tasks are considered for assignment to processors. A list (or permutation) of the indices $1, 2, \cdots, n$ is provided. Tasks are considered in the order in which they appear on this list. If $\hat{f}$ is the finish time of a list schedule and f^* that of an optimal schedule for any given task set then it is known [3] that:

$$\hat{f}/f^* \leqq 2 - \frac{1}{m}.$$

Hence, for identical processor systems LPT schedules are better than arbitrary list schedules by only a constant factor. Note that an LPT schedule is a special case of a list schedule (i.e., the case when the tasks in the list are ordered in nonincreasing order of task times). Other heuristics for identical processor systems have been studied by Coffman, Garey and Johnson [1] and Sahni [9].

* Received by the editors July 25, 1978. This research was supported in part by the National Science Foundation under Grant MCS 76-21024.

† Department of Computer Science, University of Minnesota, Minneapolis, Minnesota 55455. Presently at Seoul National University, Korea.

‡ Department of Computer Science, University of Minnesota, Minneapolis, Minnesota 55455.

Another special case of a uniform processor system is when $s_i = 1$, $1 \leqq i < m$ and $s_m > 1$. This case has been studied by Liu and Liu [8] and Gonzalez, Ibarra and Sahni [2]. Liu and Liu [8] considered a variation of the LPT rule defined here. They require a task to be assigned to that processor that becomes idle first rather than to the processor on which it will complete first. For this rule they show that

$$\hat{f}/f^* \leqq \begin{cases} 2(m-1+s_m)/(s_m+2) & \text{for } s_m \leqq 2, \\ (m-1+s_m)/2 & \text{for } s_m > 2. \end{cases}$$

Gonzalez, Ibarra and Sahni [2] show that for LPT schedules

$$\hat{f}/f^* \leqq \begin{cases} \dfrac{1+\sqrt{17}}{4}, & m = 2, \\ 3/2 - 1/(2m), & m > 2. \end{cases}$$

Finally, LPT schedules for general uniform processor systems have been analyzed by Gonzalez, Ibarra and Sahni [2]. They show that

$$\hat{f}/f^* \leqq 2m/(m+1).$$

Liu and Liu [8] have analyzed list schedules for the case when $s_i = 1$, $1 \leqq i < m$ and $s_m < 1$. Their analysis assumes that the next task on the list is to be assigned to the first idle processor. Under this assumption they obtain the bound:

$$\hat{f}/f^* \leqq s_m + \frac{m-1}{s_m + m - 1}.$$

A survey of similar results for other machine and task set models appears in [5]. In this paper we analyze arbitrary list schedules for the case of general uniform processor systems and also for the special case when $s_i = 1$, $1 \leqq i < m$ and $s_m > 1$.

In § 2 we show that for the general case

$$\hat{f}/f^* \leqq \begin{cases} (1+\sqrt{5})/2, & m = 2, \\ 1 + (\sqrt{2m-2})/2, & m > 2. \end{cases}$$

For the case of $m \leqq 6$ the bounds given above are tight in the sense that there exist task sets and lists for which $\hat{f}/f^*$ equals the stated bound. For $m > 6$ we are unable to show the bound tight and suspect that it is not tight. We also present an example that shows that $\hat{f}/f^*$ is not bounded by any constant. Hence, while for identical processors LPT schedules are better than list schedules by only a constant factor, for general uniform processor systems the ratio of the finish time of an LPT schedule to that of an arbitrary list schedule is not bounded by any constant (but by some function of m).

In § 3 we consider the special case of $s_i = 1$, $1 \leqq 1 < m$ and $s_m > 1$. For this case we show that

$$\hat{f}/f^* \leqq \begin{cases} (1+\sqrt{5})/2, & m = 2, \\ 3 - 4/(m+1), & m \geqq 3. \end{cases}$$

Furthermore, the bound is tight.

2. General uniform processor systems. In this section the following theorem is established:

THEOREM 1. *Let t_i, $1 \leqq i \leqq n$ be the task times of n independent tasks. Let s_i, $1 \leqq i \leqq m$ be the speeds of the m processors in the system. Let $\hat{f}$ be the finish time of the schedule obtained using any given list L and let f^* be the finish time of an optimal schedule for the*

task set. Then,

$$\hat{f}/f^* \begin{cases} (1+\sqrt{5})/2, & m=2, \\ 1+(\sqrt{2m-2})/2, & m \geqq 3. \end{cases}$$

Furthermore, for $m \leqq 6$ there exists task sets, lists and processor systems for which $\hat{f}/f^*$ equals the above bound.

Theorem 1 will be proved in several steps. First, we derive some relationships between $\hat{f}/f^*$ and the s_i's and t_i's. In the following it will be assumed that $s_1 \leqq s_2 \leqq \cdots \leqq s_m$, $s_1 = 1$ and $f^* = 1$. We shall also assume that the tasks have been indexed so that the list is given by $L = (1, 2, 3, \cdots, n)$. Note that these assumptions do not affect the generality of the proofs. Any problem instance that violates one or more of these assumptions may be transformed into an equivalent problem instance satisfying all the assumptions by sorting the s_i's, reordering the tasks and appropriately scaling the s_i's and t_i's. We shall also assume that in case of a tie, the list scheduling algorithm assigns the task to the processor with highest index. Since our proof will be a proof by contradiction, it is necessary to develop relationships only for the smallest n (for any given m) for which the theorem may be violated. Thus, if $t_1, \cdots, t_n$ defines a task set with least n for which the theorem does not hold then it is easy to see that the finish time of the list schedule is determined by task n. To see this, observe that if $i < n$ determines the finish time then we can eliminate tasks $i+1, \cdots, n$ and consider only tasks $1, 2, \cdots, i$. For this set, $\hat{f}$ is unchanged and f^* is not increased. So, $\hat{f}/f^*$ does not decrease and we have a smaller instance violating the bounds of the theorem. Now, since task n finishes at $\hat{f}$, we can imagine the list schedule to look like Fig. 1. Here j is any index in the range $[1, m]$ and $\hat{f} = F_j + t_n/s_j$. Let F_i be the finish time of P_i, $1 \leqq i \leqq m$, before task n is scheduled.

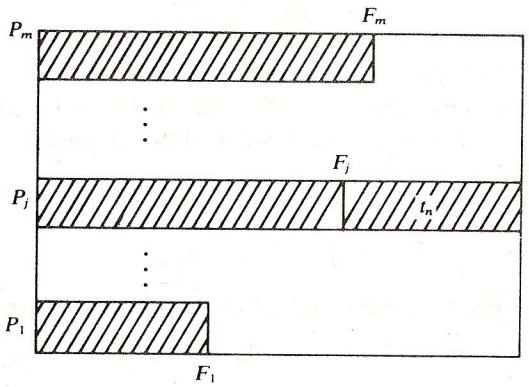

FIGURE 1

Since we are assuming $f^* = 1$, it follows that

$$\sum_1^n t_i \leqq \sum_1^m s_i, \tag{1}$$

$$t_n \leqq \max\{t_i\} \leqq s_m. \tag{2}$$

From the definition of list schedules, it follows that

$$F_i + t_n/s_i \geqq \hat{f}, \qquad 1 \leqq i \leqq m, \tag{3}$$

or $s_iF_i + t_n \geqq s_i\hat{f}$, $1 \leqq i \leqq m$. Using (3) with $i = m$ we obtain

$$s_m\hat{f} \leqq s_mF_m + t_n \leqq \sum_1^n t_i \leqq \sum_1^m s_i. \tag{4}$$

Hence, $\hat{f} \leqq \sum_i^m s_i/s_m$.

From (3), we obtain

$$s_i(\hat{f} - F_i) \leqq t_n, \qquad 1 \leqq i \leqq m, \tag{5}$$

Summing (5) for $1 \leqq i \leqq m$ and using (1), we get

$$\sum_1^m s_i\hat{f} - \sum_1^m s_iF_i \leqq (m-1)t_n + t_n$$

or

$$\begin{aligned}\sum_1^m s_i\hat{f} &\leqq (m-1)t_n + \sum_1^m s_iF_i + t_n \\ &\leqq (m-1)t_n + \sum_1^n t_i \\ &\leqq (m-1)t_n + \sum_1^m s_i.\end{aligned}$$

Hence,

$$\hat{f} \leqq 1 + (m-1)t_n \Big/ \sum_1^m s_i \tag{6a}$$

$$\leqq 1 + (m-1)s_m \Big/ \sum_1^m s_i. \tag{6b}$$

LEMMA 1. $\hat{f}/f^* \leqq (1+\sqrt{4m-3})/2$, $m \geqq 2$.

Proof. Assume the lemma is false. Consider the smallest n for which it is false. From the preceding discussion we may assume $f^* = 1$ and that the list is $(1, 2, \cdots, n)$. Hence equations (1)–(6) hold.

Using x to represent the ratio $\sum_1^m s_i/s_m$ we obtain (7) from (4) and (6b).

$$\hat{f} \leqq \min\{x, 1 + (m-1)/x\}. \tag{7}$$

The maximum value of the right hand side of (7) is obtained when

$$x = 1 + (m-1)/x \quad \text{or} \quad x^2 - x - (m-1) = 0 \quad \text{or} \quad x = (1+\sqrt{4m-3})/2.$$

Hence, $\hat{f} \leqq (1+\sqrt{4m-3})/2$. So, the lemma must be true for all m, all task sets and all lists. □

When $m = 2$, $(1+\sqrt{4m-3})/2 = (1+\sqrt{5})/2$. This observation together with the following example proves Theorem 1 when $m = 2$.

Example 1. Let $s_1 = 1$, $s_2 = (1+\sqrt{5})/2$, $t_1 = 1$ and $t_2 = (1+\sqrt{5})/2$. (Note that by assumption $L = (1, 2)$.) It is clear that $f^* = 1$. In the list schedule however, both tasks 1 and 2 get assigned to P_2 and

$$\hat{f} = (t_1 + t_2)/s_2 = (1+\sqrt{5})/2.$$

When $m = 3$, $(1+\sqrt{4m-3})/2 = 1 + (\sqrt{2m-2})/2 = 2$. This together with Example 2 establishes Theorem 1 for $m = 3$.

Example 2. Consider, $s_1 = s_2 = 1$, $s_3 = 2$, $t_1 = t_2 = 1$ and $t_3 = 2$. Again, $f^* = 1$. In the list schedule all three tasks get assigned to P_3. Hence, $\hat{f} = 2$. Note that if a different tie breaking rule is used then we may replace s_3 and t_3 by $2+\varepsilon$. In this case $\hat{f} = (4+\varepsilon)/(2+\varepsilon)$ whuch approaches 2 as $\varepsilon \to 0$.

The bound of Lemma 1 is not tight for $m > 3$. This is established by obtaining a smaller bound. First, we derive some more inequalities. We readily observe that in every list schedule, task 1 is always assigned to P_m. Since we may assume $n > 1$, it follows that P_m always has at least one task (other than task n) assigned to it. Consider the status of the list schedule just before task n is assigned to a processor. Let t' be the task time of the last task assigned to P_m (note that this task cannot be task n as it has not yet been assigned). Let G_i be the finish time of P_i, $1 \leqq i \leqq m$ just before t' was assigned to P_m. It follows that $G_i \leqq F_i$, $1 \leqq i \leqq m$. Since t' was assigned to P_m, if follows that:

$$F_i + t'/s_i \geqq G_i + t'/s_i \geqq G_m + t'/s_m = F_m, \qquad 1 \leqq i \leqq m. \tag{8}$$

Using t to denote t_n and substituting $i = m$ in (3) we obtain

$$s_m F_m + t \geqq s_m \hat{f} \quad \text{or} \quad F_m \geqq \hat{f} - t/s_m.$$

Substituting into (8) we get

$$F_i + t'/s_i \geqq \hat{f} - t/s_m \quad \text{or} \quad s_i F_i + t' \geqq s_i \hat{f} - t s_i/s_m, \qquad 1 \leqq i \leqq m. \tag{9}$$

Further, it follows from $f^* = 1$ that

$$t' + t \leqq s_{m-1} + s_m. \tag{10}$$

LEMMA 2. $\hat{f}/f^* \leqq 1 + (\sqrt{2m-2})/2$, $m \geqq 4$.

Proof. Suppose the lemma is not true. Consider the least n for which

$$\hat{f}/f^* > 1 + (\sqrt{2m-2})/2. \tag{11}$$

We may assume $f^* = 1$ and that the list is $(1, 2, \cdots, n)$. Let t' and t be as defined before. We first recall the following inequality:

$$\sum_1^m s_i F_i + t = \sum_1^n t_i \leqq \sum_1^m s_i. \tag{12}$$

From (3) with $i = m$ and $m-1$ we get

$$s_{m-1} F_{m-1} + s_m F_m + 2t \geqq \hat{f}(s_{m-1} + s_m).$$

So,

$$\sum_1^m s_i F_i + 2t \geqq \hat{f}(s_{m-1} + s_m).$$

This together with (2) and (12) yields

$$\sum_1^m s_i + s_m \geqq \hat{f}(s_{m-1} + s_m) \quad \text{or} \quad \hat{f} \leqq 1 + \frac{\sum_1^m s_i - s_{m-1}}{s_{m-1} + s_m}. \tag{13}$$

Equations (11) and (13) together yield:

$$1 + \frac{\sqrt{2m-2}}{2} < 1 + \frac{\sum_1^m s_i - s_{m-1}}{s_{m-1} + s_m}$$

or

$$(s_{m-1}+s_m)<\frac{2}{\sqrt{2m-2}}\left(\sum_1^m s_i - s_{m-1}\right). \tag{14}$$

Summing up $m-1$ inequalities from (3) (i.e. $1<i\leqq m$) and inequality (9) with $i=1$, we get

$$\sum_1^m s_iF_i+(m-1)t+t'\geqq\sum_1^m s_i\hat{f}-s_1t/s_m.$$

Substituting (12) into the above equation, we get

$$\sum_1^m s_i+(m-2)t+t'\geqq\sum_1^m s_i\hat{f}-s_1t/s_m.$$

From this and (2) we get

$$\sum_1^m s_i+(m-2)t+t'+s_1\geqq\sum_1^m s_i\hat{f}. \tag{15}$$

Also, from (6b) and (11) we get

$$\frac{\sqrt{2m-2}}{2}<(m-1)s_m\Big/\sum_1^m s_i. \tag{16}$$

The next step is to show that if (11) holds then $t>s_{m-1}$. Suppose $t\leqq s_{m-1}$ then from (15) and the knowledge $t'\leqq\max\{t_i\}\leqq s_m$ we get

$$\sum_1^m s_i+(m-2)s_{m-1}+s_m+s_1\geqq\sum_1^m s_i\hat{f}.$$

Rearranging terms, we get

$$\hat{f}\leqq 2+(m-3)s_{m-1}\Big/\sum_1^m s_i. \tag{17}$$

This together with (11) gives

$$1+(\sqrt{2m-2})/2<2+(m-3)s_{m-1}\Big/\sum_1^m s_i$$

or

$$\frac{\sqrt{2m-2}-2}{2(m-3)}<\frac{s_{m-1}}{\sum_1^m s_i}. \tag{18}$$

Adding $1/(m-1)$ times (16) to (18) we get

$$\frac{\sqrt{2m-2}-2}{2(m-3)}+\frac{\sqrt{2m-2}}{2(m-1)}<\frac{s_{m-1}+s_m}{\sum_1^m s_i}. \tag{19}$$

Combining (19) and (14) we get

$$\frac{\sqrt{2m-2}-2}{2(m-3)}+\frac{\sqrt{2m-2}}{2(m-1)}<\frac{2}{\sqrt{2m-2}}\left(1-\frac{s_{m-1}}{\sum s_i}\right).$$

Simplifying, we get

$$\frac{s_{m-1}}{\sum_1^m s_i} < \frac{\sqrt{2m-2}-2}{2(m-3)}.$$

This contradicts (18). So, $t > s_{m-1}$. Hence, task n is scheduled on P_m in f^*. If t' is also on P_m then $t' \leqq s_m - t$. Otherwise, $t' \leqq s_{m-1}$. Hence,

$$t' \leqq \max\{s_{m-1}, s_m - t\}. \tag{20}$$

We shall now show that no matter which of s_{m-1} and $s_m - t$ is maximum we arrive at contradicting relations. So, $\hat{f} \leqq 1 + (\sqrt{2m-2})/2$.

Summing (3) with $i = m$ and $m-1$ and (9) with $1 \leqq i \leqq m-2$, we obtain

$$\sum_1^m s_i F_i + 2t + (m-2)t' \geqq \hat{f} \sum_1^m s_i - \sum_1^{m-2} s_i t / s_m.$$

Substituting from (2), (10), and (12) we reduce this to

$$\sum_1^m s_i + s_{m-1} + s_m + (m-3)t' + \sum_1^{m-2} s_i \geqq \hat{f} \sum_1^m s_i$$

or

$$\hat{f} \leqq 2 + (m-3)t' \Big/ \sum_1^m s_i. \tag{21}$$

First, let us consider the case $s_{m-1} \geqq s_m - t$. (20) yields $t' \leqq s_{m-1}$. Substituting into (21) we get

$$\hat{f} \leqq 2 + (m-3)s_{m-1} \Big/ \sum_1^m s_i.$$

This is the same as (17) and together with (11) and (16) can be used to derive (19) and arrive at a contradiction as before.

If $s_{m-1} < s_m - t$ then $t' \leqq s_m - t$ and $t < s_m - s_{m-1}$. Substituting into (21) we get

$$\hat{f} \leqq 2 + (m-3)(s_m - t) \Big/ \sum_1^m s_i. \tag{22}$$

Adding (22) and $(m-3)/(m-1)$ times (6a) we get

$$\frac{2m-4}{m-1}\hat{f} \leqq 2 + \frac{m-3}{m-1} + \frac{(m-3)s_m}{\sum_1^m s_i}.$$

Substituting for $\hat{f}$ from (11) we get

$$\frac{2m-4}{m-1}\left(1 + \frac{\sqrt{2m-2}}{2}\right) < 2 + \frac{m-3}{m-1} + \frac{(m-3)s_m}{\sum_1^m s_i}$$

or

$$\frac{2m-4}{m-1} + \frac{(2m-4)\sqrt{2m-2}}{2(m-1)} - 2 - \frac{m-3}{m-1} < \frac{(m-3)s_m}{\sum_1^m s_i}$$

or

$$\frac{(m-2)\sqrt{2m-2}}{m-1} - 1 < \frac{(m-3)s_m}{\sum_1^m s_i}$$

or

$$\frac{(m-2)\sqrt{2m-2}-(m-1)}{(m-1)(m-3)} < \frac{s_m}{\sum_1^m s_i}.$$

Combining with (4) we get

$$\hat{f} < \frac{(m-1)(m-3)}{(m-2)\sqrt{2m-2}-(m-1)}. \tag{23}$$

One may easily verify that the right hand side of (23) is no more than $1+(\sqrt{2m-2})/2$ for $m \geq 4$. This contradicts (11) and establishes the lemma. □

To complete the proof of Theorem 1 we need to show that the bound is tight for $m = 4$, 5 and 6. The next three examples do this.

Example 3. $m = 4$, $s_1 = s_2 = 1$, $s_3 = \sqrt{6}/2$, $s_4 = s_3 + 1$. $t_1 = t_2 = .5$, $t_3 = 1$, $t_4 = \sqrt{6}/2$ and $t_5 = 1 + \sqrt{6}/2$. Clearly, $f^* = 1$.
Figure 2(a) shows the list schedule. $\hat{f} = 1 + \sqrt{6}/2 = 1 + (\sqrt{2m-2})/2$.

Example 4. $m = 5$, $s_1 = s_2 = s_3 = 1$, $s_4 = \sqrt{2}$, $s_5 = 1 + \sqrt{2}$. $t_1 = t_2 = t_3 = 1$, $t_4 = \sqrt{2}$, $t_5 = 1 + \sqrt{2}$. $f^* = 1$. The list schedule is shown in Figure 2(b) and $\hat{f} = 1 + \sqrt{2} = 1 + (\sqrt{2m-2})/2$.

Example 5. $m = 6$, $s_1 = s_2 = s_3 = s_4 = 1$, $s_5 = \sqrt{10}/2$, $s_6 = 1 + \sqrt{10}/2$. $t_1 = t_2 = .5$, $t_3 = t_4 = t_5 = 1$, $t_6 = \sqrt{10}/2$ and $t_7 = 1 + (\sqrt{10})/2$. Again, $f^* = 1$, $\hat{f} = 1 + \sqrt{10}/2 = 1 + (\sqrt{2m-2})/2$ (see Fig. 2(c)).

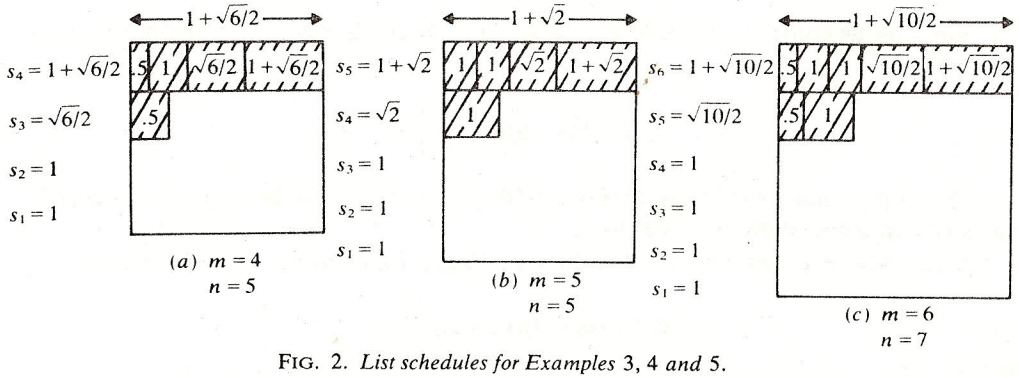

FIG. 2. *List schedules for Examples* 3, 4 *and* 5.

The question that naturally arises at this time is: What happens when $m > 6$? Is it possible for $\hat{f}/f^*$ to get as large as the bound given in Theorem 1? We have been unable to generate examples achieving this bound for $m > 6$. The worst examples we were able to generate for $m = 7$ and 8 are given in Examples 6 and 7. These were arrived at by considering a certain job distribution pattern, deriving inequalities that led to a cubic equation. A root of this equation yielded s_m and the remaining numbers were obtained by back substitution.

Example 6. $m = 7$. Let r be a real root of the cubic equation $4r^3 - 11r^2 + r - 1 = 0$. $s_1 = s_2 = s_3 = 1$, $s_4 = r + 1/r - 2$, $s_5 = r - r/(r-1)$, $s_6 = r - 1$, $s_7 = r$. $t_1 = r^2 - 3r + 2 - 1/r$, $t_2 = r^2 - 3r + r/(r-1)$, $t_3 = 1 - t_1 - t_4$, $t_4 = r^2 - 3r + 1$, $t_5 = 1 - t_2$, $t_6 = 1$, $t_7 = r + 1/r - 2$, $t_8 = r - r/(r-1)$, $t_9 = r - 1$, $t_{10} = r$. For an optimal schedule, the list is (10, 9, 8, 7, 6, 5, 4, 3, 1, 2) Clearly $f^* = 1$. An approximate value for r is 2.691. This yields $s_4 = 1.063$, $s_5 = 1.100$, $s_6 = 1.691$, $s_7 = 2.691$, $t_1 = .797$, $t_2 = .760$, $t_3 = .033$, $t_4 = .170$, $t_5 = .240$, $t_6 = 1$, $t_7 = 1.063$, $t_8 = 1.100$, $t_9 = 1.691$ and $t_{10} = 2.691$. For the list schedule consider

the list $(1, 2, \cdots, 10)$. The resulting schedule is shown in Fig. 3(a) and $\hat{f} = 2.69\ldots$. The bound of Theorem 1 when $m = 7$ is 2.732.

Example 7. $m = 8$. Let r be a real root of the cubic equation $4r^3 - 11r^2 - 1 = 0$. $s_1 = s_2 = s_3 = s_4 = 1$, $s_5 = r + 1/r - 2$, $s_6 = r - r/(r-1)$, $s_7 = r - 1$, $s_8 = r$. $t_1 = r^2 - 3r + 1 - 1/r$, $t_2 = 1$, $t_3 = r^2 - 3r + r/(r-1)$, $t_4 = r^2 - 2r - r/(r-1)$, $t_5 = 1 - t_3$, $t_6 = 1 - t_4 - t_1$, $t_7 = 1$ $t_8 = r + 1/r - 2$, $t_9 = r - r/(r-1)$, $t_{10} = r - 1$, $t_{11} = r$. For an optimal schedule, the list is (11, 10, 9, 8, 7, 6, 5, 1, 4, 3, 2). Again $f^* = 1$. An approximate value for r is 2.782. Using this, we get $s_5 = 1.141$, $s_6 = 1.221$, $s_7 = 1.782$, $s_8 = 2.782$, $t_1 = .035$, $t_2 = 1$, $t_3 = .955$, $t_4 = .615$, $t_5 = .045$, $t_6 = .350$, $t_7 = 1$, $t_8 = 1.141$, $t_9 = 1.221$, $t_{10} = 1.782$, $t_{11} = 2.782$. Assume a list schedule is constructed using the list $(1, 2, \cdots, 11)$. The resulting schedule is shown in Figure 3(b) and $\hat{f} = 2.782\ldots$. The bound of Theorem 1 is 2.87.

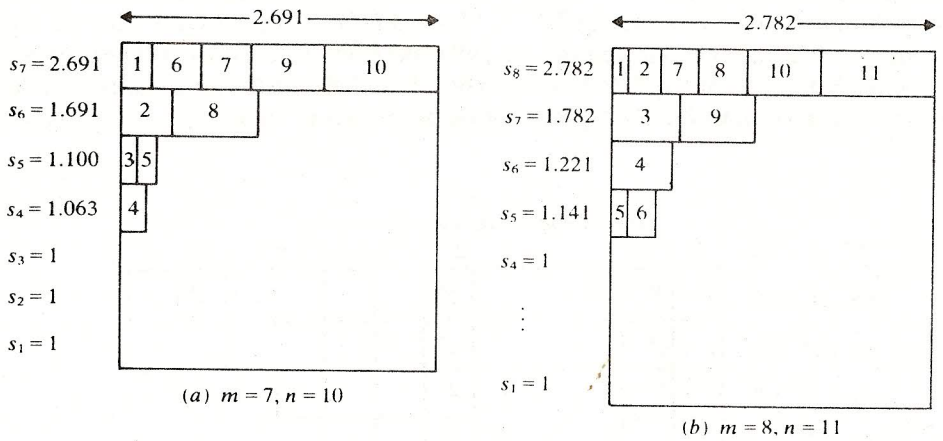

FIG. 3. *List schedules for Examples* 6 *and* 7.

While we have been unable to establish a tight bound for $m > 6$, we can show that the bound on $\hat{f}/f^*$ must increase as m increases. Hence, there is no constant k such that $\hat{f}/f^* \leqq k$ for all m. This result should be contrasted with the bound for identical processors which is itself bounded by 2.

THEOREM 2. *There exist task sets, uniform processor systems, and lists for which* $\hat{f}/f^* \geqq \lfloor (\log_2 (3m-1)+1)/2 \rfloor$.

Proof. Let $k = (\log_2 (3m-1)+1)/2$. We shall construct an example that achieves the above bound when m is such that k has integer values (i.e. when $m = 3, 11, 43, \cdots$). There are k sets of processors G_i, $1 \leqq i \leqq k$. Each processor in G_i has a speed of 2^i. $|G_i| = 2^{2k-2i-1}$, $1 \leqq i < k$ and $|G_k| = 1$. Thus, the total number of processors is $m = 1 + 2^1 + 2^3 \cdots + 2^{2k-3} = 2(4^{k-1} - 1)/3 + 1$.

We shall have k sets of tasks T_i, $1 \leqq i \leqq k$. The task time of a task in T_i, is 2^i, $1 \leqq i \leqq k$. Also, $|T_i| = 2^{2k-2i-1}$, $1 \leqq i < k$ and $|T_k| = 1$.

It is easily verified that $f^* = 1$. Consider the list in which all tasks appear as follows: T_1 tasks followed by T_2 tasks followed by T_3 tasks etc. The resulting list schedule is given in Fig. 4. The schedule consists of k columns. Each column contains tasks with identical task times. In column i, $1 \leqq i < k$ the processors in G_j, $i+1 \leqq j \leqq k$ will be processing tasks with a task time of 2^i. The number of tasks on a processor in G_j, $i+1 \leqq j \leqq k$ is 2^{j-i}. Thus, the total number of tasks scheduled on all processors in G_j will be $2^{j-i} * |G_j| = 2^{j-i} * 2^{2k-2j-1} = 2^{2k-j-i-1}$ for $i+1 \leqq j < k$ and 2^{k-i} for G_k. Therefore

the total number of tasks in column i is

$$\sum_{j=i+1}^{k-1} 2^{2k-j-i-1}+2^{k-i}=\sum_{j=1}^{k-i-1} 2^{2k-2i-1-j}+2^{k-i}$$

$$=2^{2k-2i-1}\sum_{j=1}^{k-i-1} 1/2^j+2^{k-i}$$

$$=2^{2k-2i-1}(1/2((1/2)^{k-i-1}-1)/(1/2-1))+2^{k-i}$$

$$=2^{2k-2i-1}(1-(1/2)^{k-i-1})+2^{k-1}$$

$$=2^{2k-2i-1}-2^{k-i}+2^{k-i}=2^{2k-2i-1}.$$

In column i, we need $2^{2k-2i-1}$ tasks with task time 2^i. We observe that the number of tasks with task time 2^i is exactly $2^{2k-2i-1}$ for $1 \leqq i < k$. Thus, all tasks with the same task time are scheduled in one column, i.e., all tasks from T_i, $1 \leqq i \leqq k$ will be processed in column i. Note that exactly k columns can be scheduled with the given tasks. It is clear that any task in column i, $2 \leqq i \leqq k$ cannot be processed in column j, $1 \leqq j \leqq i$. Hence, $\hat{f}=k$. □

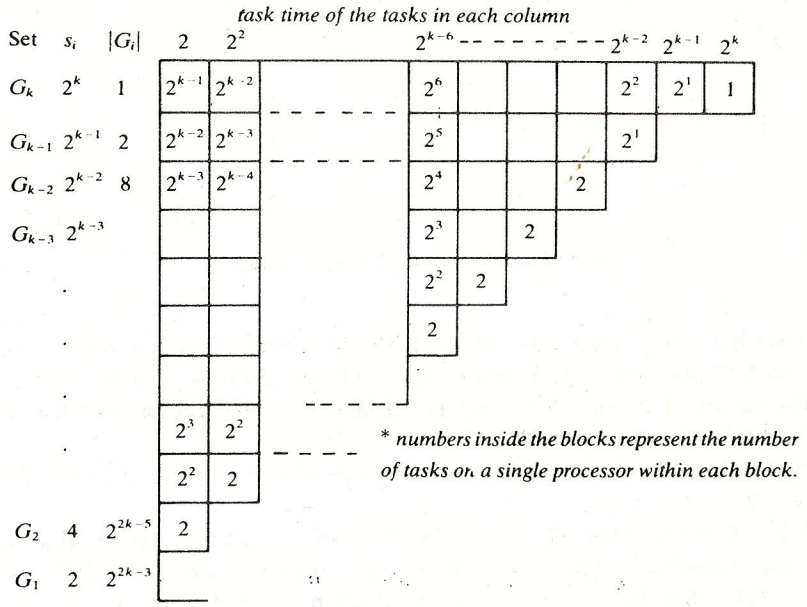

FIG. 4. *List schedule for Theorem 2.*

3. The case $s_i=1$, $1 \leqq i < m$ and $s_m > 1$. In this section the following theorem is established:

THEOREM 3. *If $s_i=1$, $1 \leqq i < m$ and $s_m > 1$ then*

$$\hat{f}/f^* \leqq \begin{cases} (1+\sqrt{5})/2, & m=2, \\ 3-4/(m+1), & m \geqq 3. \end{cases}$$

For each m, $m \geqq 2$ there exists task sets, lists and s_m for which $\hat{f}/f^*$ equals the above bound.

For $m = 2$ and 3 the bounds for Theorems 1 and 3 are the same. Moreover, the examples given in the last section for $m = 2$ and 3 can be used here too. So, the theorem needs to be proved only for $m > 3$.

For $m > 3$, consider any task set, m, s_m and list. Let $s = s_m$. Let $\hat{f}$ be the finish time of the corresponding list schedule and f^* the optimal finish time. We shall show that the following inequality holds.

$$\hat{f}/f^* \leqq 1 + \frac{(m-1)}{s+m-1} \min\{s, 2\}. \tag{24}$$

Since $s/(s+m-1)$ is an increasing function of s, it follows that the right hand side of (24) is maximized when $s = 2$. Hence, (24) reduces to

$$\hat{f}/f^* \leqq 1 + \frac{2(m-1)}{m+1}$$

$$= 3 - 4/(m+1).$$

We prove (24) by considering two cases. Let F_i be the finish time of P_i in the list schedule.

Case 1. $[\hat{f} = F_m]$. Let $F = \min_{i \neq m}\{F_i\}$. Clearly, the following inequality must hold:

$$(m-1)F + s\hat{f} \leqq \sum_1^n t_i$$

or

$$F \leqq \left(\sum_1^n t_i - s\hat{f}\right) \Big/ (m-1). \tag{25}$$

Let t be the task time of the last task assigned to P_m. We may assume this is the last task in the list. If not, we can dispense with the remaining tasks and not reduce $\hat{f}/f^*$. We observe that $F + t \geqq \hat{f}$ and $sf^* \geqq t$. Using these and (25) we get:

$$sf^* \geqq \hat{f} - F \geqq \hat{f} - (\textstyle\sum t_i - s\hat{f})/(m-1).$$

The preceding inequality together with the inequality $\sum t_i \leqq (s+m-1)f^*$ yields:

$$\hat{f}/f^* \leqq 1 + s(m-1)/(s+m-1). \tag{26}$$

Now, let t' be the length of the last task assigned to P_m in the list schedule and which was not assigned to P_m in the optimal schedule. Clearly such a task exists as otherwise $\hat{f} = f^*$. Let $SUC(t')$ be the sum of the task lengths of the tasks assigned to P_m after t' in the list schedule. Clearly, $SUC(t') \leqq sf^*$ and $t' \leqq f^*$.
Also, $F + t' \geqq \hat{f} - SUC(t')/s$. Hence,

$$s(F + t') \geqq s\hat{f} - SUC(t')$$

$$\geqq s\hat{f} - sf^*$$

or

$$F + f^* \geqq \hat{f} - f^* \quad \text{or} \quad F \geqq \hat{f} - 2f^*$$

From this and (25) we obtain

$$\sum_1^n t_i \geqq (\hat{f} - 2f^*)(m-1) + s\hat{f}. \tag{27}$$

Since $\sum t_i \leqq (s+m-1)f^*$, (27) results in (28)

$$\hat{f}/f^* \leqq 1+\frac{2(m-1)}{s+m-1}. \tag{28}$$

Combining (26) and (28) we get (24).

Case 2. [$\hat{f} \neq F_m$]. In this case, $\hat{f} > F_m$. Without loss of generality, we may assume $\hat{f} = F_{m-1}$. Let t be the length of the last task assigned by P_{m-1}. As before, we can assume that this is the last task to be assigned to any processor. Note that no F_i, $i \neq m$ may be less than $\hat{f}-t$. If any F_i, $i < m-1$ is greater than $\hat{f}-t$ then we can decrease the processing requirements of the last few tasks assigned to P_i so that F_i is now equal to $\hat{f}-t$. This decrease will not change the list schedule but may decrease f^*. So $\hat{f}/f^*$ does not decrease and we will only be considering a worse case. Hence, we may assume $F = \hat{f}-t = F_i$, $1 \leqq i \leqq m-2$.

It is easy to see that $sF_m + \hat{f} + (m-2)F = \sum_1^n t_i \leqq (s+m-1)f^*$. Since $F+t = \hat{f}$ and $sF_m + t > s\hat{f}$, it follows that $sF_m + t + (m-2)(F+t) > (s+m-2)\hat{f}$. But, $sF_m + \hat{f} + (m-2)F = \sum t_i \leqq (s+m-1)f^*$. So, $(s+m-1)f^* - \hat{f} + (m-1)t > (s+m-2)\hat{f}$. This together with the equation $t \leqq sf^*$ yields

$$\hat{f}/f^* < 1+\frac{s(m-1)}{s+m-1}. \tag{29}$$

Let t' be the length of the last task assigned to P_m in the list schedule and which is not assigned to P_m in f^*. If such a t' does not exist then $F_m \leqq f^*$. If the task with length t was on P_m in f^* then $\hat{f} \leqq F_m + t/s \leqq f^*$. Otherwise, $t \leqq f^*$ and $\hat{f} < F_m + t/s < 2f^*$. In either case $\hat{f}/f^* < 2$ and the theorem holds. So, we may assume t' exists. Let $SUC(t')$ be as defined in Case 1. Now, $F + t' \geqq \hat{f} - SUC(t')/s$. Since, $SUC(t') \leqq sf^*$, the previous inequality becomes $sF + st' \geqq s\hat{f} - sf^*$ or $F + t' \geqq \hat{f} - f^*$ or $F \geqq \hat{f} - 2f^*$.

Substituting $sF_m + t > \hat{f}$ and $\hat{f} - t = F$ into $sF_m + \hat{f} + (m-2)F \leqq (s+m-1)f^*$, we get

$$(m-1)F + s\hat{f} \leqq (s+m-1)f^*. \tag{30}$$

Using $F \geqq \hat{f} - 2f^*$ in (30) yields

$$\hat{f}/f^* \leqq 1+\frac{2(m-1)}{s+m-1}. \tag{31}$$

(29) and (31) yield (24). This completes the proof for the upper bound of Theorem 3. The next example shows that the bound is tight.

Example 8. For any fixed m, $m \geqq 3$ let $n = (m-3)(m+1)+3$ and $s_m = 2$, $s_i = 1$, $1 \leqq i < m$. The n task times are $t_i = 1/(m+1)$, $1 \leqq i \leqq n-3$, $t_{n-2} = t_{n-1} = 1$ and $t_n = 2$. By

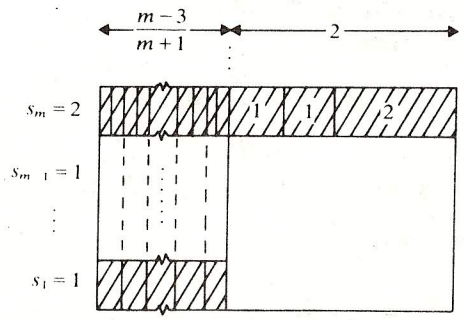

FIG. 5. *List schedule for Example* 8.

assigning task n to P_m, tasks $n-1$ and $n-2$ to P_{m-1} and P_{m-2} respectively and assigning $(m+1)$ of the remaining tasks to each of the remaining processors we get a schedule with finish time 1. It is easy to see that this is optimal and so $f^* = 1$.

If the list for scheduling is $(1, 2, \cdots, n)$ then the resulting list schedule is as in Fig. 5. For every two tasks with index $\leqq n-3$ assigned to P_m, one task with index $\leqq n-3$ is assigned to each of the remaining processors. The finish time $\hat{f}$ is $(m-3)/(m+1)+4/2 = 3-4/(m+1)$.

REFERENCES

[1] E. G. COFFMAN, JR., M. R. GAREY AND D. S. JOHNSON, *An application of bin-packing to multiprocessor scheduling*, this Journal, 7 (1978), pp. 1–17.

[2] T. GONZALEZ, O. H. IBARRA AND S. SAHNI, *Bounds for LPT schedules on uniform processors*, this Journal, 6 (1977), pp. 155–166.

[3] R. L. GRAHAM, *Bounds for certain multiprocessing anomalies*, Bell System Tech. J., 45 (1966), pp. 1563–1581.

[4] ——— *Bounds on multiprocessing timing anomalies*, SIAM J. Appl. Math., 17 (1969), pp. 263–269.

[5] R. L. GRAHAM, E. L. LAWLER, J. K. LENSTRA AND A. H. G. RINNOOY KAN, *Optimization and approximation in deterministic sequencing and scheduling: A Survey*, Mathematisch Centrum, Amsterdam, BW 82/77, 1977.

[6] E. HOROWITZ AND S. SAHNI, *Exact and Approximate Algorithms for Scheduling Non-Identical Processors*, J. Assoc. Comput. Mach., (1976), pp. 317–327.

[7] ——— *Fundamentals of Computer Algorithms*, Computer Science Press, Maryland, 1978.

[8] J. W. S. LIU AND C. L. LIU, *Bounds on Scheduling Algorithms for Heterogeneous Computing Systems*, Proc. IFIP, (1974), pp. 349–353.

[9] S. SAHNI, *Algorithms for scheduling independent tasks*, J. Assoc. Comput. Mach., 23 (1976), pp. 116–127.