MIT/LCS/TM-161

CRITICAL PATH SCHEDULING OF TASK SYSTEMS WITH RESOURCE AND PROCESSOR CONSTRAINTS·

Errol L. Lloyd

March 1980

Critical Path Scheduling of Task Systems With Resource and Processor Constraints

Errol L. Lloyd
Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts

Key Words: scheduling, task system with resources, bin packing

Abstract

Several papers over the past few years have investigated minimum execution time scheduling of unit execution time (UET) task systems with resources. Because such scheduling problems are, in general, NP-hard, a variety of heuristic methods for producing schedules have been studied, among them, critical path scheduling. The strongest results to date have been for systems where there is no processor constraint. These results may be utilized for systems with a processor constraint by treating the processors as an additional resource. Unfortunately, in those cases where the number of processors is close to the number of resources, this results in an upper bound which is somewhat misleading. In this paper we investigate the performance of critical path scheduling for UET task systems with resources and a fixed number of processors. An upper bound for the worst case performance of critical path scheduling is given. This bound depends both on the number of processors and on the number of different resources. Moreover, we show that this is the best possible (asymptotic) upper bound.

This work was partially supported by the National Science Foundation under Grant MCS78-05849.

1.1 The Model

A UET task system with resources is a system $S = \langle T, \prec, m, s \rangle$ where:

1. $T = \{T_1, \ldots, T_n\}$ is a set of tasks. Each task has an execution time of one unit.
2. $\prec$ is a partial order specifying precedence constraints between the tasks.
3. There are m identical processors.
4. s is the number of different resources. It is assumed that $s \geq 1$, that there is exactly one unit of each resource, and that each task may require any portion of that one unit for each resource.

For each task T_i and each resource v, $R_v(T_i)$ specifies the portion of resource v required by task T_i during its execution. Moreover, $R_{max}(T_i) = \max \{R_v(T_i): 1 \leq v \leq s\}$. $R_v(T_i)$ is the R_v-value of task T_i and $R_{max}(T_i)$ is the R_{max}-value of task T_i. This notation is extended to a set of tasks B, with $R_v(B) = \Sigma R_v(T)$ over all $T \in B$ and $R_{max}(B) = \Sigma R_{max}(T)$ over all $T \in B$. For convenience, if B is empty, let $R_{max}(B) = 0$. Finally, a set of tasks B, is a legal set of tasks if for each resource v, $R_v(B) \leq 1$.

With respect to the precedence constraints, we have the following definitions: If $T_i \prec T_j$, then T_j is a successor of T_i, and T_i is a predecessor of T_j. The level of a task in the precedence structure may be defined as follows: If T_i has no successors then $level(T_i) = 1$; otherwise, $level(T_i) = 1 + \max\{level(T_j): T_i \prec T_j\}$. This notion can be extended to a set of tasks B, by letting $level(B) = \max\{level(T_i) : T_i \in B\}$.

A list schedule for a task system with resources may be constructed as follows: Initially, let L be an ordered list of the tasks. If a processor becomes available at time t, the list L is instantaneously scanned from the beginning and the first task T_i (if any) which meets the following criteria is removed from L and assigned to the available processor: 1. Each task T_j such that $T_j \prec T_i$, has completed execution and 2. If $[r_1, \ldots, r_s]$ represents the total resource requirements of all tasks executing at time t, then for each resource v, $r_v + R_v(T_i) \leq 1$. This last requirement guarantees that the currently executing tasks do not require more than a total of one unit for any resource. If there is no task in L meeting the criteria then the processor is idle until time t+1. If several processors become available simultaneously, they are assigned tasks sequentially beginning with the lowest numbered processor.

A critical path schedule is a special type of list schedule where the list has the property that for any two tasks T_i and T_j, if $level(T_i) > level(T_j)$ then T_i precedes T_j on the list. Critical path schedules are of particular practical interest because they are applicable to any system having precedence constraints and are easy to construct.

There are a number of criteria for determining the quality of a schedule. In this paper we investigate the problem of minimizing the total schedule length. We let ω be the length of an arbitrary list schedule, let ω_{cp} be the length of a critical path schedule and let ω_0 be the length of an optimal schedule. A schedule of length ω is said to consist of time units $B_1, \ldots, B_\omega$ where $B_t = \{T_j \in T$: task T_j executes at time t of the schedule$\}$.

1.2 Relation to bin packing

The problem we have described here may also be formulated as a bin packing problem. In that interpretation, the T_i's are items to be placed in bins $B_1, B_2, \ldots$, under the constraint that no more than m items are to be placed in any single bin, and for each resource, the items in any single bin require no more

than one unit of that resource. In this case the objective is to pack the items into a minimum number of bins.

1.3 Background

Task systems with resources were first investigated by Garey and Graham[2,3]. They showed that for systems with arbitrary task execution times, no precedence constraints, and $m \geq 2$ processors, $\omega/\omega_0 \leq \min\{(m+1)/2, s+2-(2s+1)/m\}$, and that the result is the best possible. Garey and Johnson[5] have shown that finding optimal schedules for UET task systems is NP-hard for the case of two processors, one resource and precedence constraints restricted to be a forest, and for the case of three processors, one resource and no precedence constraints. These results make the study of heuristic algorithms particularly important, since it is unlikely that a polynomial time algorithm exists which computes minimum length schedules.

In this regard, Garey, et.al.[4] show, for arbitrary list schedules and UET task systems with resources, that $\omega/\omega_0 \leq s\omega_0 + s/2 + 1$ if $m \geq n$. They also show that this bound is the best possible. In [8], Lloyd gives tight bounds for ω/ω_0 when $m \geq 2$ under a slightly different model.

For critical path schedules and UET task systems with resources, Garey, et.al.[4] show that $\omega_{cp}/\omega_0 \leq 17s/10 + 1$ when $m \geq n$. Moreover, this result is essentially the best possible. When $m \geq 2$ (the problem we study), Yao[9] shows that $\omega_{cp}/\omega_0 \leq \min\{ m, 2 + 2s - (2s+1)/m \}$. Comprehensive surveys of scheduling results may be found in [1,6].

In this paper we investigate critical path scheduling of UET task systems with resources which have a processor constraint (i.e. $m \geq 2$). We give an upper bound for the worst case of the ratio ω_{cp}/ω_0. Moreover, we show that this is (asymptotically) the best possible upper bound.

It has been noted that results associated with task systems with no processor constraints can be applied to systems with processor constraints, merely by treating the processors as an additional resource. That is, given a system with s resources and a processor constraint, apply the results as if this was a system with $s+1$ resources and no processor constraint. However, from an intuitive viewpoint, this approach is suspect, since processors are not "just another resource". The processor resource possesses certain characteristics that are not shared by resources in general. In particular, every task requires exactly one unit of the processor resource. The processor resource, then, is unique, in that a task may not require just any portion of the resource, as was assumed for resources in general. At least intuitively, there is no reason to believe that treating the processors as an additional resource will result in accurate worst case bounds.

Our results tend to support this intuition. Consider the bound of $\omega_{cp}/\omega_0 \leq 17s/10 + 1$ for the no processor constraint case. When applied to systems with a processor constraint, this bound becomes $\omega_{cp}/\omega_0 \leq 17s/10 + 27/10$. This can be combined with the result of Yao[9] to yield a composite bound of $\omega_{cp}/\omega_0 \leq \min\{ m, 2 + 2s - (2s+1)/m, 17s/10 + 27/10\}$ for $m \geq 2$. This composite bound contains the best available information for dealing with the processor constraint case. Unfortunately, depending on the relationship of s and m, even this composite bound may be quite different from the best upper bound. For example, when $s > 6$ and $m = 1.8s + 2$, this bound indicates that $\omega_{cp}/\omega_0 \leq 17s/10 + 27/10$. The bound that we give shows that $\omega_{cp}/\omega_0 \leq 14s/10 + 3/2$. The difference between the two bounds is $3s/10 + 6/5$ -- a value which grows linearly with s. In percentages, the composite bound in this case is too large by over 21 percent. In all cases where $m \geq s+1$, our bounds improve over the composite result.

2. Weighting Functions

In this section we state the main theorem, briefly outline the proof of that theorem and describe three "weighting functions" which play a key role in the proof. The main result of this paper is:

Theorem 1: If $S = \langle T, <, m, s\rangle$ is a UET task system with resources where $s \geq 1$ and $m \geq 2$, then

$$\omega_{cp}/\omega_o \leq \begin{cases} m & \text{if } 2 \leq m < s+1 \\ (s+m+1)/2 & \text{if } s+1 \leq m < 2s+1 \\ (4s+m+3)/4 & \text{if } 2s+1 \leq m < 8s/3+1 \\ (14s+m+9)/10 & \text{if } 8s/3+1 \leq m < 3s+1 \\ 2+17s/10-(3s+1)/m & \text{if } 3s+1 \leq m \text{ and } m \geq 10 \\ 2+5s/3-(8s/3+1)/m & \text{if } 3s+1 \leq m \text{ and } m < 10 \end{cases}$$

2.1 Proof Outline

Consider any critical path schedule for the task system S. The time units of that schedule may be divided into three sets: those time units where the final task of each level executes, those where all of the processors are utilized and those where at least one processor is idle due solely to resource constraints. Call these path, full and resource time units respectively. The proof follows by bounding the number of time units of each type. The number of path time units is bounded by the length of an optimal schedule. The number of full time units can be bounded using the length of an optimal schedule and the number of tasks executed in resource and path time units. The number of resource time units can bounded by the use of a "weighting function".

A weighting function W, is a mapping from the interval [0, 1] to an interval [0, x], where the x depends on the particular weighting function. We extend the functional notation to tasks and let $W(T) = W(R_{max}(T))$. Moreover, if B is a set of tasks, then $W(B) = \Sigma\, W(T)$ over all $T \in B$. (Our use of weighting functions is motivated by and draws upon the work of Garey, et.al. [4]). Given a particular weighting function and a set of resource time units, the average weight associated with each of those time units can be bounded below (this lower bound will be 1). Moreover, by examining an optimal schedule, the total weight associated with all tasks executing in resource time units can be bounded above. Combining these two bounds gives an upper bound on the number of resource time units. The main result then follows from the upper bounds on the numbers of path, full and resource time units.

2.2 Two important properties

In this section we introduce two properties of weighting functions.

Definition: Weighting function W has Property A, if:

Given a task T' and a nonempty set of tasks B such that:

$R_{max}(T) \geq R_{max}(T')$ for each $T \in B$ and $R_{max}(T') > 1 - R_{max}(B)$,

then $W(B) \geq 1$.

Definition: Weighting function W has Property B, if:

Given a set of time units $\{B_1, \ldots, B_t\}$ with $t \geq 1$ and $Y = \cup_{i=1}^{t} B_i$, such that:

For every task $T \in B_i$, $1 < i \leq t$, and every j, $1 \leq j < i$, $R_{max}(T) > 1 - R_{max}(B_j)$,

then there exists a task $T^* \in Y$, such that $W(Y - \{T^*\}) \geq t-1$.

Intuitively, Property A states that given a set of n tasks in which the total resource requirements of the tasks exceeds one, then the total weight of the largest n-1 tasks is at least one. Property B will be used to obtain a lower bound on the average weight associated with a resource time unit.

Lemma 1: If W is a weighting function which has Property A, then W also has Property B.

Proof

Assume that W is a weighting function which has Property A, and let $\{B_1, \ldots, B_t\}$ be a set of time units with $Y = \bigcup_{i=1}^{t} B_i$ such that for every task $T \in B_i$, $1 < i \leq t$, and every j, $1 \leq j < i$, $R_{max}(T) > 1 - R_{max}(B_j)$. We want to show that there exists a task $T^* \in Y$ such that $W(Y-\{T^*\}) \geq t-1$. Without loss of generality, assume that $W(B_i) < 1$ for each time unit B_i, $1 \leq i \leq t$. The proof is by induction on t.

If $t = 1$ the lemma is immediate, so suppose that $t \geq 2$. Consider time units B_{t-1} and B_t. Let X be any task in B_t. Then $R_{max}(X) > 1 - R_{max}(B_{t-1})$. Moreover, for any task $T \in (B_{t-1} \cup \{X\})$, $R_{max}(T) > 1 - R_{max}(B_{t-1} \cup \{X\} - \{T\})$. In particular, let Z be a task in $(B_{t-1} \cup \{X\})$ with a minimal R_{max}-value. From Property A, it follows that $W(B_{t-1} \cup \{X\} - \{Z\}) \geq 1$.

Now consider the set of time units $\{B'_1, \ldots, B'_{t-1}\}$, where $B'_i = B_i$ for $1 \leq i \leq t-2$, and $B'_{t-1} = \{Z\}$. Let $Y' = \bigcup_{i=1}^{t-1} B'_i$. By induction, there exists a $T^* \in Y'$, such that $W(Y'-\{T^*\}) \geq t-2$. Thus, $W(Y - \{T^*\}) \geq W(Y' - \{T^*\}) + W(B_{t-1} \cup \{X\} - \{Z\}) \geq t-2 + 1 = t-1$. □

2.3 The Weighting Functions

Three weighting functions are used in the proof of the main theorem. Three functions are used, as opposed to just one, due to varying requirements with respect to the weights assigned in various parts of that proof. Weighting function W_1 has the property that if $\alpha_1 + \alpha_2 \leq 1$, then $W_1(\alpha_1) + W_1(\alpha_2) \leq 1.5$. Moreover, values of α_1 and α_2 exist such that $W_1(\alpha_1) + W_1(\alpha_2) = 1.5$. A similar statement can be made about weighting function W_2 and the value 1.6. Weighting function W_3 has the property that if $\alpha_1 + \ldots + \alpha_n \leq 1$, then $W_3(\alpha_1) + \ldots + W_3(\alpha_n) \leq 1.7$. These properties play a critical role in establishing various segments of the upper bound.

For each of the three weighting functions which we introduce, we give two major results. First, we give an upper bound on the weight of a legal set of tasks. As a corollary to this result we give an upper bound on the weight of any set of tasks drawn from the task system which we are considering. Both of these bounds depend upon the cardinality of the set of tasks being considered. These results will allow us to bound the total weight of the tasks executing in resource time units. Secondly, we show that the weighting function has Property B.

2.3.1 The First Weighting Function

Definition: $W_1(\alpha) = \begin{cases} 0 & \text{if } \alpha = 0 \\ 1/4 & \text{if } \alpha \in (0, 1/4] \\ 1/2 & \text{if } \alpha \in (1/4, 1/2] \\ 1 & \text{if } \alpha \in (1/2, 1] \end{cases}$

Lemma 2: If B is a legal set of tasks, then $W_1(B) \leq \min\{ (|B|+s)/2, (|B|+4s)/4 \}$.

Proof

Recall that B is a legal set of tasks if for each resource v, the total usage of v by the tasks in B does not exceed one.

Part 1: Let $X = \{T \in B: R_{max}(T) > 1/2\}$ and let $x = |X|$. Since for each resource v, there is at most one $T \in B$, such that $R_v(T) > 1/2$, it must be that $x \leq s$. Moreover, if $R_{max}(T) > 1/2$ then $W_1(T) = 1$. Each task $T \in (B - X)$ has $R_{max}(T) \leq 1/2$, hence $W_1(T) \leq 1/2$. Thus, $W_1(B)$ is bounded above by $\max[x + (|B|-x)/2]$ such that $x \leq s$. This maximum occurs at $x = s$. Therefore, $W_1(B) \leq s + (|B|-s)/2 = (|B|+s)/2$.

Part 2: Let $X = \{T \in B: R_{max}(T) > 1/2\}$, let $x = |X|$, let $Y = \{T \in B: 1/4 < R_{max}(T) \leq 1/2\}$ and let $y = |Y|$. Similarly to Case 1, we deduce that $x \leq s$ and $y \leq 3s - 2x$. Moreover, if $R_{max}(T) > 1/2$ then $W_1(T) = 1$ and if $1/4 < R_{max}(T) \leq 1/2$ then $W_1(T) = 1/2$. Each task $T \in (B - X - Y)$ has $R_{max}(T) \leq 1/4$ and $W_1(T) \leq 1/4$. Thus, $W_1(B)$ is bounded above by $\max\{x + y/2 + (|B|-x-y)/4]$ such that $x \leq s$ and $y \leq 3s - 2x$. This maximum occurs at $x = y = s$, so $W_1(B) \leq s + s/2 + (|B|-2s)/4 = (|B|+4s)/4$. □

Corollary 1: Given a set of tasks $Y \subseteq T$, then $W_1(Y) \leq \min\{ (|Y|+s\omega_o)/2, (|Y|+4s\omega_o)/4 \}$.

Proof

Let $B_1, \ldots B_{\omega_o}$ be the time units of an optimal schedule restricted to the tasks in Y. Then, $Y = \cup_{i=1}^{\omega_o} B_i$ and $W_1(Y) = \Sigma_{i=1}^{\omega_o} W_1(B_i)$.

Part 1: By Lemma 1, each $W_1(B_i) \leq (|B_i|+s)/2$. Thus, $W_1(Y) \leq \Sigma_{i=1}^{\omega_o} (|B_i|+s)/2 = s\omega_o/2 + \Sigma_{i=1}^{\omega_o} |B_i|/2 = (|Y|+s\omega_o)/2$.

Part 2: By Lemma 1, each $W_1(B_i) \leq (|B_i|+4s)/4$. Thus, $W_1(Y) \leq \Sigma_{i=1}^{\omega_o} (|B_i|+4s)/4 = (|Y|+4s\omega_o)/4$. □

Lemma 3: Weighting function W_1 has Property B.

Proof

By Lemma 1, it is sufficient to show that W_1 has Property A. Consider a task T' and a nonempty set of tasks B, such that $R_{max}(T) \geq R_{max}(T')$ for each $T \in B$ and $R_{max}(T') > 1 - R_{max}(B)$. We want to show that $W_1(B) \geq 1$.

If $R_{max}(T) > 1/2$ for any $T \in B$, then the lemma is immediate, so suppose $R_{max}(T) \leq 1/2$ for each $T \in B$. If $R_{max}(T') = 0$ then $R_{max}(B) \geq 1$, hence $W_1(B) \geq 1$, so suppose $R_{max}(T') > 0$.

Case 1: $R_{max}(T') \in (0, 1/4]$

Then $R_{max}(B) > 3/4$. Since for each $T \in B$, $0 < R_{max}(T) \leq 1/2$, we have that $|B| \geq 2$. Moreover, for $T \in B$, $W_2(T)$ is either 1/4 or 1/2. If $|B| \geq 4$, then the lemma is immediate. If $|B| = 3$ then at least one of the tasks has an R_{max}-value exceeding 1/4, hence it has a weight of 1/2. The other 2 tasks have weights of at least 1/4. Thus, $W_1(B) \geq 1$. If $|B| = 2$, then both of the tasks in B must have R_{max}-values exceeding 1/4, hence they have weights of 1/2, and $W_1(B) = 1$.

Case 2: $R_{max}(T') \in (1/4, 1/2]$

Then $R_{max}(B) > 1/2$. Hence $|B| \geq 2$, since $R_{max}(T) \leq 1/2$ for each $T \in B$. Since for each $T \in B$, $R_{max}(T) \geq R_{max}(T')$, we have: $R_{max}(T) \in (1/4, 1/2]$ and $W_1(T) = 1/2$ for $T \in B$. Thus, $W_1(B) = |B|/2 \geq 1$. □

2.3.2 The Second Weighting Function

Definition: $W_2(\alpha) =$

0 if $\alpha = 0$
$10/100$ if $\alpha \in (0, .092]$
$15/100$ if $\alpha \in (.092, .136]$
$20/100$ if $\alpha \in (.136, .182]$
$25/100$ if $\alpha \in (.182, .204]$
$30/100$ if $\alpha \in (.204, .250]$
$40/100$ if $\alpha \in (.250, .296]$
$45/100$ if $\alpha \in (.296, .318]$
$50/100$ if $\alpha \in (.318, .364]$
$55/100$ if $\alpha \in (.364, .408]$
$60/100$ if $\alpha \in (.408, .500]$
1 if $\alpha \in (.500, 1]$

The following two lemmas and corollary about W_2 are proven in the appendix.

Lemma 4: If B is a legal set of tasks, then $W_2(B) \leq (|B|+14s)/10$.

Corollary 2: Given a set of tasks $Y \subseteq T$, then $W_2(Y) \leq (|Y|+14s\omega_0)/10$.

Lemma 5: Weighting function W_2 has Property B.

2.3.3 The Third Weighting Function

Definition: $W_3(\alpha) =$

$(6/5)\alpha$ if $\alpha \in [0, 1/6]$
$(9/5)\alpha - 1/10$ if $\alpha \in (1/6, 1/3]$
$(6/5)\alpha + 1/10$ if $\alpha \in (1/3, 1/2]$
$(6/5)\alpha + 4/10$ if $\alpha \in (1/2, 1]$

This is the weighting function defined in Garey, et.al.[4]. In that paper the following corollary and lemma about W_3 are proven.

Corollary 3: Given a set of tasks $Y \subseteq T$, then $W_3(Y) \leq 17s\omega_0/10$.

Lemma 6: Given $0 \leq \alpha < 1/2$, and a set of tasks $B = \{T_1, \ldots, T_n\}$ with $n \geq 2$, such that $R_{max}(T_1) \geq R_{max}(T_2) > \alpha$ and $\alpha \geq 1 - R_{max}(B)$, then $W_3(B) \geq 1$.

A straight-forward consequence of Lemma 6 and the definition of W_3 (used to handle $|B| = 1$ and $R_{max}(T') \geq 1/2$) is that W_3 has Property A, hence:

Lemma 7: Weighting function W_3 has Property B.

3. The Main Result

In this section we complete the proof of Theorem 1. Assume that a UET task system with resources $S = \langle T, <, m, s\rangle$ is given. Let ω_{cp} be a critical path schedule and let ω_o be an optimal schedule for this system. The time units of ω_{cp} are partitioned into the following three sets:

$P = \{B_i \in \omega_{cp}: (\forall j > i)[\text{level}(B_i) > \text{level}(B_j)]\}$

$F = \{B_i \in \omega_{cp} : |B_i| = m \text{ and } B_i \notin P\}$

$H = \{B_i \in \omega_{cp}: |B_i| < m \text{ and } B_i \notin P\}$

The time units in P are path time units, those in F are full time units, and those in H are resource time units. Clearly, $\omega_{cp} = |P| + |F| + |H|$.

Let $Q = \{T \in T: T \in B_i \text{ and } B_i \in H\}$ (i.e. Q consists of all tasks executing in resource time units of ω_{cp}). Clearly, $|P| \leq \omega_o$ and $|F| \leq \omega_o - |P|/m - |Q|/m$. The number of resource time units |H|, can be bounded by use of the following lemma (adapted from a lemma given by Garey, et.al.[4]).

Lemma 8: If W is a weighting function which has Property B, then there exists a set of tasks $Q' \subseteq T$ with $|Q'| = |Q|$ such that $|H| \leq W(Q')$.

Proof

Assume that W is a weighting function which has Property B. Let k be the maximum level of any task in T. For each level l, $1 \leq l \leq k$, there is one time unit $B_l \in P$ with $\text{level}(B_l) = l$. Let T_l be any task in B_l with $\text{level}(T_l) = l$. Moreover, for each level l, $1 \leq l \leq k$, define the following two sets:

$A_l = \{B_i \in H : \text{level}(B_i) = l\}$

$L_l = \{T: \text{level}(T) = l \text{ and } (\exists B_i \in A_l)[T \in B_i]\} \cup \{T_l\}$.

Thus, A_l contains all of the resource time units where the highest level of any task executing in the time unit is l. Likewise, L_l contains task T_l and all level l tasks executing in a resource time unit where the highest level of any task executing in the time unit is l. Figure 1 shows the correspondence between L_l, T_l and A_l.

Consider any set A_l. We claim that there exists a task $X_l \in L_l$ such that $W(L_l - \{X_l\}) \geq |A_l|$. If $|A_l| = 0$ then the result is immediate, so assume that $|A_l| \geq 1$. Let $B_1, \ldots, B_{|A_l|}$ be the time units in A_l. For each $B_i \in A_l$, let $B_i' = B_i \cap L_l$. There is one B_i' for each B_i, and each B_i' contains at least one task. Also, let $B_{|A_l|+1}' = \{T_l\}$. Note that $\bigcup_{i=1}^{|A_l|} B_i' = L_l$. Moreover, each B_i' contains only level l tasks.

Now consider any B_j' and B_i', with $j < i$. Let T be any task in B_i'. When T was scheduled all tasks with levels larger than l must have already been scheduled in time units prior to B_j. Moreover, the only tasks already scheduled in time unit B_j were level l tasks. Thus, T was not scheduled to execute in B_j due solely to resource constraints imposed by the level l tasks in B_j. This means that for $T \in B_i'$, $R_{max}(T) > 1 - R_{max}(B_j')$ for all $j < i$. Thus, the B_j' s form a set of time units for which the conditions

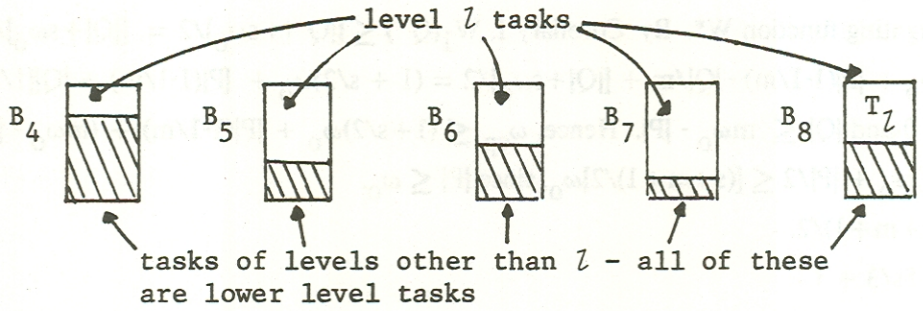
FIGURE 1: An example of the sets A_l and L_l, and the task T_l.

Assume that B_8 has a level of l and is a path time unit. This means that the task in B_8 of the highest level has level l, and that all tasks executing in time units after B_8 have levels less than l.

Some unmber of time units immediately preceding B_8 also have a level of l. Assume that these are time units B_4, B_5, B_6, and B_7. The set The set A_l consists of these 4 time units. The set L_l consists of all of the level l tasks which execute in these 4 time units, along with task T_l.

$A_l = \{B_4, B_5, B_6, B_7\}$ $\quad$ level(B_i) = l $\quad$ for i = 4, 5, 6, 7, 8

B_8 is B_l in this instance



B_4, B_5, B_6, B_7 are resource time units and B_8 is a path time unit

$L_l = \{T: \text{level}(T) = l \text{ and } T \text{ is in a time unit in } A_l\} \cup \{T_l\}$

The tasks in the non-shaded portions of B_4, B_5, B_6, B_7 and B_8 are the tasks in L_l.

given in the definition of Property B hold. Then, since weighting function W has Property B there exists a task $X_l \in L_l$ (since $L_l = \cup_{i=1}^{|A_l|} B_i^l$) such that $W(L_l - \{X_l\}) \geq |A_l|$, and the claim is proved.

Finally, let $Q' = (Q \cup \{T_l : 1 \leq l \leq k\}) - \{X_l : 1 \leq l \leq k\}$. Clearly, $|Q'| = |Q|$.

$\therefore |H| = \Sigma_{l=1}^{k} |A_l| \leq \Sigma_{l=1}^{k} W(L_l - \{X_l\}) \leq W(Q')$, since $\cup_{l=1}^{k} (L_l - \{X_l\}) \subseteq Q'$. □

From Lemma 8, it follows that given a particular weighting function W* which has Property B, there exists a set of tasks $Q' \subseteq T$ such that $|Q'| = |Q|$ and $|H| \leq W^*(Q')$.

Thus, $\omega_{cp} = |P| + |F| + |H| \leq |P| + (\omega_0 - |P|/m - |Q|/m) + W^*(Q')$, and with a reordering of terms,

$$\omega_{cp} \leq \omega_0 + |P|(1-1/m) - |Q|/m + W^*(Q'). \qquad (I)$$

There are six cases to consider based on the relative values of s and m.

Case 1: $2 \leq m < s+1$

Then $\omega_{cp} \leq m\omega_0$ since at least one task must execute during each time unit of ω_{cp}.

Case 2: $s+1 \leq m < 2s+1$

Let W_1 be the weighting function W*. By Corollary 1, $W_1(Q') \leq [|Q'| + s\omega_0]/2 = [|Q| + s\omega_0]/2$. Thus from (I), $\omega_{cp} \leq \omega_0 + |P|(1-1/m) - |Q|/m + [|Q| + s\omega_0]/2 = (1 + s/2)\omega_0 + |P|(1-1/m) + |Q|[1/2 - 1/m]$. But, $1/2 - 1/m \geq 0$ and $|Q| \leq m\omega_0 - |P|$. Hence, $\omega_{cp} \leq (1+s/2)\omega_0 + |P|(1-1/m) + (m\omega_0 - |P|)[1/2 - 1/m] = [(s+m)/2)\omega_0 + |P|/2 \leq [(s+m+1)/2]\omega_0$, since $|P| \leq \omega_0$.

$\therefore \omega_{cp}/\omega_0 \leq (s+m+1)/2$.

Case 3: $2s+1 \leq m < 8s/3 + 1$

First assume that $m \geq 4$. Let W_1 be the weighting function W*. Then by Corollary 1, $W_1(Q') \leq [|Q'| + 4s\omega_0]/4$. Similarly to Case 2, we derive from (I) that $\omega_{cp}/\omega_0 \leq (4s+m+3)/4$.

Now assume that $m < 4$. The only combination of s and m to lie in this range is $s=1$ and $m=3$. But, from Case 2 (since the assumption that $m < 2s+1$ was not used in that proof), $\omega_{cp}/\omega_0 \leq (s+m+1)/2 = (4s+m+5)/4$ when $s=1$ and $m=3$.

Case 4: $8s/3 + 1 \leq m < 3s+1$

First assume that $m \geq 10$. Let W_2 be the weighting function W*. Then by Corollary 2, $W_2(Q') \leq [|Q'| + 14s\omega_0]/10$. Similarly to Case 2 we derive from (I) that $\omega_{cp}/\omega_0 \leq (14s+m+9)/10$.

Now assume that $m < 10$. The only combination of s and m to lie in this range is $s=3$ and $m=9$. But, from Case 3, $\omega_{cp}/\omega_0 \leq (4s+m+3)/4 = (14s+m+9)/10$ when $s=3$ and $m=9$.

Case 5: $3s+1 \leq m$ and $m \geq 10$

First assume that $|Q| \geq 3s\omega_0$. Let W_3 be the weighting function W*. Then by Corollary 3, $W_3(Q') \leq 17s\omega_0/10$. Thus, from (I), $\omega_{cp} \leq \omega_0 + |P|(1 - 1/m) - |Q|/m + 17s\omega_0/10$. But $-|Q| \leq -3s\omega_0$ and $|P| \leq \omega_0$, so $\omega_{cp} \leq \omega_0 + \omega_0(1-1/m) - 3s\omega_0/m + 17s\omega_0/10 = \omega_0[2 + 17s/10 - (3s+1)/m]$.

Now assume that $|Q| < 3s\omega_0$. Let W_2 be the weighting function W*. Then by Corollary 2, $W_2(Q') \leq [|Q'| + 14s\omega_0]/10 = [|Q| + 14s\omega_0]/10$. Thus from (I), $\omega_{cp} \leq \omega_0 + |P|(1-1/m) - |Q|/m + [|Q| +$

$14s\omega_0]/10 = \omega_0[1 + 14s/10] + |P|(1-1/m) + |Q|[1/10 - 1/m]$. But $1/10 - 1/m \geq 0$, $|Q| < 3s\omega_0$ and $|P| \leq \omega_0$. Hence, $\omega_{cp} \leq \omega_0[1+14s/10] + \omega_0(1-1/m) + 3s\omega_0[1/10 -1/m] = \omega_0[2 + 17s/10 - (3s+1)/m]$. Thus, $\omega_{cp}/\omega_0 \leq 2 + 17s/10 - (3s+1)/m$.

Case 6: $3s+1 \leq m$ and $m < 10$

First assume that $|Q| \geq (8s/3)\omega_0$. Let W_2 be the weighting function W*. Then, by Corollary 2, $W_2(Q') \leq [|Q'| + 14s\omega_0]/10$. Similarly to Case 5 we derive from (I) that $\omega_{cp}/\omega_0 \leq 2 + 5s/3 - (8s/3+1)/m$.

Now assume that $|Q| < (8s/3)\omega_0$. Let W_1 be the weighting function W*. Then by Corollary 1, $W_1(Q') \leq [|Q'|+4s\omega_0]/4$. Similarly to Case 5 we derive from (I) that $\omega_{cp}/\omega_0 \leq 2 + 5s/3 - (8s/3 + 1)/m$.

This completes the proof of Theorem 1. □

4. The Achievable Bounds

In this section we show that the upper bound given in Theorem 1 is (asymptotically) the best possible upper bound. For each possible combination of s and m, we exhibit a UET task system with resources, $S = \langle T, \prec, m, s\rangle$, a critical path schedule for that system, and an optimal schedule for that system such that the ratio ω_{cp}/ω_o is arbitrarily close to the appropriate upper bound. As in Theorem 1, there are six cases to consider based on the relationship between s and m. The constructions that we use in the six cases are similar, but not identical. They make use of task systems which differ primarily in the resource usages of certain tasks in the system. The overall precedence structures of these systems are the same, as are the resource usages of several of the tasks. Thus, before proving each of the lemmas, this general task system structure is introduced. The aspects of the system which are the same in all cases are specified. We indicate which parameters will be specified within the proofs of the individual lemmas. We also sketch optimal and critical path schedules for this general system. The exact nature of these schedules will, of course, depend upon the values assigned to the unspecified parameters within the proofs of the individual lemmas.

4.1 A general task system structure

Assume that $s \geq 1$ and $m \geq 2$, with $m \geq s+1$, are given (in the next section we will indicate how to handle the case of $m \leq s$). Integers x and z are to be specified later, as is ϵ, a positive constant. Consider a task system S^* with the following tasks:

1. D_i for $1 \leq i \leq x$, such that $R_1(D_i) = \epsilon$ and $R_v(D_i) = 0$ for $v \neq 1$.
2. B_0 such that $R_1(B_0) = 1$ and $R_v(B_0) = 0$ for $v \neq 1$.
3. B_i for $1 \leq i \leq s$, such that $R_i(B_i) = 1$ and $R_v(B_i) = 0$ for $v \neq i$.
4. C_i for $1 \leq i \leq s$. These tasks require no resources.
5. A^i_j for $1 \leq i \leq s$ and $1 \leq j \leq z$. For $v \neq i$, $R_v(A^i_j) = 0$. The usage of resource i by each task A^i_j (its R_i-value) will be specified later (it will be a non-zero requirement). Tasks $A^i_1, \ldots, A^i_z$ are called A^i-tasks.

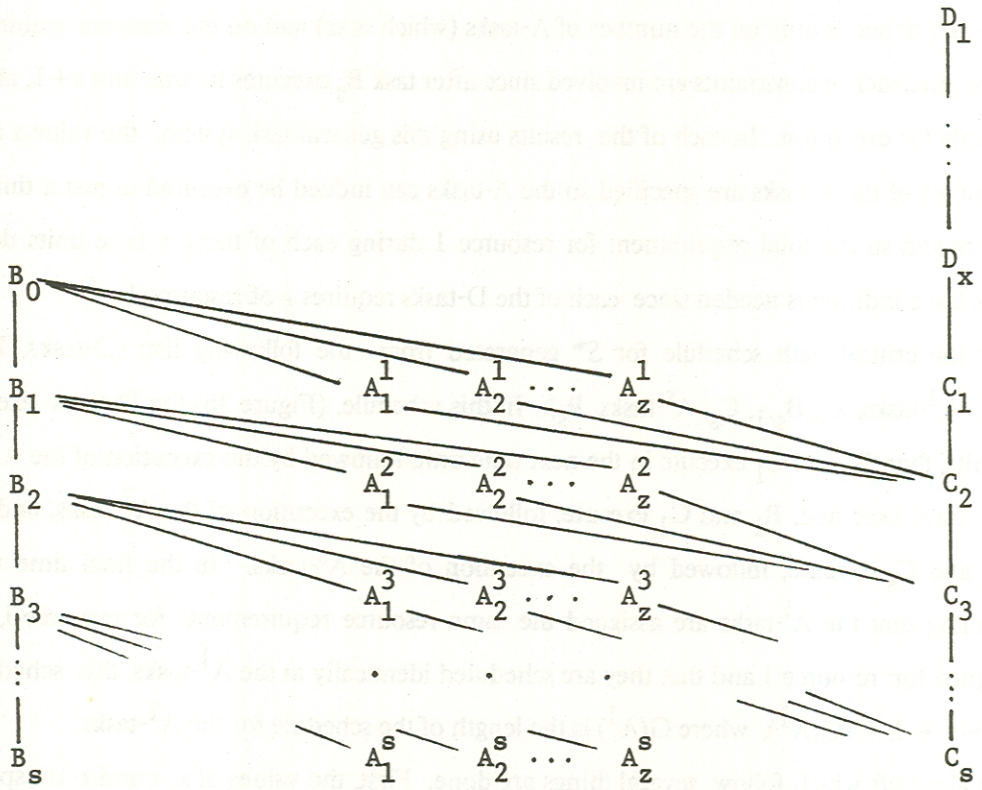
This task system has the following precedence constraints:

1. For $1 \leq i \leq x-1$, $D_i \prec D_{i+1}$. Moreover, $D_x \prec C_1$.
2. For $0 \leq i \leq s-1$, $B_i \prec B_{i+1}$ and $B_i \prec A^{i+1}_j$ for $1 \leq j \leq z$.
3. For $1 \leq i \leq s-1$ and $1 \leq j \leq z$, $A^i_j \prec C_{i+1}$.
4. For $1 \leq i \leq s-1$, $C_i \prec C_{i+1}$.

The precedence structure of this system is shown in Figure 2.

Assuming that the constants x, z and ϵ have been specified, consider the following schedule for S^* (Figure 3a): In the first $s+1$ time units execute the B-tasks. In the next x time units execute the D-tasks on processor

FIGURE 2: The general task system structure used for the lower bounds.



The non-zero resource requirements of these tasks are:

Each D-task requires ε of resource 1.

B_0 requires all of resource 1.

B_i requires all of resource i.

Each A^i-task requires a non-zero portion of resource 1.

m, and execute all of the A-tasks on the other m-1 processors. In the final s time units execute the C-tasks. Such a schedule has length x + 2s + 1. The assumption that the A-tasks can all be executed in time units s+2 through x+s+1 depends only on the number of A-tasks (which is sz) and on the resource requirements of the A-tasks - no precedence constraints are involved since after task B_s executes in time unit s+1, all of the A-tasks are available for execution. In each of the results using this general task system, the value z and the resource requirements of the A-tasks are specified so the A-tasks can indeed be executed in just x time units on m-1 processors and so the total requirement for resource 1 during each of those x time units does not exceed 1 - ϵ. This last condition is needed since each of the D-tasks requires ϵ of resource 1.

Now consider the critical path schedule for S^* generated from the following list: (D-tasks, B_0, C_1, A^1-tasks, B_1, C_2, A^2-tasks, ... , B_{s-1}, C_s, A^s-tasks, B_s). In this schedule, (Figure 3b) the D-tasks execute in the first x time units, then B_0 and C_1 execute in the next time unit, followed by the execution of the A^1-tasks. After those tasks have executed, B_1 and C_2 execute, followed by the execution of the A^2-tasks, and so on. Eventually, B_{s-1} and C_s execute, followed by the execution of the A^s-tasks. In the final time unit B_s executes. Assuming that the A^i-tasks are assigned the same resource requirements for resource i, as the A^1-tasks are assigned for resource 1 and that they are scheduled identically to the A^1-tasks, this schedule has length $\omega_{cp} = x + s + 1 + sG(A^1)$, where $G(A^1)$ is the length of the schedule for the A^1-tasks.

In the individual proofs which follow, several things are done. First, the values of x, z and ϵ are specified, and the remaining resource requirements for the A-tasks are given. We then show that the A-tasks can be executed on m-1 processors in x time units with the total requirement for resource 1 by the A-tasks, in each of those x time units, not exceeding 1 - ϵ. This establishes that $\omega_o \leq x + 2s + 1$. The value of $G(A^1)$ is then derived by analyzing a particular list schedule for the A^1-tasks, establishing that $\omega_{cp} \geq x + s + 1 + sG(A^1)$. The lower bound for the worst case of ω_{cp}/ω_o is then obtained by combining the bounds for ω_o and ω_{cp}. □

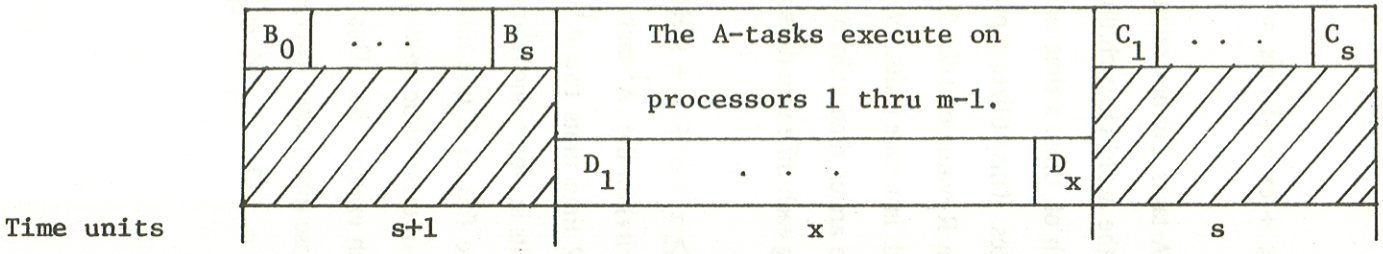
4.2 The Simple Cases

Lemma 9: If $2 \leq m < s + 1$, then ω_{cp}/ω_o can be arbitrarily close to m.

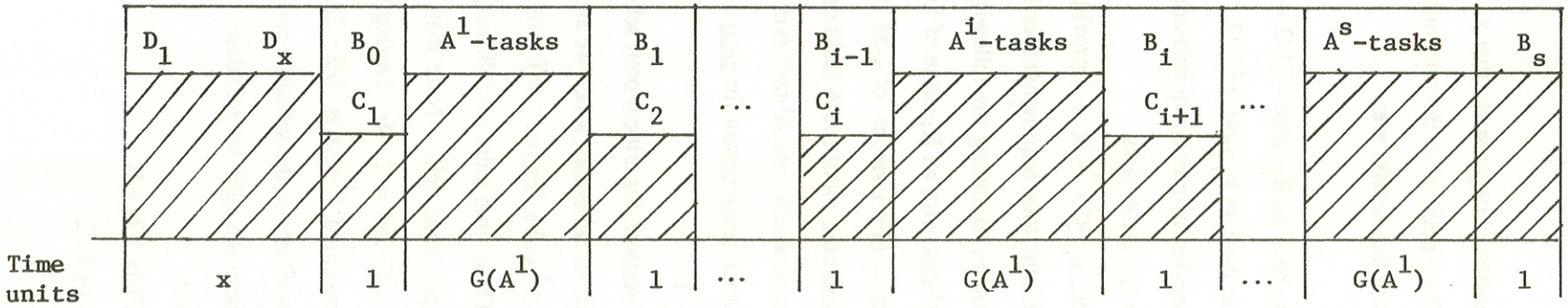
Proof

Assume that there are only m-1 resources. That is, assume s = m-1. (i.e. in the task system used to show that the upper bound of m may be approached arbitrarily closely, the tasks require only the first m-1 resources). The next lemma shows that in this case (i.e. $m \geq s+1$), that ω_{cp}/ω_o can be arbitrarily close to (s+m+1)/2. But, if m = s+1, then (s+m+1)/2 = m. □

FIGURE 3: Two schedules for the genral task system structure.



3a) An optimal schedule -- length = x + 2s + 1.



3b) A critical path schedule -- length = x + s + 1 + $sG(A^1)$

Lemma 10: If $s + 1 \leq m < 2s + 1$ then ω_{cp}/ω_0 can be arbitrarily close to $(s+m+1)/2$.

Proof

Let $c = (m-s-1)/s$. Let x be a positive integer such that $x = 0 \bmod 2s$, let $z = [1+c]x$ and let $\epsilon < 1/12$. Now consider the task system S^* as specified in the previous section, using these values of x, z and ϵ. The remaining resource requirements of the A-tasks are:

For each i, $1 \leq i \leq s$,

x of the A^i-tasks have an R_i-value of $1/2 + \epsilon$

cx of the A^i-tasks have an R_i-value of $1/2 - 2\epsilon$.

Note that for each i, we have specified resource requirements for exactly $x + cx = [1+c]x = z$ A^i-tasks. As desired, in total there are $zs = (m-1)x$ A-tasks.

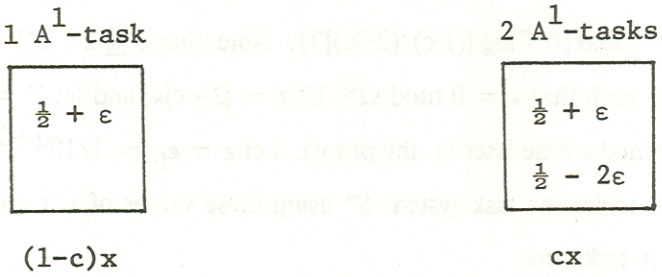
As noted in the previous section, $\omega_0 \leq x + 2s + 1$ provided all of the A-tasks can be executed on m-1 processors in just x time units, with the total requirement for resource 1 by the A-tasks in each time unit not exceeding $1 - \epsilon$. This can be done by executing the following tasks at each of those x time units (Figure 4a): For each i, $1 \leq i \leq s$, an A^i-task with an R_i-value of $1/2 + \epsilon$ executes. This utilizes s processors at each time unit. Moreover, for $cs = m-s-1$ values of i, an A^i-task with an R_i-value of $1/2 - 2\epsilon$ executes. Since m-1 A-tasks execute per time unit, all of the A-tasks can be executed in x time units. Note that for each i, there are (1-c)x time units in which one A^i-task executes and there are cx time units in which two A^i-tasks execute. Moreover, the total requirement for <u>each</u> resource during each time unit does not exceed $1 - \epsilon$. Therefore, $\omega_0 \leq x + 2s + 1$.

Also as noted in the previous section, for critical path schedules, $\omega_{cp} \geq x + s + 1 + sG(A^1)$, where $G(A^1)$ is the length of a particular list schedule (which we are about to specify) for the A^1-tasks. Consider the following schedule for the A^1-tasks (Figure 4b): In the first cx/2 time units two A^1-tasks with R_1-values of $1/2 - 2\epsilon$ execute. These time units are followed by x time units in which one A^1-task with an R_1-value of $1/2 + \epsilon$ executes per time unit. Note that in each of the first cx/2 tme units the total requirement for resource 1 is $2(1/2 - 2\epsilon) = 1 - 4\epsilon$. During the execution of these time units the smallest resource requirement of any unexecuted A^1-task is $1/2 - 2\epsilon$, a value which exceeds 4ϵ. This means that none of the A^1-tasks which execute later in the schedule can execute in these time units. This assures that the schedule we have described here is a valid list schedule. Thus, $G(A^1) = cx/2 + x$, and $\omega_{cp} \geq x + s + 1 + s[cx/2 + x] > x(m+s+1)/2$.

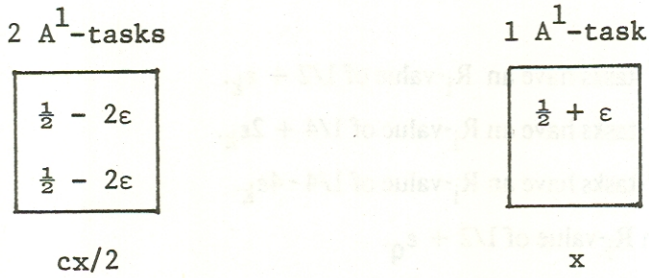
$\therefore \omega_{cp}/\omega_0 \geq (x(m+s+1)/2)/(x+2s+1)$

$\text{limit}_{x \to \infty} \omega_{cp}/\omega_0 = (m+s+1)/2$. □

FIGURE 4: The schedules used in Lemma 10.



4a) An optimal schedule - for each other resource v, A^v-tasks execute (in a similar manner) with these A^1-tasks.



4b) The schedule used for $G(A^1)$. These tasks execute alone.

In each of the above figures, the values inside of the boxes indicate the R_1-values of the tasks executing in those time units. The values under the boxes indicate the number of time units where tasks with those particular R_1-values execute.

Lemma 11: If $2s + 1 \leq m < 8s/3 + 1$, then ω_{cp}/ω_0 can be arbitrarily close to $(4s+m+3)/4$.

Proof

Let $c = (m-2s-1)/s$ and let $q = \max\{0, \lceil \log [(1-c)/(2-3c)] \rceil\}$. Note that $0 \leq c < 2/3$. Moreover, $q = 0$ iff $c \leq 1/2$. Let x be an integer such that $x = 0 \bmod s2^q$, let $z = [2+c]x$, and let $Y = 3c-2 + (1-c)/2^{q-1}$. (The origin of Y will be explained a little later in the proof). Let $\varepsilon = \varepsilon_0 = 1/10^{q+2}$. Also, for $1 \leq k \leq q$, let $\varepsilon_k = 10\varepsilon_{k-1}$. Now consider the task system S^* using these values of x, z and ε. The remaining resource requirements of the A-tasks are:

For each i, $1 \leq i \leq s$:

1. $(1-c)x$ of the A^i-tasks have an R_i-value of $1/2 + \varepsilon_0$

   $(1-c)x$ of the A^i-tasks have an R_i-value of $1/2 - 2\varepsilon_0$.

2. For $0 \leq k \leq q-1$,

   $(1-c)x/2^k$ of the A^i-tasks have an R_i-value of $1/2 + \varepsilon_k$.

   $(1-c)x/2^k$ of the A^i-tasks have an R_i-value of $1/4 + 2\varepsilon_k$.

   $(1-c)x/2^k$ of the A^i-tasks have an R_i-value of $1/4 - 4\varepsilon_k$.

3. Yx of the A^i-tasks have an R_i-value of $1/2 + \varepsilon_q$.

   Yx of the A^i-tasks have an R_i-value of $1/4 + 2\varepsilon_q$.

   Yx of the A^i-tasks have an R_i-value of $1/4 - 4\varepsilon_q$.

There are two cases to be considered here:

1. If $q = 0$, then no tasks are assigned resource requirements in part 2 of the above specifications. In this instance $Y = c$.
2. If $q > 0$, then some tasks are assigned resource requirements in part 2 of the specifications. Note that $Y \geq 0$, since $q < 1 + \log [(1-c)/(2-3c)]$.

In both cases, resource requirements are specified for exactly z A^i-tasks. The constant Y was chosen so that this was the case. Intuitively, in part 2 of the specifications, we assign R_i-values to the tasks in a series of sets of tasks. The number of tasks in each set is one half the number of tasks in the preceding set. Since there are only $[2+c]x = z$ A^i-tasks, the series must be terminated at an appropriate point. In this instance, that is after q sets. The value $3Yx$ is the number of A^i-tasks whose R_i-value has not been specified when the series is terminated. These $3Yx$ tasks are the tasks assigned R_i-values in part 3 of the specifications.

As before, $\omega_0 \leq x + 2s + 1$ provided all of the A-tasks can be executed in x time units with the total requirement for resource 1 by the A-tasks in each time unit not exceeding $1 - \varepsilon$. This can be done by executing the following tasks at each of those x time units (Figure 5a): For each i, $1 \leq i \leq s$, either 2 or 3 A^i-tasks execute at each of the x time units. More specifically, for $(1-c)s = 3s-m+1$ values of i, two A^i-tasks execute. They have R_i-values of $1/2 + \varepsilon_0$ and $1/2 - 2\varepsilon_0$. For the <u>other</u> $cs = m-2s-1$ values of i,

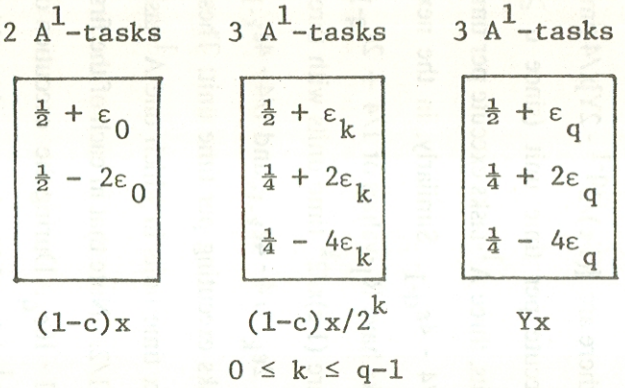
three A^i-tasks execute. They have R_i-values of $1/2 + \varepsilon_k$, $1/4 + 2\varepsilon_k$ and $1/4 - 4\varepsilon_k$, for some k, $0 \leq k \leq q$. Since at each time unit $2(1-c)s + 3cs = m-1$ tasks execute, all of the A-tasks can be executed in x time units. Note that for each i, there are $(1-c)x$ time units in which two A^i-tasks execute and there are cx time units in which three A^i-tasks execute. Moreover, since $\varepsilon_k \geq \varepsilon_0 = \varepsilon$ for $0 \leq k \leq q$, the total requirement for any resource during each time unit does not exceed $1 - \varepsilon$. Thus, the A-tasks can be executed in just x time units, and $\omega_o \leq x + 2s + 1$.

For critical path schedules, $\omega_{cp} \geq x + s + 1 + sG(A^1)$. There are two cases to consider based on the value of q (i.e. $q = 0$ and $q > 0$).

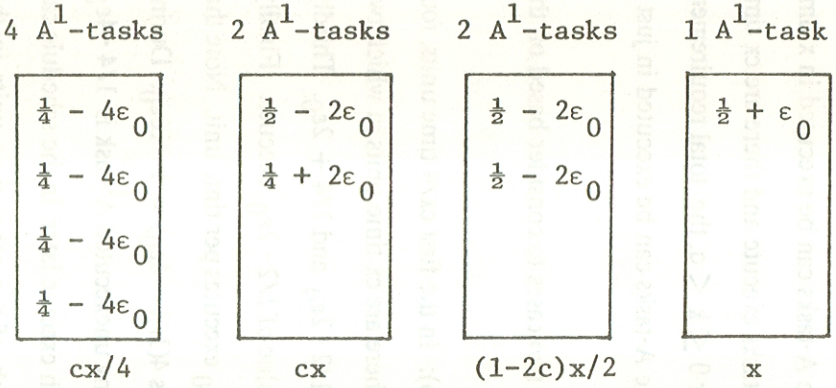
If $q = 0$, consider the following schedule for the A^1-tasks (Figure 5b): In the first $cx/4$ time units, four A^1-tasks with R_1-values of $1/4 - 4\varepsilon_0$ execute in each time unit. Next there are cx time units in which two A^1-tasks execute during each time unit. These tasks have R_i-values of $1/2 - 2\varepsilon_0$ and $1/4 + 2\varepsilon_0$. Thirdly, there are $(1-2c)x/2$ time units in which two A^1-tasks, each with an R_i-value of $1/2 - 2\varepsilon_0$, execute. Finally, there are x time units in which one A^1-task with an R_i-value of $1/2 + \varepsilon_0$ executes per time unit. Note that in each of the first $cx/4$ time units the total requirement for resource 1 is $4(1/4 - 4\varepsilon_0) = 1 - 16\varepsilon_0$. During the execution of these time units the smallest resource requirement of any unexecuted A^i-task is $1/4 - 4\varepsilon_0$, a value which exceeds $16\varepsilon_0$. This means that none of the A^1-tasks which execute later in the schedule can execute in these time units. Similar remarks can be made about each of the other time units in this schedule. This assures that the schedule we have described here is a valid list schedule. Thus, $G(A^1) = cx/4 + cx + (1-2c)x/2 + x = [3/2 + c/4]x$.

If $q > 0$, consider the following schedule for the A^1-tasks (Figure 5c): In the first $Yx/4$ time units four A^1-tasks with R_1-values of $1/4 - 4\varepsilon_q$ execute in each time unit. Next, there are $[(1-c)/2^{q-1} - 2Y]x/4$ time units in which four A^1-tasks with R_1-values of $1/4 - 4\varepsilon_{q-1}$ execute per time unit (since $q \geq \log[(1-c)/(2-3c)]$ this quantity is non-negative). In the next Yx time units, three A^1-tasks execute per time unit: these tasks have R_1-values of $1/4 + 2\varepsilon_q$, $1/4 - 4\varepsilon_{q-1}$, and $1/4 - 4\varepsilon_{q-1}$. Similarly, in the next $(1-c)x/2^{q-1}$ time units three A^1-tasks execute per time unit. These tasks have R_1-values of $1/4 + 2\varepsilon_{q-1}$, $1/4 - 4\varepsilon_{q-2}$, and $1/4 - 4\varepsilon_{q-2}$. Generally, for k, $q-1 \geq k \geq 1$, there are $(1-c)x/2^k$ time units with three A^1-tasks executing per time unit. These tasks have R_1-values of $1/4 + 2\varepsilon_k$, $1/4 - 4\varepsilon_{k-1}$, and $1/4 - 4\varepsilon_{k-1}$. Following these time units there are $(1-c)x$ time units with two A^1-tasks executing per time unit: These tasks have R_1-values of $1/2 - 2\varepsilon_0$ and $1/4 + 2\varepsilon_0$. Finally, there are x time units in which one A^1-task executes per time unit. Each of these tasks has an R_1-value exceeding 1/2. Note that in each of the first $Yx/4$ time units the total requirement for resource 1 is $4(1/4 - 4\varepsilon_q) = 1 - 16\varepsilon_q$. During the execution of these time units the smallest resource requirement of any unexecuted A^1-task is $1/4 - 4\varepsilon_q$, a value which

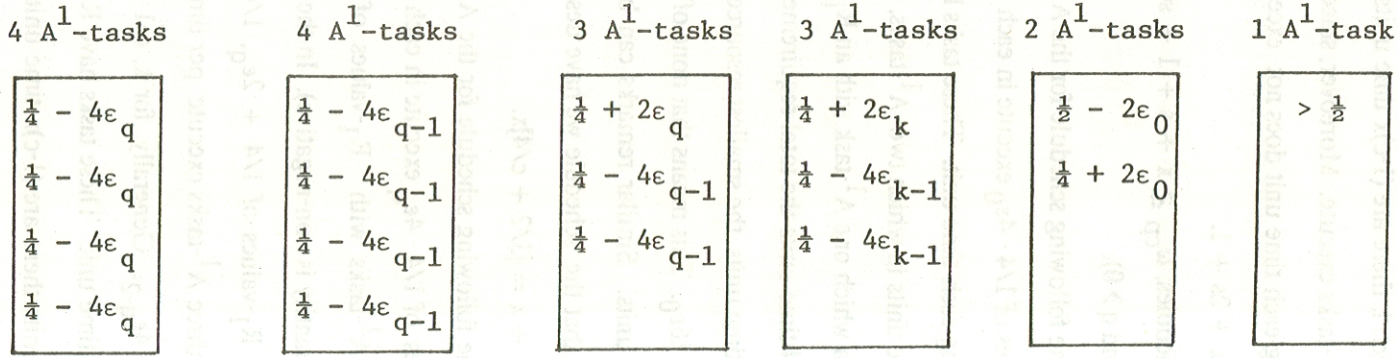
FIGURE 5: The schedules used in Lemma 11.



5a) An optimal schedule - for each other resource v, A^v-tasks execute (in a similar manner) with these A^1-tasks.



5b) The schedule used for $G(A^1)$ when q = 0. These tasks execute alone.



5c) The schedule used for $G(A^1)$ when q > 0. These tasks execute alone.

In each of the above figures, the values inside of the boxes indicate the R_1-values of the tasks executing in those time units. The values under the boxes indicate the number of time units where tasks with those particular R_1-values execute.

exceeds $16\varepsilon_q$. This means that none of the A^1-tasks which execute later in the schedule can execute in these time units. Similar remarks can be made about each of the other time units in this schedule. This assures that the schedule we have described here is a valid list schedule. Thus, $G(A^1) = [Y/4 + ((1-c)/2^{q-1} - 2Y)/4 + Y + \Sigma_{k=1}^{q-1}(1-c)/2^k + (1-c) + 1]x = [3/2 + c/4]x$.

$\therefore$ In both cases, $G(A^1) = [3/2 + c/4]x$ and $\omega_{cp} \geq x + s + 1 + s[3/2 + c/4]x > x(4s+m+3)/4$.

$\therefore \omega_{cp}/\omega_o \geq (x[4s+m+3]/4)/(x + 2s + 1)$

$\text{limit}_{x \to \infty}\ \omega_{cp}/\omega_o = (4s+m+3)/4$. □

4.3 A useful set of independent tasks

In the next two lemmas, we make use of a set of tasks originally described by Johnson, et.al.[7]. We have modified this set of tasks slightly to better suit our purposes.

Given some resource (say, resource 1) and an integer y, we will describe a set of 3y - 1 independent tasks. Each task requires some non-zero portion of the resource. These tasks can be grouped into three sets of tasks: In the first set all of the tasks have R_1-values of approximately 1/6; in the second set the tasks have R_1-values of approximately 1/3; and in the third set the tasks have R_1-values exceeding 1/2. Within each set the tasks differ slightly in their resource requirements. For instance, in the first set some of the tasks have resource requirements exceeding 1/6 and some have requirements less than 1/6. There are y tasks in each of the first two sets and y - 1 tasks in the third.

More formally, assume that an integer y, with $y = 0 \bmod 10$ is given. Let δ be such that $0 < \delta \ll 18^{-y/10}$. Also, let $\delta_i = \delta\ 18^{y/10 - i}$ for $1 \leq i \leq y/10$. Consider the following three sets of tasks:

1. The first set contains y tasks, T^1_{ji} for $0 \leq j \leq 9$ and $1 \leq i \leq y/10$. These tasks have the following resource requirements for $1 \leq i \leq y/10$:

   $R_1(T^1_{0i}) = 1/6 + 33\delta_i$

   $R_1(T^1_{1i}) = 1/6 - 3\delta_i$

   $R_1(T^1_{2i}) = R_1(T^1_{3i}) = 1/6 - 7\delta_i$

   $R_1(T^1_{4i}) = 1/6 - 13\delta_i$

   $R_1(T^1_{5i}) = 1/6 + 9\delta_i$

   $R_1(T^1_{6i}) = R_1(T^1_{7i}) = R_1(T^1_{8i}) = R_1(T^1_{9i}) = 1/6 - 2\delta_i$

2. The second set contains y tasks, T^2_{ji} for $0 \leq j \leq 9$ and $1 \leq i \leq y/10$. These tasks have the following resource requirements for $1 \leq i \leq y/10$:

   $R_1(T^2_{0i}) = 1/3 + 46\delta_i$

   $R_1(T^2_{1i}) = 1/3 - 34\delta_i$

$R_1(T^2_{2i}) = R_1(T^2_{3i}) = 1/3 + 6\delta_i$

$R_1(T^2_{4i}) = 1/3 + 12\delta_i$

$R_1(T^2_{5i}) = 1/3 - 10\delta_i$

$R_1(T^2_{6i}) = R_1(T^2_{7i}) = R_1(T^2_{8i}) = R_1(T^2_{9i}) = 1/3 + \delta_i$

3. The third set contains y - 1 tasks, T^3_i for $1 \leq i \leq y-1$. Each task requires $1/2 + \delta$ of resource 1.

An optimal schedule for these 3y-1 tasks has length y. It consists of time units with the following tasks:

1. For $2 \leq j \leq 9$ and $1 \leq i \leq y/10$, a T^3-task and T^1_{ji} and T^2_{ji}
2. For $1 \leq i \leq y/10$, a T^3-task and T^1_{0i} and T^2_{1i}
3. For $1 \leq i < y/10$, a T^3-task and T^1_{1i} and $T^2_{0,i+1}$
4. $T^1_{1,y/10}$ and T^2_{01}

Now consider the list $(T^1_{01}, \ldots, T^1_{91}, T^1_{02}, \ldots, T^1_{92}, \ldots, T^1_{0,y/10}, \ldots, T^1_{9,y/10}, T^2_{01}, \ldots, T^2_{91}, \ldots, T^2_{0,y/10}, \ldots, T^2_{9,y/10}, T^3_1, \ldots, T^3_{y-1})$. This list results in a schedule with length 17y/10 - 1. This follows easily from the results in [7]. We give an informal description of the schedule here. The schedule has y/5 time units in which 5 tasks from the first set execute per time unit and in which the total resource requirement in each of the time units exceeds 5/6; y/2 time units in which 2 tasks from the second set execute per time unit and in which the total resource requirement in each of the time units exceeds 2/3; and, y - 1 time units in which one task from the third set executes per time unit.

Now assume that y is fixed. Since each task in the system requires a non-zero portion of the resource, and since (in both of the schedules given above) each time unit has 5 or fewer executing tasks, there exists a $\beta_y > 0$, such that the resource requirement of every task can be reduced by β_y without changing either of the two schedules. Moreover, this implies that the total resource usage during any single time unit in these two schedules does not exceed $1 - \beta_y$.

In the next result, some A^i-tasks are assigned R_i-values in a manner similar to those assigned in previous lemmas, and some are assigned R_i-values similar to the resource requirements of the J-tasks.

4.4 The remaining cases

Lemma 12: If $8s/3 + 1 \leq m < 3s + 1$, then ω_{cp}/ω_o can be arbitrarily close to $(14s+m+9)/10$.

Proof

Let $c = (m-2s-1)/s$ and let $q > 0$ be an arbitrary integer. Note that $2/3 \leq c < 1$. Let $x = 20s2^{q-1}$, let $z = [2+c]x - 1$ and let $Y = 3c-2 + (1-c)/2^{q-1}$. The value Y will serve a purpose in this result similar to what it served in the previous result. Also similarly to the previous result, let $\varepsilon = \varepsilon_0 << \min\{ \beta_{Yx}, 1/10^{q+2} \}$

and for $1 \leq k \leq q$, let $\epsilon_k = 10\epsilon_{k-1}$. Now consider the task system S^* using these values of x, z, and ϵ. The remaining resource requirements of the A-tasks are as follows:

For each i, $1 \leq i \leq s$,

1. (1-c)x of the A^i-tasks have an R_i-value of $1/2 + \epsilon_0$.

   (1-c)x of the A^i-tasks have an R_i-value of $1/2 - 2\epsilon_0$.

2. For $0 \leq k \leq q-1$,

   $(1-c)x/2^k$ of the A^i-tasks have an R_i-value of $1/2 + \epsilon_k$.

   $(1-c)x/2^k$ of the A^i-tasks have an R_i-value of $1/4 + 2\epsilon_k$.

   $(1-c)x/2^k$ of the A^i-tasks have an R_i-value of $1/4 - 4\epsilon_k$.

3. $3Yx - 1$ of the A^i-tasks are assigned R_i-values equal to the R_1-values of the tasks in a set of $3Yx - 1$ J-tasks. These A^i-tasks will be called <u>type J</u> A^i-tasks.

An optimal schedule for this task system has a similar form for the execution of the A-tasks as the optimal schedules in the previous lemma. As before, $\omega_0 \leq x + 2s + 1$ provided all of the A-tasks can be executed in x time units on m-1 processors. This can be done by executing the following tasks at each of those x time units: For $(1-c)s = 3s-m+1$ values of i, two A^i-tasks execute: these tasks have R_i-values of $1/2 + \epsilon_0$ and $1/2 - 2\epsilon_0$. For the <u>other</u> $cs = m-2s-1$ values of i, either:

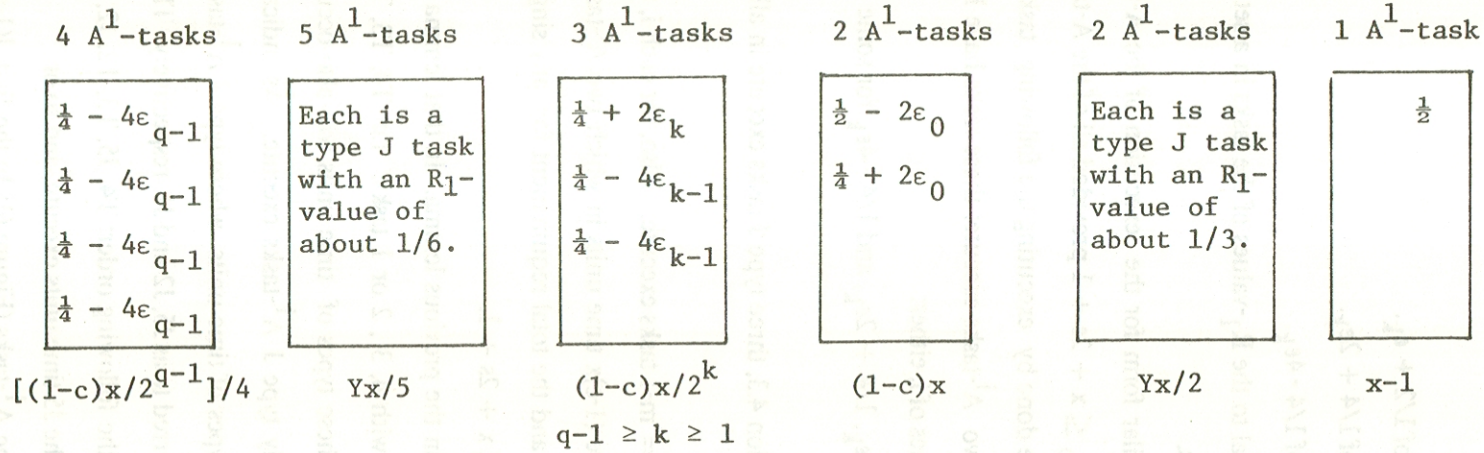
1. Three A^i-tasks execute having R_i-values of $1/2 + \epsilon_k$, $1/4 + 2\epsilon_k$, and $1/4 - 4\epsilon_k$ for some k, $0 \leq k \leq q-1$, or
2. Two or three type J tasks execute (as noted in section 4.3, three type J tasks execute in all but one of these time units).

Note that at each time unit no more than $2(1-c)s + 3cs = m-1$ tasks execute. Also, for each i, there are cx time units in which three A^i-tasks execute and there are (1-c)x time units in which two A^i-tasks execute. Thus, the A-tasks can be executed in just x time units and the total requirement for any single resource during each time unit does not exceed $1 - \epsilon$. Thus, $\omega_0 \leq x + 2s + 1$.

The execution of the A^1-tasks is also similar to that in the previous lemma. In that lemma (for $q > 0$), there were essentially four types of time units: those with 4, 3 , 2 or 1 tasks. Let T4, T3, T2 and T1 designate all of the time units of each type. Each of these types of time units will also occur here. In addition, in this proof, we have time units where only type J A^1-tasks execute. As indicated in our discussion in the previous section, there will be three types of time units where type J A^1-tasks execute. These time units contain 5, 2 and 1 tasks, and will be referred to as J5, J2 and J1, respectively. The schedule used to derive $G(A^1)$ consists all of these time units in the following order: T4, J5, T3, T2, J2, J1 and T1. That is, first all of the T4 time units execute, then all of the J5 time units execute, and so on.

More formally, consider the following schedule for the A^1-tasks (Figure 6): In the first $\lfloor (1-c)x/2^{q-1} \rfloor /4$

FIGURE 6: The schedule used for $G(A^1)$ in Lemma 12. These tasks execute alone.



The values inside of the boxes indicate the R_1-values of the tasks executing in those time units. The values under the boxes indicate the number of time units where tasks with those particular R_1-values execute.

time units four A^1-tasks, each with an R_1-value of $1/4 - 4\varepsilon_{q-1}$, execute in each time unit. Next, there are Yx/5 time units in which five type J tasks execute - as noted in the previous section, each of these tasks has an R_1-value of approximately 1/6. Next, similarly to the critical path schedule described in Lemma 11, for $q-1 \geq k \geq 1$, there are $(1-c)x/2^k$ time units with three tasks executing per time unit. These tasks have R_1-values of $1/4 + 2\varepsilon_k$, $1/4 - 4\varepsilon_{k-1}$, and $1/4 - 4\varepsilon_{k-1}$. Following these time units there are (1-c)x time units with two A^1-tasks executing per time unit. These tasks have R_1-values of $1/2 - 2\varepsilon_0$ and $1/4 + 2\varepsilon_0$. Next, there are Yx/2 time units with two type J tasks executing per time unit - as noted in the previous section, these tasks have R_1-values of approximately 1/3. Finally, there are x-1 time units in which one A^1-task executes per time unit. Each of these tasks has an R_1-value exceeding 1/2. Note that in each of the first $[(1-c)/2^{q-1}]x/4$ time units the total requirement for resource 1 is $4(1/4 - 4\varepsilon_{q-1}) = 1 - 16\varepsilon_{q-1}$. During the execution of these time units the smallest resource requirement of any unexecuted A^i-task is approximately 1/6 (actually, just a little less than 1/6). But, ε_{q-1} was chosen such that $1/6 \gg 16\varepsilon_{q-1}$. This means that none of the A^1-tasks which execute later in the schedule can execute in these time units. Similar remarks can be made about each of the other time units in this schedule. This assures that the schedule we have described here is a valid list schedule. Thus, $G(A^1) = ([(1-c)/2^{q-1}]/4 + Y/5 + \sum_{k=1}^{q-1} (1-c)/2^k + (1-c) + Y/2 + 1)x - 1 = [(16+c)/10 - (1-c)/(20\ 2^{q-1}]x - 1$. Hence, $\omega_{cp} \geq x + s + 1 + sx[(16+c)/10 - (1-c)/(20\ 2^{q-1})] - s$. But, $x = 20s2^{q-1}$, so $\omega_{cp} > x[s(16+c)/10 + 1] - s^2$.

$\therefore \omega_{cp}/\omega_0 \geq (x[s(16+c)/10 + 1] - s^2)/(x+2s+1)$

$\text{limit}_{x \to \infty}\ \omega_{cp}/\omega_0 \geq (14s+m+9)/10.$ □

Lemma 13: If $3s + 1 \leq m$ and $m \geq 10$, then ω_{cp}/ω_0 can be arbitrarily close to $2 + 17s/10 - (3s+1)/m$.

Proof

Let $x = 0 \bmod 10m$, let $z = 3x - 1$ and let $\varepsilon = \beta_x$. Consider the task system S^* using these values of x, z and ε. For each i, $1 \leq i \leq s$, the A^i-tasks are assigned R_i-values equal to the R_1-values of the tasks in a set of z J-tasks. In addition to the usual tasks in S^* the following tasks are added to S^*:

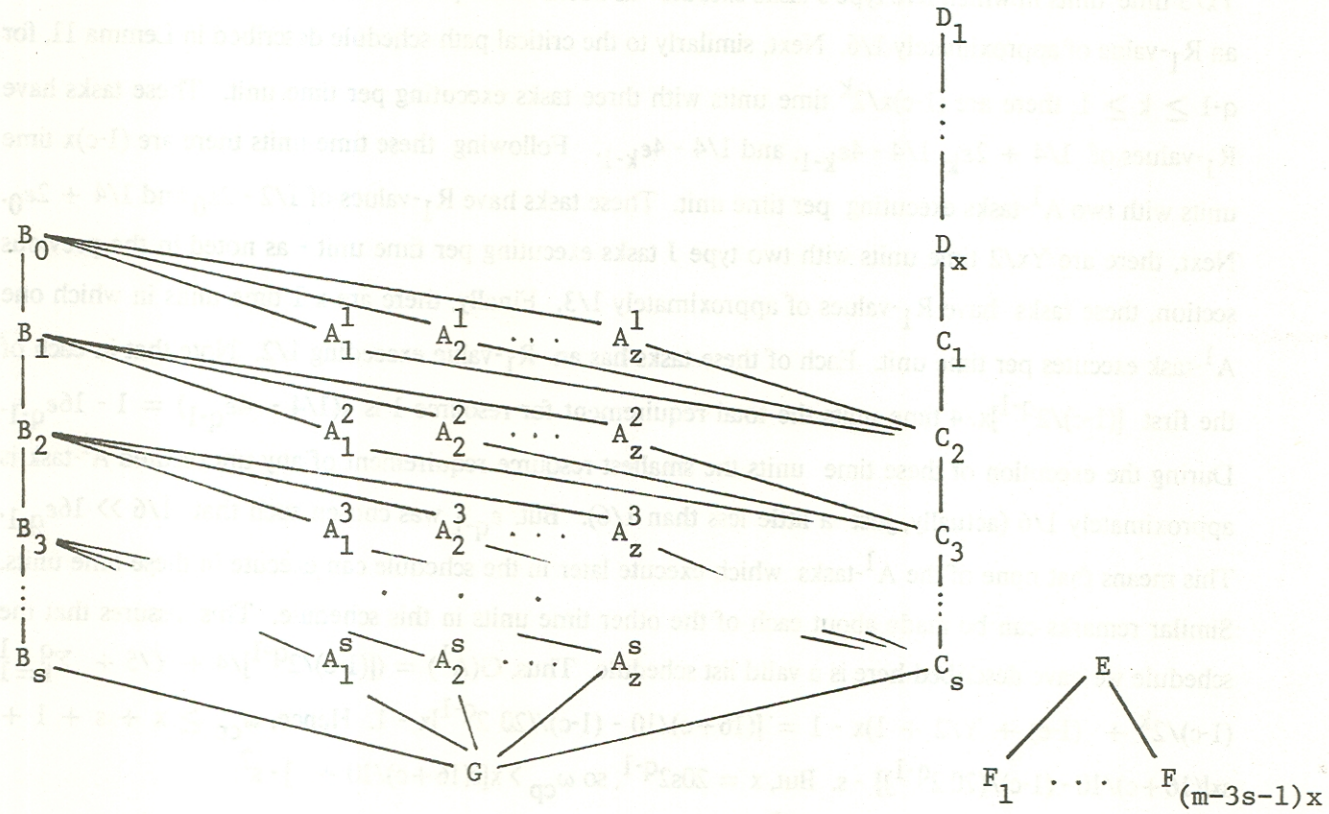
1. G, a task which requires no resources.
2. F_j for $1 \leq j \leq (m-3s-1)x$. These tasks require no resources.
3. E with $R_i(E) = 1$ for $1 \leq i \leq s$.

The following precedence constraints are also added to the system:

1. For $1 \leq j \leq z$, $A_j^s < G$.
2. $B_s < G$, and $C_s < G$.
3. For $1 \leq j \leq (m-3s-1)x$, $E < F_j$.

The precedence structure of this task system is shown in Figure 7.

FIGURE 7: The task system used in Lemma 13.



The non-zero resource requirements of these tasks are as follows:

Each D-task requires ε of resource 1.

B_0 requires all of resource 1.

B_i requires all of resource i, i > 0.

Each A^i-task requires a non-zero portion of resource i.

E requires all the resources.

G, the C-tasks and the F-tasks require no resources.

An optimal schedule for this system is: In the first s+2 time units execute the B-tasks and task E. In the next x time units the A-tasks, D-tasks and F-tasks are executed (1 D-task, m-3s-1 F-tasks and no more than 3s A-tasks per time unit). For each i, there are x-1 time units where three A^i-tasks execute and there is one time unit where two A^i-tasks execute. In the final s+1 time units execute the C-tasks followed by task G. Thus $\omega_o \leq s + 2 + x + s + 1 = x + 2s + 3$.

Now consider the following critical path schedule: Execute the D-tasks and tasks B_0 and C_1 in the first x+1 time units. In the next 17x/10 - 1 time units execute the A^1-tasks. Then, execute B_1 and C_2, followed by the A^2-tasks in the next 17x/10 - 1 time units, and so on, until B_s executes. Then execute E and G. In the final (m-3s-1)x/m time units execute the F-tasks. Thus, $\omega_{cp} \geq x + 1 + 17xs/10 + 1 + (m-3s-1)x/m > x[2 + 17s/10 - (3s+1)/m]$.

$\therefore \omega_{cp}/\omega_o \geq x[2 + 17s/10 - (3s+1)/m]/(x + 2s + 3)$

$\text{limit}_{x \to \infty}\ \omega_{cp}/\omega_o = 2 + 17s/10 - (3s+1)/m.$ □

Lemma 14: If $3s + 1 \leq m$ and $m < 10$, then ω_{cp}/ω_o can be arbitrarily close to $2 + 5s/3 - (8s/3 + 1)/m$.

Proof

The task system we describe here combines various aspects of the systems used in Lemmas 11 and 13. We use the task system structure from Lemma 13 (i.e. with the added tasks) and we assign the A-tasks resource requirements as was done in Lemma 11.

More formally, assume s and m are given. Let $c \in (1/2, 2/3)$ and let $q = \lceil \log[(1-c)/(2-3c)] \rceil$. Let x be an integer such that $x = 0 \bmod sm2^q$, let $z = [2+c]x$ and let $Y = 3c-2 + (1-c)/2^{q-1}$. Let $\varepsilon = \varepsilon_0 = 1/10^{q+2}$. Also, for $1 \leq k \leq q$, let $\varepsilon_k = 10\varepsilon_{k-1}$. Consider the task system S^* using these values of x, z and ε.

For each i, $1 \leq i \leq s$:

1. (1-c)x of the A^i-tasks have an R_i-value of $1/2 + \varepsilon_0$
   (1-c)x of the A^i-tasks have an R_i-value of $1/2 - 2\varepsilon_0$.
2. For $0 \leq k \leq q-1$,
   $(1-c)x/2^k$ of the A^i-tasks have an R_i-value of $1/2 + \varepsilon_k$.
   $(1-c)x/2^k$ of the A^i-tasks have an R_i-value of $1/4 + 2\varepsilon_k$.
   $(1-c)x/2^k$ of the A^i-tasks have an R_i-value of $1/4 - 4\varepsilon_k$.
3. Yx of the A^i-tasks have an R_i-value of $1/2 + \varepsilon_q$.
   Yx of the A^i-tasks have an R_i-value of $1/4 + 2\varepsilon_q$.
   Yx of the A^i-tasks have an R_i-value of $1/4 - 4\varepsilon_q$.

These are exactly the same specifications for the R_i-values of the A^i-tasks as given in Lemma 11.

In addition to the usual tasks in S^*, the following tasks are added to S^*:

1. G, a task which requires no resources.
2. F_j for $1 \leq j \leq (m-[2+c]s-1)x$. These tasks require no resources.
3. E with $R_i(E) = 1$ for $1 \leq i \leq s$.

The following precedence constraints are also added to the system:

1. For $1 \leq j \leq z$, $A_j^s \prec G$.
2. $B_s \prec G$, and $C_s \prec G$.
3. For $1 \leq j \leq (m-3s-1)x$, $E \prec F_j$.

An optimal schedule for this system is similar to that for the system used in the proof of the previous lemma. The B-tasks and task E are executed in the first s+2 time units. In the next x time units the A-tasks, D-tasks and F-tasks are executed. In each of those x time units, [2+c]s A-tasks, 1 D-task and (m-[2+c]-1) F-tasks execute. For each i, there are (1-c)x time units where two A^i-tasks execute and there are cx time units where three A^i-tasks execute. In the final s time units the C-tasks are executed. Thus, $\omega_o \leq x + 2s + 2$.

Now consider the following critical path schedule: Execute the D-tasks and tasks B_0 and C_1 in the first x+1 time units. In the next [3/2 + c/4]x time units execute the A^1-tasks (this follows from the proof of Lemma 11, where $G(A^1) = [3/2 + c/4]x$). Then execute B_1 and C_2, followed by the A^2-tasks in the next [3/2 + c/4]x time units, and so on, until B_s executes. Next execute E and G. Finally, execute the F-tasks in the final (m-[2+c]s-1)x/m time units. Thus, $\omega_{cp} \geq x + 1 + ([3/2 + c/4]x + 1)s + 1 + (m-[2+c]s-1)x/m > x[2 + 3s/2 - (2s+1)/m + cs(1/4 - 1/m)]$.

$\therefore \omega_{cp}/\omega_o \geq x[2 + 3s/2 - (2s+1)/m + cs(1/4 - 1/m)]/(x + 2s + 2)$

$\text{limit}_{c \to 2/3}\, \omega_{cp}/\omega_o \geq x[2 + 5s/3 - (8s/3 + 1)/m]/(x + 2s + 2)$

$\text{limit}_{x \to \infty}\, \omega_{cp}/\omega_o = 2 + 5s/3 - (8s/3 + 1)/m$ □

Acknowledgements: We would like to thank Ron Rivest for several helpful discussions which greatly improved the presentation of this paper. Also, the MACSYMA system at the MIT Laboratory for Computer Science was used to verify the entries in Table I.

5. References

1. Coffman, E.G., editor. Computer and Job-Shop Scheduling Theory. Wiley (1976).

2. Garey, M.R. and Graham, R.L., "Bounds on Scheduling With Limited Resources," Operating Systems Review, 7, 4(1973), 104-111.

3. Garey, M.R. and Graham, R.L., "Bounds for Multiprocessing Scheduling with Resource Constraints," SIAM Journal on Computing, 4(1975), 187-200.

4. Garey, M.R., Graham, R.L., Johnson, D.S., and Yao, A.C., "Resource Constrained Scheduling as Generalized Bin Packing," Journal of Combinatorial Theory, Series A. 3(1976), 257-298.

5. Garey, M.R. and Johnson, D.S., "Complexity Results for Multiprocessor Scheduling under Resource Constraints." Proceedings of the 8th Annual Princeton Conference on Information Sciences and Systems (1974) 168-172.

6. Graham, R.L., Lawler, E.L., Lenstra, J.K., and Rinnooy Kan, A.H.G., "Optimization and Approximation in Deterministic Sequencing and Scheduling: A Survey", BW 82/77, Mathematisch Centrum, Amsterdam, The Netherlands.

7. Johnson, D.S., Demers, A., Ullman, J.D., Garey, M.R. and Graham, R.L., "Worst-case Performance Bounds for Simple One-dimensional Packing Algorithms," SIAM Journal of Computing 3(1974), 299-325.

8. Lloyd, E.L., "List Scheduling Bounds for UET Systems with Resources," To appear in Information Processing Letters.

9. Yao, A.C., "Scheduling Unit-Time Tasks With Limited Resources," Proceedings of the Sagamore Computer Conference, Springer-Verlag, (1974) 17-36.

6. Appendix

In this section we prove two lemmas and a corollary about weighting function W_2. For convenience, the definition of W_2 is repeated here.

Definition: $W_2(\alpha) =$

0 if $\alpha = 0$

$10/100$ if $\alpha \in (0, .092]$

$15/100$ if $\alpha \in (.092, .136]$

$20/100$ if $\alpha \in (.136, .182]$

$25/100$ if $\alpha \in (.182, .204]$

$30/100$ if $\alpha \in (.204, .250]$

$40/100$ if $\alpha \in (.250, .296]$

$45/100$ if $\alpha \in (.296, .318]$

$50/100$ if $\alpha \in (.318, .364]$

$55/100$ if $\alpha \in (.364, .408]$

$60/100$ if $\alpha \in (.408, .500]$

1 if $\alpha \in (.500, 1]$

We have the following facts which follow by inspection from the definition of W_2.

Fact 1: If $\alpha \in (.092, .500]$, then $W_2(\alpha) \leq (1.64)\alpha$.

Fact 2: If $|B| = 3$ and $R_v(B) \leq 1$, then $W_2(R_v(B)) \leq 17/10$.

Fact 3: If $|B| = 2$ and $R_v(B) \leq 1$, then $W_2(R_v(B)) \leq 16/10$.

Fact 4: If $|B| = 2$ and $R_v(B) \leq .500$, then $W_2(R_v(B)) \leq 7/10$.

The following claim is useful in proving Lemma 4:

Claim A: If B is a set of tasks such that $R_v(B) \leq 1$ and $|B| \geq 2$ then $W_2(R_v(B)) \leq (|B|+14)/10$.

Proof

If $|B| \leq 3$ then the claim follows from Facts 2 and 3, so, assume that $|B| \geq 4$. Define the following two sets of tasks:

$Y = \{T \in B: R_v(T) > .500\}$

$X = \{T \in B: .092 < R_v(T) \leq .500\}$

Clearly, $W_2(R_v(B)) = W_2(R_v(Y)) + W_2(R_v(X)) + W_2(R_v(B\text{-}X\text{-}Y))$. Note that if $T \in Y$, then $W_2(R_v(T)) = 1$ and if $T \in B\text{-}X\text{-}Y$ then $W_2(R_v(T)) \leq 10/100$. Thus,

$W_2(R_v(B)) \leq |Y| + W_2(R_v(X)) + (|B| - |X| - |Y|)/10$.

Case 1: $|Y| = 0$

Then, $W_2(R_v(B)) = W_2(R_v(X)) + (|B| - |X|)/10$.

If $|X| \leq 2$, then since for each $T \in X$, $W_2(R_v(T)) \leq 60/100$, we have $W_2(R_v(B)) \leq (60/100)|X| + (|B|-|X|)/10 = 5|X|/10 + |B|/10 < (|B|+14)/10$.

If $|X| > 2$, then by Fact 1, $W_2(R_v(X)) \leq 1.64$, hence $W_2(R_v(B)) \leq 1.64 + (|B| - |X|)/10 \leq 1.64 + [|B| - 3]/10 < (|B|+14)/10$.

Case 2: $|Y| = 1$

Note that $R_v(X) < .500$ and

$$W_2(R_v(B)) \leq 1 + W_2(R_v(X)) + [|B|-|X|-1]/10 \qquad \text{(III)}$$

If $|X| = 0$, then from (III), $W_2(R_v(B)) \leq 1 + (|B|-1)/10 < (|B|+14)/10$.

If $|X| = 1$, then $W_2(R_v(X)) \leq 60/100$, so from (III), $W_2(R_v(B)) \leq 1 + 60/100 + (|B|-2)/10 = (|B|+14)/10$.

If $|X| = 2$, then by Fact 4, $W_2(R_v(X)) \leq 7/10$, so from (III),

$W_2(R_v(B)) \leq 1 + 7/10 + (|B|-3)/10 = (|B|+14)/10$.

If $|X| = 3$, then let $\max_v(X) = \max\{R_v(T): T \in X\}$.

If $\max_v(X) > .318$ then the other two tasks in X have R_v-values totaling less than .182, since $R_v(X) < .500$. Then at least one of these other two tasks must have an R_v-value less than .091. But, by definition each task in X has an R_v-value exceeding .092. Thus, $\max_v(X) \leq .318$.

If $\max_v(X) \in (.250, .318]$, then $W_2(\max_v(X)) = 45/100$. The other two tasks in X have R_v-values not exceeding .136 and .182 respectively, hence they have a total weight not exceeding 35/100. Thus, $W_2(R_v(X)) \leq 80/100$.

If $\max_v(X) \in (.092, .250]$, then $W_2(\max_v(X)) \leq 30/100$. The other two tasks in X have R_v-values not exceeding .204, hence they have a total weight not exceeding 50/100. Thus, $W_2(R_v(X)) \leq 80/100$.

Thus, if $|X| = 3$ then $W_2(R_v(X)) \leq 80/100$, hence $W_2(R_v(B)) \leq 1 + 80/100 + [|B|-4]/10 = (|B|+14)/10$.

If $|X| \geq 4$, then $W_2(R_v(X)) \leq 1.64R_v(X) \leq .82$. Then from (III), $W_2(R_v(B)) \leq 1 + .82 + [|B|-|X|-1]/10 \leq 1.82 + [|B|-5]/10 < (|B|+14)/10$. □

Lemma 4: If B is a legal set of tasks, then $W_2(B) \leq (|B|+14s)/10$.

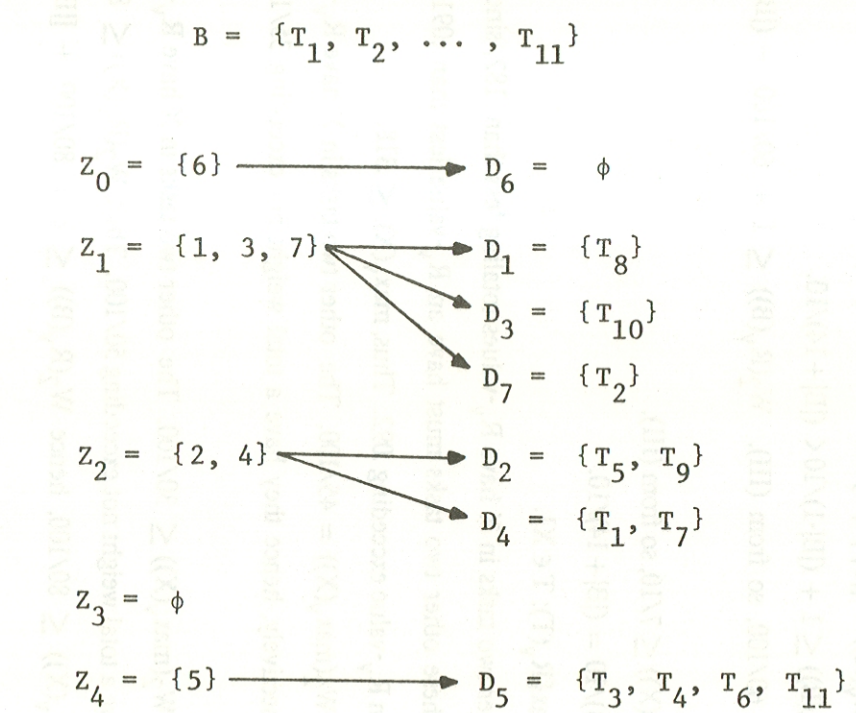
Proof

Partition the tasks in B into s sets $D_1, \ldots, D_s$, where $T \in D_v$ if and only if v is the minimum value such that $R_v(T) = R_{max}(T)$. Clearly, $W_2(B) = \Sigma_{v=1}^{s} W_2(R_v(D_v))$. Now partition the resources into sets $Z_0, \ldots, Z_n$, according to the sizes of the respective D_v sets. That is, resource v is placed into set $Z_{|D_v|}$ (Figure 8). Thus, $W_2(B) = \Sigma_{j=0}^{n} (\Sigma_{v \in Z_j} W_2(R_v(D_v)))$. From the definition of W_2 it follows that for each $v \in Z_0$, $W_2(R_v(D_v)) = 0$ and for each $v \in Z_1$, $W_2(R_v(D_v)) \leq 1$. Moreover, from Claim A, it follows that for each $j \geq 2$, and each $v \in Z_j$, $W_2(R_v(D_v)) \leq (j + 14)/10$ and $\Sigma_{v \in Z_j} W_2(R_v(D_v)) = [(j+14)/10]|Z_j|$. Thus, $W_2(B) \leq |Z_1| + \Sigma_{j=2}^{n} [(j + 14)/10]|Z_j| = \Sigma_{j=1}^{n} j|Z_j|/10 + \Sigma_{j=1}^{n} 14|Z_j|/10 - |Z_1|/2$. But, the Z_i's are a partition of the resources, so $\Sigma_{j=1}^{n} |Z_j| \leq s$. Moreover, that partition is based on a partition of the tasks such that $\Sigma_{j=1}^{n} j\,|Z_j| \leq |B|$. Also, $|Z_1| \geq 0$.

FIGURE 8: Partitioning the resources in Lemma 4. An example with 7 resources and a set B of 11 tasks.

	Resource						
	1	2	3	4	5	6	7
T_1 : [	-	-	-	(.7)	-	-	-]
T_2 : [	-	-	-	-	-	-	(.6)]
T_3 : [	-	-	-	-	(.5)	.2	-]
T_4 : [	-	-	-	-	(.1)	.1	-]
T_5 : [	.1	(.3)	-	-	-	-	-]
T_6 : [	-	-	-	-	(.1)	-	-]
T_7 : [	-	-	-	(.2)	-	-	.2]
T_8 : [	(.8)	-	-	.1	-	-	-]
T_9 : [	-	(.6)	-	-	-	-	-]
T_{10} : [	-	-	(.7)	-	-	.4	-]
T_{11} : [	-	-	-	-	(.3)	-	-]

These are the resource vectors associated with the tasks. A zero requirement is shown as a dash. The largest requirement for each task is circled.



Resource partition Task partition

$\therefore W_2(B) \leq |B|/10 + 14s/10 - 0/10 = (|B| + 14s)/10$ □

Corollary 2: Given a set of tasks $Y \subseteq T$, then $W_2(Y) \leq (|Y| + 14s\omega_o)/10$.

Let $B_1, \ldots, B_{\omega_o}$ be the time units of an optimal schedule restricted to the tasks in Y. By Lemma 4, each $W_2(B_i) \leq (|B_i| + 14s)/10$. Thus $W_2(Y) = \Sigma_{i=1}^{\omega_o} W_2(B_i) \leq \Sigma_{i=1}^{\omega_o} [|B_i| + 14s]/10 = 14s\omega_o/10 + \Sigma_{i=1}^{\omega_o} |B_i|/10 = (|Y| + 14s\omega_o)/10$. □

Lemma 5: Weighting function W_2 has Property B.

Proof

By Lemma 1 it is sufficient to show that W_2 has Property A. Consider a task T' and a nonempty set of tasks B such that $R_{max}(T) \geq R_{max}(T')$ for $T \in B$, and $R_{max}(T') > 1 - R_{max}(B)$. We want to show that $W_2(B) \geq 1$.

If $|B| = 1$, the result is immediate, so assume that $|B| \geq 2$. Let $\min(B) = \min\{R_{max}(T): T \in B\}$. If there is only one resource in the task system, then min(B) is the smallest resource requirement of any task in B. Given a time unit B, it is possible to compute a lower bound for $W_2(B)$ based on |B|, min(B) and $R_{max}(B)$. In particular, Table I gives various combinations of |B|, min(B) and $R_{max}(B)$, each of which implies that $W_2(B) \geq 1$.

Now consider the possible values of $W_2(T')$. If $W_2(T') \geq 50/100$, then for each $T \in B$, $W_2(T) \geq 50/100$. Since $|B| \geq 2$, we have $W_2(B) \geq 1$. If $W_2(T') = 10/100$, then $0 < R_{max}(T') \leq .092$. But this implies that $R_{max}(B) > .908$ and $\min(B) > 0$ hence from Table I, $W_2(B) \geq 1$. If $R_{max}(T) = 0$, then $R_{max}(B) \geq 1$, hence $W_2(B) \geq 1$.

There are six remaining possibilities for $W_2(T')$: 15/100, 20/100, 25/100, 30/100, 40/100, and 45/100. Associated with each of these weights there is a range $(\alpha_1, \alpha_2]$ in which $R_{max}(T')$ must lie. Moreover, in each instance it follows that $\min(B) > \alpha_1$ and that $R_{max}(B) > 1 - \alpha_2$. For each $(\alpha_1, \alpha_2]$ pair, an examination of the "relevant" entries in Table I, shows that $W_2(B) \geq 1$ in all instances. A guide to the "relevant" entries of Table I is given in Table II. In Table II, for each of the six possible values of $W_2(T')$, we give the values α_1, α_2, the subsequent lower bounds on min(B) and $R_{max}(B)$ and the entries of Table I that need to be examined. Note that entries are not listed for each size of |B| in every case. In particular, for each $W_2(T')$ possibility, only one entry of the form (|B|, min(B), 0) is given. Such an entry implies that $W_2(B) \geq |B|\, W_2(\min(B)) \geq 1$. Thus, for any larger |B|, we also have $W_2(B) \geq 1$.

For example, when $W_2(T') = 25/100$, $R_{max}(T') \in (.182, .204]$. Thus, $\min(B) > .182$ and $R_{max}(B) > 1 - .204 = .796$. If $|B| \geq 4$, it follows from |B| and $\min(B) > .182$ that $W_2(B) \geq 4\, W_2(\min(B)) \geq 4\,(25/100) = 1$. If $|B| < 4$, the entries (2, 0 , .750) and (3, .182, .750) in Table I indicate that $W_2(B) \geq 1$. □

Table I

$\|B\|$	min(B)	$R_{max}(B)$	$\|B\|$	min(B)	$R_{max}(B)$	$\|B\|$	min(B)	$R_{max}(B)$
2	0	.750	4	0	.820	7	0	.868
2	.250	.704	4	.136	.818	7	.092	0
2	.296	.682	4	.182	0			
						8	0	.870
3	0	.820	5	0	.864	8	.092	0
3	.136	.772	5	.136	0			
3	.182	.750				9	0	.872
3	.250	0	6	0	.866	9	.092	0
			6	.092	.862			
			6	.136	0	10	0	0

An entry **(i, x, y)** in this table is interpreted as follows: If B is a set of tasks such that $|B| = i$, $\min(B) > x$, and $R_{max}(B) > y$, **then** $W_2(B) > 1$.

Table II

$W_2(T')$	(α_1, α_2)	min(B)>	$R_{max}(B)$>	Relevant Entries
15/100	(.092, .136]	.092	.864	(2, 0, .750), (3, 0, .820), (4, 0, .820), (5, 0, .864), (6, .092, .862), (7, .092, 0)
20/100	(.136, .182]	.136	.818	(2, 0, .750), (3, .136, .772), (4, .136, .818), (5, .136, 0)
25/100	(.182, .204]	.182	.796	(2, 0, .750), (3, .182, .750), (4, .182, 0)
30/100	(.204, .250]	.204	.750	(2, 0, .750), (3, .182, .750), (4, .182, 0)
40/100	(.250, .296]	.250	.704	(2, .250, .704), (3, .250, 0)
45/100	(.296, .318]	.296	.682	(2, .296, .682), (3, .250, 0)