JOURNAL OF COMBINATORIAL THEORY (A) 21, 257–298 (1976)

Resource Constrained Scheduling as Generalized Bin Packing

M. R. GAREY, R. L. GRAHAM, AND D. S. JOHNSON

Bell Laboratories, Murray Hill, New Jersey 07974

AND

ANDREW CHI-CHIH YAO

University of Illinois, Urbana, Illinois

Communicated by the Managing Editors

Received May 27, 1975

1. INTRODUCTION

The problem we consider in this paper may be described abstractly as follows. We are given a finite set $L = \{x_1, x_2, \ldots, x_n\}$, a partial order $\prec$ defined on L, and a finite set $\{R_j : 1 \leqslant j \leqslant s\}$ of functions $R_j : L \to [0, 1]$. We wish to find an ordered partition $P = \langle B_1, B_2, \ldots, B_m \rangle$ of L into nonempty sets such that:

(i) For all i and j, $1 \leqslant i \leqslant m$, $1 \leqslant j \leqslant s$, $\sum_{x \in B_i} R_j(x) \leqslant 1$.

(ii) If $x, y \in L$, $x \prec y$ and $x \in B_j$, $y \in B_k$ then $j < k$.

(iii) For any partition P' satisfying (i) and (ii), $| P | \leqslant | P' |$.

This problem is a special case of the general multiprocessor scheduling problem with resource constraints [4, 5]. We may think of the $x \in L$ as unit-time tasks to be executed, the partial order as a precedence constraint, and the functions R_j as giving resource usages for the tasks (normalized so that all resource bounds equal 1). The desired partition then furnishes us with a minimum time schedule, with the tasks of B_i all being concurrently executed during the ith time slice (the assumption being that there are sufficiently many processors available to accommodate any number of tasks we might wish to execute simultaneously).

If $\prec$ is empty, i.e., there are no precedence constraints, the problem may also be thought of as a form of "multidimensional" bin packing. In the one-dimensional bin packing problem [10, 11, 13], we are given a list L of real numbers between 0 and 1 which we wish to assign to unit

capacity "bins," so that no bin receives numbers totaling more than 1, and a minimum number of bins is used. Here, the items packed can be thought of as vectors $\langle R_1(x), R_2(x),..., R_s(x)\rangle$ and we wish to assign them to bins so that the vector sum of the items received by any bin is dominated by $\langle 1, 1,..., 1\rangle$.

For convenience, we shall borrow some of the more useful terminology from each of these two interpretations. L will be called a *set of tasks* and will interchangeably be thought of as a list $L = \langle x_1, x_2,..., x_n\rangle$ with the order of the list determined by the indexing of the set. The functions R_j will be called *resources* and for each task $x \in L$, $R_j(x)$ will be the *R_j-resource usage* of x. The requirement that $\sum_{x\in B_i} R_j(x) \leqslant 1$ will be called a *resource constraint*. Similarly, the partial order $\prec$ and condition (ii) give the *precedence constraints.* In an ordered partition $P = \langle B_1, B_2,..., B_m\rangle$ obeying (i) and (ii), the sets B_i will be called *bins* and P itself will be called a *packing of L.* An *optimal packing* will be one which in addition satisfies (iii). Finally, we define L^* to be the value of $| P |$, the number of sets of P, for an optimal packing P of L.

Now even in the one-dimensional case, the problem of finding an optimal packing P can involve the solution of the *NP*-complete PARTITION problem (see [1], [13]) and hence is likely to be computationally intractable for even relatively short lists L. Thus the approach that has been taken in the literature for this and related problems (e.g., [2, 3, 6, 7, 9, 11–13, 16]) has been to relax condition (iii), and use efficient algorithms which, while not guaranteeing an optimal packing, do generate packings which are never too far from optimal. One of the principal algorithms of this type which has been studied is FIRST FIT (FF) (see [11, 13]). In the one-dimensional case this simple heuristic proceeds as follows. Assign the x_i to bins in order, first assigning x_1 to bin B_1, and, thereafter, assuming x_i has been assigned, assigning x_{i+1} to the bin B_j of minimal index to whose contents x_{i+1} can be added without violating the bin capacity constraint.

The worst-case behavior of FF has been extensively analyzed in the one-dimensional, no precedence constraint case. Let FF(L) be the value of $| P |$ when P is the packing of L generated by FF. In [13] it is shown that $\mathrm{FF}(L) \leqslant (17/10)\, L^* + 2$, for any list L (assuming $s = 1$, $\prec = \varnothing$), and that arbitrarily long lists L exist with $\mathrm{FF}(L) \geqslant (17/10)\, L^* - 2$. The algorithm behaves even better when the list is in nonincreasing order, i.e., $R_1(x_1) \geqslant R_1(x_2) \geqslant \cdots \geqslant R_1(x_n)$. In this case we have $\mathrm{FF}(L) \leqslant (11/9)\, L^* + 4$, with arbitrarily long lists existing for which $\mathrm{FF}(L) = (11/9)\, L^*$.

In this paper we generalize the algorithm FIRST FIT so that it may apply to the more complicated multidimensional case with precedence constraints, and prove corresponding bounds on FF(L) in the more general

situation. The generalized algorithm proceeds by constructing the bins $B_1, B_2, \ldots$, each in turn, proceeding inductively as follows.

Suppose all bins preceding B_i have been constructed and there are still some tasks remaining to be packed. Let L_i be the list obtained from L by deleting all tasks already packed, while retaining the relative order of the remaining tasks in the list. (Initially, $L_1 = L$.) We now further modify L_i by deleting all tasks x which have *predecessors* in L_i, i.e., those $x \in L_i$ for which there is a $y \in L_i$ with $y \prec x$. Since $\prec$ is a partial order and so contains no cycles, some tasks must still remain in L_i, which we can write as $\langle y_1, y_2, \ldots, y_k \rangle$. Assign y_1 to B_i. In general, if y_j can be added to B_i without violating any resource constraints, do so. Otherwise delete it from the list and go on to y_{j+1}. Once every y_j has either been assigned or deleted, the packing of B_i is complete. If any tasks now still remain unpacked, we then proceed to pack B_{i+1}. This process is continued until all tasks are packed.

The reader may verify that, when restricted to the case of $s = 1$ and $\prec = \varnothing$, this generalized algorithm will indeed yield the same packing (although generated in a different way) as the original FF algorithm.

We also consider two variants of the generalized FF algorithm, both of which operate by applying FF to the list after the list has been specially preordered. FIRST FIT DECREASING (FFD) is a generalization of the corresponding algorithm in the one-dimensional case. For each $x \in L$, define

$$R_{\max}(x) = \max\{R_j(x): 1 \leqslant j \leqslant s\}. \tag{1}$$

FFD reorders L into the form $\langle y_1, y_2, \ldots, y_n \rangle$, where $R_{\max}(y_1) \geqslant R_{\max}(y_2) \geqslant \cdots \geqslant R_{\max}(y_n)$ and, if $R_{\max}(y_i) = R_{\max}(y_{i+1})$, then y_i precedes y_{i+1} in the original ordering of L.[1]

The LEVEL algorithm (FFL) reorders L according to *precedence level*. A *chain of tasks* of L is a sequence $\langle z_1, z_2, \ldots, z_k \rangle$ of tasks where $z_1 \prec z_2 \prec \cdots \prec z_k$. The *head* of the chain is z_1, and the *length* of the chain is k. The *level* of a task $x \in L$ is the length of the longest chain which has x as head. FFL reorders L into the form $\langle y_1, y_2, \ldots, y_n \rangle$ where level $(y_1) \geqslant$ level $(y_2) \geqslant \cdots \geqslant$ level (y_n) and, if level $(y_i) =$ level (y_{i+1}), then y_i comes before y_{i+1} in the original ordering (if $\prec$ is empty, FFL will leave the original ordering of L unchanged since all tasks $x \in L$ have level $(x) = 1$.)

One might remark at this point that the LEVEL algorithm yields optimal packings (see [8]) if $s = 0$ (no resource constraints) and $\prec$ is a *forest*,

[1] A slightly different form of "decreasing" order is discussed in [16]. The present definition yields worst-case behavior which is at least as good and is easier to deal with.

i.e., each task $x \in L$ has at most one immediate predecessor[2] y, i.e., $y \prec x$ and, if $z \prec x$ then $z = y$ or $z \prec y$. However, our primary interest is in the case $s \geqslant 1$. This paper will be devoted to determining the worst-case behavior of the algorithms under consideration as a function of the number s of resources. To this end, define

$$\widehat{\mathrm{FF}}(k) = \max\{\mathrm{FF}(L): L^* = k\}, \tag{2}$$

with similar definitions for $\widehat{\mathrm{FFD}}$ and $\widehat{\mathrm{FFL}}$. We consider cases both with and without precedence constraints, and our results can be summarized as follows.

(A) If $s \geqslant 1$ and $\prec$ is empty then

(1) $\lim_{k\to\infty}(\widehat{\mathrm{FF}}(k)/k) = s + 7/10$,

(2) $s + ((s-1)/s(s+1)) \leqslant \lim_{k\to\infty}(\widehat{\mathrm{FFD}}(k)/k) \leqslant s + 1/3$

(with somewhat stronger lower bounds for $s \leqslant 3$).

(B) If $s \geqslant 1$ and $\prec$ is any partial order, then

(1) $\lim_{k\to\infty}(\widehat{\mathrm{FF}}(k)/k) = (s/2)\,k + (s/2) + 1$,

(2) $(1.69)\,s + 1 \leqslant \lim_{k\to\infty}(\widehat{\mathrm{FFD}}(k)/k) \leqslant (17/10)\,s + 1$,

(3) $\lim_{k\to\infty}(\widehat{\mathrm{FFL}}(k)/k) = (17/10)\,s + 1$.

We point out that $A(1)$ is a straightforward generalization of the corresponding result for the one-dimensional case, and many of the other bounds are also seen to be related. In fact, many of the ideas occurring in our proofs have roots which come from the one-dimensional case, and we begin in Section 2 by presenting improved versions of these earlier results (cf. [13]). In Sections 3 and 4, these are then used to prove the results in (A) and (B). The concluding Section 5 discusses results for certain more restricted cases, such as requiring $R_{\max}(x) \leqslant 1/k$ for all $x \in L$, and when the additional constraint is added that for any bin B_i, the number $|B_i|$ of tasks in B_i cannot exceed some fixed upper bound.

2. The Weighting Function

In many of the upper bound proofs which follow, we shall make use of a specially designed weighting function W: $[0, 1] \to [0, 8/5]$. This function is derived from that used in [13], but has been modified so as to

[2] This is also true if each $x \in L$ has at most one immediate successor.

yield somewhat stronger results. It is defined as follows (Fig. 1 provides a pictorial description).

$$
\begin{aligned}
W(\alpha) &= (6/5)\,\alpha && \text{for } 0 \leqslant \alpha \leqslant 1/6,\\
&= (9/5)\,\alpha - 1/10 && \text{for } 1/6 < \alpha \leqslant 1/3,\\
&= (6/5)\,\alpha + 1/10 && \text{for } 1/3 < \alpha \leqslant 1/2,\\
&= (6/5)\,\alpha + 4/10 && \text{for } 1/2 < \alpha \leqslant 1.
\end{aligned}
$$

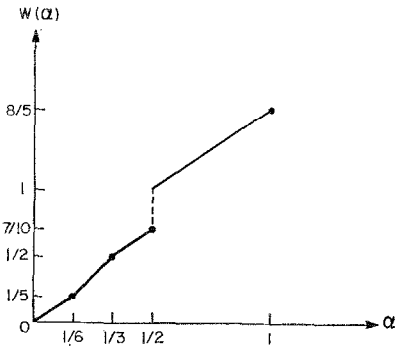

FIGURE 1

Suppose L is a list of tasks and we are given a map $R\colon L \to [0, 1]$ which is not identically 0. Extend R to sets of tasks S by setting $R(S) = \sum_{x \in S} R(x)$. We then have the following useful (though perhaps at this point, obscure) lemmas about W.

LEMMA 1. *If $B \subseteq L$ and $R(B) \leqslant 1$ then*

(a) $\sum_{b \in B} W(R(b)) \leqslant 17/10$,

(b) *If* $\{R(b)\colon b \in B\} \subseteq [0, 1/2]$ *then* $\sum_{b \in B} W(R(b)) \leqslant 3/2$.

LEMMA 2. *If $B \subseteq L$ and $R(B) > 1$ then*

(a) $\sum_{b \in B} W(R(b)) > 6/5$,

(b) *If* $\max\{R(b)\colon b \in B\} > 1/2$ *then* $\sum_{b \in B} W(R(b)) > 8/5$.

LEMMA 3. *If $0 \leqslant \alpha < 1/2$ and $B = \{b_1, b_2, \ldots, b_m\}$, $m \geqslant 2$, has $R(b_1) \geqslant R(b_2) > \alpha$ and $R(B) \geqslant 1 - \alpha$, then $\sum_{b \in B} W(R(b)) \geqslant 1$.*

LEMMA 4. *If $0 \leqslant \alpha < 1/2$ and $B = \{b_1, b_2, \ldots, b_m\}$, $m \geqslant 1$, has $R(b_1) \geqslant R(b_2) \geqslant \cdots \geqslant R(b_m) > \alpha$ and $\sum_{b \in B} W(R(b)) = 1 - \beta$ for some $\beta > 0$, then either*

(a) $m = 1$ *and* $R(b_1) \leqslant 1/2$, *or*

(b) $R(B) \leqslant 1 - \alpha - (5/6)\,\beta$.

LEMMA 5. *Suppose* $Y \subseteq L$ *and* $\mathscr{B} = \{B_1, B_2, \ldots, B_t\}$ *is a partition of* Y *into disjoint nonempty sets such that for all integers* i *and* j *with* $1 \leqslant i < j \leqslant t$, $y \in B_j$ *implies* $R(y) > 1 - R(B_i)$. *Then*

$$\sum_{y \in Y} W(R(y)) > t - 1 + \sum_{i=1}^{t} \max\left\{0, \left[\sum_{b \in B_i} W(R(b)) - 1\right]\right\}.$$

These five lemmas are all based on similar claims made in the proof of [13, Theorem 2.2], and are proved in much the same way, although the weighting function W is slightly different, and in some cases the claims made here are a bit stronger. The interested reader is referred to the Appendix for the proofs. That the current lemmas are indeed more powerful than those in [13] can be seen from the fact that we can use them to improve Theorem 2.2 of that paper. That result stated that for all lists L with one resource and an empty partial order, $\mathrm{FF}(L) \leqslant (17/10)\,L^* + 2$. We have the following:

THEOREM 1. *If* L *is a list of tasks in a system with* $s = 1$ *and* $\prec$ *is empty then*

$$\mathrm{FF}(L) < (17/10)\,L^* + 1.$$

Proof. We can think of an optimal packing of L as a partition of L into L^* sets, each set B of which has $R_1(B) \leqslant 1$. By Lemma 1, we have

$$\sum_{x \in L} W(R_1(x)) \leqslant (17/10)\,L^*. \tag{3}$$

On the other hand, we can think of the FF packing of L as a partition of L into $\mathrm{FF}(L)$ sets, say $\{B_1, B_2, \ldots, B_{\mathrm{FF}(L)}\}$, with the FF rule assuring that for all i and j, $1 \leqslant i < j \leqslant \mathrm{FF}(L)$, $b \in B_j$ implies $R_1(b) > 1 - R_1(B_i)$. Thus, by Lemma 5,

$$\sum_{x \in L} W(R_1(x)) > \mathrm{FF}(L) - 1. \tag{4}$$

The theorem follows from (3) and (4). ∎

Note that if L^* is a multiple of 10 then in fact $\mathrm{FF}(L) \leqslant (17/10)\,L^*$. More generally, the conclusion of Theorem 1 can be written as $\mathrm{FF}(L) \leqslant \lceil (17/10)\,L^* \rceil$.

3. The Case of $\prec$ Empty

In this section we examine the situation in which there are no precedence constraints on the tasks of L.

THEOREM 2. *If L is any list in a system with $s \geqslant 1$ resources and $\prec$ is empty, then*

$$\mathrm{FF}(L) \leqslant (s + 7/10)\, L^* + 5/2. \tag{5}$$

Proof. The theorem will be proved by induction on s. Theorem 1 shows that it is true for $s = 1$. Suppose (5) holds for the values $1, 2, \ldots, s-1$, but *not* for s. In that case we have the following.

CLAIM 1. *There is a list L with*

$$(s + 7/10))\, L^* + 5/2 < \mathrm{FF}(L) \leqslant (s + 7/10)\, L^* + 7/2. \tag{6}$$

To verify this claim, let $\mathscr{L} = \{L\colon \mathrm{FF}(L) > (s + 7/10)\, L^* + 5/2\}$, and let $l^* = \min\{L^*\colon L \in \mathscr{L}\}$. Now choose an $L \in \{\mathscr{L}\colon L^* = l^*\}$ for which $\mathrm{FF}(L)$ is minimal. This L will be our desired list. For suppose $\mathrm{FF}(L) > (s + 7/10)\, L^* + 7/2$. Consider the list $\bar{L}$ obtained from L by deleting those tasks contained in the first bin of the FF packing of L. Clearly, $\mathrm{FF}(\bar{L}) = \mathrm{FF}(L) - 1$, and $\bar{L}^* \leqslant L^*$. Thus $\bar{L} \in \mathscr{L}$ and hence $\bar{L}^* = l^*$, contradicting the assumption that L was a list of this type with minimal $\mathrm{FF}(L)$. Thus, we must have $\mathrm{FF}(L) \leqslant (s + (7/10))\, L^* + 7/2$ and L is our desired list. ∎

Let L be a list as specified by the preceding claim. We shall now focus on the FF packing of L, and, partitioning the nonempty bins B into sets, we shall prove a number of preliminary results about the partition (Claims 2 through 7). Using these, it will then be possible to proceed with a weighting function proof analogous to that of Theorem 1. The partition is defined as follows:

$$\Delta = \{B\colon R_j(B) > 1/2 \text{ for at least two distinct values of } j,\ 1 \leqslant j \leqslant s\},$$
$$S_i(1) = \{B\colon |B| = 1,\ R_i(B) > 1/2 \text{ and } R_j(B) \leqslant 1/2 \text{ for } j \neq i\},\ 1 \leqslant i \leqslant s,$$
$$S_i(2) = \{B\colon |B| \geqslant 2,\ R_i(B) > 2/3 \text{ and } R_j(B) \leqslant 1/2 \text{ for } j \neq i\},\ 1 \leqslant i \leqslant s,$$
$$X_i = \{B\colon |B| \geqslant 2,\ 1/2 < R_i(B) \leqslant 2/3 \text{ and } R_j(B) \leqslant 1/2 \text{ for } j \neq i\},\ 1 \leqslant i \leqslant s,$$
$$X_0 = \{B\colon R_i(B) \leqslant 1/2 \text{ for all } i,\ 1 \leqslant i \leqslant s\}.$$

That these $3s + 2$ sets of bins do indeed form a partition of the FF packing is clear. Define

$$S_i = S_i(1) \cup S_i(2), \qquad 1 \leqslant i \leqslant s, \qquad \text{and} \qquad X = \bigcup_{i=0}^{s} X_i .$$

We relabel the resources if necessary so that $| S_1 | \geqslant | S_2 | \geqslant \cdots \geqslant | S_s |$. Figure 2 gives a schematic of the partition, with the bins arranged in a

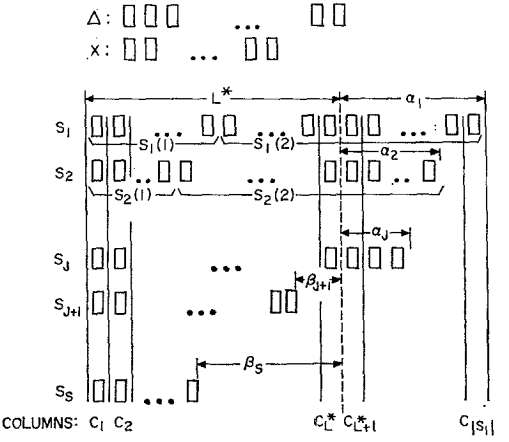

FIGURE 1

pattern which will be quite useful in what follows. The bins of Δ and X are set apart with the bins in the S_i more carefully arranged. Each of the sets S_i is laid out in a row, with all the bins of $S_i(1)$ preceding all the bins of $S_i(2)$. The rows are left justified so that $| S_1 |$ columns of bins are created, the jth column C_j, $1 \leqslant j \leqslant | S_1 |$, consisting of the jth bin in each S_i for which $| S_i | \geqslant j$. Let $J = \max\{j: | S_j | \geqslant L^*\}$. We define α_i and β_i, $1 \leqslant i \leqslant s$, as follows.

$$\text{If } i \leqslant J \text{ then } \alpha_i = | S_i | - L^*, \beta_i = 0;$$
$$\text{If } i > J \text{ then } \alpha_i = 0, \beta_i = L^* - | S_i |.$$

We now proceed to verify a number of claims about our partition of the FF packing, many of which will make use of the diagram in Fig. 2.

CLAIM 2. $| X_i | \leqslant 1$ *for* $0 \leqslant i \leqslant s$, *and hence*, $| X | \leqslant s + 1$.

Proof. If $| X_0 | > 1$ then X_0 would have to contain at least two bins, one of higher index than the other. But then the first task placed in the higher indexed bin would have fit into the lower indexed bin, since it would

have size no more than 1/2 in any resource. The fact that the task did not go there violates the FF rule. If $|X_i| > 1$ for some i, $1 \leqslant i \leqslant s$, then let B_j and B_k be two bins in X_i with $j < k$ and let a and b be two tasks in bin B_k. The only way a and b can have been excluded from bin B_j (since they must have size at most 1/2 in all resources except R_i) is if $R_i(a) + R_i(B_j) > 1$ and $R_i(b) + R_i(B_j) > 1$. But then both $R_i(a)$ and $R_i(b)$ exceed 1/3, so that $R_i(B_k) \geqslant R_i(a) + R_i(b) > 2/3$, a contradiction to the definition of X_i. This proves the claim. ∎

CLAIM 3.

$$\sum_{i=1}^{s} |S_i(1)| \leqslant L^*. \tag{7}$$

Proof. If there were more than L^* one-task bins, then at least two of the single tasks would have had to come from the same bin in an optimal packing of L. Hence, the task in the higher indexed bin in the FF packing would have been illegally placed. ∎

CLAIM 4.

$$|X| + |\Delta| \geqslant \sum_{i=1}^{s} \beta_i. \tag{8}$$

Proof. This will be the only place in which the induction hypothesis (that Theorem 2 holds for any number of resources less than s) is used. Suppose $|X| + |\Delta| < \sum_{i=1}^{s} \beta_i$. Then we must have $\sum_i \beta_i > 0$ and, hence, there are some resources for which $|S_i| < L^*$ and so $J < s$. Let us create a new partition of the bins of the FF packing by assigning the bins in $\Delta \cup X$ to the sets S_i. To be specific, let

$$S_i' = S_i \cup \{B \in \Delta \cup X : i = \max\{j : R_j(B) > 1/2\}\}, \qquad 1 \leqslant i \leqslant s.$$

We then have

$$\begin{aligned} \sum_{i=J+1}^{s} |S_i'| &\leqslant \sum_{i=J+1}^{s} |S_i| + |\Delta| + |X| \\ &\leqslant (s - J) L^* - \sum_{i=1}^{s} \beta_i + |\Delta| + |X| \leqslant (s - J) L^*. \end{aligned} \tag{9}$$

Furthermore, all bins in $\bigcup_{i=1}^{J} S_i'$ still have resource levels of no more than 1/2 in resources R_{J+1} through R_s. Consider the list $\bar{L}$ obtained from L by deleting all tasks in the bins in $\bigcup_{i=J+1}^{s} S_i'$. This list must be packed

by FF into a collection of bins identical to $\bigcup_{i=1}^{J} S_i' \cup X_0$, and moreover, resources R_{J+1} through R_s can have had no effect on the packing, since all tasks and bins had resource usages of at most 1/2 in these resources. Since $J < s$, the induction hypothesis therefore applies and we have

$$\begin{aligned} \mathrm{FF}(\bar{L}) = \sum_{i=1}^{J} |S_i'| + |X_0| &\leqslant (J + 7/10)\,\bar{L}^* + 5/2 \\ &\leqslant (J + 7/10)\,L^* + 5/2. \end{aligned}$$

But then

$$\begin{aligned} \mathrm{FF}(L) &= \mathrm{FF}(\bar{L}) + \sum_{i=J+1}^{s} |S_i'| \\ &\leqslant (J + 7/10)\,L^* + 5/2 + (s - J)\,L^* \\ &= (s + 7/10)\,L^* + 5/2, \end{aligned}$$

which contradicts our assumption on FF(L). This proves the claim. ∎

The next claim we make will actually be used like a lemma when evaluating overall resource usage for the bins in a column of the diagram.

CLAIM 5. *Let $I \subseteq \{1, 2, \ldots, s\}$ be nonempty, and suppose that for each $i \in I$ we are given a bin $B_i \in S_i$. Then there is an $r \in I$ such that*

$$\sum_{i \in I} \sum_{j \in I - \{r\}} R_j(B_i) \geqslant |I| - 1. \tag{10}$$

Proof. The claim is immediate for $|I| = 1$. Suppose that for some k, $2 \leqslant k < s$, the claim holds for all I with $|I| = k - 1$. Consider an example for which $|I| = k$. Let u and v be distinct elements of I. We must have either $R_u(B_u) + R_u(B_v) > 1$ or $R_v(B_u) + R_v(B_v) > 1$, since otherwise no task from the higher indexed bin could have been prevented from going into the lower indexed bin. (Resources other than R_u and R_v would be of no avail since both bins have level at most 1/2 for any such resource). We may assume without of loss of generality that the first case holds. But then, since $I' = I - \{u\}$ has $|I'| = k - 1$, the induction hypothesis implies that there exists an $r \in I'$ such that

$$\sum_{i \in I'} \sum_{j \in I' - \{r\}} R_j(B_i) \geqslant k - 2$$

and hence,

$$\begin{aligned} \sum_{i \in I} \sum_{j \in I - \{r\}} R_j(B_i) &\geqslant k - 2 + R_u(B_u) + R_u(B_v) \\ &\geqslant k - 1 = |I| - 1. \end{aligned}$$

This proves the claim. ∎

COROLLARY 1. *If I and $\{B_i : i \in I\}$ are as in Claim 5, and, in addition, there is a δ, $0 < \delta \leqslant 1/2$, such that $R_i(B_i) > 1 - \delta$ for all $i \in I$, then*

$$\sum_{i \in I} \sum_{j \in J} R_j(B_i) > |I| - \delta. \tag{11}$$

CLAIM 6.

$$\alpha_1 > (3/5)\, L^* + 3. \tag{12}$$

Proof. First we recall that by assumption $\mathrm{FF}(L) - sL^* > (7/10)L^* + 5/2$. By the definition of our partition we have

$$\mathrm{FF}(L) = |\Delta| + |X| + sL^* - \sum_{i=1}^{s} \beta_i + \sum_{i=1}^{s} \alpha_i . \tag{13}$$

This implies

$$\sum_{i=1}^{s} \alpha_i > (7/10)\, L^* + 5/2 + \sum_{i=1}^{s} \beta_i - |\Delta| - |X|. \tag{14}$$

Now consider the partition of the bins given in Fig. 2. The kth column, C_k, satisfies the hypotheses of Claim 5 with $I_k = \{1, 2, \ldots, |C_k|\}$ and $B_{k,i}$ being the bin in row S_i and column C_k. Similarly, by Claim 2, $X - X_0$ satisfies the hypotheses with $I_x = \{i > 0 : X_i \neq \varnothing\}$. Thus we can apply Corollary 1 to each of these sets of bins, with an appropriate value for δ. By the definition of S_i and X_i, we can take S to be $1/2$ for $X - X_0$ and for each C_k, $1 \leqslant k \leqslant L^*$. Furthermore, since by Claim 3 and the construction of the diagram, every $B_{k,i}$ for $k > L^*$ belongs to $S_i(2)$ and hence has $R_i(B_{k,i}) > 2/3$, we can take S to be $1/3$ for each C_k, $L^* < k \leqslant |S_1| = L^* + \alpha_1$. Thus we can conclude that

$$\begin{aligned} SL^* &\geqslant \sum_{\text{all } B} \sum_{j=1}^{s} R_j(B) \\ &\geqslant \sum_{k=1}^{|S_1|} |C_k| - (1/2)\, L^* - (1/3)\, \alpha_1 + |X| - (3/2) + |\Delta| \\ &\geqslant SL^* + \sum_{i=1}^{s} \Delta_i - \sum_{i=1}^{s} \beta_i + |X| + |\Delta| - (1/2)\, L^* - (1/3)\, \alpha_1 - (3/2). \end{aligned}$$

Substituting for $\sum_i \alpha_i$ using (14) yields the desired result.

CLAIM 7.

$$\alpha_2 < (1/10)\, L^* + 1/2. \tag{15}$$

Proof. By (6), (8) and (13) we have

$$\sum_{i=1}^{s} \alpha_i \leqslant (7/10)\, L^* + \left(\sum_{i=1}^{s} \beta_i - |\, \Delta\,| - |\, X\,|\right) + 7/2 \qquad (16)$$

$$\leqslant (7/10)\, L^* + 7/2.$$

Thus, using Claim 6, we have

$$\alpha_2 \leqslant \sum_{i=2}^{s} \alpha_i \leqslant (7/10)\, L^* + 7/2 - (3/5)\, L^* - 3$$

$$\leqslant (1/10)\, L^* + 1/2.$$

This proves the claim. ∎

We have now gotten the preliminaries out of the way and we are ready to begin the more direct mechanisms of the proof. As in Theorem 1, we shall use a weighting function w: $L \to \mathbb{R}$, where $\mathbb{R}$ denotes the set of real numbers. This time the two inequalities which we shall prove are the following.

(A) $w(L) \leqslant (s + 7/10)\, L^*$;

(B) $\mathrm{FF}(L) \leqslant w(L) + 5/2$.

Theorem 2 will then follow immediately.

First we must define w, which we do in terms of functions w_i, $1 \leqslant i \leqslant s$, specified as follows.

$w_1(b) = W(R_1(b))$ where W is as defined in Theorem 1 (see Fig. 1),

$w_i(b) = R_i(b), \qquad 2 \leqslant i \leqslant s,$

$$w(b) = \sum_{i=1}^{s} w_i(b).$$

As usual, we extend the domains of definition to sets B by letting $w_i(B) = \sum_{b \in B} w_i(b)$, etc.

CLAIM 8. *For any* FF *packed bin* B:

(i) $w(B) \leqslant s + 7/10$;

(ii) *If* $R_1(b) \leqslant 1/2$ *for all* $b \in B$ *then* $w(B) \leqslant s + 1/2$,

Proof. Any validly packed bin B must have $R_i(B) \leqslant 1$, $1 \leqslant i \leqslant s$. Thus,

$$\sum_{i=2}^{s} w_i(B) = \sum_{i=2}^{s} R_i(B) \leqslant s - 1.$$

Moreover, by Lemma 1, $R_1(B) \leqslant 1$ implies that $w_1(B) \leqslant 17/10$ and, if the hypothesis of case (ii) holds in addition, then $w_1(B) \leqslant 3/2$. This proves the claim. ∎

Inequality (A) follows immediately from the preceding claim. The proof of (B) is considerably more involved. We first show that we can restrict our attention to the bins of S_1 and one other bin for each column in our diagram.

Let us first look at the columns of the diagram, this time ignoring the elements of row S_1. That is, we consider $C_k' = C_k - S_1$, $1 \leqslant k \leqslant |S_2|$. Each C_k' satisfies the hypotheses of Claim 5 with $I_k' = \{2, 3, \ldots, |C_k|\}$, and $B_{i,k}$ being the element of row S_i in column C_k', $1 \leqslant k \leqslant |S_2|$, $i \in I_k'$. Thus, there exists an index $r(k) \in I_k'$ such that for all k, $1 \leqslant k \leqslant |S_2|$, we have

$$\sum_{i \in I_k'} \sum_{j \in I_k' - \{r(k)\}} R_j(B_{i,k}) \geqslant |C_k'| - 1 = |C_k| - 2.$$

Letting $T = \bigcup_{i=2}^{s} S_i = \bigcup_{k=1}^{|S_2|} C_k'$, we thus have proved:

CLAIM 9.

$$\sum_{B \in T} w(B) \geqslant |T| - |S_2| + \sum_{k=1}^{|S_2|} [w_1(B_{r(k),k}) + R_{r(k)}(B_{r(k),k})]. \tag{17}$$

Next, we look at S_1. If we let L_1 be the list obtained from L by deleting all tasks which do not go into bins of S_1, then by the operation of the FF algorithm, we see that $\mathrm{FF}(L) = |S_1|$ and, in fact, the bins of the FF packing of L_1 are packed in exactly the same way as those in S_1. Hence by Lemma 5 we have

CLAIM 10.

$$\sum_{B \in S_1} w(B) \geqslant |S_1| - 1 + \sum_{\substack{B \in S_1 \\ w_1(B) > 1}} (w_1(B) - 1) + \sum_{B \in S_1} \sum_{i=2}^{s} R_i(B).$$

Finally, since $w_i(b) \geqslant R_i(b)$, $1 \leqslant i \leqslant s$, we have by the definition of Δ and X, and by Claim 2, and Corollary 1 to Claim 5 the following results.

CLAIM 11.

$$\sum_{B \in \Delta} w(B) \geqslant |\Delta|.$$

CLAIM 12.

$$\sum_{B \in X} w(B) \geqslant |X| - 3/2.$$

Let $V = \{B_{r(k),k} : 1 \leqslant k \leqslant |S_2|\}$ and let $R_B \equiv R_{r(k)}$ if $B = B_{r(k),k} \in V$ Then by combining the preceding four claims we have

$$\begin{aligned} w(L) &= \sum_{B \in S_1} w(B) + \sum_{B \in T} w(B) + \sum_{B \in \Delta} w(B) + \sum_{B \in X} w(B) \\ &\geqslant \mathrm{FF}(L) - 5/2 - |S_2| + \sum_{B \in V} (w_1(B) + R_B(B)) \\ &\quad + \sum_{\substack{B \in S_1 \\ w_1(B) > 1}} (w(B) - 1) + \sum_{B \in S_1} \sum_{i=2}^{s} R_i(B). \end{aligned} \tag{18}$$

Thus, inequality (B) will follow if we can show

$$\text{(C)} \quad |S_2| \leqslant \sum_{B \in V} [w_1(B) + R_B(B)] + \sum_{B \in S_1} \left[\max\{0, w_i(B) - 1\} + \sum_{i=2}^{s} R_i(B) \right].$$

We shall show how to partition $S_1 \cup V$ into sets of bins $A_1, A_2, \ldots, A_m$ such that for each A_k

$$\text{(D)} \quad \sum_{B \in A_k \cap V} [w_1(B) + R_B(B)] + \sum_{B \in A_k \cap S_1} \left[\max\{0, w_1(B) - 1\} + \sum_{i=2}^{s} R_i(B) \right] \geqslant |A_k \cap V|.$$

Since $|V| = |S_2|$, the desired inequality (C) will follow. If we find a set A_k which satisfies (D), we will say it has been "cancelled." We find sets to include in our partition by using "cancellation rules," proceeding inductively. We start with $V_1 = V \cup S_1$ and, in general, we choose a set A_k from V_k, showing that it satisfies (*D*). If we ever reach a point when $V_k \cap V = \varnothing$, we set $A_k = V_k$ which obeys (D) trivially and the partition will be complete. Our goal will be to show that this must eventually happen. We start with $k = 1$ and the following cancellation rule.

Cancellation Rule 1. If there are one-task bins $B_i \in V_k \cap V$ and $B_j \in V_k \cap S_1$, set $A_k = \{B_i, B_j\}$, $V_{k+1} = V_k - A_k$.

CLAIM 11. *If A_k is formed by Rule* 1, *then it satisfies* (D).

Proof. Let b_i be the one task in B_i and let b_j be the one task in B_j. Task b_i exceeds 1/2 only in resource R_{B_i}, and task b_j exceeds 1/2 only in resource R_1, and yet, the two tasks were placed by FF into distinct one-task bins. Thus we must have either

$$R_1(b_i) + R_1(b_j) > 1 \qquad \text{or} \qquad R_{B_i}(b_i) + R_{B_i}(b_j) > 1.$$

In the second case, (D) is immediate. In the first case, since $R_1(b_j) > 1/2$, Lemma 2 implies that $w_1(B_i) + w_1(b_j) \geqslant 8/5$ so that

$$\begin{aligned} w_1(B_i) + \max\{0, w_1(B_j) - 1\} + R_B(B_i) \\ \geqslant 8/5 + 1/2 - 1 > 1 = |A_k \cap V|. \end{aligned}$$

Thus, (D) also holds in this case and the claim is proved.

Apply Cancellation Rule 1 repeatedly until it can no longer be applied, and let $k(1)$ be the index of V_k at this time. If $V_{k(1)} \cap V = \varnothing$ we are done. Otherwise, we at least have

$$V_{k(1)} \cap \{B_i \in V: B_i \text{ is a one-task bin}\} = \varnothing,$$

as the next claim will imply.

CLAIM 12.

$$|S_1(1)| - 1 > (1/2)\, L^* > \sum_{i=2}^{s} |S_i(1)|.$$

Proof. Let L_1 be as above, i.e., the list obtained from L by deleting all tasks not in bins of S_1. Let J be the set of tasks b from L_1 which have $R_1(b) > 1/2$. By Lemma 1,

$$\sum_{B \in S_1} w_1(B) \leqslant (3/2)\, L_1^* + (1/5)\,|J| \leqslant (3/2)\, L^* + (1/5)\,|J|. \qquad (19)$$

Now, let $J_1 = S_1(1)$, and let $J_2 \subseteq S_1(2)$ be those bins of $S_1(2)$ which contain an element of J. Observe that $|J| = |J_1| + |J_2|$. By Lemma 5 we have

$$\sum_{B \in S_1} w_1(B) \geqslant |S_1| - 1 + \sum_{B \in S_1} (\max\{0, w_1(B) - 1\}). \qquad (20)$$

Consider any pair of distinct bins B_1, $B_2 \in J_2$ with B_2 having higher index in the FF packing of L. Let a_1 and a_2 be the tasks of J in B_1 and B_2, respectively, and let b_2 be a second task from B_2 (which must exist since $B_2 \in J_2$). Since b_2 did not go into the earlier bin B_1 during the FF packing

of L_1, we must have $R_1(B_1) + R_1(b_2) > 1$, and hence, by Lemma 2, $w_1(B_1) + w_1(b_2) \geqslant 8/5$. Thus,

$$\max\{0, w_1(B_1) - 1\} + \max\{0, w_1(B_2) - 1\} \geqslant 3/5.$$

This implies

$$\sum_{B \in S_1} (\max\{0, w_1(B) - 1\}) \geqslant \lfloor (1/2) \mid J_2 \mid \rfloor \cdot (3/5) > (3/10) \mid J_2 \mid - 1. \quad (21)$$

Thus, by (12) and (20), we obtain

$$\sum_{B \in S_1} w_1(B) \geqslant (8/5)\, L^* + (3/10) \mid J_2 \mid + 1 \quad (22)$$

and so, by (19),

$$(3/2)\, L^* + (1/5) \mid J \mid \geqslant (8/5)\, L^* + (3/10) \mid J_2 \mid + 1,$$

and hence

$$\mid S_1(1) \mid = \mid J_1 \mid = \mid J \mid - \mid J_2 \mid \geqslant (1/2)\, L^* + (1/2) \mid J_2 \mid + 5, \quad (23)$$

which is the first half of Claim 12. The second half follows from (23) and Claim 3. This proves Claim 12. ∎

Thus, as long as there is a one-task bin left in $V_k \cap V$ (and hence, in $\bigcup_{i=2}^{s} S_i(1)$), there must also be a one-task bin left in $V_k \cap S_1$ and so we have established the following result.

CLAIM 13.

$$V_{k(1)} \cap \{B_i \in V\colon B_i \text{ is a one-task bin}\} = \varnothing.$$

After we can no longer apply Cancellation Rule 1, we begin to use the following rule.

Cancellation Rule 2. If B_j is a one-task bin in $V_k \cap S_1(1)$ and $B_i \in V_k \cap V$, set $A_k = \{B_i, B_j\}$, $V_{k+1} = V_k - A_k$.

CLAIM 14. *If A_k is formed by Rule* 2, *then it satisfies* (D).

Proof. Let b_j be the one task in bin B_j. Again, we must have either $R_1(B_i) + R_1(b_j) > 1$ or $R_{B_i}(B_i) + R_{B_i}(b_j) > 1$. The claim now follows by exactly the same arguments as used in the proof of Claim 11, since we still have $R_1(b_j) > 1/2$. ∎

Apply Cancellation Rule 2 repeatedly until it no longer can be applied, and let $k(2)$ be the index of V_k at that time. If $V_{k(2)} \cap V = \varnothing$, we are done. Otherwise, by Claim 12, the following fact holds.

CLAIM 15.

$$k(2) = |S_1(1)| + 1 \leqslant (1/2)\, L^* + 2.$$

After we can no longer apply Rule 2, we begin to use the following rule.

Cancellation Rule 3. If $B_i \in V_k \cap V$ and $B_j \in V_k \cap S$, and $R_{B_i}(B_i) + R_{B_i}(B_j) > 1$, set $A_k = \{B_i, B_j\}$, $V_{k+1} = V_k - A_k$. A pair formed by Rule 3 automatically obeys (D). We apply this rule repeatedly until it can no longer be applied, and we let $k(3)$ be the index of V_k at that time. If $V_{k(3)} \cap V = \varnothing$ we are done. Otherwise we proceed with the next rule, after making note of the following fact.

CLAIM 16. *If* $B_i \in V_{k(3)} \cap V$ and $B_j \in V_{k(3)} \cap S_1$, then $j < i$.

Proof. Suppose not, so that bin B_i was found before bin B_j. Since B_j must contain more than one task (by Rule 2), it must contain a task b with $R_1(b) \leqslant 1/2$. Since b did not go in the earlier bin B_i, and since, by Rule 3, $R_{B_i}(B_i) + R_{B_i}(b) \leqslant 1$, we must have $R_1(B_i) > 1 - R_1(b) \geqslant 1/2$. But this means that B_i has resource usage exceeding 1/2 in two resources (R_1 and R_{B_i}) and this contradicts the fact that $B_i \notin \Delta$. ∎

Cancellation Rule 4. If $B_i \in V_k \cap V$ and $B_j \in V_k \cap S_1$, and $R_1(B_j) \leqslant 8/9$, then set $A_k = \{B_i, B_j\}$, $V_{k+1} = V_k - A_k$.

CLAIM 17. If A_k is formed by Rule 4, then it obeys (D).

Proof. By Claim 12 we know that B_1 contains at least two tasks, say a and b. Since it contains more than one task, we also know by the definition of S_{B_i} that $R_{B_i}(B_i) > 3/2$. By Claim 16 we know that B_j has a lower index and hence is the earlier bin. By Rule 3 we see that $R_{B_i}(B_i) + R_{B_i}(B_j) \leqslant 1$. Thus, since neither a nor b went into the earlier bin B_j, we must have $R_1(a) + R_1(B_j) > 1$ and $R_1(b) + R_1(B_j) > 1$, and hence both $R_1(a)$ and $R_1(b)$ must exceed 1/9. Since $W(\alpha) \geqslant (6/5)\,\alpha$ for all $\alpha \in [0, 1]$, we therefore have

$$w_1(B_j) + w_1(B_i) \geqslant (6/5)(10/9) = 4/3.$$

Therefore,

$$(w_1(B_j) - 1) + w_1(B_i) + R_{B_i}(B_i) \geqslant 4/3 - 1 + 2/3 = 1$$

and (D) holds. ∎

Let $k(4)$ be the index of V_k at the time Rule 4 can no longer be applied. If $V_{k(4)} \cap V = \varnothing$, we are done. Otherwise, we note that each of our cancellation rules so far removes one bin from $V_k \cap S_1$ and one bin from $V_k \cap V$, and thus by Claims 6, 7, and 15, we have

$$\begin{aligned} &| V_{k(4)} \cap S_1 |/| V_{k(4)} \cap V | \\ &\quad \geqslant \frac{([(8/5)\, L^* + 3] - [(1/2)\, L^* + 1])}{([(11/10)\, L^* + 1/2] - [(1/2)\, L^* + 1])} \\ &\quad \geqslant ((11/10)\, L^* + 2)/((6/10)\, L^* - 1/2) > 11/6. \end{aligned} \tag{24}$$

Also observe that by Claim 16 and Rule 4, all the bins B of $V_{k(4)} \cap S_1$ must be earlier in the FF packing than any of the bins of $V_{k(4)} \cap V$, and all must have $R_1(B) > 8/9$. Further, by Claim 13 all the bins of $V_{k(4)} \cap V$ must contain at least two tasks. We now apply a final cancellation rule.

Cancellation Rule 5.

(a) If $| V_k \cap V | \geqslant 3$ and $| V_k \cap S_1 | \geqslant 5$, let $B_{i(1)}$, $B_{i(2)}$, $B_{i(3)} \in V_k \cap V$ and $B_{j(1)}$, $B_{j(2)}$, $B_{j(3)}$, $B_{j(4)}$, $B_{j(5)} \in V_k \cap S_1$ be distinct elements of V_k and set A_k to be the set consisting of all eight of these tasks.

(b) If $| V_k \cap V | = 2$ and $| V_k \cap S_1 | \geqslant 4$, let $B_{i(1)}$, $B_{i(2)} \in V_k \cap V$ and $B_{j(1)}$, $B_{j(2)}$, $B_{j(3)}$, $B_{j(4)} \in V_k \cap S_1$ be distinct elements of V_k and set A_k to be the set consisting of all six of these tasks.

(c) If $| V_k \cap V | = 1$ and $| V_k \cap S_1 | \geqslant 2$, let $B_{i(1)} \in B_k \cap V$ and $B_{j(1)}$, $B_{j(2)} \in V_k \cap S_1$ be distinct elements of V_k and set A_k to be the set consisting of all three of these tasks.

Set $V_{k+1} = V_k - A_k$.

CLAIM 18. *If A_k is formed by Rule 5 then it obeys* (D).

Proof. Consider case (a). By Claim 13, $B_{i(1)}$, $B_{i(2)}$ and $B_{i(3)}$ all contain at least two tasks, say $a(1)$, $b(1)$, $a(2)$, $b(2)$ and $a(3)$, $b(3)$. Also, $R_{B_{i(h)}}(B_{i(h)}) > 2/3$, $1 \leqslant h \leqslant 3$. Since the bins of $V_k \cap S_1$ precede the bins $B_{i(h)}$, we must have, by Rule 3,

$$\begin{aligned} &R_1(B_{j(1)}) + R_1(a(1)) > 1, \qquad R_1(B_{j(2)}) + R_1(a(2)) > 1, \\ &R_1(B_{j(3)}) + R_1(a(3)) > 1, \qquad R_1(B_{j(4)}) + R_1(b(1)) > 1, \\ &R_1(B_{j(5)}) + R_1(b(2)) > 1. \end{aligned}$$

Thus, since $W(\alpha) \geqslant (6/5)\,\alpha$ for all $\alpha \in [0, 1]$, we have

$$\sum_{h=1}^{5} (w_1(B_{j(h)}) - 1) + \sum_{h=1}^{3} w_1(B_{i(h)}) + \sum_{h=1}^{3} R_{B_{i(h)}}(B_{i(h)})$$
$$> 5(6/5 - 1) + 3(2/3) = 3 = |\, V \cap A_k \,|.$$

and so (D) holds. Similar arguments apply to cases (b) and (c) and the claim is proved. ∎

Let $k(5)$ be the index of V_k when Rule 5 can no longer be applied.

CLAIM 19. $V_{k(5)} \cap V = \varnothing$.

Proof. By (24) we know that initially

$$|\, V_{k(5)} \cap S_1 \,|/|\, V_k \cap V \,| \geqslant 11/6.$$

So long as $|\, V_k \cap S_1 \,|/|\, V_k \cap V \,|$ remains at least 11/6 and $V_k \cap V \neq \varnothing$, we can apply Rule 5, for then

(a) $|\, V_k \cap V \,| \geqslant 3$ implies $|\, V_k \cap S_1 \,| \geqslant \lceil (11/6) \cdot 3 \rceil = 6 > 5$,

(b) $|\, V_k \cap V \,| = 2$ implies $|\, V_k \cap S_1 \,| \geqslant \lceil (11/6) \cdot 2 \rceil = 4$,

(c) $|\, V_k \cap V \,| = 1$ implies $|\, V_k \cap S_1 \,| \geqslant \lceil 11/6 \rceil = 2$.

But if we ever apply (b) or (c) to V_k we will immediately get $V_{k+1} \cap V = \varnothing$ and we will be done. On the other hand, if (a) is applied, the ratio can only *increase*, since in this case,

$$|\, V_{k+1} \cap S_1 \,|/|\, V_{k+1} \cap V \,| = (|\, V_k \cap S_1 \,| - 5)/(|\, V_k \cap V \,| - 3)$$
$$> |\, V_k \cap S_1 \,|/|\, V_k \cap V \,| \geqslant 11/6$$

since $5/3 < 11/6$. Thus, by induction the ratio will remain at least 11/6 as long as $V_k \cap V \neq \varnothing$. Hence, when Rule 5 can no longer be applied, we must have $V_{k(5)} \cap V = \varnothing$. This proves the claim. ∎

Since $V_{k(5)} \cap V = \varnothing$, we then complete our partition of $V \cap S_1$ by setting $A_{k(5)} = V_{k(5)}$. Thus, all sets A_k in the partition obey (D) and hence, (C). Therefore (B) holds and the proof of Theorem 2 is complete. ∎

We shall next show how to construct lists L for which $\mathrm{FF}(L)/L^*$ is arbitrarily close to $s + 7/10$, showing that the upper bound in Theorem 2 is essentially best possible.

We begin by recalling that for the case $s = 1$, [13] gives for each $k \geqslant 1$, a list L_1 with $L_1^* = k$ such that $\mathrm{FF}(L_1) \geqslant (17/10)\, L_1^* - 8$. For a fixed k, let $y_1, y_2, \ldots, y_p$ be the list given in that construction and let a_i be the

requirement of y_i for that single resource. We shall use this list as the basis for our construction with $s > 1$.

Let $L = \{x_1, x_2, \ldots, x_n\}$ where $n = 2(s-1)k + p$. The corresponding resource requirements are as follows.

$$\begin{aligned} R_1(x_i) &= a_j && \text{for } i = n - p + j,\ 1 \leqslant j \leqslant p, \\ &= (s+1)\epsilon && \text{for } 1 \leqslant i \leqslant n-p,\ i \text{ even}, \\ &= 0 && \text{for } 1 \leqslant i \leqslant n-p,\ i \text{ odd}, \end{aligned}$$

and, for $2 \leqslant j \leqslant s$,

$$\begin{aligned} R_j(x_i) &= 1 - (s+1)\epsilon && \text{for } 2(j-2)k < i < 2(j-1)k,\ i \text{ odd}, \\ &= (s+1)\epsilon && \text{for } 1 \leqslant i \leqslant n-p,\ i \text{ even}, \\ &= \epsilon && \text{otherwise}, \end{aligned}$$

where ϵ is chosen so that $0 < \epsilon < (ns^2)^{-1}$.

Basically there are three types of tasks in L; one type with all resource requirements equal to $(s+1)\epsilon$; one type with zero requirement for R_1, requirement $1 - (s+1)\epsilon$ in some resource other than R_1, and requirement ϵ for all other resources; and one type with requirement a_i, for some i, for R_1 and ϵ for all other resources. Each task of the last type corresponds to a unique task in the original single resource example.

It is not hard to verify that an optimal packing requires at most $k+1$ bins. One way to achieve this is as follows. Place all tasks having requirement $(s+1)\epsilon$ for all resources in a single bin. Fill each of the remaining $k = L_1^*$ bins with $s+2$ tasks, three of which correspond to tasks packed together in the optimal packing for the single resource example and, for each k with $2 \leqslant k \leqslant s$, one task which has requirement $1 - (s+1)\epsilon$ for resource R_k.

However, in the FF packing for L, we will have $(s-1)k$ bins each containing two tasks, one task with all requirements equal to $(s+1)\epsilon$ and one task having requirement $1 - (s+1)\epsilon$ in exactly one resource. Following these willl be at least $(17/10)k - 8$ bins, each containing a set of tasks corresponding to a set of tasks placed together by the FF packing of L_1. We therefore have

$$\begin{aligned} \mathrm{FF}(L)/L^* &\geqslant ((s-1)k + (17/10)k - 8)/(k+1) \\ &= s + 7/10 - (s + 87/10)/(k+1). \end{aligned}$$

Since k was arbitrary, this gives the desired result.

The next result shows that by preprocessing the list in this no precedence constraint case, we can improve the worst-case behavior of FF somewhat.

THEOREM 3. *If L is any list arranged in decreasing order by $R_{\max}(x)$ in a system with $s \geqslant 1$ resources and $\prec$ is empty, then*

$$\mathrm{FF}(L) \leqslant (s + 1/3)\, L^*. \tag{25}$$

Proof. Suppose we have a list L for which $\mathrm{FF}(L) > (s + 1/3)\, L^*$. Let the FF packing of L consist of bins $B_1, B_2, \ldots, B_{\mathrm{FF}(L)}$, and let $S = \{B_i : 1 \leqslant i \leqslant sL^*\}$ and $T = \{B_i : sL^* < i \leqslant \mathrm{FF}(L)\}$, with $L_S = \bigcup_{B \in S} B$ and $L_T = \bigcup_{B \in T} B$.

CLAIM 20. *For all $x \in L_T$, $R_{\max}(x) \leqslant 1/3$.*

Proof. Suppose there were an $x \in L_T$ with $R_{\max}(x) > 1/3$. Since L is in decreasing order by $R_{\max}(x)$, we may assume without loss of generality that x is the first task in L to go into a bin of T. Let $\bar{L}$ be the list obtained from L by deleting all tasks that follow x in L. By the operation of FF, we will then have $\mathrm{FF}(\bar{L}) = sL^* + 1$. Let $C_1, C_2, \ldots, C_{sL^*+1}$ be the bins of the FF packing of $\bar{L}$, and let $D_1, D_2, \ldots, D_{\bar{L}^*}$ be the bins of an optimal packing $\bar{L}$, and let $\mathscr{C} = \{y \in \bar{L}$: for some i, $1 \leqslant i \leqslant sL^*$, $C_i = \{y\}\}$.

$\mathscr{C}$ is the set of all tasks that are in one-task bins in the FF packing of $\bar{L}$. Since no two such tasks can fit together (or else they would not have gone into separate one-task bins), each $y \in \mathscr{C}$ must come from a distinct bin $D(y)$ in the optimal packing. We shall now show that for all optimal bins D_i, $|\, D_i \,| \leqslant 2s$, and if $D_i = D(y)$ for some $y \in \mathscr{C}$, $|\, D_i \,| \leqslant 2s - 1$.

The first bound is easy. For each $z \in \bar{L}$, let the *maximal index of* z, denoted by $m(z)$, be defined by

$$m(z) = \min\{i\colon 1 \leqslant i \leqslant s \text{ and } R_i(z) = R_{\max}(z)\}.$$

Since all tasks z in $\bar{L}$ have $R_{\max}(z) > 1/3$, no optimal bin D_i can contain more than two tasks with the same maximal index, and hence $|\, D_i \,| \leqslant 2s$. If $D_i = D(y)$ for some $y \in \mathscr{C}$, and $|\, D_i \,| > 2s - 1$, we would have to have $|\, D_i \,| = 2s$, and hence no task $w \in D_i$ could have $R_i(w) > 1/3$ for any $i \neq m(w)$. Furthermore, there would have to be a task $z \neq y$ with $z \in D_i$ and $m(z) = m(y)$. But then we would have

$$R_{m(y)}(z) = R_{\max}(z) \geqslant R_{\max}(x) \geqslant R_{m(y)}(x),$$

and so,

$$R_{m(y)}(y) + R_{m(y)}(x) \leqslant R_{m(y)}(y) + R_{m(y)}(z) \leqslant 1.$$

Moreover, since there can be no more than sL^* tasks u with $R_{\max}(u) > 1/2$, we must have $R_{\max}(x) \leqslant 1/2$ and therefore,

$$R_j(y) + R_j(x) \leqslant 1/3 + 1/2 < 1 \qquad \text{for all } j \neq m(y),\ 1 \leqslant j \leqslant s.$$

Thus, x would have fit as the second task in the one-task bin containing y in the FF packing of $\bar{L}$, a contradiction. We therefore can conclude that $|D(y)| \leqslant 2s - 1$ for all $y \in \mathscr{C}$.

We now have two contradictory bounds on $|\bar{L}|$. From the optimal packing we have

$$|\bar{L}| \leqslant (2s)(\bar{L}^* - |\mathscr{C}|) + (2s-1)(|\mathscr{C}|) = 2s\bar{L}^* - |\mathscr{C}| \leqslant 2sL^* - |\mathscr{C}|.$$

But from the FF packing and the definition of $\mathscr{C}$ we have

$$|\bar{L}| \geqslant 2(sL^* - |\mathscr{C}|) + |\mathscr{C}| + 1 = 2sL^* - |\mathscr{C}| + 1.$$

This contradiction proves the claim. ∎

As a straightforward corollary of Claim 20 and the assumption that $\mathrm{FF}(L) > (s + 1/3)\,L^*$ we have the following fact.

CLAIM 21. $|L_T| > L^*$.

Let us now reexamine the FF packing of the original list L. Partition S as follows.

$$S_0 = \left\{B_i \in S: \sum_{j=1}^{s} R_j(B_i) > 1\right\},$$

$$S_j = \{B_i \in S - S_0: R_j(B_i) > 1/2\}, \qquad 1 \leqslant j \leqslant s.$$

Since L_T is nonempty and, by Claim 20, contains only tasks x with $R_{\max}(x) \leqslant 1/3$, all bins in S must have level exceeding 2/3 in some resource, and so $\{S_0, S_1, \ldots, S_s\}$ is indeed a partition of S.

CLAIM 22. *If* $B \in S_j$, $1 \leqslant j \leqslant s$, *and* $x \in L_T$, *then* $R_j(B) + R_j(x) > 1$.

Proof. Since x did not go into bin $B \in S_j$ but instead into a higher indexed bin in T, we must have $R_i(B) + R_i(x) > 1$ for *some* i, $1 \leqslant i \leqslant s$. However, for each $i \neq j$, $R_i(B) \leqslant 1/2$ by definition of S_j, and $R_i(x) \leqslant 1/3$ by Claim 20. Thus we must have $R_j(B) + R_j(x) > 1$ as desired. ∎

Now let us relabel the bins in the set S_j, and the tasks in L_T. For each j, $1 \leqslant j \leqslant s$, arbitrarily label the bins in S_j as $B_{1,j}, B_{2,j}, \ldots, B_{|S_j|,j}$. Label the elements of L_T arbitrarily as $X_1, X_2, \ldots, X_{|L_T|}$.

CLAIM 23. *For all* j, $1 \leqslant j \leqslant s$, $|S_j| < L^*$.

Proof. If $|S_j| \geqslant L^*$, consider the pairs $\langle B_{i,j}, X_i \rangle$, $1 \leqslant i \leqslant L^*$. By Claim 21, $|L_T| > L^*$, so such pairs exist. For each pair, we have by Claim 22 that $R_j(B_{i,j}) + R_j(x_i) > 1$. But this means that

$$R_j(L) \geqslant \sum_{i=1}^{L^*} [R_j(B_{i,j}) + R_j(x_i)] > L^*$$

which is a contradiction. ∎

Thus, we can repartition S as follows: Let $V_0 = S_0$ and for each i, $1 \leqslant i \leqslant L^*$, let $V_i = \{B_{i,j} : |S_j| \geqslant i\}$. By Claim 23, every bin in $S - S_0$ will be in some V_i, and so $\{V_0, V_1, V_2, \ldots, V_{L^*}\}$ is a partition of S. Now let $L(V_i) = \bigcup_{B \in V_i} B \cup \{x_i\}$, $1 \leqslant i \leqslant L^*$, and let $L(V_0) = \bigcup_{B \in V_0} B$. Since the V_i's form a partition of S, and all the X_i's are all distinct members of L_T (and hence not in L_S), we then have that the $L(V_i)$, $0 \leqslant i \leqslant L^*$, are all disjoint subsets of L. Thus we have

$$\sum_{j=1}^{s} R_j(L) \geqslant \sum_{i=0}^{L^*} \sum_{j=1}^{s} R_j(L(V_i)).$$

However, by the definition of $V_0 = S_0$, we have

$$\sum_{j=1}^{s} R_j(L(V_0)) = \sum_{B \in V_0} \sum_{j=1}^{s} R_j(B) > \sum_{B \in V_0} (1) = |V_0|.$$

Moreover, by Claim 2 and the labelling of the $B_{i,j} \in V_i$, we have for each i, $1 \leqslant i \leqslant L^*$,

$$\sum_{j=1}^{s} R_j(L(V_i)) \geqslant \sum_{j=1}^{|V_i|} [R_j(B_{i,j}) + R_j(X_i)] > \sum_{j=1}^{|V_i|} (1) = |V_i|.$$

Thus,

$$\sum_{j=1}^{s} R_j(L) > \sum_{i=0}^{L^*} |V_i| = |S| = sL^*.$$

However, this contradicts the fact that no resource can have total usage exceeding L^*. Thus, the assumption that $\mathrm{FF}(L) > (s + 1/3)\,L^*$ leads to a contradiction. This proves the theorem. ∎

Our lower bound for the case considered in Theorem 3 is not equal to the upper bound in (25). The best result we have obtained is that the ratio $\mathrm{FF}(L)/L^*$ can be made arbitrarily close to $s + (s - 1)/(s^2 + s)$. We now describe this construction.

Choose an arbitrary positive integer k which is a multiple of $s(s+1)$. The list L will be composed of s regions, with all tasks in region i occurring before all tasks in region $i+1$ in L, $1 \leqslant i \leqslant s$. The tasks in region i will be denoted by

$$x_{i,1}, x_{i,2}, \ldots, x_{i,q(i)},$$

where $q(i) = (i+1)(k-1)$ for $1 \leqslant i < s$ and $q(s) = (s+1)k$. For $1 \leqslant i < s$, the resource requirements for the tasks in region i are given as follows, where ϵ satisfies $0 < \epsilon < k^{-3}$:

$$\begin{aligned} R_i(x_{i,j}) &= (i+1)^{-1} + t_{i,j}\epsilon \qquad \text{if } 1 \leqslant j \leqslant i(k-1) \\ &\qquad\qquad\qquad\qquad \text{where } t_{i,j} = k - \lfloor (j-1)/i \rfloor, \\ &= (i+1)^{-1} - it'_{i,j}\epsilon \qquad \text{if } i(k-1) < j \leqslant q(i) \\ &\qquad\qquad\qquad\qquad \text{where } t'_{i,j} = j + 1 - i(k-1). \end{aligned}$$

For $1 \leqslant l \leqslant s$, $l \neq i$, $R_l(x_{i,j}) = \epsilon/s^2$. The resource requirements for the tasks in region s are as follows.

$$\begin{aligned} R_s(x_{s,j}) &= (s+1)^{-1} + 2\epsilon \qquad \text{for } 1 \leqslant j \leqslant k(s-1), \\ &= (s+1)^{-1} - s\epsilon \qquad \text{for } k(s-1) < j \leqslant q(s). \end{aligned}$$

For $1 \leqslant l \leqslant s-1$, $R_l(x_{s,j}) = \epsilon/s^2$.

The FF packing of L also consists of s regions, each containing only tasks from the corresponding region of L. For $1 \leqslant i \leqslant (s-1)$, each bin in region i will contain exactly i tasks with requirement $(i+1)^{-1} + t\epsilon$ for R_i and one task with requirement $(i+1)^{-1} - it\epsilon$ for R_i (for an appropriate t), thus completely using up resource R_i. This gives a total of $(k-1)(s-1)$ bins. Region s will contain $k(s-1)/s$ bins which each have exactly s tasks with requirement $(s+1)^{-1} + 2\epsilon$ for R_s and $2k/(s+1)$ bins which each have exactly $s+1$ tasks with requirement $(s+1)^{-1} - s\epsilon$ for R_s. Thus

$$\mathrm{FF}(L) = (k-1)(s-1) + k(s-1)/s + 2k/(s+1). \tag{26}$$

However, we have $L^* \leqslant k$, as can be seen from the following packing of L. Each bin will contain $s+1$ tasks from region s, $s-1$ tasks with requirement $(s+1)^{-1} + 2\epsilon$ for R_s and two tasks with requirement $(s+1)^{-1} - s\epsilon$ for R_s. In addition, each bin will contain tasks from each of the other regions. The first bin will contain the first i tasks from each region i; these are the tasks with requirement $(i+1)^{-1} + k\epsilon$ for R_i. The last bin will contain the single task from each region i having require-

ment $(i+1)^{-1} - 2i\epsilon$ for R_i. The remaining bins each contain $i+1$ tasks from each region i, consisting of i tasks with requirement $(i+1)^{-1} + t\epsilon$ for R_i and one task with requirement $(i+1)^{-1} - i(t+1)\epsilon$ for R_i (for an appropriate t). Notice that in no bin do all the tasks from region i use more than a total of $1-\epsilon$ of resource R_i. This fact allows tasks from other regions, all of which require ϵ/s^2 units of resource R_i, to fit into that bin. We leave to the reader the straightforward verification that the two preceding packings are indeed valid.

Combining this information, we obtain

$$\begin{aligned} \mathrm{FF}(L)/L^* &\geqslant (1/k)((k-1)(s-1) + k(s-1)/s + 2k/(s+1)) \\ &= s + (s-1)/(s^2+s) - (s-1)/k. \end{aligned}$$

Of course, this can be made arbitrarily close to $s + (s-1)/(s^2+s)$ by choosing k sufficiently large.

We remark that although this gives the best general lower bound we know, slightly better bounds can be obtained for the specific cases $1 \leqslant s \leqslant 3$. For $s = 1$, the lower bound is an 11/9 ratio (see [13]). For $s = 2$ or 3, the lower bound of $s + 11/60$ can be obtained, based on a construction similar to the above but with a different list of tasks for region s (see the 71/60 example in [13]).

4. The Case of Arbitrary $\prec$

We next turn to the situations in which the partial order $\prec$ is allowed to be arbitrary. As is to be expected, the worst-case behavior is considerably worse than for the case of $\prec$ empty.

THEOREM 4. *If L is any list of tasks having an arbitrary partial order $\prec$ in a system with $s \geqslant 1$ resources then*

$$\mathrm{FF}(L) \leqslant (s/2)(L^*)^2 + ((s/2)+1)\,L^*. \tag{27}$$

Proof. Let us first divide L into "big" and "small" tasks. The set of big tasks is

$$X = \{x \in L\colon R_i(x) > 1/2 \text{ for some } i,\ 1 \leqslant i \leqslant s\}.$$

The remaining tasks are small in the sense that for no resource do they have size exceeding 1/2. We can also divide the bins of the FF packing of L into large and small bins, the small bins being those nonempty bins B with $R_i(B) \leqslant 1/2$ for all $1 \leqslant i \leqslant s$. We now make some observations about small and large bins and tasks.

For each k, $1 \leqslant k \leqslant \mathrm{FF}(L)$, let B_k be the bin with index k in the FF packing of L, and define $p(k) = \max\{p < k: B_p \text{ is a small bin}\}$, with $p(k)$ taken as 0 if no such index p exists. In other words, $p(k)$ is the index of the most recent small bin preceding bin B_k, if one exists.

CLAIM 24. *If $y \in L - X$ is in bin B_k and $p(k) > 0$, then there is a task $z \in L$ and an index j, $p(k) \leqslant j < k$, such that z is in B_j and $z \prec y$.*

The claim is immediate, since y, being small, could not be prevented from going in the small bin by any *resource constraints*. By the transitivity of $\prec$, the above claim leads to the following.

CLAIM 25. *If $y \in L - X$ is in bin B_k, $p(k) > 0$, and for no task z in bin $B_{p(k)}$ does $z \prec y$, then there is a big task $x \in X$ and a j, $p(k) < j < k$, such that x is in B_j and $x \prec y$.*

We shall now select a number of disjoint *chains* from L, which will help us characterize the FF packing. A chain of tasks is a sequence of tasks $\langle x_1, x_2, \ldots, x_m \rangle$ where $x_i \prec x_{i+1}$, $1 \leqslant i < m$. The first and last elements of a chain are its *head* and *tail*, respectively.

Choose for the tail of our first chain any task y in the highest indexed small bin. If $\langle y_1, y_2, \ldots, y_m \rangle$ is the chain constructed so far, and y_1 is in small bin B_k, with $p(k) > 0$, consider bin $B_{p(k)}$. If $B_{p(k)}$ contains any task y with $y \prec y_1$, let the updated chain be $\langle y, y_1, y_2, \ldots, y_m \rangle$ and continue. If $B_{p(k)}$ contains no such y, then there must be an $x \in X$ in a bin between $B_{p(k)}$ and B_k such that $x \prec y_1$, by Claim 25. Choose an x of this form which occurs in the earliest bin. Finalize the current chain as $\langle x, y_1, \ldots, y_m \rangle$ and start a *new* chain with any $y \in B_{p(k)}$ as its tail. In this case we call x a "big-head." If $p(k) = 0$ (this can only happen once, after which the process is complete since B_k must be the earliest small bin), we finalize the chain as $\langle y_1, \ldots, y_m \rangle$ and say that it has a "small-head."

By the above procedure, we create a number of chains all but one of which are big-headed, and it is clear that we have the following.

CLAIM 26. (A) *No bin of the* FF *packing contains more than one element which belongs to a chain.*

(B) *Every small bin of the* FF *packing contains a chain element.*

Let $\mathscr{C}$ be the set of all the big-headed chains that we constructed. We can partition the big-headed chains using the following claim.

CLAIM 27. *If $\langle y_1, y_2, \ldots, y_m \rangle \in \mathscr{C}$ is a big-headed chain and y_1 is in bin B_k, then for some i, $1 \leqslant i \leqslant s$, $R_i(B_k) + R_i(B_{p(k)}) > 1$.*

Proof. Since the chain is big-headed, we know that there must be a y_2, and $p(k) > 0$. If y_1 is prevented from going into bin $B_{p(k)}$ by a precedence constraint, let y be the task with $y \prec y_1$ which occurs in the earliest bin B_j, $p(k) \leqslant j < k$. By transitivity of $\prec$, $y \prec y_2$ and so we cannot have $y \in B_{p(k)}$, for that would mean that y should have been chosen for the chain rather than y_1. If $y \in L - X$, then by the above and Claim 25, there must be an $x \in X$ in a bin $B_{j'}$ with $p(k) < j' < j$ and $x \prec y \prec y_1$, a contradiction of our choice of y. Thus we must have $y \in X$, but this too leads to a contradiction, since it means that y should have been chosen as the big-head of the chain, rather than y_1. Thus, no such y exists and y_1 must have been prevented from going into bin $B_{p(k)}$ by some resource constraint and the claim is proved. ∎

Thus, we can assign the big-headed tasks to resources as follows.

$$C_i = \{\langle y_1, \ldots, y_m \rangle \in \mathscr{C}\text{: for the } k \text{ such that } y_1 \in B_k,\ i = \min\{j\text{: } R_j(B_k) + R_j(B_{p(k)}) > 1\}\}.$$

Clearly we will have $\mathscr{C} = \bigcup_{i=1}^{s} \mathscr{C}_i$ and the union is disjoint. Using this partition we can obtain an upper bound on FF(L). Let C_0 be the one small-headed chain, if it exists, and let the length of a chain C (the number of tasks it contains) be denoted by $| C |$.

CLAIM 28.

$$\mathrm{FF}(L) \leqslant | C_0 | + \sum_{i=1}^{s} \left\lceil \sum_{C \in \mathscr{C}_i} | C | + 2(L^* - | \mathscr{C}_i |) - 1 \right\rceil. \tag{28}$$

Proof. Let $\mathscr{B}$ be the set of bins containing elements of chains. By Claim 26, we know that

$$| C_0 | + \sum_{i=1}^{s} \sum_{C \in \mathscr{C}_i} | C | = | \mathscr{B} |$$

and so that sum counts the total number of small bins plus the total number of bins containing big-heads. All the remaining bins of the FF packing must exceed 1/2 in some resource. Call the set of such bins $\mathscr{A}$, and partition $\mathscr{A}$ into $\mathscr{A}_i$, $1 \leqslant i \leqslant s$, where

$$\mathscr{A}_i = \{B \in \mathscr{A}\text{: } i = \min\{j\text{: } R_j(B) > 1/2\}\}.$$

The total amount of resource R_i available to bins of $\mathscr{A}_i$ is clearly bounded above by $L^* - \sum_{B \in \mathscr{B}} R_i(B)$. But note that by Claim 27 and the definition of $\mathscr{C}_i$, $\sum_{B \in \mathscr{B}} R_i(B) > | \mathscr{C}_i |$. Thus, we cannot have $| \mathscr{A}_i | > 2(L^* - | C_i |)$ and the claim follows. ∎

To go from the preceding claim to the desired bound on FF(L), we look at an optimal packing of L which uses precisely L^* bins. For each $x \in X$, let $f(x) = (i, k)$, where $i = \min\{j: R_j(x) > 1/2\}$ and x is contained in bin B_k of the optimal packing. Since no bin can contain two tasks that exceed 1/2 in the same resource, then for distinct $x, y \in X$ we must have $f(x) \neq f(y)$. Moreover, if $f(x) = (i, k)$, the longest chain that x can head has length $L^* - k + 1$. We thus have for each i, $1 \leqslant i \leqslant s$:

$$\sum_{C \in \mathscr{C}_i} |C| + 2(L^* - |\mathscr{C}_i|) - 1 \leqslant \sum_{h=0}^{|\mathscr{C}_i|-1} (L^* - h) + 2(L^* - |\mathscr{C}_i|) - 1$$

$$= \sum_{h=0}^{|\mathscr{C}_i|+1} (L^* - h) = \binom{L^* + 1}{2} - \binom{L^* - |\mathscr{C}_i| - 1}{2} \leqslant \binom{L^* + 1}{2}.$$

Substituting this into Claim 28, together with the fact that we must have $|C_0| \leqslant L^*$ since *no* chain can exceed *that* length, we get

$$\mathrm{FF}(L) \leqslant L^* + \sum_{i=1}^{s} (L^*(L^* + 1)/2) = (s/2)(L^*)^2 + ((s/2) + 1)\, L^*.$$

This proves the theorem. ∎

The upper bound given by Theorem 4 is essentially best possible, as we show by describing a method for constructing lists L for which $\mathrm{FF}(L)/L^*$ is arbitrarily close to $sL^*/2 + 1 + s/2$. Choose an integer $k \geqslant 3$. The list L, which will have $L^* \leqslant k$, is composed of $s + 1$ regions. The regions will be numbered from 0 through s, with all tasks from region i occurring before all tasks from region $i + 1$ in L. Choose ϵ so that $0 < \epsilon < (2sk^2)^{-1}$.

The first region (region 0) is composed of $2k$ tasks in the order

$$w_1, w_1', w_2, w_2', \ldots, w_k, w_k'.$$

The only nonzero resource requirements for these tasks are:

$$R_l(w_i') = s\epsilon, \qquad 1 \leqslant l \leqslant s, \quad 1 \leqslant i \leqslant k.$$

The only precedence constraints involving them are:

$$w_i \prec w_{i+1}, \qquad 1 \leqslant i < k;$$
$$w_i \prec w_j', \qquad 1 \leqslant i < j \leqslant k.$$

Observe that the FF packing will place these tasks in the first k bins, each bin containing some pair w_i and w_i'. The remaining tasks will be constructed so that no other tasks will go in these k bins.

Each of the remaining s regions of L will correspond to one of the s resources and they will all have essentially the same structure. For $1 \leqslant l \leqslant s$, the tasks in region l will occur in the following sequence in L.

$x_l(1), y_l(1, 1), z_l(1, 1),$
$x_l(2), y_l(2, 1), z_l(2, 1), y_l(2, 2), z_l(2, 2),$
$x_l(3), y_l(3, 1), z_l(3, 1), y_l(3, 2), z_l(3, 2), y_l(3, 3), z_l(3, 3),$
$\vdots$
$x_l(k-1), y_l(k-1, 1), z_l(k-1, 1), \ldots, y_l(k-1, k-1), z_l(k-1, k-1).$

The only nonzero resource requirements for these tasks are:

$$\begin{aligned} R_h(z_l(i,j)) &= s\epsilon, \qquad 1 \leqslant h \leqslant s,\ 1 \leqslant j \leqslant i \leqslant k-1, \\ &\qquad\qquad \textit{except } R_l(z_l(k-2, k-2)) = 0; \\ R_l(x_l(i)) &= 1 - (s-1)\epsilon, \qquad 1 \leqslant i \leqslant k-2; \\ R_h(x_l(i)) &= \epsilon, \qquad h \neq l,\ 1 \leqslant h \leqslant s,\ 1 \leqslant i \leqslant k-2; \\ R_l(x_l(k-1)) &= 1. \end{aligned}$$

The precedence constraints involving them are as follows.

$$\begin{aligned} x_l(i) &\prec y_l(i, 1), && 1 \leqslant i \leqslant k-1; \\ y_l(i, j) &\prec y_l(i, j+1), && 1 \leqslant i \leqslant k-1,\ 1 \leqslant j \leqslant i-1; \\ x_l(i) &\prec z_l(i, j), && 1 \leqslant i \leqslant k-1,\ 1 \leqslant j \leqslant i; \\ y_l(i, j) &\prec z_l(i, j+1), && 1 \leqslant i \leqslant k-1,\ 1 \leqslant j \leqslant i-1. \end{aligned}$$

In addition, $x_l(k-1)$ is required to precede all tasks in each region h with $h > l$, except for $x_h(k-1)$.

We now consider the FF packing for the tasks in region 1. First, note that every y_1 task and z_1 task is preceded by some x_1 task. Since each x_1 task requires at least $1 - (s-1)\epsilon$ of resource R_1, no task from region 1 can be placed in any of the first k bins containing tasks from region 0. Observe next that no x_1 task can be placed in a bin containing a z_1 task, because of resource R_1, except that $x_1(k-1)$ and $z_1(k-2, k-2)$ may be placed together. With this fact in mind, it is not difficult to see that the FF packing will place each $x_1(i)$, $i \neq k-1$, by itself in a new bin, followed immediately by i bins which each contain two tasks $y_1(i, j)$ and $z_1(i, j)$. Finally $x_1(k-1)$ will be placed into the same bin with $y_1(k-2, k-2)$ and $z_1(k-2, k-2)$, followed immediately by $k-1$ bins which each contain two tasks $y_1(k-1, j)$ and $z_1(k-1, j)$. Thus the tasks from region 1 will use a total of $(k^2 + k - 4)/2$ bins.

In fact, we claim that the tasks from each remaining region will be placed in an identical manner, with no bin containing tasks from two different regions. This will follow immediately from the definitions of the different regions if we can show that no task from region 2 will be placed in a bin containing a task from region 1. To see this, first observe that $x_2(k-1)$ cannot fit into any earlier bin because of its requirement for R_2. Every other task in region 2 must follow $x_1(k-1)$ because of the precedence constraints. However, each of the $k-1$ bins following the bin containing $x_1(k-1)$ uses $s\epsilon$ of resource R_2, so that no x_2 task can be placed there. Since every y_2 and z_2 task has some x_2 task which must precede it, we see that no task from region 2 can be placed in a bin containing a task from region 1. Thus the tasks from each region i, $1 \leqslant i \leqslant s$, will be placed in a separate set of $(k^2+k-4)/2$ bins.

Combining these facts, we have

$$\mathrm{FF}(L) = s(k^2+k-4)/2 + k.$$

We now must show that $L^* \leqslant k$, by giving a packing of L into k bins. It goes as follows. Bin 1 contains w_1 and all $x_l(k-1)$, $1 \leqslant l \leqslant s$. Bin 2 contains w_2, all $x_l(k-2)$, and all $y_l(k-1, k-1)$. For $2 < i \leqslant k-1$, Bin i contains w_i, all $x_l(k-i)$, and all $y_l(k-j, i-j)$, $1 \leqslant j < i$, and $1 \leqslant l \leqslant s$. Finally, Bin k contains all the remaining tasks: w_k, all w_i' tasks, all z_l tasks, and all $y_l(j,j)$, $1 \leqslant j \leqslant k-1$. We leave to the reader the verification that this packing meets all required constraints.

We conclude that our constructed list L satisfies

$$\frac{\mathrm{FF}(L)}{L^*} \geqslant \frac{s(k^2+k-4)/2+k}{k} \geqslant \left(\frac{s}{2}\right) L^* + \left(\frac{s}{2}\right) + 1 - \frac{2s}{k},$$

which can be made arbitrarily close to $sL^*/2 + s/2 + 1$ by choosing k suitably large.

THEOREM 5. *If L is any list of tasks having an arbitrary partial order $\prec$ and L is arranged in decreasing order by $R_{\max}(x)$ in a system with $s \geqslant 1$ resources, then*

$$\mathrm{FF}(L) \leqslant ((17/10)\, s + 1)\, L^*. \tag{29}$$

Proof. Our proof is analogous to that of Theorems 1 and 2, only we use two new weighting functions based on W and the R_i. For each $x \in L$, we recall the definition $R_{\max}(x) = \max\{R_i(x): 1 \leqslant i \leqslant s\}$. Define $w_1(x) = W(R_{\max}(x))$, $w_2(x) = \sum_{i=1}^{s} W(R_i(x))$. Clearly we must have $w_1(x) \leqslant w_2(x)$ for all $x \in L$. Extend the definitions of w_1, w_2 and $R_{\max}$ to

packed bins and other sets in the standard way, i.e., $w_1(S) = \sum_{x \in S} w_1(x)$, etc. The following inequality is an immediate consequence of Lemma 1.

$$w_2(L) \leqslant (17/10)\, sL^*. \tag{30}$$

Since $w_1(L) \leqslant w_2(L)$, we can complete the proof by showing the following inequality:

$$\mathrm{FF}(L) \leqslant w_1(L) + L^*. \tag{31}$$

Let B_k be the kth nonempty bin of the FF packing of L, $1 \leqslant k \leqslant \mathrm{FF}(L)$, and let $\mathscr{B}$ be the set of bins B for which $w_1(B) < 1$. The following claim will be the mainstay of our proof of (31).

CLAIM 29. *Suppose $x \in L$ is in a bin B_k, $0 \leqslant l < k$, and no bin B_j, $l < j < k$, contains any task y with $y \prec x$. Then $\{B_j : l < j < k\} \cap \mathscr{B} = \varnothing$.*

Proof. Let j be such that $l < j < k$. We must show that $w_1(B_j) \geqslant 1$. But observe that, due to our hypothesis, x cannot have been prevented from going into B_j by any precedence constraints. Thus if we let $Y \subseteq B_j$ be the set of tasks in bin B_j which precede x in the list L, we must have $R_i(Y) > 1 - R_i(x)$ for some i, $1 \leqslant i \leqslant s$. This in turn means that $R_{\max}(Y) > 1 - R_{\max}(x)$. Moreover, by the order of L, for all $y \in Y$ we have $R_{\max}(y) \geqslant R_{\max}(x)$. Thus, either $Y = \{y\}$ and $R_{\max}(Y) = R_{\max}(y) > 1/2$ and so $w_1(B_j) \geqslant w_1(Y) > 1$, or $|Y| \geqslant 2$ in which case Lemma 3 applies with $R = R_{\max}$ and $\alpha = R_{\max}(x)$. (If $\alpha > 1/2$ then the conclusion is also immediate.) ∎

We prove that (31) holds by constructing a chain of tasks which contains, among other tasks, one task from each bin $B \in \mathscr{B}$. We will then have $|\mathscr{B}| \leqslant L^*$ since no chain can have length exceeding L^*. This in turn will imply

$$\mathrm{FF}(L) - L^* \leqslant \mathrm{FF}(L) - |\mathscr{B}| \leqslant w_1(L),$$

from which (31) is immediate.

To construct our chain, we proceed inductively. Let x be any task in the bin $B \in \mathscr{B}$ with highest index. Our initial chain is just $\langle x \rangle$. Suppose the chain so far is $\langle x_1, x_2, \ldots, x_m \rangle$, x_1 is in bin B_k, and for all $B_j \in \mathscr{B}$ with $j \geqslant k$, B_j contains an element of the chain (a hypothesis which is clearly true for the initial chain $\langle x \rangle$). Let $l = \max[\{j < k$: bin B_j contains a task y with $y \prec x_1\} \cup \{0\}]$. If $l = 1$, then by Claim 29, no bin B_j is in $\mathscr{B}$, $0 < j < k$, and so we are done and $\langle x_1, x_2, \ldots, x_m \rangle$ is our desired chain. If $l > 0$, then let $y \in B_l$ be a task with $y < x_1$ and update our current chain to $\langle y, x_1, x_2, \ldots, x_m \rangle$. By Claim 29, the induction hypothesis

will still hold. Since FF(L) is finite the process must terminate, and so we can construct the desired chain and the theorem is proved. ∎

Our best lower bound in the case of Theorem 5 is not quite equal to the upper bound. Crucial to our construction in this case is the sequence of integers $\{a_i\}$ defined by

$$a_1 = 1;$$
$$a_{i+1} = a_i(a_i + 1), \qquad i \geqslant 1.$$

The construction will give lists L, each in the proper nondecreasing order, for which the ratio FF(L)/L^* is arbitrarily close to

$$\left(\sum_{i=1}^{\infty} a_i^{-1}\right) s + 1 = (1.69\ldots)s + 1.$$

The construction begins by choosing integers $B \geqslant 2$ and $t > 0$ with $t + 1 \equiv 0 \pmod{a_1 a_2 \cdots a_B}$. The list L will be divided into $sB + 1$ regions. Each of the first sB regions is indexed by two parameters and referred to as region (l, i), where $1 \leqslant l \leqslant s$ and $1 \leqslant i \leqslant B$. The last region is called region E. All tasks from region (l, i) will occur before all tasks from region (j, k) in L whenever either $i < k$ or, $i = k$ and $l < j$. The tasks in region E follow all other tasks in L.

The sequence of tasks in L from region (l, i) is $y_1(l, i)$, $y_2(l, i)$,..., $y_t(l, i)$, $x(l, i)$. Their only nonzero resource requirements are

$$R_l(y_j(l, i)) = (a_i + 1)^{-1} + \epsilon, \qquad 1 \leqslant j \leqslant t,$$
$$R_l(x(l, i)) = (a_i + 1)^{-1} + \epsilon,$$

for a fixed ϵ satisfying $0 < \epsilon < (Ba_{B+1})^{-1}$. The sequence of tasks in region E is $z_1, z_2, \ldots, z_t$ and all these tasks have requirement 0 for every resource. The precedence constraints are as follows.

$x(l, i) \prec x(j, k)$, if either $i < k$ or, $i = k$ and $l < j$;
$x(l, i) \prec y_h(j, k)$, $1 \leqslant h \leqslant t$, if either $i < k$ or, $i = k$ and $l < j$;
$z_j \prec z_{j+1}$, $1 \leqslant j < t$;
$x(s, B) \prec z_1$.

The FF packing of L will place successive sets of a_i tasks per bin from each region (l, i). No more tasks from that region can be placed in a single bin without exceeding 1 in resource R_i. Furthermore, since $x(l, i)$ will go in the last bin for region (l, i), the precedence constraints insure that no task from another region can be placed in the bins containing tasks from

region (l, i). Finally, the t tasks from region E will go into the last t bins, one per bin by the precedence constraints. Thus, we have

$$\mathrm{FF}(L) = t + s(t+1)\sum_{i=1}^{B} a_i^{-1}.$$

We now give a packing of L which shows that $L^* \leqslant t + sB$. The tasks $x(l, i)$, $1 \leqslant l \leqslant s$, $1 \leqslant i \leqslant B$, go into the first sB bins, one per bin in the order required by the precedence constraints. For $1 \leqslant j \leqslant t$, bin $sB + j$ contains the task z_j along with all the tasks $y_j(l, i)$, $1 \leqslant l \leqslant s$, $1 \leqslant i \leqslant B$. The choice of values for $\{a_i\}$ and ϵ insure that no resource bound will be exceeded, since

$$\sum_{i=1}^{B} (a_i + 1)^{-1} = 1 - (1/a_{B+1}) < 1 - B\epsilon.$$

The reader may check that all other requirements are met by this packing of L.

We therefore have

$$\frac{\mathrm{FF}(L)}{L^*} \geqslant \frac{t + s(t+1)\sum_{i=1}^{B} a_i^{-1}}{t + s \cdot B}$$

$$= 1 + s\sum_{i=1}^{B} a_i^{-1} - \frac{s((sB-1)\sum_{i=1}^{B} a_i^{-1} + B)}{t + sB}.$$

For any fixed s and B, the last term may be made arbitrarily small by choosing t sufficiently large. Thus, the desired result then follows by letting B tend to infinity.

If L is a list with partial order $\prec$ and $x \in L$, define *level* (x) to be the length of the longest chain headed by x. From this definition, it is immediate that $1 \leqslant \mathrm{level}\,(x) \leqslant L^*$ for all $x \in L$.

THEOREM 6. *If L is any list having an arbitrary partial order $\prec$ and L is arranged in decreasing order by level, in a system with $s \geqslant 1$ resources, then*

$$\mathrm{FF}(L) \leqslant ((17/10)\, s + 1)\, L^*.$$

Proof. This proof is remarkably similar to that of Theorem 5, although perhaps this is not so remarkable considering the fact that the upper

bounds are the same. We use the same definitions of $R_{\max}$, w_1 and w_2. Again, one equality is immediate from Lemma 1:

$$w_2(L) \leqslant (17/10)\, sL^*. \tag{32}$$

And again the proof is completed by showing the following inequality.

$$\mathrm{FF}(L) \leqslant w_1(L) + L^*. \tag{33}$$

It is at this point that the proofs of Theorems 5 and 6 diverge. We first note that $x \prec y$ implies level $(x) \geqslant$ level $(y) + 1$. Thus, all x with $x \prec y$ must precede y in list L. Therefore, in the FF packing of L, the assignment of y to a bin cannot be delayed because one of y's predecessors in the partial order has not yet been assigned. Thus, if we let B_k be the kth bin of the FF packing, and define, for each k, $1 \leqslant k \leqslant \mathrm{FF}(L)$,

$$\text{level } (B_k) = \max\{\text{level } (x): x \in B_k\},$$

then we have the following claim.

CLAIM 30. *For all j, k, $1 \leqslant j < k \leqslant \mathrm{FF}(L)$,*

$$\text{level } (B_j) \geqslant \text{level } (B_k). \tag{34}$$

Therefore, there is a sequence of integers $0 = a_1 \leqslant b_1 \leqslant a_2 \leqslant b_2 \leqslant \cdots \leqslant a_{L^*} \leqslant b_{L^*} = L^*$ such that for each l, $1 \leqslant l \leqslant L^*$, $\{B_j : \text{level } (B_j) = l\} = \{B_j : a_l < j \leqslant b_l\}$, a set which we shall call $\mathscr{B}_l$. The next claim will lead directly to (33).

CLAIM 31. $\sum_{B_j \in \mathscr{B}_l} w_1(B_j) \geqslant |\, \mathscr{B}_l \,| - 1$ *for all* l, $1 \leqslant l \leqslant L^*$.

Proof. We may assume that $|\, \mathscr{B}_l \,| \geqslant 2$ since otherwise the result is trivial. Let $L_l = \{x: \text{level } (x) = l \text{ and } x \in B_j \text{ for some } B_j \in \mathscr{B}_l\}$. Each $B_j \in \mathscr{B}_l$ must contain at least one $x \in L_l$. In fact, when the last task in L_l was assigned, the bins in $\mathscr{B}_l$ could have contained *only* elements of L_l, since no task of lower level could yet have been assigned, and no task of higher level can have been in any of the bins (by the definition of $\mathscr{B}_l$). For each $B_j \in \mathscr{B}_l$, let $B_j' = B_j \cap L_l$. Since no member of L_l was prevented from going into any bin of $\mathscr{B}_l$ by a precedence constraint, we must have for all j, k, $a_l < j < k \leqslant b_l$, and any $y \in B_k'$, there is a resource R_i such that $R_i(B_j') + R_i(y) > 1$. Hence we also have $R_{\max}(B_j') + R_{\max}(y) > 1$. Thus for each $y \in L_l$, if $y \in B_k'$,

$$R_{\max}(y) > 1 - R_{\max}(B_j) \qquad \text{for all } j,\ a_l < j < k.$$

Thus, Lemma 5 applies with $R(x) = R_{\max}(x)$ and we have

$$\sum_{B_j \in \mathscr{B}_l} w_1(B_j) \geqslant \sum_{B_j \in \mathscr{B}_l} w_1(B_j') \geqslant |\mathscr{B}_l| - 1$$

and Claim 31 is proved. ∎

By Claim 31, we now can conclude that

$$w_1(L) = \sum_{j=1}^{\mathrm{FF}(L)} w_1(B_j) = \sum_{l=1}^{L^*} \left(\sum_{B_j \in \mathscr{B}_l} w_1(B_j) \right) \geqslant \sum_{l=1}^{L^*} (|\mathscr{B}_l| - 1)$$
$$= \mathrm{FF}(L) - L^*,$$

and so (33) is proved. This proves the theorem. ∎

The upper bound given by Theorem 6 is essentially the best possible. As in proving the lower bound for Theorem 2, this construction will be based on the construction in [13] which gives for each $k \geqslant 1$, a list L_1 with $L_1^* = k$, $s = 1$, $<$ empty, and $\mathrm{FF}(L_1) \geqslant (17/10)\, L_1^* - 8$. An important fact about that construction is that if every task with resource usage 1 is replaced by an identical task with usage $1 - \epsilon$, for a suitably small $\epsilon > 0$, the same packings still result and no bin in the optimal packing contains sets of tasks whose total resource usage exceeds $1 - \epsilon$.

Choose a fixed $k \geqslant 10$ and let $y_1, y_2, \ldots, y_p$ be the list given by the modified construction described above. Let a_i denote the requirement of y_i for that single resource and let $\epsilon > 0$ be such that no bin in the optimal packing has total resource usage exceeding $1 - \epsilon$. Our list L will consist of $s + 1$ regions, with all tasks in each region i occurring before all tasks in region $i + 1$ of L. The sequence of tasks in the first region is $w_1, w_2, \ldots, w_{k-1}, x_1, w_k$. The resource requirements for these tasks are

$$R_l(w_i) = \epsilon, \qquad 1 \leqslant l \leqslant s,\ 1 \leqslant i < k;$$
$$R_l(w_k) = 0, \qquad 1 \leqslant l \leqslant s;$$
$$R_1(x_1) = 1;$$
$$R_l(x_1) = 0, \qquad 2 \leqslant l \leqslant s.$$

Each of the regions 2 through $s + 1$ has a similar structure. The sequence of tasks in region l, $2 \leqslant l \leqslant s + 1$, is $y_l(1), y_l(2), \ldots, y_l(p), x_l, z_l$. The only nonzero resource requirements for these tasks are

$$R_{l-1}(y_l(i)) = a_i, \qquad 1 \leqslant i \leqslant p;$$
$$R_{l-1}(x_l) = 1 - \epsilon.$$

Note that in our indexing there is no y_1 task and no z_1 task.

The precedence constraints are as follows.

$$
\begin{aligned}
w_i &\prec w_{i+1}, && 1 \leqslant i \leqslant k-1;\\
w_k &\prec z_2; &&\\
z_i &\prec z_{i+1}, && 2 \leqslant i \leqslant s;\\
x_l &= x_{l+1}, && 1 \leqslant l \leqslant s;\\
x_l &\prec y_{l+1}(i), && 1 \leqslant l \leqslant s,\ 1 \leqslant i \leqslant p;\\
y_l(i) &\prec z_l, && 2 \leqslant l \leqslant s+1,\ 1 \leqslant i \leqslant p.
\end{aligned}
$$

Notice that the list L is ordered as required in Theorem 6. Each task z_i has level $s+2-i$; each task w_i has level $k+s+1-i$; each task $y_l(i)$ has level $s+3-l$; and each task x_l has level $s+3-l$, except for x_{s+1} which has level 1.

We now examine the FF packing of L. The tasks in region 1 will use the first k bins, with w_1 through w_{k-1} each occupying a single bin and w_k together with x_1 occupying bin k. Since x_1 is required to precede all remaining tasks, none of those remaining tasks can be placed in the first k bins. Next consider the tasks in region 2. By the choice of their requirements for R_1, $y_2(1)$ through $y_2(p)$ will be placed in the next m_2 bins where $m_2 \geqslant (17/10)k - 8$. Since z_2 is required to follow all the $y_2(i)$ tasks, it will be placed in the next bin along with x_2 whose requirement for R_1 prevents it from going in the same bin with any of the $y_2(i)$ tasks (recall from [13] that each a_i exceeds 1/7). Since x_2 is required to precede all remaining tasks, none of those remaining tasks will be placed in the first $k+m_2+1$ bins. The tasks from each of the remaining regions will be packed in the same manner, the tasks in region l using m_l+1 bins where $m_l \geqslant (17/10)k - 8$. Thus, we have

$$\mathrm{FF}(L) = k + \sum_{l=2}^{s+1} (m_l + 1) \geqslant k + (17/10)ks - 7s.$$

We next give a packing for L which shows that $L^* \leqslant k+s+2$. Let $I_1, I_2, \ldots, I_k$ be an optimal packing of the tasks $y_1, y_2, \ldots, y_p$ from the construction in [13], where each I_i is the set of indices of the tasks which go into bin i. Our packing for L is as follows. For $1 \leqslant i \leqslant s+1$, bin i contains the task x_i. For $3 \leqslant i \leqslant k+2$, bin i contains the task w_{i-2}. For $k+3 \leqslant i \leqslant k+s+2$, bin i contains the task z_{i-k-1}. Finally, for each i and l, $1 \leqslant i \leqslant k$ and $2 \leqslant l \leqslant s+1$, the tasks in the set $\{y_l(t): t \in I_i\}$ go into bin $l+i$. We omit the straightforward verification that this is a valid packing for L.

We conclude that

$$(\mathrm{FF}(L)/L^*) \geqslant (k + (17/10)\,ks - 7s)/(k + s + 2)$$
$$= 1 + (17/10)\,s - ((s + 2)(1 + (17/10)\,s - 7s)/(k + s + 2).$$

The last term can be made arbitrarily close to zero by choosing k sufficiently large, which gives the desired result.

5. Concluding Remarks

The major results presented in this paper generalize the bin packing results of [11, 13] to the problem of packing vectors into vector-capacity bins, with or without precedence constraints. We can also generalize the results of [11, 13] for the case where the resource requirements for individual tasks are restricted, in particular, for the case when the range of each $R_i : L \to [0, 1]$ is constrained to lie in the smaller range $[0, 1/n]$ for a fixed $n \geqslant 2$. The best bounds on $\lim_{k\to\infty} (\widehat{\mathrm{FF}}(k)/k)$ currently known are given by the following (proofs omitted).

(A) If $s \geqslant 1$, $\prec$ is empty, and $R_i : L \to [0, 1/n]$, $1 \leqslant i \leqslant s$, for $n \geqslant 2$, then

(1) $\lim_{k\to\infty} (\widehat{\mathrm{FF}}(k)/k) = s + (1/n)$,

(2) $$s + \frac{s + n - 1}{(s + n)(s + n + 1)} \leqslant \lim_{k\to\infty} \frac{\widehat{\mathrm{FFD}}(k)}{k} \leqslant s + \frac{1}{n + 1},$$

(B) If $s \geqslant 1$, $\prec$ is any partial order, and $R_i : L \to [0, 1/n]$, for $1 \leqslant i \leqslant s$ and $n \geqslant 2$, then

(1) $$\lim_{k\to\infty} \frac{\widehat{\mathrm{FF}}(n)}{k} = \left(\frac{n}{n - 1}\right) s + 1,$$

(2) $$\left(1 + \frac{n + 3}{n^2 + 3n + 2}\right) s + 1 < \lim_{k\to\infty} \frac{\widehat{\mathrm{FFD}}(k)}{k} \leqslant \left(\frac{n + 1}{n}\right) s + 1,$$

(3) $$\left(\frac{n + 1}{n}\right) s \leqslant \lim_{k\to\infty} \frac{\widehat{\mathrm{FFL}}(k)}{k} \leqslant \left(\frac{n + 1}{n}\right) s + 1.$$

Slightly better lower bounds than that given in (A)(2) are known when $s + n \leqslant 3$. The lower bound in (B)(2) is not quite the best known, the actual bound being a rather complicated limit, similar in construction to the lower bound for this algorithm in the general resource case.

One may also ask about the worst-case behavior of these algorithms when the number of tasks per bin is limited. This situation corresponds in the multiprocessing interpretation of the problem to the case when there is a fixed number of processors, say m, and hence only m tasks may be simultaneously executed. In [15] this case is studied when there is only one resource. The problem for general $s \geqslant 1$ remains open, although [16] gives some weak upper bounds, and we do have the following result. If $m = rsL^*$ where $r \in [0, 1]$, then in the case where $\prec$ can be arbitrary,

$$\lim_{k\to\infty} \frac{\mathrm{FF}(k)}{k} = \frac{1-(1-r)^2}{2} sL^{*2} + \frac{r}{2} sL^* + rL^*,$$

a generalization of our Theorem 4. (Of course, there is also the trivial upper bound $\mathrm{FF}(L) \leqslant mL^*$.)

Appendix

Proof of Lemma 1. Suppose $B = \{b_1, b_2, \ldots, b_m\} \subseteq L$ and $\sum_{i=1}^{m} R(b_i) \leqslant 1$. We must show that $\sum_{i=1}^{m} W(R(b_i)) \leqslant 17/10$, and in fact is bounded by $3/2$ if no b_i has $R(b_i) > 1/2$. The latter bound is immediate, since the reader may verify that $W(\alpha) \leqslant (3/2)\,\alpha$ for all $\alpha \in [0, 1/2]$. For the first bound, let us assume that $R(b_1) > 1/2 > R(b_2) \geqslant \cdots \geqslant R(b_m)$. Since the slope of W is the same in the region $[0, 1/6]$ and $(1/2, 1]$, we can replace b_1 without loss of generality by four tasks, b_1', c_1, c_2, and c_3, where $R(b_1) = 1/2 + \epsilon$, and $R(c_1) = R(c_2) = R(c_3) = [R(b_1) - (1/2 + \epsilon)]/3$ for $1/2 + \epsilon < R(b_1)$. Moreover, since the slope of W is also the same in the region $[1/3, 1/2]$ as it is in $[0, 1/6]$, we can replace any b_i with $R(b_i) \in [1/3, 1/2]$ by two tasks b_i' and c_4, with $R(b_i') = 1/3$ and $R(c_4) = R(b_i) - 1/3$. Furthermore, if neither $R(b_i)$ nor $R(b_j)$ exceeds $1/6$ they can be combined into a single task and $\sum_i W(R(b_i))$ will not decrease (in fact it may increase). We consequently have reduced the proof to the consideration of four cases:

(1) $m = 2$, $R(b_2) \leqslant 1/3$,

(2) $m = 3$, $1/6 < R(b_3) \leqslant R(b_2) \leqslant 1/3$,

(3) $m = 3$, $R(b_3) \leqslant 1/6 < R(b_2) \leqslant 1/3$, and

(4) $m = 4$, $R(b_4) \leqslant 1/6 < R(b_3) \leqslant R(b_2) \leqslant 1/3$.

In each case, since $W(R(b_1)) = 1 + (6/5)\,\epsilon$ and ϵ can be as small as we like, all we need to show is that

$$\sum_{i>2} W(R(b_i)) \leqslant 7/10.$$

This is immediate in (1) since there we will have

$$W(R(b_2)) \leqslant W(1/3) = 1/2.$$

In (2),

$$\begin{aligned} W(R(b_2)) + W(R(b_3)) &= (9/5)[R(b_2) + R(b_3)] - 1/5 \\ &\leqslant (9/5)(1/2) - 1/5 = 7/10. \end{aligned}$$

For (3),

$$\begin{aligned} W(R(b_2)) + W(R(b_3)) &= (6/5)\, R(b_3) + (9/5)\, R(b_2) - 1/10 \\ &\leqslant (6/5)(1/6) + (9/5)(1/3) - 1/10 = 7/10. \end{aligned}$$

And finally, in (4),

$$\begin{aligned} \sum_{i=2}^{4} W(R(b_i)) &= (6/5)\, R(b_4) + (9/5)[R(b_3) + R(b_2)] - 1/5 \\ &= (9/5) \sum_{i=2}^{4} R(b_i) - (3/5)\, R(b_4) - 1/5 < 9/10 - 1/5 = 7/10, \end{aligned}$$

since

$$\sum_{i=2}^{4} R(b_i) < 1/2.$$

This proves the lemma. ∎

Proof of Lemma 2. Suppose $B = \{b_1, b_2, \ldots, b_m\} \subseteq L$ and $\sum_{i=1}^{m} R(b_i) > 1$. Since $W(\alpha) \geqslant (6/5)\,\alpha$ for all $\alpha \in [0, 1]$, we immediately have (a):

$$\sum_{i=1}^{m} W(R(b_i)) > 6/5.$$

If one of the b_i, say b_1, has $R(b_1) > 1/2$, then

$$\begin{aligned} \sum_{i=1}^{m} W(R(b_i)) &= 1 + (6/5)[R(b_1) - 1/2] + \sum_{i=2}^{m} W(R(b_i)) \\ &> 1 + (6/5)(1/2) = 8/5, \end{aligned}$$

and so (b) holds. The lemma is proved. ∎

Proof of Lemma 3. Since for all $\alpha \in [0, 1]$, $W(\alpha)$ as defined here is at least as large as the $W(\alpha)$ defined in [13], this lemma follows immediately from the proof of Claim 2.2.3 in that paper. ∎

Proof of Lemma 4. Let $\alpha \in [0, 1/2]$, $R(b_1) \geqslant R(b_2) \geqslant \cdots \geqslant R(b_m) > \alpha$ for $B = \{b_1, b_2, \ldots, b_m\} \subseteq L$ and

$$\sum_{i=1}^{m} W(R(b_i)) = 1 - \beta \qquad \text{for some} \quad \beta > 0.$$

If $m = 1$, then (a) must hold, since $R(b_1) > 1/2$ would imply $W(R(b_1)) > 1$ by the definition of W. So assume $m \geqslant 2$, in which case we must prove

(b) $$\sum_{i=1}^{m} R(b_i) \leqslant 1 - \alpha - (5/6)\beta.$$

Let

$$\sum_{i=1}^{m} R(b_i) = 1 - \alpha - \gamma.$$

By Lemma 3 we cannot have

$$\sum_{i=1}^{m} R(b_i) \geqslant 1 - \alpha,$$

so we know that $0 < \gamma < 1$. Form a list $\bar{L} = L \cup \{d_1, d_2, d_3, d_4, d_5, d_6\}$ and extend the domain of R to $\bar{L}$ by letting $R(d_i) = \gamma/6$, $1 \leqslant i \leqslant 6$. Then the set

$$C = \{b_1, b_2, \ldots, b_m, d_1, d_2, \ldots, d_6\} \subseteq \bar{L}$$

has

$$R(b_1) \geqslant R(b_2) > \alpha$$

and

$$\sum_{c \in C} R(c) = 1 - \alpha,$$

so that Lemma 3 applies to it and yields

$$\sum_{c \in C} W(R(c)) \geqslant 1.$$

Since for $1 \leqslant i \leqslant 6$, $R(d_i) = \gamma/6 < 1/6$, we have by definition of W that $W(R(d_i)) = (6/5)\, R(d_i)$. Thus

$$1 \leqslant \sum_{i=1}^{m} W(R(b_i)) + \sum_{i=1}^{6} W(R(d_i))$$
$$= \sum_{i=1}^{m} W(R(b_i)) + (6/5) \sum_{i=1}^{6} R(d_i) = 1 - \beta + (6/5)\gamma,$$

and so $\gamma \geqslant (5/6)\,\beta$. Thus

$$\sum_{i=1}^{m} R(b_i) = 1 - \alpha - \gamma \leqslant 1 - \alpha - (5/6)\beta,$$

as desired. ∎

Proof of Lemma 5. Suppose $Y \subseteq L$ and $\mathscr{B} = \{B_1, B_2, \ldots, B_t\}$ is a partition of Y into disjoint nonempty sets such that for all i and j with $1 \leqslant i < j \leqslant t$, $b \in B_j$ implies $R(b) > 1 - R(B_i)$. We wish to establish a lower bound on $\sum_{y \in Y} W(R(y))$ in terms of $t = |\mathscr{B}|$. For each B_i, $1 \leqslant i \leqslant t$, define

$$\gamma_i = \max\left\{0, \sum_{b \in B_i} W(R(b)) - 1\right\}$$

and

$$\delta_i = \max\left\{0, 1 - \sum_{b \in B_i} W(R(b))\right\}.$$

Since $\mathscr{B}$ is a partition of Y, we then have

$$\sum_{y \in Y} W(R(y)) = t + \sum_{i=1}^{t} \gamma_i - \sum_{i=1}^{t} \delta_i .$$

The desired lower bound will follow if we can show that $\sum_{i=1}^{t} \delta_i \leqslant 1$.

Let $\mathscr{C} = \{B_i \in \mathscr{B}: \delta_i > 0\}$ and relabel the sets in $\mathscr{C}$ as $C_1, C_2, \ldots, C_m$, with the associated δ_i's appropriately relabelled also, and with the sets C_i retaining the same relative order they hold in $\mathscr{B}$. All we need now is to show that $\sum_{i=1}^{m} \delta_i \leqslant 1$.

For each $C_i \in \mathscr{C}$, define the *coarseness* α_i of C_i to be $\max\{\alpha$: for some j, $1 \leqslant j < i$, $R(C_j) = 1 - \alpha\}$, with α_1 taken to be 0. By our assumption about the sets B_i, we know that for $1 \leqslant i \leqslant m$ and all $b \in C_i$, $R(b) > \alpha_i$. This in turn means that Lemma 4 applies to each C_i, with $\alpha = \alpha_i$ and $\beta = \delta_i$, and so either 4(a) or 4(b) must hold for each C_i. If 4(a) were to hold for some C_i, $1 \leqslant i < m$, we would have $R(C_i) \leqslant 1/2$ and hence $\alpha_{i+1} \geqslant 1/2$. Thus C_{i+1} would have to contain a task c with $R(c) > 1/2$ and hence $W(R(c)) > 1$, a contradiction to our choice of the C_i's. Thus, 4(b) holds for $1 \leqslant i < m$ and we have $R(C_i) \leqslant 1 - \alpha_i - (5/6)\,\delta_i$ for these i. But this means that $\alpha_{i+1} \geqslant 1 - R(C_i) \geqslant \alpha_i + (5/6)\,\delta_i$, for $1 \leqslant i < m$. Thus

$$\sum_{i=1}^{m-1} \delta_i \leqslant (6/5) \sum_{i=1}^{m-1} (\alpha_{i+1} - \alpha_i) = (6/5)(\alpha_m - \alpha_1).$$

Moreover, $\delta_m < 1 - (6/5)\,\alpha_m$ since C_m contains at least one task b, and that task must have $R(b) > \alpha_m$ and hence $W(R(b)) > (6/5)\,\alpha_m$. Thus,

$$\sum_{i=1}^{m} \delta_i < (6/5)(\alpha_m - \alpha_1) + 1 - (6/5)\,\alpha_m = 1 - (6/5)\,\alpha_1 = 1$$

and the lemma is proved. ∎

REFERENCES

1. A. V. AHO, J. E. HOPCROFT, AND J. D. ULLMAN, "The Design and Analysis of Computer Algorithms," Chap. 10, Addison-Wesley, Reading, Mass., 1974.
2. A. K. CHANDRA AND C. K. WONG, Worst case analysis of a placement algorithm related to storage allocation, *SIAM J. Computing* **4** (1975), 249–263.
3. E. G. COFFMAN, JR. (Ed.), "Computer and Job-Shop Scheduling Theory," Wiley, New York, 1975.
4. M. R. GAREY AND R. L. GRAHAM, Bounds for multiprocessor scheduling with resource constraints, *SIAM J. Computing* **4** (1975), 187–200.
5. M. R. GAREY AND D. S. JOHNSON, Complexity results for multiprocessor scheduling under resource constraints, *SIAM J. Computing* **4** (1975), 397–411.
6. R. L. GRAHAM, Bounds for certain multiprocessing anomalies, *Bell System Tech. J.* **45** (1966), 1563–1581.
7. R. L. GRAHAM, Bounds on multiprocessing timing anomalies, *SIAM J. Appl. Math.* **17** (1969), 416–429.
8. T. C. HU, Parallel scheduling and assembly line problems, *Operations Res.* **9** (1961), 841–848.
9. O. H. IBARRA AND C. H. KIM, "Fast approximation algorithms for the knapsack and sum of subset problems," *J. Assoc. Comput. Mach.*, to appear.
10. D. S. JOHNSON, Near-optimal bin packing algorithms, Doctoral thesis, M. I. T., Cambridge, Mass., 1973.
11. D. S. JOHNSON, Fast algorithms for bin packing, *J. Comput. System Sci.* **8** (1974), 272–314.
12. D. S. JOHNSON, Approximation algorithms for combinatorial problems, *J. Comput. System Sci.* **9** (1974), 256–278.
13. D. S. JOHNSON, A. DEMERS, J. D. ULLMAN, M. R. GAREY, AND R. L. GRAHAM, Worst-case performance bounds for simple one-dimensional packing algorithms, *SIAM J. Computing* **3** (1974), 299–325.
14. R. M. KARP, Reducibility among combinatorial problems, *in* "Complexity of Computer Computations" (R. E. Miller and J. W. Thatcher, Eds.), pp. 85–104, Plenum Press, New York, 1972.
15. K. L. KRAUSE, Analysis of computer scheduling with memory constraints, Doctoral thesis, Computer Science Department, Purdue University, 1973.
16. A. C. YAO, Scheduling unit-time tasks with limited resources, *in* "Proceedings Sagamore Computer Conference," 1974.