

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 , $m \geq 1$, processors P_1, P_2, \dots, P_m . Associated with each processor is a speed s_i , $s_i \geq 1$. In one unit of time P_i can carry out s_i units of processing. Without loss of generality, we may assume $s_i \leq s_{i+1}$, $1 \leq i < m$ 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 $s_i = 1$, $1 \leq i \leq m$ 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 $m \geq 2$ 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

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

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 $1, 2, \dots, n$ 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:

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

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 $s_i = 1$, $1 \leq i < m$ 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

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

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

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

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

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

Liu and Liu [8] have analyzed list schedules for the case when $s_i = 1$, $1 \leq i < m$ 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:

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

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 $s_i = 1$, $1 \leq i < m$ and $s_m > 1$.

In § 2 we show that for the general case

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

For the case of $m \leq 6$ 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 $s_i = 1$, $1 \leq i < m$ and $s_m > 1$. For this case we show that

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

Furthermore, the bound is tight.

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

THEOREM 1. *Let t_i , $1 \leq i \leq n$ be the task times of n independent tasks. Let s_i , $1 \leq i \leq m$ 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,

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

Furthermore, for $m \leq 6$ 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 $s_1 \leq s_2 \leq \dots \leq s_m$, $s_1 = 1$ and $f^* = 1$. We shall also assume that the tasks have been indexed so that the list is given by $L = (1, 2, 3, \dots, n)$. 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, \dots, 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 $i+1, \dots, n$ and consider only tasks $1, 2, \dots, i$. 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 , $1 \leq i \leq m$, before task n is scheduled.

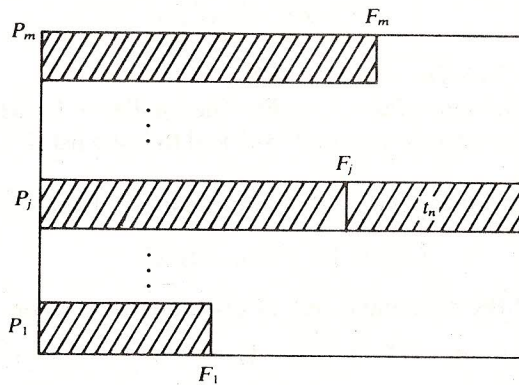


FIGURE 1

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

$$(1) \quad \sum_{i=1}^n t_i \leq \sum_{i=1}^m s_i,$$

$$(2) \quad t_n \leq \max \{t_i\} \leq s_m.$$

From the definition of list schedules, it follows that

$$(3) \quad F_i + t_n/s_i \geq \hat{f}, \quad 1 \leq i \leq m,$$

