

NEAR-OPTIMAL BIN PACKING ALGORITHMS

by

DAVID STIFLER JOHNSON

B.A., Amherst College

1967

S.M., Massachusetts Institute of Technology

1968

SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENTS FOR THE

DEGREE OF DOCTOR OF

PHILOSOPHY

at the

MASSACHUSETTS INSTITUTE OF

TECHNOLOGY

June, 1973

Signature of Author.....

Department of Mathematics

May 18, 1973

Certified by.....

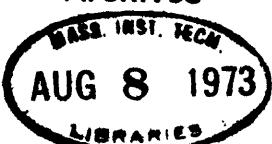
Thesis Supervisor

Accepted by.....

Chairman, Department Committee

on Graduate Students

Archives



ABSTRACT

The Bin Packing problem is a model for a number of problems occurring in industry and computer science. Suppose we are given a list of pieces with sizes between 0 and 1, and a sequence of unit-capacity bins. Our goal is to pack the pieces into as few bins as possible. All known algorithms for finding optimal solutions to this problem require exponential time. In this thesis we study instead algorithms which generate near-optimal solutions and which run in low-order polynomial time.

The previously analyzed FIRST FIT and BEST FIT packing rules belong to a more general class of packing rules, the AAF packing rules, which we show all have the same worst case behavior. We extend these results to include the case when numbers in the input list are restricted to any given sub-interval of $(0,1]$. No algorithm which implements any packing rule in the class can use less than $O(n \log n)$ time, and we give implementations for several of the rules, including FIRST FIT and BEST FIT, which realize this bound. We then introduce linear time algorithms whose worst case behavior is the same as that of the above algorithms in many restricted situations and is never known to be worse.

It has previously been shown that if the input list is in decreasing order, FIRST FIT can asymptotically require no more than $5/4$ times the optimal number of bins. We show that this result extends to an even larger class of algorithms, the AF algorithms, and generalize our proof to get results for lists with numbers in restricted ranges. We then show that in fact FIRST FIT and BEST FIT asymptotically require no more than $11/9$ times the optimal number of bins, and this is the best bound possible. Other algorithms are suggested that may do even better.

Finally, we report the results of an empirical study using randomly generated lists to get a picture of the average case behavior of the algorithms studied. The average case behavior is very much better than our worst case results might have led one to expect.

ACKNOWLEDGMENTS

The author wishes to thank Ron Graham and Mike Garey for their interest in these results and suggestions for simplifying some of the proofs. Thanks also to Daniel Kleitman who provided encouragement and a receptive ear, and to Albert Meyer who originally suggested that I pursue this topic, whose intuitions led to some of my earlier results, and who supplied me with the phrase "notorious for their computational intractability," which I liked so much that I have managed to use it in every paper I have written since, including this one. In particular, I wish to express my thanks to my advisor Michael Fischer, who has been a demanding task-master but in so being has helped me clarify many of my ideas, and also taught me a lot about the writing of readable proofs, some of which hopefully shows in this thesis. He also has provided intuitions that led to new results, and was the one who saw how my implementation of GROUP FIT led to an nlogn implementation of FIRST FIT. And finally, I should like to dedicate this thesis to my wife Susan, whose patience, understanding, and chocolate chip cookies have seen me through.

TABLE OF CONTENTS

INTRODUCTION.....	6
CHAPTER 1. DEFINITIONS AND ILLUSTRATIONS.....	15
SECTION 1.1. Definitions and Worst Case Measures.....	15
SECTION 1.2. NEXT FIT as an Illustration.....	23
CHAPTER 2. ANY FIT ALGORITHMS.....	39
SECTION 2.1. FIRST FIT, BEST FIT, and their Generalizations.....	39
SECTION 2.2. The Interval Problem.....	76
SECTION 2.3. Implementation of ANY FIT Algorithms.....	91
CHAPTER 3. ANY FIT DECREASING ALGORITHMS COMPARED.....	105
SECTION 3.1. Preprocessing Rules.....	105
SECTION 3.2. A Proof Method.....	108
SECTION 3.3. Results for Arbitrary ANY FIT Algorithms..	119
SECTION 3.4. Results for FF, BF, and Less Restricted Lists.....	149
CHAPTER 4. WORST CASE BEHAVIOR OF AFD ALGORITHMS.....	162
SECTION 4.1. Lower Bounds.....	162
SECTION 4.2. Weak Upper Bounds for all S ∈ AF.....	167
CHAPTER 5. EXACT UPPER BOUNDS.....	209
SECTION 5.1. A Weighting Function.....	209
SECTION 5.2. The 71/60 Theorem.....	253
SECTION 5.3. The 11/9 Theorem.....	273
CHAPTER 6. OTHER OFF-LINE ALGORITHMS.....	344
SECTION 6.1. The GROUPING Rule.....	344
SECTION 6.2. The INCREASING Rule.....	362
SECTION 6.3. More Complicated Algorithms.....	367
CHAPTER 7. EMPIRICAL TESTS OF AVERAGE CASE BEHAVIOR.....	375
SECTION 7.1. The Experimental Environment.....	375
SECTION 7.2. Average Case Results.....	380
BIBLIOGRAPHY.....	398
BIOGRAPHY.....	400

FIGURES AND TABLES

FIGURE 1.1	...	25	FIGURE 5.1	...	215
FIGURE 1.2	...	28	FIGURE 5.2	...	220
FIGURE 1.3	...	34	FIGURE 5.3	...	276
FIGURE 1.4	...	37	FIGURE 5.4	...	286
			FIGURE 5.5	...	287
FIGURE 2.1	...	53	FIGURE 5.6	...	291
FIGURE 2.2	...	56	FIGURE 5.7	...	299
FIGURE 2.3	...	66	FIGURE 5.8	...	299
FIGURE 2.4	...	67	FIGURE 5.9	...	300
FIGURE 2.5	...	71			
FIGURE 2.6	...	72	FIGURE 6.1	...	346
FIGURE 2.7	...	73	FIGURE 6.2	...	348
FIGURE 2.8	...	74	FIGURE 6.3	...	350
FIGURE 2.9	...	75	FIGURE 6.4	...	352
FIGURE 2.10	...	79	FIGURE 6.5	...	357
FIGURE 2.11	...	94	FIGURE 6.6	...	364
			FIGURE 6.7	...	369
FIGURE 3.1	...	121	FIGURE 6.8	...	372
FIGURE 3.2	...	127	FIGURE 6.9	...	373
FIGURE 3.3	...	130			
FIGURE 3.4	...	135	FIGURE 7.1	...	377
FIGURE 3.5	...	136	FIGURE 7.2	...	377
FIGURE 3.6	...	138	FIGURE 7.3	...	377
FIGURE 3.7	...	140	FIGURE 7.4	...	388
FIGURE 3.8	...	150	FIGURE 7.5	...	388
FIGURE 3.9	...	150	FIGURE 7.6	...	393
FIGURE 3.10	...	151	FIGURE 7.7	...	393
FIGURE 3.11	...	154	FIGURE 7.8	...	395
FIGURE 3.12	...	156			
FIGURE 3.13	...	158			
FIGURE 3.14	...	160			
FIGURE 3.15	...	160			

TABLES

FIGURE 4.1	...	163	TABLE 5.1	...	311
FIGURE 4.2	...	165	TABLE 5.2	...	319
FIGURE 4.3	...	166	TABLE 6.1	...	366
FIGURE 4.4	...	169	TABLE 7.1	...	381
FIGURE 4.5	...	170	TABLE 7.2	...	384
FIGURE 4.6	...	175	TABLE 7.3	...	386
FIGURE 4.7	...	178	TABLE 7.4	...	390
FIGURE 4.8	...	184			

INTRODUCTION

Suppose you are a carpenter whose job is to build a large and complicated structure. Your plans call for a large number of various length pieces of wood, but your local lumberyard only sells standard 8 foot boards. Your problem: figure out how few of the boards you can buy and still be able to cut all the required lengths from them.

As a humble carpenter, you may not realize it, but you are faced with a problem that occurs in more or less complicated guises in many industrial situations [Br1]. Steel companies that supply steel bars of different lengths for reinforcing concrete; paper companies that produce rolls of newsprint cut to widths as specified by a wide range of customers; even television networks that wish to pack as many commercials into station breaks as possible; all must come to grips with the same issues. Computer programmers may find themselves with similar worries in situations where memory space is costly and comes in standard sized portions, such as tracks on a disk or pages in low speed memory.

The problem has been called among other things "stock-cutting," "box-packing," and, as we shall refer to it here, "bin packing." In its simplest form it can be formally described as follows:

Suppose we are given a list $L = \langle a_1, a_2, \dots, a_n \rangle$ of numbers in the range $(0,1]$, and a sequence of unit-capacity bins, BIN_1, BIN_2, \dots , extending from left to right. Find an assignment of numbers to bins so that for no bin does the sum of the numbers assigned to it exceed 1, and such that the number of bins used, that is, with at least one number assigned to them, is minimized.

Unfortunately, as a carpenter you have little use for an abstract formulation of your difficulties, and knowing that you are in good company is not much solace. The only way you can see to find the minimal number of boards required is to examine all possible ways of cutting boards into your lengths, and then choose the best one. You can't think of any substantial short-cuts. It looks like a lot of work.

Such an offhand judgment of the complexity of the problem turns out to be supported by theoretical results. The bin packing problem is a member of a class of problems first described by Cook [Co1] and Karp [Ka1], the polynomial complete problems. This class is such that if there is a method for solving any member of the class that takes time bounded by a polynomial in the size of the input, then every one of the problems has such a polynomial time solution. Since problems

notorious for their computational intractability, such as the traveling salesman problem and the question of whether a logical expression is a tautology, populate this class, it seems probable that in fact no member of the class can be solved in polynomial time.

Indeed, although the industrial bin packers do have money to spend on computers and IBM experts who can apply mathematical programming techniques to the problem [Gal], they have found that although their investment pays for itself on reasonably small problems, the time required by even these sophisticated methods appears to grow exponentially with an increase in problem size [Br1].

Thus your carpenter's common sense is probably correct when it tells you to forget about figuring out the absolute minimum number of boards required, and to just use some simple heuristics that intuitively would seem to be of help in keeping the wastage down. You buy a board and cut the first length on your list of specifications from it, starting a scrap pile with the portion of the board left over. As you continue, you keep the scrap pile ordered by size, and cut each new length from the smallest scrap which is still long enough. If all scraps are too short, you buy a new board. After a little experience with this method, you find that you get better results if you first preorder your list of specifications so that the lengths are in

decreasing order, with largest first.

When you are done, you notice that only about 2% of the total amount of wood you had to pay for is left in bits and pieces on the floor, and as you sweep it up, you think to yourself, "This is minimal enough for me. Who needs mathematical experts?"

Although the basic heuristic just described has been known for quite some time under the name of BEST FIT [Kn1], it is only in the last few years that researchers have attempted a rigorous analysis of this and other intuitive heuristics that carpenters and others might use when they are willing to settle for "almost minimal." In the context of the abstract bin packing problem, Ullman [U11] and Demers [De1] studied BEST FIT and a similar heuristic known as FIRST FIT, and showed that these could require as many as 70% excess bins, but never any more than that. Garey, Graham, and Ullman [Ga1] studied the modification to the method in which the list of lengths is first put in decreasing order, and showed that this method could require as many as 22.2% ($2/9$) excess bins, but no more than 25%. These worst case results are a far cry from the observed average case behavior, but they do have practical significance.

A bound of 25% gives a rigorous assurance that the method can never be terribly bad. Moreover, knowing that more than

one algorithm has the same worst case behavior allows the practical bin packer to tailor his choice of algorithm to his particular situation. In the same vein, knowing that a simple basic algorithm is sound, he may add special case heuristics designed to meet the exigencies of the moment, and still have confidence in the resulting algorithm, even though it is now far too complicated to analyze. And the proofs of the worst case results, dealing as they must with the issue of what it is about an input that can cause the algorithm to behave poorly, can provide insights into precisely what these "exigencies of the moment" are.

There is also theoretical importance in these results, for they are examples of a fairly new approach to studying hard problems, and can serve as a case study in how to analyze heuristics whose goal is only to generate near-optimal solutions. Some of the first work in bounding the worst case behavior of near-optimal algorithms was done as early as 1966 by Graham [Gr1,Gr2], for a problem in multiprocessor scheduling quite similar to the bin packing problem. However to date most hard combinatorial problems have only been studied with the goal of finding algorithms for optimal solutions. This author has presented some tentative work on simple heuristics for polynomial complete problems [Jo1], but the field is still wide open, and hopefully the wealth and variety of results about bin

packing will help inspire others.

It is hoped that this thesis may serve as a sort of compendium for this "wealth and variety of results." The research herein reported extends the previous results for FIRST and BEST FIT to entire classes of algorithms, studies the relation between the range of the numbers in the input list and the behavior of the algorithms, and introduces new and faster algorithms whose worst case behavior is comparable to that of those already studied. In addition, it presents the results of extensive empirical tests of the various algorithms, to indicate how the worst case results actually do compare to the average results on randomly generated lists.

Although in a general introduction of this type it would not be appropriate to go into detailed statements of all of our results, we can present a general picture of the organization of this rather massive opus:

The first two chapters are devoted to algorithms which, like our original BEST FIT, operate on-line, without preordering the input list. Chapter 1 presents formal definitions of list, packing, and worst case behavior, and illustrates these by analyzing a fairly simple algorithm. Chapter 2 then turns to FIRST and BEST FIT, and shows how they belong to a large class

of algorithms, all with the same worst case behavior. It also characterizes the worst case behavior of these algorithms when the numbers in the input list are restricted to an arbitrary subinterval of $(0,1]$. Finally, all algorithms in the class are shown to require $O(n \log n)$ time, where n is the length of the input list, and linear time approximations to the algorithms are introduced whose worst case behavior is just as good.

Chapters 3 thru 5 deal with the above algorithms in the case when the list is initially put in decreasing order. In Chapter 3, we show that under certain restrictions on the range of the numbers, many of these algorithms will give nearly identical results when applied to the same list. These results serve as useful lemmas for the next two chapters, which return to the task of putting bounds on worst case behavior. Chapter 4 shows that the 25% bound for BEST FIT DECREASING extends to an even larger class of algorithms than did the on-line result, and finds a corresponding bound for the case when no number in the input list is larger than a given maximum. Chapter 5 presents what might be termed the major result of this thesis, by erasing the gap between the 25% and 22.2% upper and lower bounds for FIRST and BEST FIT DECREASING, and showing that in fact neither algorithm can require more than $11/9$ times the optimal number of bins.

Chapter 6 briefly treats a number of other off-line

algorithms, including linear time algorithms which improve on those presented in Chapter 2, and suggests some algorithms which might be even better than FIRST FIT DECREASING. Included in this chapter is a table which summarizes all our worst case results and in so doing ranks the various algorithms.

Chapter 7 puts the rest of the thesis in perspective by presenting the results of empirical tests of the various algorithms on randomly generated lists.

One of our hopes for this thesis has failed to be realized. This was that the proofs of our results, and the ideas involved in them, might be of use to researchers investigating other problems. Unfortunately, the best proofs we could find have turned out to be quite domain dependent, and even worse, the major proofs are exceedingly long. Although we are not prepared to say that such length and complexity are inherent in the nature of the problem, we fear this may well be the case.

Thus the reader will be faced with some rather difficult proofs, especially in Chapters 3, 4, and 5. We have attempted where possible to present the intuitions behind our constructions, but under the necessity of insuring that the results be at least checkable step by step, and while operating within time constraints made worse by the sheer bulk of this thesis, some of these intuitions may have become lost in the

INTRODUCTION - Page 14

details. Thus an interested reader may find it fruitful to skim through the thesis as a first approximation, reading some of the shorter arguments, and only then tackle the details of any of the major proofs he may be particularly interested in.

CHAPTER 1. DEFINITIONS AND ILLUSTRATIONS

SECTION 1.1. Definitions and Worst Case Measures

In this section we shall present formal definitions of list, packing, bin packing algorithm, and a method for describing worst case behavior, along with a variety of auxiliary concepts that will be of use in the presentation of our results. In the next section we will illustrate these definitions by studying a simple algorithm, NEXT FIT, whose worst case behavior can be analysed in a straightforward manner, and then show how a slight modification of the algorithm can cause a marked improvement in this worst case behavior.

Our definitions will be considerably more complicated than our original statement of the problem in the Introduction would have suggested. This will make it easier for us to describe the ideas involved in our proofs.

A list L will consist of a finite set $\text{PIECES}(L)$ of pieces, together with two maps:

$$\begin{aligned} \text{rank}_L: \text{PIECES}(L) &\longrightarrow \{1, 2, \dots, |\text{PIECES}(L)|\}, \\ \text{size}_L: \text{PIECES}(L) &\longrightarrow (0, 1], \end{aligned}$$

where $\text{rank}_L(x)$ is a 1-1 function which gives the rank (i.e., position) of piece x in L , and $\text{size}(x)$ gives its size. Usually we can drop the subscript L without causing confusion. By $W(L)$ we will mean $\sum_{x \in \text{PIECES}(L)} \text{size}(x)$.

This definition corresponds to our former notion of a list as a sequence of real numbers, for we may also write $L = \langle a_1, \dots, a_n \rangle$, where $n = |\text{PIECES}(L)|$, and $a_i = \text{size}_L(\text{rank}_L^{-1}(i))$. We shall use the two notations interchangeably, the latter being especially useful for giving examples of lists, such as " $L = \langle .01, .5, .25, .01 \rangle$," and for talking about the concatenation $L_1 \bullet L_2$ of two lists L_1 and L_2 .

Our definition of a packing will have considerably more structure to it than the simple notion of a map from pieces to bins, but can easily be shown to be equivalent: A packing P of a list L will consist of a natural number $N(P)$, a set of positions

$$\text{POS}(P) = \{(j, h) : 1 \leq j \leq N(P), 1 \leq h\}$$

and a 1-1, onto partial function

$$\text{piece}_P : \text{POS}(P) \dashrightarrow \text{PIECES}(L),$$

satisfying

(1) For each j , $1 \leq j \leq N(P)$,

$$\sum_{(j,h) \in \text{Domain}(\text{piece}_P)} \text{size}(\text{piece}(j,h)) \leq 1.$$

An ordered packing of L obeys the additional properties:

(2) $(j,h) \in \text{Domain}(\text{piece}_P)$ and $h > 1$

$$\implies (j,h-1) \in \text{Domain}(\text{piece}_P),$$

(3) $(j,h), (j,h') \in \text{Domain}(\text{piece}_P)$ and $h > h'$

$$\implies \text{rank}(\text{piece}_P(j,h)) \geq \text{rank}(\text{piece}_P(j,h')).$$

We say that $\{(j,h) : h \geq 1\}$ is the set of positions in BIN_j , with $(j,1)$ the bottom position. If $(j,h) \in \text{Domain}(\text{piece}_P)$ then we say that position (j,h) is filled, and that $\text{piece}_P(j,h)$ fills it and is contained in BIN_j . Properties (2) and (3) assert that in an ordered packing the pieces contained in BIN_j fill the bottom-most positions, and do so by order of rank. Most of the packings we consider will be ordered packings of some list (though not necessarily the original list in its original order). We shall often display packings by means of pictures,

with the bins drawn as upright rectangles, the pieces as smaller rectangles fitted within the bins by order of position, $\text{piece}_P(j,1)$ on the bottom. See for instance Figures 1.1 and 1.2 in the next section.

In order to describe the contents of a bin, we introduce the following notation:

$$\text{cont}_P(i) = \{b : b = \text{piece}(j,h) \text{ for some } h\}$$

$$\text{height}_P(i) = |\text{cont}_P(i)|,$$

$$\text{level}_P(i) = \sum_{b \in \text{cont}_P(i)} \text{size}(b), \text{ and}$$

$$\text{gap}_P(i) = 1 - \text{level}_P(i).$$

BIN_j is used or non-empty if $\text{level}_P(j) > 0$, otherwise it is empty. The rightmost non-empty bin in a packing will be called the last bin.

Note that to any packing P of a list L there corresponds an ordered packing P' of L which uses the same bins and has $\text{cont}_{P'}(j) = \text{cont}_P(j)$ for all BIN_j used in P . Merely reassign the pieces within each bin so that packing properties (2) and (3) are satisfied.

A segment P_1 of packing P will be any set of consecutive bins from P . $\#P_1$ is the number of nonempty bins in P_1 .

$\text{cont}(P_1) = \{x : x \in \text{cont}_P(j) \text{ and } \text{BIN}_j \text{ in } P_1\}$. $W(P_1) = \sum_{x \in \text{cont}(P_1)} \text{size}(x)$. Since P may be considered a segment of itself,

all these definitions apply to P as well, and so $\#P$ is the number of bins used in P .

In general a bin packing algorithm is any method which, given a list L , produces a packing P of L . However, the algorithms we shall spend most of our time studying all fit into a specific two-part format and will be called two-part algorithms. The first part will be a reordering of the list according to the preprocessing rule of the algorithm, the second the generation of a packing of the reordered list using the packing rule of the algorithm. A two-part algorithm is completely specified by giving these two rules, and the precise way in which they are applied is as follows:

PART 1 (Reordering the list via the preprocessing rule):

Reorder L (construct a new rank function) to obtain a permutation $L' = \langle b_1, b_2, \dots, b_n \rangle$ that obeys the preprocessing rule.

PART 2 (Generation of a packing of L' via the packing rule):

1. Let P be a sequence of n empty bins. Set $i = 1$. P may be thought of as a packing of the empty list, and in what follows, as a packing of $\langle b_1, \dots, b_{i-1} \rangle$, with $\text{height}_P(j)$ and $\text{level}_P(j)$ defined for P as above. At this initial step we have $\text{height}_P(j) = \text{level}_P(j) = 0$, $1 \leq j \leq n$.

2. If $i > n$, halt and return P .
3. Let $b \in \text{PIECES}(L')$ be that piece with rank i . Using the packing rule of the algorithm, which bases its decision solely on $\text{size}(b)$ and the current values of $\text{level}_P(j)$, choose a j such that

b will fit in BIN_j (such that $\text{level}_P(j) + \text{size}(b) \leq 1$).

4. Set $P =$ the packing of $\langle b_1, \dots, b_i \rangle$ obtained by putting b in position $(j, \text{height}_P(j)+1)$ of P and leaving all other pieces in their old positions. Set $i = i+1$ and go to 2.

If the algorithm in question is S , then a packing resulting from the above will be called an S -packing of L . Note that it is in fact an ordered packing of L' , since each piece goes in the first unfilled position above the positions in the bin already filled with earlier pieces in the list.

Similarly, if S is any bin-packing algorithm (not necessarily two-part), an S -packing of L will be any packing generable by S applied to L . We then define

$$S(L) = \max_{\{P\}} P : P \text{ is an } S\text{-packing of } L$$

Note that we have explicitly allowed for the possibility that S , whether two-part or not, may not be completely determined, and hence might yield a number of different packings depending on

which choices are made at the points in the algorithm when the next step is not uniquely determined. However, since no packing of L can use more than $|PIECES(L)|$ bins, there must be some S -packing which uses precisely $S(L)$ bins.

To measure how "good" a particular packing P of L is, we can compare $\#P$ with

$$L^* = \min_{\{P: P \text{ is a packing of } L\}} \#P$$

A packing P of L for which $\#P = L^*$ (there must be at least one) will be called an optimal packing. We shall evaluate algorithms as to the worst case behavior of the ratio $S(L)/L^*$. For an algorithm S , let

$$\begin{aligned} R[S](k) &= \max_{\{L: L \text{ is a list with } L^* = k\}} S(L)/L^*, \\ R[S] &= \limsup R[S](k). \end{aligned}$$

$R[S]$ is an asymptotic measure. Such a measure is required because for small L^* , edge effects may have a large effect on the ratio. In all our results, the \limsup could actually be shown to be a limit, although there might well be algorithms for which this is not the case.

We shall also present results about restricted lists. If $X \subseteq (0,1]$, then we define

$$R[S, X](k) = \max_{\{L: Range(size_L) \subseteq X \text{ and } L^* = k\}} S(L)/L^*,$$

$$R[S, X] = \limsup R[S, X](k).$$

A simple special case will be when $X = (0, t]$ for $t \leq 1$, in which case we will write $R[S, t]$ for $R[S, (0, t)]$. Note that $R[S, 1] = R[S]$.

LEMMA 1.1. Suppose S is a bin packing algorithm and $X \subseteq (0, 1]$.

(A) If there exists a K such that for all L with $Range(size_L) \subseteq X$, $S(L) \leq rL^* + K$, then $R[S, X] \leq r$.

(B) If there exists a K' and a sequence of lists L_n with $\lim_{n \rightarrow \infty} L_n^* = \infty$, $Range(size_{L_n}) \subseteq X$, and $S(L_n) \geq rL_n^* - K'$, then $R[S, X] \geq r$.

Proof. We shall just prove (A). (B) is similar. Since for all L with pieces in the required range, $S(L) \leq rL^* + K$, we have $R[S, X](k) \leq r + K/k$, and hence

$$R[S, X] = \limsup R[S, X](k) \leq r. \quad \square$$

SECTION 1.2. NEXT FIT as an Illustration

In this and the next chapter, we shall restrict our attention to on-line algorithms, by which we mean two-part algorithms in which the preprocessing rule is vacuous, and hence PART 1 yields $L' = L$. Such algorithms are hence completely specified by their packing rules. In this section we shall examine a particularly simple one, so as to illustrate the definitions presented in the previous section, without having to face the complicated proof methods required for the more interesting algorithms to follow. The NEXT FIT packing rule is given by

NEXT FIT (NF): Let $j = \text{MAX}\{j': \text{BIN}_{j'} \text{ is used in } P\}$. If b fits in BIN_j ($\text{size}(b) + \text{gap}_P(j) \leq 1$), choose j , else choose $j+1$.

Note that this algorithm is not only on-line in the sense that the pieces could be input one at a time and packed as they are received, but it is also on-line with respect to the bins, in that we are through with each bin as soon as the next one is started. The value of $R[NF]$ can be precisely determined:

THEOREM 1.2. $R[NF] = 2.$

Proof. For the upper bound, let P_N be the NF-packing of a list L (There is only one possible packing since NF is completely determined). If BIN_j is not the last bin in P_N , then

$$\begin{aligned} \text{level}_{P_N}(j) + \text{level}_{P_N}(j+1) \\ > 1 - \text{gap}_{P_N}(j) + \text{size}(\text{piece}_{P_N}(j+1,1)) \\ > 1. \end{aligned}$$

Thus, $W(L) > \lfloor \#P_N/2 \rfloor$. Since no bin in an optimal packing of L can contain pieces whose total size exceeds 1.0 by packing property (1), we must have $L^* \geq W(L)$. Thus we can conclude that $NF(L) = \#P_N \leq 2L^* + 1$, and the upper bound follows by Lemma 1.1.

For the lower bound, consider lists of the form

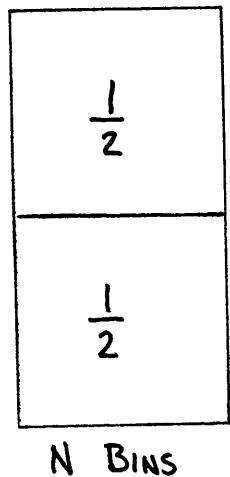
$$L = \langle 1/2, \langle 2N-1 \text{ repetitions of } \langle 1/(2N), 1/2 \rangle \rangle \rangle.$$

Figure 1.1 shows optimal and NF-packings of such lists, demonstrating that $L^* = N+1$, and $NF(L) = 2N$. The bound again follows by Lemma 1.1. \square

Note that since the ϵ in Figure 1.1 can actually be made arbitrarily small, we have the following:

OPTIMAL PACKING

$$L^* = N + 1$$



NF - PACKING

$$NF(L) = 2N$$

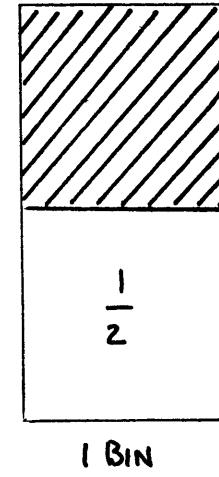
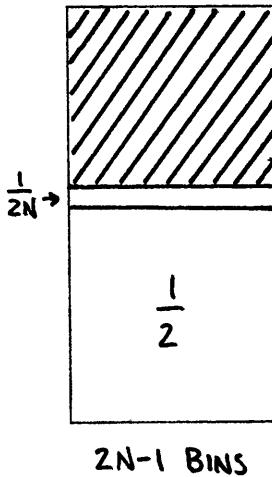
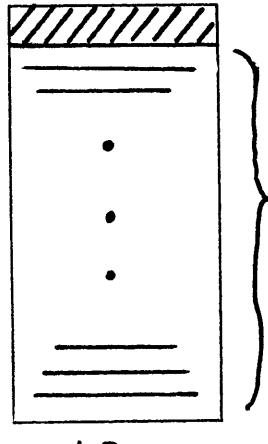


FIGURE 1.1. Lists L for which $NF(L)/L^* \rightarrow 2$.

COROLLARY 1.2.1. $R[NF, t] = 2$, for $1/2 < t \leq 1$.

NEXT FIT, like all the algorithms we shall study, has the property that if we restrict the maximum piece size to $1/2$ or smaller, we can achieve a considerable improvement in worst case behavior. The practical significance of this is that, by increasing the bin size with respect to maximum piece size, we may lower the proportion of the total capacity of the bins in the packing which goes unused. For instance, if we are cutting various length pieces of wood from a standard size board, we can expect to waste less wood if we double the size of the standard length. The result for NEXT FIT is given in general form by the following theorem:

THEOREM 1.3. $R[NF, t] = 1/(1-t)$, for $0 < t \leq 1/2$.

Proof. For the upper bound, let PN be a packing of a list L with $\text{Range}(\text{size}_L) \subseteq (0, t]$. Then since no piece in L has size exceeding t , all bins in PN except the last must have level exceeding $1-t$, and hence $W(L) > (1-t)(\#PN - 1)$, so that $L^* \geq W(L) > (1-t)(\#PN - 1)$ and the bound follows by Lemma 1.1.

For the lower bound, choose any $\epsilon > 0$ and let n be large enough so that $(1-2t)/n \leq \text{MIN}(\epsilon, t)$. Consider lists of the form

$L = \langle 2N \text{ repetitions of } \langle t, \langle n+1 \text{ reps of } (1-2t)/n \rangle \rangle \rangle,$

Figure 1.2 shows optimal and NF-packings of such lists.

Note that each bin in the NF-packing has level $t + n[(1-2t)/n] + (1-2t)/n \leq 1 - t + \epsilon$, and so we must have $W(L) \leq NF(L)[1-t+\epsilon]$.

The optimal packing is as shown, since the number of pieces of size $(1-2t)/n$ is $2N(n+1) > Nn$, which is the number of such pieces required to fill up the N bins containing two pieces of size t , each of which requires exactly n of the pieces. the first N bins all have level 1, and all the remaining bins except the last have as many pieces of size $(1-2t)/n$ as will fit and hence have level exceeding $1-\epsilon$. Thus $W(L) \geq (L^* - 1)(1-\epsilon)$.

Therefore

$$\begin{aligned} NF(L) &\geq (L^* - 1)[1-\epsilon]/[1-t+\epsilon] \\ &\geq (L^* - 1)[1/(1-t) - 6\epsilon] \end{aligned}$$

(since $t \leq 1/2$), and so by Lemma 1.1, $R[NF, t] \geq 1 - t - 6\epsilon$.

Since ϵ can be arbitrarily small, the lower bound follows. \square

NEXT FIT can be generalized by the following packing rule:

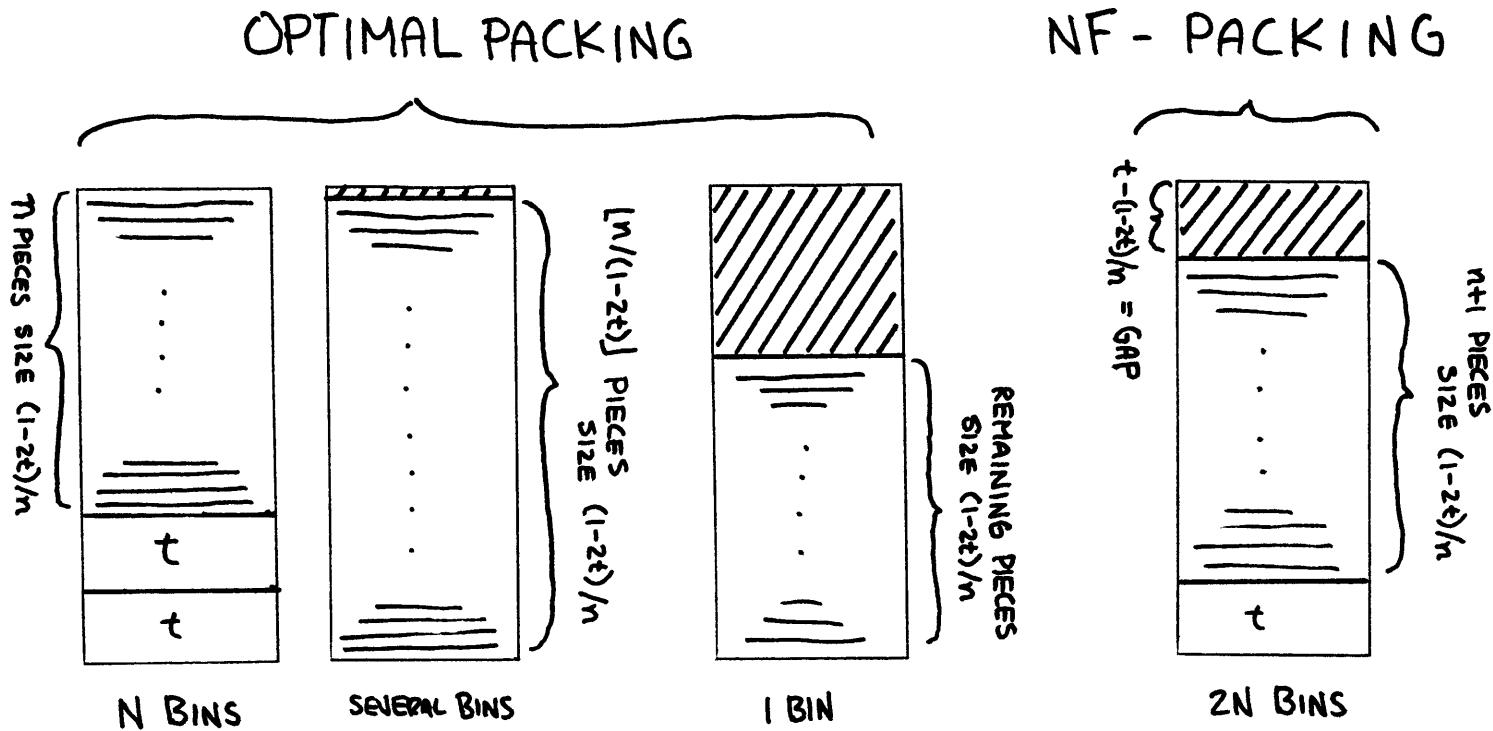


FIGURE 1.2. Lists L with $\text{Range}(\text{size}_L) \subseteq (0, t]$, $t \leq 1/2$,
and $\text{NF}(L)/L^* \dashrightarrow 1/(1-t)$.

NEXT- k FIT (NkF): Choose

$$j = \min\{j' : \text{size}(b) \leq \text{gap}_P(j') \text{ and } j' > \max\{0, #P - k\}\}.$$

NEXT FIT is thus the same as NEXT-1 FIT. For $k \geq 2$, we get results which are a distinct improvement over those for NEXT FIT, however once $k = 2$, no further increase in k causes any further improvement in worst case behavior. The following Lemma tells us a key fact about NkF-packings which will allow us to derive the value of $R[NkF, t]$ for $k \geq 2$ and $t \leq 1/2$.

LEMMA 1.4. Suppose $k \geq 2$, $m \geq 2$, and P_N is an NkF-packing of a list L with $\text{Range}(\text{size}_L) \subseteq (0, 1/m]$. Then if $j < #P_N$,

- (A) $\text{level}_{P_N}(j) > (m-1)/m$ and
- (B) BIN_j contains at least m pieces of size exceeding $\max\{\text{gap}_{P_N}(j') : j' > 0 \text{ and } j-k < j' < j\}$.

Proof. Since $j < #P_N$, position $(j+1, 1)$ is filled in P_N . Let $b = \text{piece}_{P_N}(j+1, 1)$. When b was assigned, we must have had $\text{level}_P(j) + \text{size}(b) > 1$, so since $\text{size}(b) \leq 1/m$, we must have had $\text{level}_P(j) > (m-1)/m$. Thus, since no piece has size $> 1/m$, BIN_j must have at that time already contained at least m pieces. By the NkF rule, these all must have had size exceeding $\text{gap}_P(j')$, for all $j' > 0$ such that $j-k < j' < j$. Since for no bin does the gap ever increase, the lemma follows. \square

THEOREM 1.5. Suppose $k \geq 2$, $0 < t \leq 1/2$, and $m = \lfloor 1/t \rfloor$.

Then $R[NkF, t] = 1 + 1/m$.

Proof. For the upper bound, let P_N be the NkF -packing of a list L obeying with $\text{Range}(\text{size}_L) \subseteq (0, t]$. We use Lemma 1.4 to show that the following induction hypothesis holds for all j , $0 \leq j \leq \#P_N - 2$.

$$(H1.5) \text{ Either } \sum_{j'=1}^j \text{level}_{P_N}(j') \geq jm/(m+1) \text{ or} \\ \sum_{j'=1}^{j+1} \text{level}_{P_N}(j') \geq (j+1)m/(m+1).$$

Since $\sum_{j=1}^0 \text{level}_{P_N}(j') = 0 = 0 \cdot [m/(m+1)]$, (H1.5) holds vacuously for $j = 0$. Suppose it holds for $j \leq \#P_N - 3$. If $\sum_{j'=1}^{j+1} \text{level}_{P_N}(j') < (j+1)m/(m+1)$, then by (H1.5) $\sum_{j'=1}^j \text{level}_{P_N}(j') \geq jm/(m+1)$, so $\text{level}_{P_N}(j+1) < m/(m+1)$. Let $\text{gap}_{P_N}(j+1) = d$. We thus have $d = 1/(m+1) + \epsilon$ for some $\epsilon > 0$, so by Lemma 1.4, BIN_{j+2} contains at least m pieces of size exceeding d and hence $\text{level}_{P_N}(j+2) > md = m/(m+1) + m\epsilon$. Thus

$$\begin{aligned} & \sum_{j'=1}^{j+2} \text{level}_{P_N}(j') \\ &= \sum_{j'=1}^j \text{level}_{P_N}(j') + \text{level}_{P_N}(j+1) + \text{level}_{P_N}(j+2) \\ &> jm/(m+1) + m/(m+1) - \epsilon + m/(m+1) + m\epsilon \end{aligned}$$

$> (j+2)|m/(m+1)|,$

and so (H1.5) holds for $j+1$.

By induction (H1.5) holds for $j = \#PN-2$, and hence $L^* \geq W(L) > (\#PN-2)m/(m+1)$, and the upper bound follows by Lemma 1.1.

We now turn to the task of giving examples to show that the lower bound holds. (Theorem 3 of [Ga1], although it is presented without proof and refers to a different algorithm, suggests that similar examples were known to the authors of that paper. However, we have not seen their examples and this is essentially an independent result.) We will first consider the special case when $1/3 < t \leq 1/2$ and hence $m = \lfloor 1/t \rfloor = 2$, and then show how to generalize our examples to show the bound holds for $0 < t \leq 1/3$. To prove that the bound holds in the special case, we present for each $N > 0$ lists L for which $L^* = 2N + 1$, and yet $NkF(L) = 3N$. From these it will follow by Lemma 1.1 that $R[NkF, t] \geq 3/2 = 1 + 1/m$.

First, let us choose a d such that $0 < d < 1/t - 1/3$. Then, for $1 \leq i \leq N$, let $d(i) = d \cdot 6^{-i}$. There will be $6N$ pieces which we shall index $a_{k,j,i}$, for $1 \leq k \leq 2$, $1 \leq j \leq 3$, and $1 \leq i \leq N$. If we let $a(k,j,i) = \text{size}(a_{k,j,i})$, the sizes of the pieces are given by:

$$\begin{aligned}
 a(1,1,i) &= 1/3 + 5d(i) \\
 a(2,1,i) &= 1/3 - 3d(i) \\
 a(1,2,i) &= 1/3 + d(i) \\
 a(2,2,i) &= 1/3 + d(i) \\
 a(1,3,i) &= 1/3 - d(i) \\
 a(2,3,i) &= 1/3 + 2d(i)
 \end{aligned}$$

The ranking of the pieces will be determined by $\text{rank}(a_{k,j,i}) < \text{rank}(a_{k',j',i'}) \iff i < i'$, or $i = i'$ and $j < j'$, or $i = i'$, $j = j'$, and $k < k'$. In sequence notation we thus have

$$L = \langle a(1,1,1), \dots, a(2,3,1), a(1,1,2), \dots, a(2,3,2), \dots, a(1,1,N), \dots, a(2,3,N) \rangle,$$

During the generation of the NKF-packing P, pieces $a_{1,j,i}$ and $a_{2,j,i}$ will fill up a new bin for each j and i, and we will have

$$\begin{aligned}
 \text{level}_P(3i-2) &= a(1,1,i) + a(2,1,i) = 2/3 + 2d(i), \\
 \text{level}_P(3i-1) &= a(1,2,i) + a(2,2,i) = 2/3 + 2d(i), \text{ and} \\
 \text{level}_P(3i) &= a(1,3,i) + a(2,3,i) = 2/3 + d(i).
 \end{aligned}$$

This happens because after the first bin has two pieces, all remaining pieces in the list exceed $1/3 - 2d(i)$ in size, and after the third has two pieces, all remaining pieces exceed $1/3 - 3d(i+1) = 1/3 - d(i)/2$ in size. Thus $NkF(L) = 3N$, no matter what $k > 1$ is. Figure 1.3 indicates how the packing would look.

However, now note that

$$\begin{aligned} a(2,1,i) + a(2,2,i) + a(2,3,i) &= 1, \text{ and} \\ a(1,1,i) + a(1,2,i) + a(1,3,i-1) &= 1, \text{ for } i > 1. \end{aligned}$$

Thus the pieces, with the exception of $a_{1,1,1}$, $a_{1,2,1}$, and $a_{1,3,N}$, can fit three to a bin, for a total of $2N-1$ bins. The three exceptions will require two additional bins between them, and thus we have $L^* = 2N + 1$.

For the general case, with $1/(m+1) < t \leq 1/m$ for $m \geq 2$, we present for each $N > 0$ a list L with $NkF(L) = (m+1)N$ and $L^* = mN + 1$, with the desired bound again following by Lemma 1.1.

This time we choose d such that $0 < d < 1/t - 1/(m+1)$, and we let $d(i) = d \cdot [m(m+1)]^{-i}$ for $1 \leq i \leq N$. Then we let the $m(m+1)$ pieces for each i have sizes

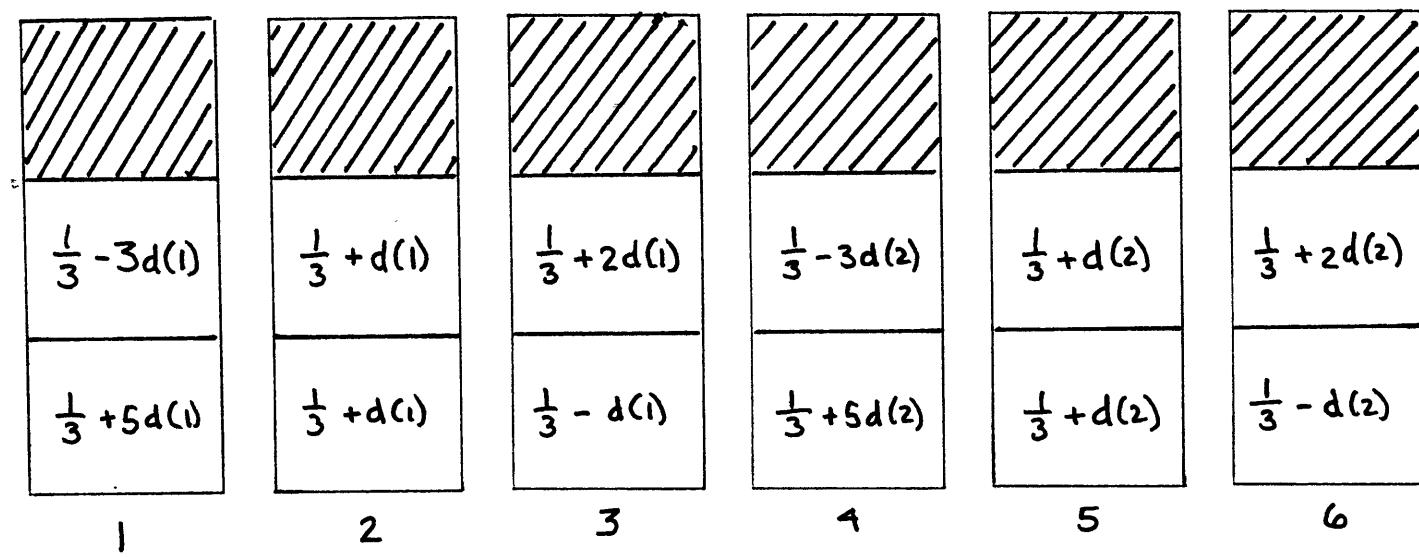


FIGURE 1.3. NkF-Packing of list for Theorem 1.5 lower bound.

$$a(1,1,i) = 1/(m+1) + (m^2+1)d(i),$$

$$a(k,1,i) = 1/(m+1) - (m+1)d(i), \text{ for } 2 \leq k \leq m,$$

$$a(1,m+1,i) = 1/(m+1) - d(i), \text{ and}$$

$$a(k,j,i) = 1/(m+1) + d(i), \text{ for all } j, k,$$

$1 \leq j \leq m+1, 1 \leq k \leq m$, except those mentioned above.

Now if we order these pieces in an analogous way to the way we ordered the pieces in the special case, NkF will yield a packing in which all bins are filled only a little above the $m/(m+1)$ level, and $(m+1)N$ bins are used. This is because, for each i we first have

$$\begin{aligned} \sum_{k=1}^m a(k,1,i) &= m/(m+1) + [(m^2+1)-(m-1)(m+1)] \\ &= m/(m+1) + 2d(i), \end{aligned}$$

and there are no pieces left less than $1/(m+1) - d(i)$. Then

$$\begin{aligned} \sum_{k=1}^m a(k,j,i) &= m/(m+1) + md(i), \text{ for } 2 \leq j \leq m, \text{ and} \\ \sum_{k=1}^m a(k,m+1,i) &= m/(m+1) + (m-2)d(i). \end{aligned}$$

for a total of $m+1$ bins for each i , yielding a grand total of $NkF(L) = (m+1)N$.

On the other hand, as in the special case, we can fit the pieces together much more efficiently, because for $2 \leq k \leq m$, $i \geq 1$, we have

$$\sum_{j=1}^{m+1} a(k, j, i) = 1 + [m - (m+1)]d(i) = 1 - d(i),$$

and for all $i > 1$,

$$\sum_{j=1}^m a(1, j, i) + a(m+1, 1, i-1) = 1.$$

The only pieces left out of the $mN-1$ bins thus formed are $a_{1,m+1,N}$ and $a_{1,1,1}$ thru $a_{1,m,1}$, and they require only two additional bins. Thus the optimal packing yields $L^* = mN + 1$.

□

As a result of Theorems 1.3 and 1.5, we can conclude that for $k \geq 2$, NEXT- k FIT has better worst case behavior than NEXT FIT for almost every t in the interval $(0, 1/2]$. Figure 1.4 illustrates this by graphing $R[NkF, t]$ and $R[NF, t]$ as functions of t in the given interval. We do not know the value of $R[NkF, t]$ for $k \geq 2$ and $t \in (1/2, 1]$, however the upper bound arguments in the proof of Theorem 1.2 clearly apply to NkF as well as NF , and so we know that $R[NkF, t] \leq 2$ for all such t . Examples we shall present in the next chapter will show that $R[NkF, t] \geq 1.7$ for such t , and we conjecture that this is the actual value.

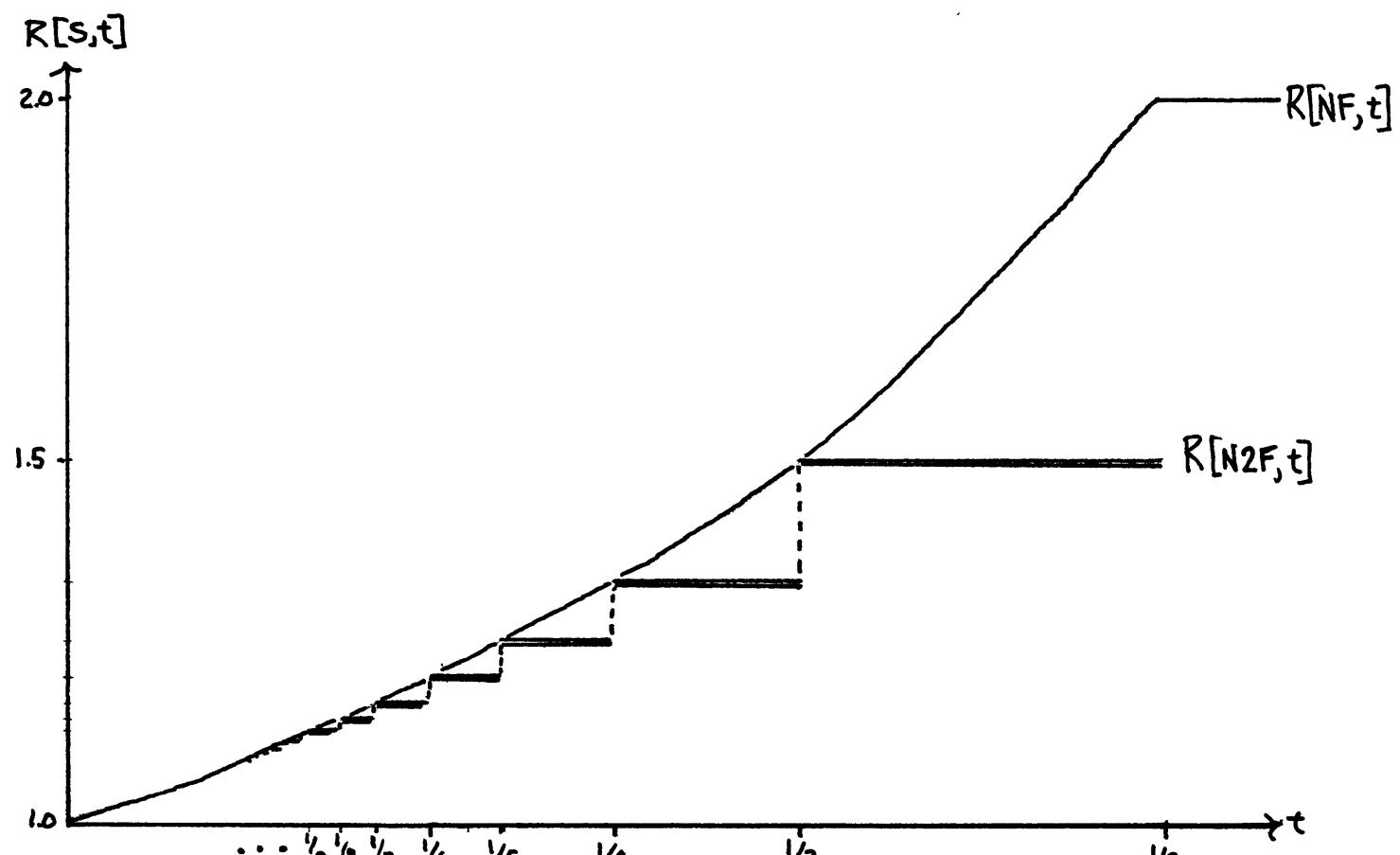


FIGURE 1.4. Comparison of $R[NF, t]$ and $R[N2F, t]$, $0 < t \leq 1/2$.

To conclude this section, we consider the question of the ratio $L^*/W(L)$. Since the upper bound proofs in Theorems 1.2 and 1.5 bounded $NF(L)$ and $NkF(L)$ in terms of $W(L)$, and $L^* \leq S(L)$ for all algorithms S , we have

COROLLARY 1.5.1. For all L with $\text{Range}(\text{size}_L) \subseteq (0, t]$, $t > 1/2$, $L^* \leq \lceil 2W(L) \rceil - 1$, and there are arbitrarily long lists obeying these size restrictions which attain this bound.

For all L with $\text{Range}(\text{size}_L) \subseteq (0, t]$ for $m = \lfloor 1/t \rfloor \geq 2$, $L^* \leq (1 + 1/m)W(L) + 2$, and there are arbitrarily long lists obeying these size restrictions with $L^* = \lceil (1 + 1/m)W(L) \rceil - 1$.

Proof. The upper bounds follow as stated from the upper bound proofs in Theorems 1.2 and 1.5. For an arbitrarily long list attaining the first lower bound, choose ϵ such that $0 < \epsilon < 1/t - 1/2$, and consider lists of length $2N$ made up entirely of pieces of size $1/2 + \epsilon$. For the second lower bound, choose ϵ such that $0 < \epsilon < 1/t - 1/(m+1)$ and consider lists of length mN made up entirely of pieces of size $1/(m+1) + \epsilon$. \square

CHAPTER 2. ANY FIT ALGORITHMS

SECTION 2.1. FIRST FIT, BEST FIT, and their Generalizations

In the last chapter we introduced a sequence of algorithms, NEXT- k FIT for $k \geq 1$. Although all the NEXT- k FIT algorithms for $k \geq 2$ were shown to have the same worst case behavior over lists with maximum piece size $t \leq 1/2$, and conjectured to have the same worst case behavior no matter what the maximum piece size, it seems likely that their average case behavior might improve as k increased. This is because under NkF no bin becomes inaccessible to further pieces until k additional bins have been started, and thus each bin would be able to accept new pieces (if they fit) for a longer period of time if k were larger. From this point of view, it would be ideal if no bin ever became inaccessible to new pieces.

In order to guarantee that no bin ever becomes inaccessible during the generation of the NkF-packing of a particular list L , it would be sufficient to take $k = |\text{PIECES}(L)|$; however, in order to guarantee the same thing for all lists L , no matter how long, we would have to use a limiting algorithm, which might be called NEXT- ∞ FIT, but which has already been introduced in the literature [Gal,U11] as

FIRST FIT (FF): Choose

$j = \text{MIN}\{j' : \text{level}_P(j') + \text{size}(b) \leq 1\}$ (Assign b to the leftmost bin into which it will fit).

In order to obtain a possible improvement in average case behavior, we thus sacrifice our ability to output bins after a sufficient number of new ones have started, and hence must keep all of them around to the bitter end. This will have consequences as far as the time required by an implementation of the algorithm, but we shall postpone our discussion of such matters until Section 2.3. The question of average case behavior will be studied in Chapter 7.

A second algorithm which has been studied [Gal,De1] and in which all bins remain accessible until the packing is completed, is given by the packing rule

BEST FIT (BF): Choose that j for which $\text{level}_P(j) + \text{size}(b)$ is closest to, without exceeding, 1. (Assign b so as to maximize the level of the bin into which it goes.) If more than one j yields this maximum, choose the least one.

The last part of the rule is not essential, but will aid us in our later proofs by rendering the rule completely determined. The results discovered by the earlier researchers [Gal,Ull,De1]

can be summarized in our terminology by the following:

THEOREM: $R[FF, t] = R[BF, t] = 1 + 1/m$, for $m = \lfloor 1/t \rfloor \geq 2$, and
 $R[FF, t] = R[BF, t] = 17/10$, for $1/2 < t \leq 1$.

We see that except for the $17/10$ result, these are exactly the same bounds as we had for NEXT- k FIT, $k \geq 2$, as we might have suspected for FIRST FIT. It is almost equally reasonable that they should hold for BEST FIT, for in a sense that algorithm is almost as much a generalization of NEXT-2 FIT as is FIRST FIT, since NEXT-2 FIT is not concerned with the bins other than the last two, and among these two the first fit will be the same as the best fit, at least until the second bin has level exceeding $1-t$. This suggests that the behavior of BEST and FIRST FIT does not depend on their particular choice strategies, but on the fact that they obey certain general constraints.

One constraint they both clearly obey is the following:

CONSTRAINT 1: If BIN_j is empty, j cannot be chosen unless b will not fit in any bin to the left of BIN_j .

Constraint 1 will often be referred to as the ANY FIT Constraint, and we will denote by AF the set of all packing rules which obey it. Any such packing rule, when thought of as

an on-line algorithm, will be called an ANY FIT algorithm.

The first thing we can observe about AF algorithms is the following [De1]:

LEMMA 2.0. If $S \in AF$ and $X \subseteq (0,1]$, then

$$R[S, X] \geq R[FF, X].$$

Proof. This follows from the definitions and the fact that for any list L there exists a permutation L' of L such that for any $S \in AF$, the S -packing of L' will use the same number of bins as the FF-packing PF of L . Simply construct a rank function for L' so that all elements of $\text{cont}_{PF}(1)$ have lower rank than all elements of $\text{cont}_{PF}(2)$ etc. S will have to put the elements of $\text{cont}_{PF}(1)$ into BIN_1 , and then, since all the remaining pieces in L' were to the right of BIN_1 in PF, none of them can fit in the gap, so next the pieces from $\text{cont}_{PF}(2)$ all go in BIN_2 , and so on. \square

Unfortunately, the converse to Lemma 2.0 is not true, and obedience to the AF Constraint is not sufficient to guarantee behavior as good as FIRST FIT. In particular, consider the following algorithm:

WORST FIT (WF): Let j be the index of a bin with minimum non-zero level. If b will fit in BIN_j , choose j ; otherwise, choose $\#P+1$, (because b will not fit in any non-empty bin).

WORST FIT can do as badly as NEXT FIT, and in fact will do so on the lists we used for the proofs of Theorems 1.2 and 1.3, and so is significantly worse than FF and BF. However, the upper bound proofs we used for those theorems will also apply, and so we do have

THEOREM 2.1: If $S \in \text{AF}$ and $0 < t \leq 1$, then

$$R[S, t] \leq R[NF, t],$$

and moreover, there is an AF algorithm, namely WORST FIT, such that, for all t , $0 < t \leq 1$,

$$R[WF, t] = R[NF, t].$$

The ANY FIT constraint being insufficient, there must be something further that distinguishes FIRST FIT and BEST FIT from WORST FIT. That "something further" is the following:

CONSTRAINT 2: If BIN_j is the unique bin with lowest non-zero level, j cannot be chosen unless b will not fit in any bin to the left of BIN_j .

Constraint 2 is really just a natural extension of Constraint 1, despite its ad hoc appearance. This is because the unique bin it mentions will usually just be the most recently started one $\text{BIN}_{\#p}$, in other words, the bin which until just a few pieces ago could not have any pieces placed in it because of Constraint 1. The only time this might not be the case would be after said bin had received enough pieces to no longer have the lowest level. In fact it is only for technical reasons involved in the proof of Theorem 2.6 that we did not use " $\text{BIN}_{\#p}$ " instead of "unique bin with lowest non-zero level" in the statement of the Constraint.

Constraint 2 will often be referred to as the ALMOST ANY FIT Constraint, since it allows b to be placed in almost any non-empty bin. The set of all packing rules obeying both Constraints 1 and 2 will be called AAF, and a member of AAF, considered as an on-line algorithm, will be called an ALMOST ANY FIT algorithm. Clearly, FF and BF belong to AAF, and just as clearly, WF does not. However, a simple modification yields

ALMOST WORST FIT (AWF): Let BIN_j be a bin with minimum non-zero level, and $BIN_{j'}, j' \neq j$, be a non-empty bin whose level is such that no bin other than possibly BIN_j has lower non-zero level. Choose the first of the following three bins into which b will fit: $BIN_{j'}, BIN_j, BIN_{\#P+1}$.

Fixing an arbitrary AF algorithm so that it will obey the AAF Constraint may not seem as if it should make much difference, but we find in Theorem 2.5 and 2.6 that it does, at least as far as the worst case is concerned. However, first we shall prove some extremely useful lemmas about AF and AAF algorithms and the packings they generate.

LEMMA 2.2. If P is an S -packing for $S \in AF$, and $b = piece_P(j,1)$ for some $(j,1) \in POS(P)$, then for all j' , $1 \leq j' < j$,

$$gap_P(j') + \sum_{\substack{a \in cont(j') \\ rank(a) > rank(b)}} size(a) < size(b).$$

Proof. Immediate from the definition of AF, since pieces with $rank > rank(b)$ are packed after b has been packed, and b was the first piece to go in BIN_j . \square

COROLLARY 2.2.1. If P and b are as above, then for all j' , $1 \leq j' < j$, $\text{rank}(\text{piece}_P(j', 1)) < \text{rank}(b)$.

LEMMA 2.3. Suppose for $m \geq 2$ that L is a list no piece of which has size $> 1/m$, $S \in \text{AF}$, and PS is an S -packing of L . If $b = \text{piece}_{PS}(j, 1)$ for $j \leq \#PS$, then when b was assigned we had for all j' , $1 \leq j' < j$,

- (a) $\text{level}_P(j') > (m-1)/m$ and
- (b) $\text{height}_P(j') \geq m$.

Proof. (a) follows from the facts that S obeys the AF constraint and that no piece exceeds $1/m$. (b) follows from (a) and the fact that $m-1$ times $1/m$ cannot exceed $(m-1)/m$. \square

LEMMA 2.4. Suppose L , S , and PS are as above and in addition $S \in \text{AAF}$, $1 \leq j < \#PS$, and $d = \text{MAX}\{\text{gap}_{PS}(j'): 1 \leq j' < j\}$. Then BIN_j contains m pieces larger than d in PS .

Proof. Since $j+1 \leq \#PS$, Lemma 2.3 tells us that BIN_j must contain at least m pieces. Let $b = \text{piece}_{PS}(j, h)$, $1 \leq h \leq m$.

If $h = 1$, $\text{size}(b) > \text{gap}_{PS}(j')$ by Lemma 2.2. If $2 \leq h \leq m$, then at the time b was assigned, although BIN_j was not empty, it contained fewer than m pieces. Thus by Lemma 2.3,

$\text{piece}_{PS}(j+1,1)$ could not yet have been assigned and so BIN_j must have been the rightmost non-empty bin at the time. Since $\text{piece}_{PS}(j,1)$ had already been assigned, all previous bins must already have had $\text{level}_P(j') > (m-1)/m$, again by Lemma 2.3. Since BIN_j can have had $\text{level}_P(j)$ at most $(m-1)/m$, it was hence the unique bin with lowest non-zero level, and so b must have been larger than all preceding gaps (and hence d) by the AAF constraint. \square

THEOREM 2.5. If $S \in \text{AAF}$ and $m = \lfloor 1/t \rfloor \geq 2$, then

$$R[S, t] = R[N2F, t] = 1 + 1/m.$$

Proof. The lower bound examples in the proof of Theorem 1.5 are such that any AF algorithm would yield the same packing as N2F, for as we observed at the time, once a new bin is started during the generation of the packing, all of the old bins have gaps too small for any of the pieces remaining in the list. Thus the lower bound follows by Lemma 1.1 as in that proof.

For the upper bound, since Lemmas 2.3 and 2.4 say the same thing (in fact, something even stronger) about AAF-packings as Lemma 1.4 did about NkF-packings, we could proceed exactly as in the proof of the upper bound in Theorem 1.5. However, since we can say something slightly stronger about AAF-packings than

SECTION 2.1 - Page 48

about NkF -packings (Claim 2.5.1), and certain features of the details will have added significance later on in Section 2.3, we shall present the proof in its entirety. So let L be a list with $\text{Range}(\text{size}_L) \subseteq (0, t]$, and let PS be an S -packing of L , with $\#PS = S(L)$.

CLAIM 2.5.1. Except for the last bin ($\text{BIN}_{\#PS}$), there is at most one bin in PS with level $\leq m/(m+1)$.

For let BIN_j be the leftmost bin such that $\text{level}_{PS}(j) \leq m/(m+1)$, and hence $\text{gap}_{PS}(j) \geq 1/(m+1)$. Then by Lemma 2.4, for every j' , $j < j' < \#PS$, $\text{BIN}_{j'}$ contains m pieces all with size $> \text{gap}_{PS}(j) \geq 1/(m+1)$, and hence $\text{level}_{PS}(j') > m/(m+1)$.

Thus all but two bins of PS have level exceeding $m/(m+1)$, and hence $L^* \geq W(L) > [m/(m+1)] \cdot [S(L) - 2]$, and the upper bound follows by Lemma 1.1. \square

Thus the worst case behavior of FF and BF is indistinguishable from that of an arbitrary AAF algorithms when lists are restricted to those whose maximum piece size is no more than $1/2$. There remains to be considered the case when we allow pieces larger than $1/2$. We were unable to derive a precise formula for $R[N2F, t]$ when $1/2 < t \leq 1$, but we are more fortunate

with the AAF algorithms, which again have the same worst case behavior as was previously proved for FIRST and BEST FIT:

THEOREM 2.6: If $S \in \text{AAF}$ and $1/2 < t \leq 1$, then

$$R[S, t] = 17/10.$$

Lower Bound Proof. We will first show the lower bound, as the example lists, from [Gal,U11], are similar to the ones we used in proving Theorem 1.5, and in fact are the source of the idea which led to the invention of those examples. For this presentation, we have slightly modified the original examples so as to emphasize the resemblance to the ones used in Theorem 1.5.

For each $N > 0$, we shall present a list L with $30N$ pieces, such that $S(L) = 17N$ and $L^* = 10N + 1$, and the lower bound will follow by Lemma 1.1. This time, instead of having all our pieces roughly the same size, they will be divided into three classes, one with all pieces close to $1/6$ in size, one with all close to $1/3$, and one with all close to $1/2$. Each class will contain $10N$ pieces divided into N blocks of 10 pieces.

Turning to the details, we first choose a and d such that $0 < d < \min(t-1/2, 1/6)$, and for each i , $1 \leq i \leq N$, set $d(i) = d \cdot (1/18)^{-i}$. The i^{th} block of $1/6$ size pieces will consist of pieces $a_{j,i}$, $1 \leq j \leq 10$, with $\text{size}(a_{j,i}) = a(j, i)$ and

SECTION 2.1 - Page 50

```

a(1,i) = 1/6 + 3d(i)
a(2,i) = 1/6 - 3d(i)
a(3,i) = 1/6 - 7d(i)
a(4,i) = 1/6 - 7d(i)
a(5,i) = 1/6 - 13d(i)
a(6,i) = 1/6 + 9d(i)
a(7,i) = a(8,i) = a(9,i) = a(10,i) = 1/6 - 2d(i)

```

The $10N$ pieces with lowest rank in our list L will be these pieces, ordered so that $\text{rank}(a_{j,i}) < \text{rank}(a_{j',i'}) \iff i < i'$, or $i = i'$ and $j < j'$. Now since

$$a(1,i) + a(2,i) + a(3,i) + a(4,i) + a(5,i) = 5/6 + 3d(i),$$

and

$$a(6,i) + a(7,i) + a(8,i) + a(9,i) + a(10,i) = 5/6 + d(i),$$

and since all the pieces that follow $a_{5,i}$ in the list L have size exceeding $1/6 - 3d(i)$, and the smallest of the pieces that follow $a_{10,i}$ is $a_{5,i+1}$, with size $a(5,i+1) = 1/6 - 13d(i+1) = 1/6 - (13/18)d(i) > 1/6 - d(i)$, we can conclude that S will be forced to pack these pieces 5 to a bin, thus using up $2N$ bins.

Following these $1/6$ size pieces will be the N blocks of 10 size $1/3$ pieces, those in block i being $b_{j,i}$, $1 \leq j \leq 10$, with $\text{size}(b_{j,i}) = b(j,i)$ and

$b(1,i) = 1/3 + 46d(i)$
 $b(2,i) = 1/3 - 34d(i)$
 $b(3,i) = 1/3 + 6d(i)$
 $b(4,i) = 1/3 + 6d(i)$
 $b(5,i) = 1/3 + 12d(i)$
 $b(6,i) = 1/3 - 10d(i)$
 $b(7,i) = b(8,i) = b(9,i) = b(10,i) = 1/3 + d(i)$

These $10N$ pieces will follow the $a_{j,i}$'s in L , and be ordered in accordance with an analogous ranking property. Since all the previous bins in the S-packing are filled to level exceeding $5/6$, the $b_{j,i}$'s will all go in new bins, and in fact will fill up $5N$ of these new bins since

$$\begin{aligned}
 b(1,i) + b(2,i) &= 2/3 + 12d(i) \\
 b(3,i) + b(4,i) &= 2/3 + 12d(i) \\
 b(5,i) + b(6,i) &= 2/3 + 2d(i) \\
 b(7,i) + b(8,i) &= 2/3 + 2d(i) \\
 b(9,i) + b(10,i) &= 2/3 + 2d(i)
 \end{aligned}$$

and as before, after each of these bins has two pieces, the gap is too small for the remaining pieces in the list.

At the end of the list are the N blocks of 10 pieces, $c_{l,i}$

SECTION 2.1 - Page 52

thru $c_{10,i}$, all of size $1/2 + d(i)$. These will not go in any of the previous bins, and will not go two to a bin themselves, so they will take up $10N$ bins in the S-packing, giving us a total of $17N = S(L)$. See Figure 2.1.

However, a much more efficient packing is possible. For note that

$$a(1,i) + b(2,i) + c(2,i) = 1, \text{ for } 1 \leq i \leq N,$$

$$a(2,i) + b(1,i-1) + c(1,i) = 1, \text{ for } 2 \leq i \leq N,$$

and for $3 \leq j \leq 10$, $1 \leq i \leq N$,

$$a(j,i) + b(j,i) + c(j,i) = 1.$$

This leaves $a_{2,1}$, $b_{1,N}$ and $c_{1,1}$ unpacked, but they will require only two more bins, for a total of $10N + 1$, as claimed, and so $S(L) = (17/10)L^* - 17/10$, and the lower bound $R[S,t] \geq 17/10$, follows by Lemma 1.1. \square

Since the lists presented in the above proof would be packed by NkF for all $k \geq 2$ in the same way that they are packed by S , we have

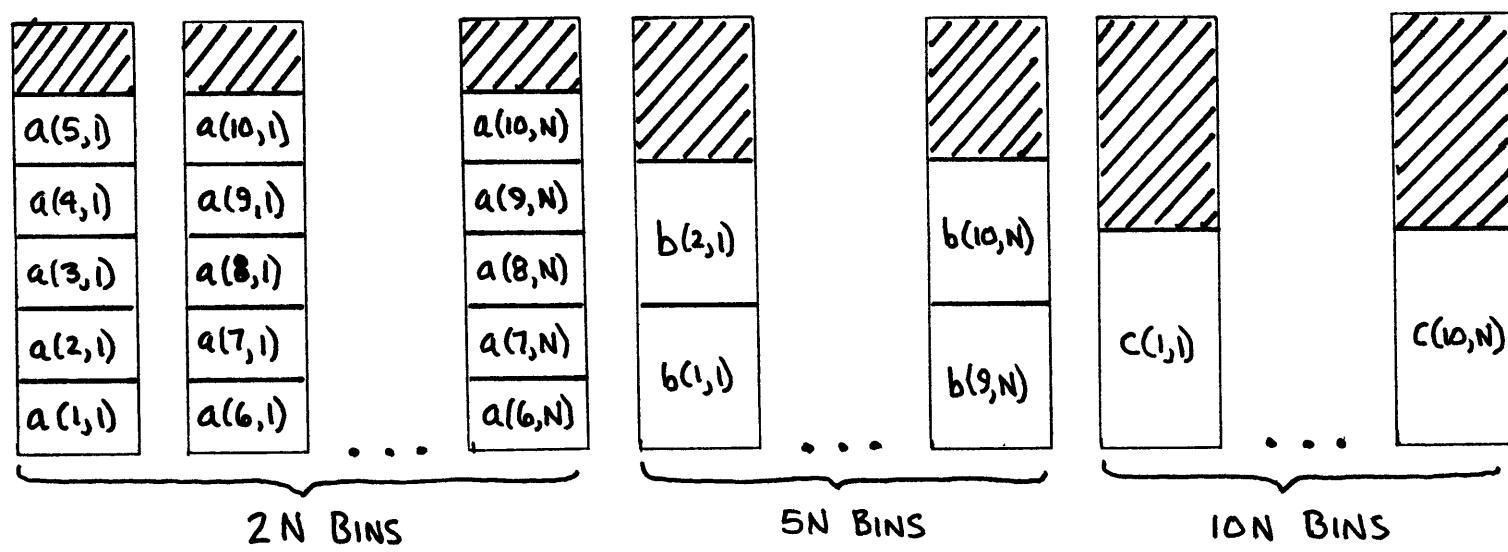


FIGURE 2.1. S-Packing of lists for which $S(L)/L^* \rightarrow 17/10$, $S \subset AAF$.

Corollary 2.6.1. For $k \geq 2$, $t > 1/2$,

$$\frac{17}{10} \leq R[NkF, t] \leq 2.$$

The upper bound of $R[S, t] \leq 17/10$ was originally proved by Ullman [Gal,U11] for $S = \text{FIRST FIT}$, and extended to $S = \text{BEST FIT}$ by Demers [De1]. We now extend it to S and arbitrary AAF algorithm, by slightly tightening the arguments in Ullman's original proof.

Upper Bound Proof. Let L be an arbitrary list with $\text{Range}(\text{size}_L) \subseteq (0, t]$, P^* an optimal packing of L , and PS an S -packing of L using $S(L)$ bins. The general strategy of the proof will be to define a weighting function on the pieces, and then compare the total weight per bin in the optimal packing P^* and in the S -packing PS . We will show that each bin in P^* contains no more than a total "weight" of $17/10$, and yet the total weight of all the pieces must be at least $S(L) - 2$. From this we will conclude that

$$S(L) \leq (17/10)L^* + 2,$$

and hence by Lemma 1.1 that the upper bound, $R[S, t] \leq 17/10$, holds, thus completing the proof of Theorem 1.6.

Actually, we have used this type of procedure before in our upper bound proofs for Theorems 1.2, 1.3, 1.5, and 2.5. There we used the elementary weighting function $W(b) = \text{size}(b)$. In this proof the weighting function f will be more complicated, in fact, a piecewise linear function of the piece size, described graphically in Figure 2.2, and formally by the following: If $a \in \text{PIECES}(L)$, then

$$\begin{aligned} f(a) &= (6/5)\text{size}(a), && \text{for } 0 < \text{size}(a) \leq 1/6 \\ f(a) &= (9/5)\text{size}(a) - 1/10, && \text{for } 1/6 \leq \text{size}(a) \leq 1/3 \\ f(a) &= (6/5)\text{size}(a) + 1/10, && \text{for } 1/3 \leq \text{size}(a) \leq 1/2 \\ f(a) &= 1, && \text{for } 1/2 < \text{size}(a) \leq 1 \end{aligned}$$

We shall proceed by a series of Claims. In our arguments, we shall represent members of $\text{PIECES}(L)$ by subscripted variables a_i , but the subscripts are not assumed to bear any relation to the rankings of the corresponding pieces in L . They are merely introduced for the sake of distinguishing the various pieces the argument is dealing with.

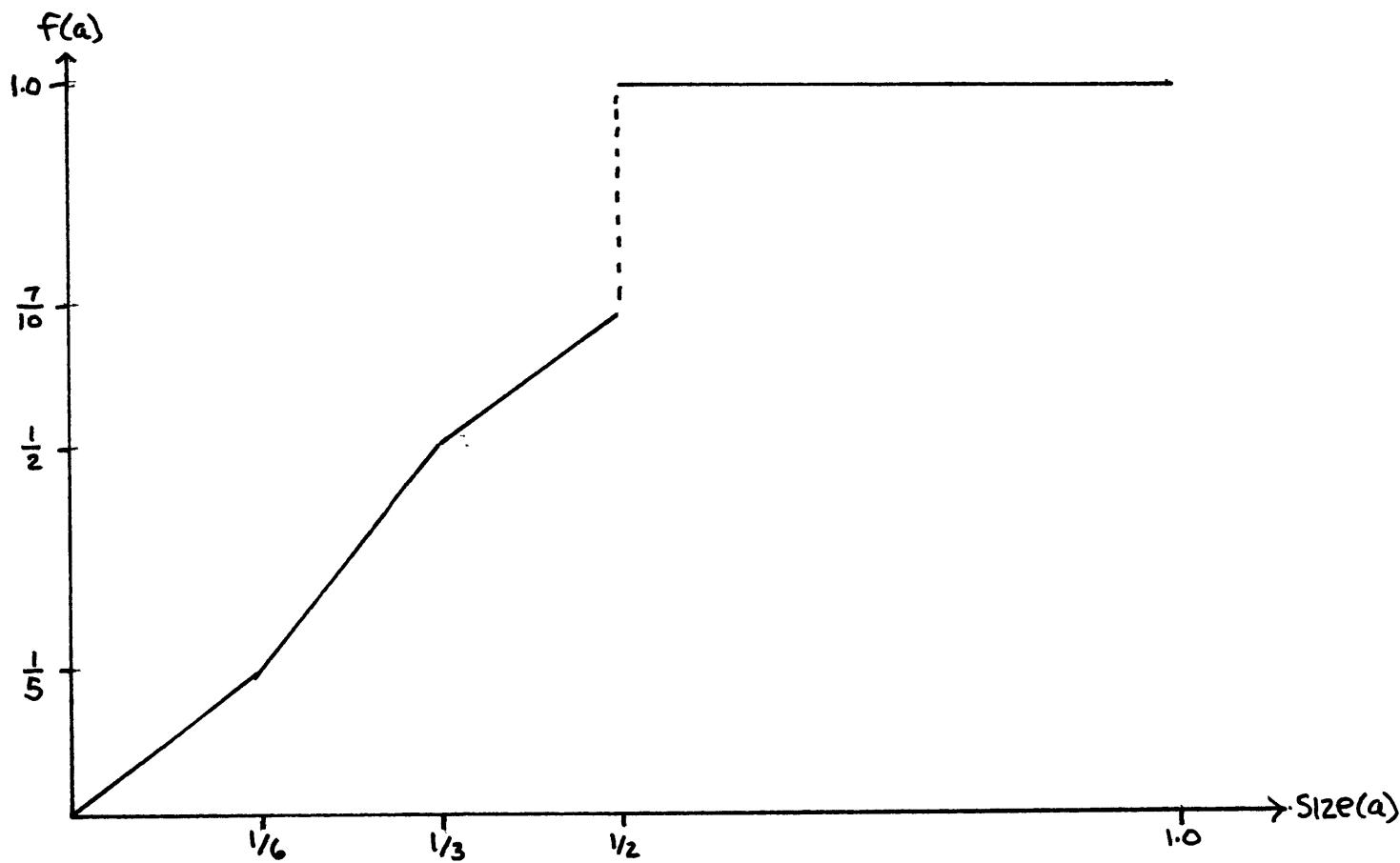


FIGURE 2.2. Weighting function f for Theorem 2.6.

CLAIM 2.6.1. If $\sum_{i=1}^n \text{size}(a_i) \leq 1$, then
 $\sum_{i=1}^n f(a_i) \leq 17/10$.

Proof of Claim. We assume the pieces are ordered by decreasing size. If $\text{size}(a_1) \leq 1/2$, then for all the pieces we must have $f(a_i)/\text{size}(a_i) \leq 3/2$, so the Claim is immediate unless $\text{size}(a_1) > 1/2$. Assuming this to be the case, we now show that if

$$\sum_{i=2}^n \text{size}(a_i) < 1/2, \text{ then } \sum_{i=2}^n f(a_i) \leq 7/10.$$

Now since the slope of f is the same in the region $(0, 1/6]$ as it is in $[1/3, 1/2]$, any a_i with size in $(1/3, 1/2]$ can be replaced without loss of generality by two pieces, one of size $1/3$ and the other of size $\text{size}(a_i) - 1/3$, and we may therefore assume that $\text{size}(a_i) \leq 1/3$ for $2 \leq i \leq n$. Moreover, if a_j and a_k both have size $\leq 1/6$, they can be combined into one piece with size = $\text{size}(a_j) + \text{size}(a_k)$, with no loss, and possibly an increase, in the total weight of the pieces. Thus we may also assume that all of the a_i 's except a_1 have sizes in the range $(1/6, 1/3]$.

This reduces the proof to the consideration of four cases:

SECTION 2.1 - Page 58

- (1) $n = 2$, $\text{size}(a_2) \leq 1/3$,
- (2) $n = 3$, $\text{size}(a_3) \leq 1/6 < \text{size}(a_2) \leq 1/3$,
- (3) $n = 3$, $1/6 \leq \text{size}(a_3) \leq \text{size}(a_2) \leq 1/3$, and
- (4) $n = 4$, $\text{size}(a_4) \leq 1/6 < \text{size}(a_3) \leq \text{size}(a_2) \leq 1/3$.

Since the first and second cases will follow from the third, we shall restrict our attention to that case and the fourth. In (3), $f(a_2) + f(a_3) = (9/5)(\text{size}(a_2) + \text{size}(a_3)) - 1/5$. Since $\text{size}(a_2) + \text{size}(a_3) \leq 1/2$, we have $f(a_2) + f(a_3) \leq 7/10$, as desired. In (4)

$$\begin{aligned}
 & f(a_2) + f(a_3) + f(a_4) \\
 & \leq (6/5)\text{size}(a_4) + (9/5)(\text{size}(a_2) + \text{size}(a_3)) - 1/5 \\
 & \leq (9/5)(\text{size}(a_2) + \text{size}(a_3) + \text{size}(a_4)) - 1/5 \\
 & \leq (9/5)(1/2) - 1/5 = 7/10,
 \end{aligned}$$

so Claim 2.6.1 holds. \square

Thus we can conclude as claimed that the total weight under f of the pieces in any bin of the optimal packing P^* can be no more than $17/10$, and hence the total weight under f of all the pieces in L can be at most $(17/10)L^*$.

We now turn to the S-packing PS . Let us define the coarseness of BIN_j in PS , $1 < j \leq \#PS$, to be $\text{MAX}\{g_{appS}(j'): 1 \leq j' < j\}$. The coarseness of BIN_1 is by convention 0.

CLAIM 2.6.2. If BIN_j has coarseness d , $b = \text{piece}_{\text{PS}}(j, h')$, and $\sum_{h=1}^{h'-1} \text{size}(\text{piece}_{\text{PS}}(j, h)) \leq 1/2$, then $\text{size}(b) > d$.

Proof of Claim. When b was assigned to BIN_j during the generation of PS, we had $\text{level}_{\text{P}}(j) = \sum_{h=1}^{h'-1} \text{size}(\text{piece}_{\text{PS}}(j, h)) \leq 1/2$, and so BIN_j was either empty, or had non-zero level $\leq 1/2$. In the latter case, since there cannot have been two non-empty bins with levels $\leq 1/2$ at the time without the rightmost having had its bottom piece placed in violation of the AF Constraint, BIN_j must at that time have been the unique bin with lowest non-zero level. Thus either Constraint 1 or 2 applied, and b cannot have fit in any bin to the left of BIN_j , and hence $\text{size}(b) > \text{MAX}\{g_{\text{app}}(j'): 1 \leq j' < j\} \geq d$ by definition of coarseness and the fact that gaps never get any bigger during the generation of a packing. \square

CLAIM 2.6.3. Suppose some BIN_j , $1 \leq j \leq \#PS$, has coarseness $d < 1/2$, and $\text{cont}_{\text{PS}}(j) = \{a_1, \dots, a_n\}$. If $\sum_{i=1}^n \text{size}(a_i) \geq 1-d$, then $\sum_{i=1}^n f(a_i) \geq 1$.

Proof of Claim. If $\text{size}(a_i) > 1/2$ for any i , the result is immediate, since $f(a_i) = 1$. We may therefore assume that $\text{size}(a_i) \leq 1/2$ for all i , and hence, since $\sum_{i=1}^n \text{size}(a_i) \geq 1-d >$

SECTION 2.1 - Page 60

$1/2$, that $n \geq 2$. Note that this means by Claim 2.6.2 that at least two of the a_i 's [$\text{piece}_{PS}(j,1)$ and $\text{piece}_{PS}(j,2)$] have size $> d$. In the following let a_1 be a piece with maximum size in $\text{cont}_{PS}(j)$, and a_2 be a second piece with size $> d$. We consider several cases, depending on the range of d .

Case 1: $d \leq 1/6$. Then $\sum_{i=1}^n \text{size}(a_i) \geq 1-d \geq 5/6$. Since $f(a)/\text{size}(a) \geq 6/5$ in the range $0 < \text{size}(a) \leq 1/2$, we immediately have $\sum_{i=1}^n f(a_i) \geq (6/5)(5/6) = 1$.

Case 2: $1/6 < d \leq 1/3$. We consider subcases, depending on the value of n .

$n = 2$: If both a_1 and a_2 have size $\geq 1/3$, then $f(a_1) + f(a_2) \geq 2[(6/5)(1/3)+(1/10)] = 1$. Both cannot have size $< 1/3$, since then $\text{size}(a_1) + \text{size}(a_2) < 2/3 < 1-d$, which is impossible. The only other possibility is $1/6 \leq d < \text{size}(a_2) < 1/3 \leq \text{size}(a_1) \leq 1/2$, in which case $f(a_1) + f(a_2)$

$$\begin{aligned} &= (9/5)\text{size}(a_2) - 1/10 + (6/5)\text{size}(a_1) + 1/10 \\ &= (6/5)[\text{size}(a_1) + \text{size}(a_2)] + (3/5)\text{size}(a_2) \\ &\geq (6/5)(1-d) + (3/5)d \\ &\geq 1 + |1/5 - (3/5)d| \geq 1, \text{ since } d \leq 1/3. \end{aligned}$$

$n \geq 3$: As in the previous case, if two of the a_i 's have size $\geq 1/3$, the result is immediate. If only one has size $\geq 1/3$, then as above we have

$$\begin{aligned} 1/6 &\leq d < \text{size}(a_2) < 1/3 \leq \text{size}(a_1) \leq 1/2, \text{ and so } \sum_{i=1}^n f(a_i) \\ &\geq (6/5)\text{size}(a_1) + 1/10 + (9/5)\text{size}(a_2) - 1/10 \\ &\quad + (6/5) \left[\sum_{i=3}^n \text{size}(a_i) \right] \\ &\geq (6/5) \left[\sum_{i=1}^n \text{size}(a_i) \right] + (3/5)\text{size}(a_2) \\ &\geq (6/5)(1-d) + (3/5)d = 1 + 1/5 - (3/5)d \geq 1. \end{aligned}$$

If none of the a_i 's have size $\geq 1/3$, then we have

$$\begin{aligned} 1/6 &\leq d < \text{size}(a_2) \leq \text{size}(a_1) < 1/3, \text{ so } \sum_{i=1}^n f(a_i) \\ &\geq (9/5) [\text{size}(a_1) + \text{size}(a_2)] - 1/5 \\ &\quad + (6/5) \left[\sum_{i=3}^n \text{size}(a_i) \right] \\ &\geq (6/5)(1-d) + (3/5)(2d) - 1/5 = 1 + (6/5)(d-d) = 1. \end{aligned}$$

Case 3: $1/3 < d < 1/2$. We must have in this case

$1/3 < d < \text{size}(a_2) \leq \text{size}(a_1)$, and as argued in Case 2, $n = 2$, the result is immediate.

This exhausts the possibilities and completes the proof of
Claim 2.6.3. \square

Claim 2.6.4. If some BIN_j , $1 \leq j \leq \#PS$, has coarseness $d < 1/2$, $\text{cont}_{PS}(j) = \{a_1, \dots, a_n\}$, and $\sum_{i=1}^n f(a_i) = 1 - b$, where $b > 0$, then either

- (A) $n = 1$ and $\text{size}(a_1) \leq 1/2$, or
- (B) $\sum_{i=1}^n \text{size}(a_i) \leq 1 - d - (5/9)b$.

Proof of Claim. Again let a_1 be a piece with maximum size in $\text{cont}_{PS}(j)$. If $\text{size}(a_1) > 1/2$, it is impossible that $b > 0$. Therefore, if (A) does not hold, we may assume that $n \geq 2$, and by Claim 2.6.2 that at least two of the a_i 's have size exceeding d , so let a_2 be a second such piece different from a_1 . (B) can fail to hold only if $c = 1 - d - \sum_{i=1}^n \text{size}(a_i) > 0$. In this case, let x_1 and x_2 be two invented pieces designed so that $\text{size}(x_1) > \text{size}(a_1)$, $\text{size}(x_2) > \text{size}(a_2)$, $\text{size}(x_1) + \text{size}(x_2) = \text{size}(a_1) + \text{size}(a_2) + c$, and neither x_1 nor x_2 exceeds $1/2$ in size. We would then have $\sum_{i=3}^n \text{size}(a_i) + \text{size}(x_1) + \text{size}(x_2) = 1 - d$, and thus by the arguments used in the proof of Claim 2.6.3, $\sum_{i=3}^n f(a_i) + f(x_1) + f(x_2) \geq 1$. But since the slope of f in the range $(0, 1/2]$ does not exceed $9/5$, it follows that $f(x_1) + f(x_2) \leq f(a_1) + f(a_2) + (9/5)c$. Therefore, $c \geq (5/9)b$, and (B) holds.

□

CLAIM 2.6.5. $\sum_{a \in \text{PIECES}(L)} f(a) \geq S(L) - 2.$

Proof of Claim. Suppose that in PS, $\text{BIN}_{J_1}, \dots, \text{BIN}_{J_m}$, $J_1 < \dots < J_m$, are all the bins which contain at least one piece, but for which $\sum_{a \in \text{contps}(J_k)} f(a) = 1 - b_k$, for $b_k > 0$. Let d_k be the coarseness of BIN_{J_k} . By the definition of coarseness, the d_k 's would form an non-decreasing sequence. No $d_k \geq 1/2$, since by Claim 2.6.2, BIN_{J_k} must have a bottom piece of size $> 1/2$, and hence could not have $\sum f(a) < 1$. By Claims 2.6.3 and 2.6.4 and the definition of coarseness, we thus have

$$d_k \geq d_{k-1} + (5/9)b_{k-1}, \text{ for } 1 < k \leq m, \text{ and so}$$

$$\begin{aligned} \sum_{k=1}^{m-1} b_k &\leq (9/5) \sum_{k=2}^m (d_k - d_{k-1}) \\ &\leq (9/5)(d_m - d_1) \leq (9/5)(1/2) < 1. \end{aligned}$$

Since b_m cannot exceed 1, we thus have

$$\sum_{k=1}^m b_k \leq 2,$$

and so $\sum_{a \in \text{PIECES}(L)} f(a) \geq \#\text{PS} - \sum_{k=1}^m b_k \geq S(L) - 2.$

Thus Claim 2.6.5 is proven. \square

We now can complete the upper bound proof. By Claim 2.6.1,

$$\sum_{a \in \text{PIECES}(L)} f(a) \leq (17/10)L^*.$$

Thus by Claim 2.6.5, $S(L) \leq (17/10)L^* + 2$. Since L was an arbitrary list with $\text{Range}(\text{size}_L) \subseteq (0, t]$, Lemma 1.1 thus tells us that $R[S, t] \leq 17/10$. \blacksquare

There is an intuition behind the number $17/10$ that appears in Theorem 2.6. It is based on an observation about our worst case examples: pieces of size about $1/J$ can be made to go $J-1$ per bin, if we are clever enough, but we do not seem to be able to get away with fewer than $j-1$. The number $17/10$ is then derived as follows:

Let a sequence of integers $D = \langle j_1, \dots, j_N \rangle$ be called a subdivision of 1 if $\sum_{i=1}^n [1/j_i] = 1$, and let total(D) = $\sum_{i=1}^n [1/(j_i - 1)]$. For each subdivision of 1 D with $|D| \geq 3$, we can construct a series of worst case examples made up of pieces of sizes near $1/j_i$ for each j_i in the subdivision, which by Lemma 1.1 can be used to show that $R[S] \geq \text{total}(D)$.

The justification for the upper bound comes from the fact (easily derived) that

$$\text{MAX}\left\{\text{total}(D) : D \text{ is a subdivision of } 1 \text{ and } |D| \geq 3\right\} = 17/10$$

where the subdivision that realizes this max is $\langle 2, 3, 6 \rangle$. If we further restrict the subdivisions by requiring that all the j_i be such that $1/j_i < t \leq 1/2$, the MAX becomes $1 + 1/m$ where $m = \lfloor 1/t \rfloor$, and so this idea also helps to explain our results for restricted lists.

One would hope that some simple argument, based perhaps on the above intuition, will eventually be discovered to replace the present rather complicated proof of Theorem 2.6. One problem, however, is that a number of would-be lemmas are just not true. For instance, decreasing the size of all the pieces in a given list may actually increase the number of bins used in the FF-packing, as may simply deleting a single piece. Figure 2.3 presents an example of the first type of misbehavior, and Figure 2.4 an example of the second. These examples originally appeared in [Gr3].

The results of this chapter provide us with a wide variety of algorithms which are competitive with FIRST and BEST FIT as far as worst case behavior. This allows the practical bin-packer to make his choice of which algorithm to use on the basis of other considerations. An application of Theorems 2.1,

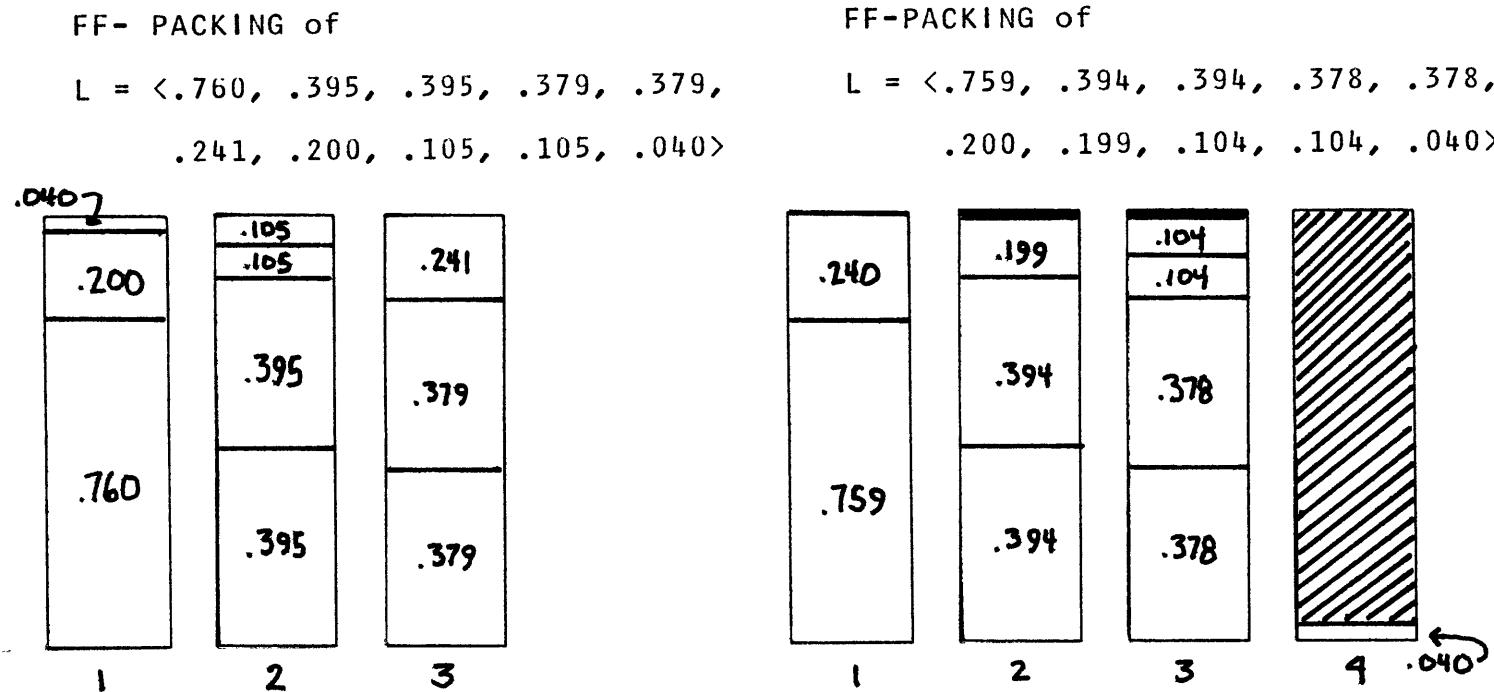
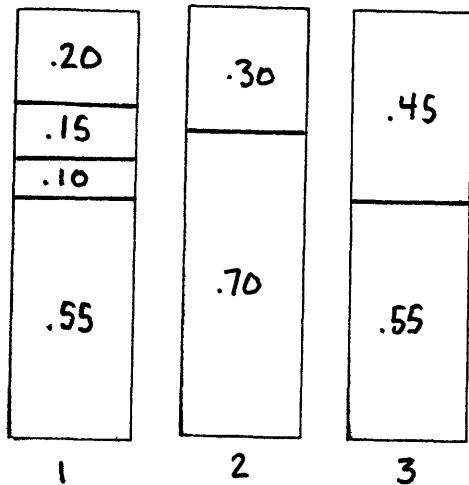


FIGURE 2.3. Example showing that decreasing the sizes of the pieces can increase the number of bins used by FIRST FIT.

FF-PACKING of

$L = \langle .55, .70, .55, .10,$
 $.45, .15, .30, .20 \rangle$



FF-PACKING of

$L = \langle .55, .70, .55,$
 $.45, .15, .30, .20 \rangle$

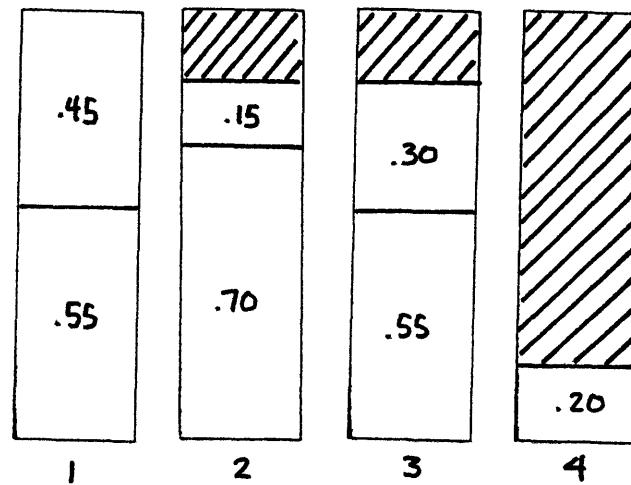


FIGURE 2.4. Example showing that deleting a piece from L can increase the number of bins used by FIRST FIT.

2.5, and 2.6 in all their generality goes as follows:

The completely determined nature of the FIRST and BEST FIT algorithms raises an objection to their use in certain practical situations, especially when the environment is dynamic. For instance, suppose we have a computer system with a large amount of relatively slow storage, and a relatively small amount of high-speed core storage, and suppose the slower storage is divided into a number of pages of a fixed size, which currently are filled to various levels.

A bin packing situation would arise if a sequence of new files of various length were being input, each file having to be assigned to some page of the low-speed storage. One of our goals might well be to keep the number of pages used low, but we might also be acting under an additional constraint, that of time. Presumably we can keep a small fixed number of pages in core at one time; however, there may be no way to predict which pages will be in high speed memory when a given file is to be stored, and so a deterministic algorithm might choose a page not currently in high speed memory, which then must be fetched at high cost. Therefore, partially determined algorithms, which allow a wide choice of possible assignments at any given point, may be more desirable, so that we may use pages already in high-speed memory whenever possible. Our results tell us that we can, without making a sacrifice in worst case behavior, use

SECTION 2.1 - Page 69

the following very unrestrictive packing rule instead of FIRST FIT:

ALMOST ANY FIT: Choose any j such that b fits in BIN_j and Constraints 1 and 2 are not violated.

ALMOST ANY FIT is the least restrictive rule in AAF. Similarly, ANY FIT, thought of as the least restrictive rule in AF, would guarantee only slightly worse behavior than FF, with the difference diminishing as the size of the pieces (files) decreases with respect to bin (page) size.

Note however that, although we now know that a wide variety of algorithms have the same worst case behavior, we cannot be certain they will all behave the same way on a given list. In fact, the variations can be quite extreme. Figures 2.5 thru 2.9 give examples of lists yielding the most extreme ratios $S_1(L)/S_2(L)$ for S_1 and S_2 among the algorithms we have studied. Summarizing, we have

THEOREM 2.7. There are lists L with L^* arbitrarily large such that (for $k \geq 2$):

$$(A) \quad \frac{BF(L)}{FF(L)} = \frac{BF(L)}{NkF(L)} = \frac{BF(L)}{AWF(L)} = \frac{4}{3}$$

$$(B) \frac{FF(L)}{BF(L)} = \frac{NkF(L)}{BF(L)} = \frac{AWF(L)}{BF(L)} = \frac{3}{2}$$

$$(C) \frac{AWF(L)}{FF(L)} = \frac{5}{4}$$

$$(D) \frac{FF(L)}{AWF(L)} \longrightarrow \frac{9}{8}$$

$$(E) \frac{NkF(L)}{FF(L)} = \frac{NkF(L)}{AWF(L)} \longrightarrow \frac{3}{2}$$

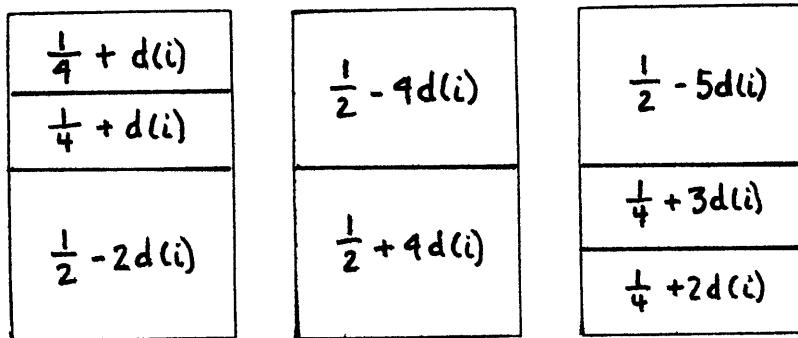
We conjecture that these are the best bounds possible and that there are no lists L for which $N2F(L) < FF(L)$.

SECTION 2.1 - Page 71

$d(1) = 3 \quad , \quad d(i) = 3d(i-1), \quad 1 < i \leq N$

BLOCK i : $\langle 1/2 - 2d(i), \quad 1/2 + 4d(i), \quad 1/4 + d(i), \quad 1/4 + d(i),$
 $1/2 - 4d(i), \quad 1/4 + 2d(i), \quad 1/4 + 3d(i), \quad 1/2 - 5d(i) \rangle$

FF & NkF & AWF - PACKINGS of BLOCK i : 3 Bins



All Bins have gaps \leq

BF - PACKING of BLOCK i : $1/4 + 2d(i) < 1/4 + d(i+1) \leq$

4 Bins

size of all remaining pieces

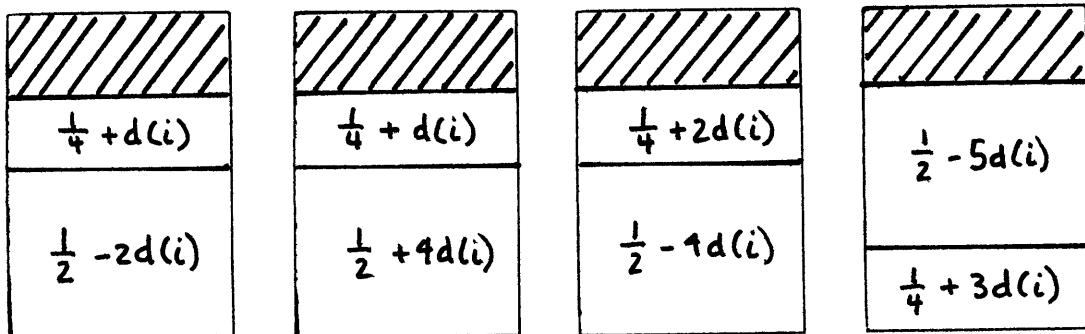


FIGURE 2.5. Building BLOCKS of lists L for which

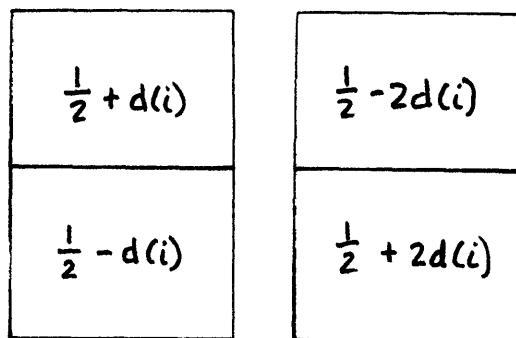
$$\frac{BF(L)}{FF(L)} = \frac{BF(L)}{NkF(L)} = \frac{BF(L)}{AWF(L)} = \frac{4}{3}$$

SECTION 2.1 - Page 72

$$d(1) = 1/32, \quad d(i) = d(i-1)/4, \quad 1 < i \leq N$$

BLOCK i: $\langle 1/2 - d(i), 1/2 + 2d(i), 1/2 - 2d(i), 1/2 + d(i) \rangle$

BF - PACKING of BLOCK i: 2 Bins



FF & NkF & AWF - PACKINGS of BLOCK i: 3 Bins

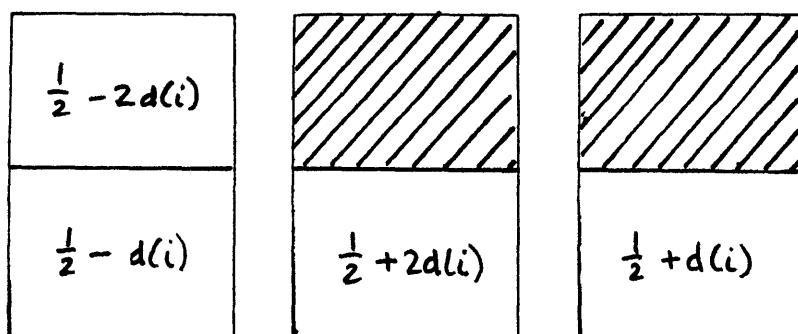
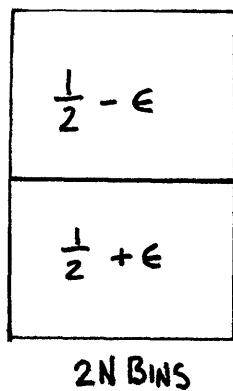


FIGURE 2.6. Building BLOCKS of lists L for which

$$\frac{FF(L)}{BF(L)} = \frac{NkF(L)}{BF(L)} = \frac{AWF(L)}{BF(L)} = \frac{3}{2}.$$

$$L = \underbrace{\langle 1/2 + \epsilon, \dots, 1/2 + \epsilon \rangle}_{2N \text{ Pieces}}, \underbrace{\langle 1/2 - \epsilon, \dots, 1/2 - \epsilon \rangle}_{2N \text{ Pieces}}$$

FF & AWF - PACKINGS: $FF(L) = AWF(L) = 2N$



NKF - PACKING: $NkF(L) = 3N - k/2$ (we assume k is even)

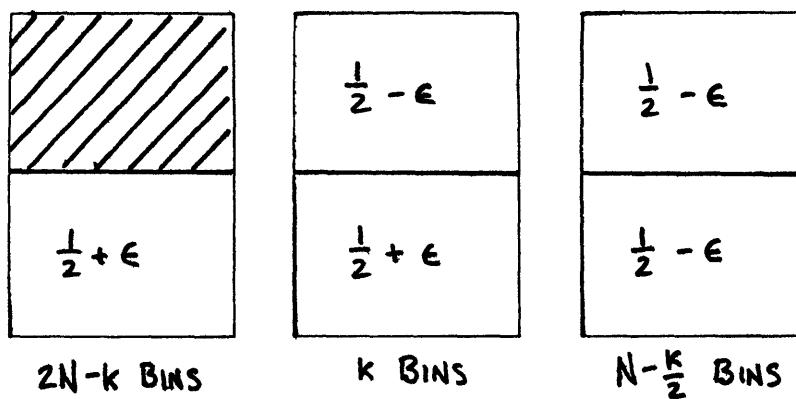


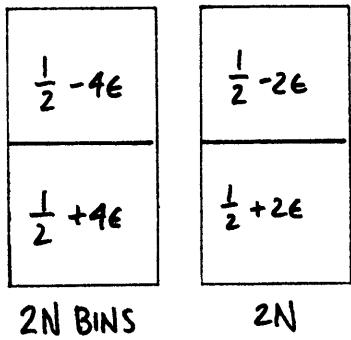
FIGURE 2.7. Lists L for which $\frac{NkF(L)}{FF(L)} = \frac{NkF(L)}{AWF(L)} \rightarrow \frac{3}{2}$,

for k even (Similar examples exist for k odd).

SECTION 2.1 - Page 74

$$L = \left\langle \underbrace{\frac{1}{2} + 4\epsilon, \dots, \frac{1}{2} + 4\epsilon}_{2N \text{ Pieces}}, \underbrace{\frac{1}{2} + 2\epsilon, \dots, \frac{1}{2} + 2\epsilon}_{2N \text{ Pieces}}, \underbrace{\frac{1}{2} - 2\epsilon, \dots, \frac{1}{2} - 2\epsilon}_{2N \text{ Pieces}}, \underbrace{\frac{1}{2} - 4\epsilon, \dots, \frac{1}{2} - 4\epsilon}_{2N \text{ Pieces}} \right\rangle$$

FF - PACKING: $FF(L) = 4N$



AWF - PACKING: $AWF(L) = 5N$

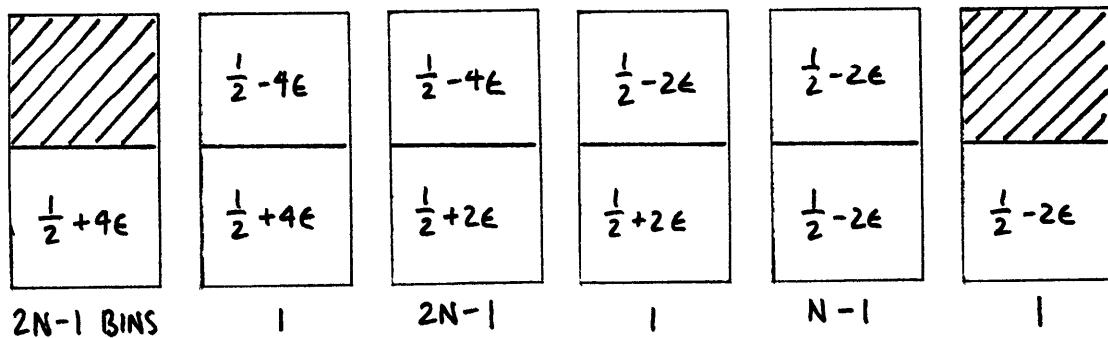


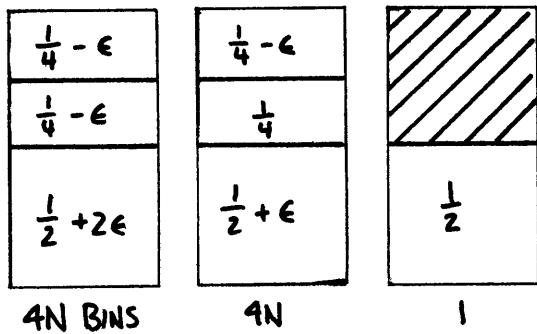
FIGURE 2.8. Lists L for which $AWF(L)/FF(L) = 5/4$.

SECTION 2.1 - Page 75

$$L = \left\langle \frac{1}{2} + 2\epsilon, \dots, \frac{1}{2} + 2\epsilon, \frac{1}{2} + \epsilon, \dots, \frac{1}{2} + \epsilon, \frac{1}{2}, \frac{1}{4}, \dots, \frac{1}{4}, \frac{1}{4}, \dots, \frac{1}{4} \right\rangle$$

4N Pieces 4N Pieces 4N Pieces 4N Pieces

AWF - PACKING: $AWF(L) = 8N + 1$



FF - PACKING: $FF(L) = 9N + 1$

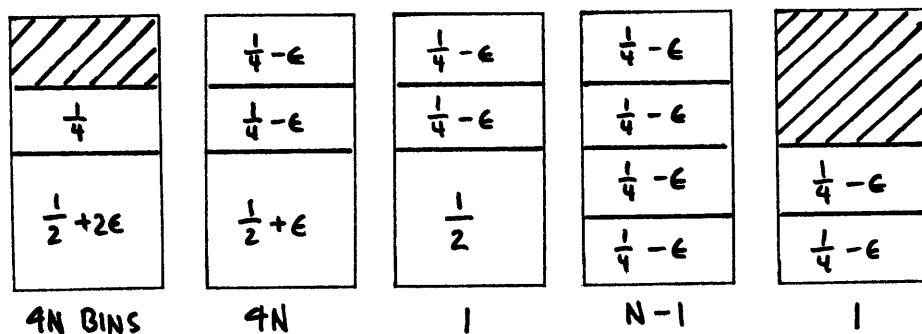


FIGURE 2.9. Lists L for which $FF(L)/AWF(L) \rightarrow 9/8$.

SECTION 2.2. The Interval Problem

In our analysis to date we have only considered worst case behavior for lists with piece sizes restricted to intervals of the form $(0, t]$. In this section, we consider the more general problem of lists with piece sizes restricted to an arbitrary interval $X \subseteq (0, 1]$ and derive the values of $R[FF, X]$ for all such X . These values will also hold if FF is replaced by any $S \in AAF$, but in certain cases the proof for FF is slightly simpler, and we shall leave the extensions, which would proceed much the same way we extended Ullman's 17/10 result, to the reader. We summarize our results in the following overall Theorem:

THEOREM 2.8. Let $X \in \{(p, q), (p, q], [p, q), [p, q]\}$ be a subinterval of $(0, 1]$, and let $m = \lfloor 1/q \rfloor$.

I. If $q \leq 1/2$, and

A) If $p = 1/(m+1)$ and $X = (p, q)$ or $(p, q]$,

or $p > 1/(m+1)$,

then $R[FF, X] = 1$.

B) If $p = 1/(m+1)$, and $X = [p, q)$ or $[p, q]$,
then $R[FF, X] = 1 + 1/[m+1+2/(m-1)]$.

C) If $p < 1/(m+1)$,
then $R[FF, X] = 1 + 1/m$.

II. If $q > 1/2$, and

A) If $p \geq 1/2$,
then $R[FF, X] = 1$.

B) If $1/4 \leq p < 1/2$,
then $R[FF, X] = 3/2$.

C) If $1/6 \leq p < 1/4$,
then $R[FF, X] = 5/3$.

D) If $0 < p < 1/6$, or $X = (0, q)$ or $(0, q]$,
then $R[FF, X] = 17/10$.

Proof. Case IA is immediate, since in the FF-packing every bin except possibly the last will contain m pieces, and in the optimal packing no bin can contain more than m pieces. Thus $FF(L) = L* = \lceil PIECES(L)/m \rceil$.

IC and IID are immediate consequences of Theorems 1.5 and 1.6, respectively. We treat the remaining cases individually.

Case 1B. To show the lower bound, we construct arbitrarily long lists made up of $k(m-1)(m+1)m$ pieces of size $1/(m+1)$, which we will call small-pieces, and $k(m+1)m$ big-pieces of size

$1/(m+1) + d < q$, where $d > 0$. Figure 2.10 illustrates the optimal and FF-packings when the pieces are ordered

$\langle k(m+1)m \text{ repetitions of } \langle 1/(m+1) + d, \langle m-1 \text{ reps of } 1/(m+1) \rangle \rangle \rangle$.

Note that

$$\begin{aligned} \text{FF}(L)/L^* &= (m^2+m)/(m^2+1) = 1 + (m-1)/(m^2+1) \\ &= 1 + 1/\left[m+(m+1)/(m-1)\right] = 1 + 1/\left[m+1+2/(m-1)\right]. \end{aligned}$$

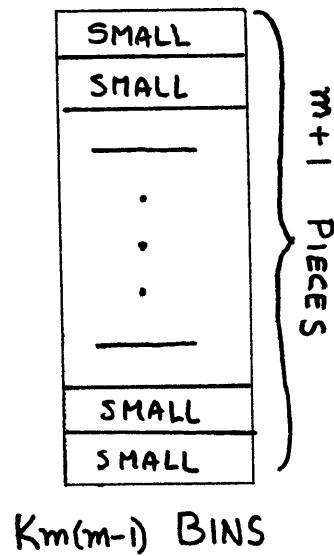
To prove the upper bound, suppose we have a list L of n pieces, with $\text{Range}(\text{size}_L) \subseteq [1/(m+1), 1/m]$, for which the ratio $\text{FF}(L)/L^*$ is the maximum for all lists of n or fewer pieces. Let

$$\begin{aligned} \text{SMALL-PIECES} &= \{b \in \text{PIECES}(L) : \text{size}(b) = 1/(m+1)\}, \\ \text{BIG-PIECES} &= \{b \in \text{PIECES}(L) : \text{size}(b) > 1/(m+1)\}, \end{aligned}$$

and let $S = |\text{SMALL-PIECES}|$, $B = |\text{BIG-PIECES}|$. Essentially all we need to know about a piece in order to determine how it will behave in a packing is whether it is a big-piece or a small-piece. Any bin containing a big-piece can contain only $m-1$ other pieces, and if a bin contains only $m-1$ pieces, any other piece will fit. On the other hand, any bin containing m small pieces has room for one more small-piece and no

OPTIMAL PACKING

$$L^* = K(m^2 + 1)$$



FF - PACKING

$$FF(L) = K(m^2 + m)$$

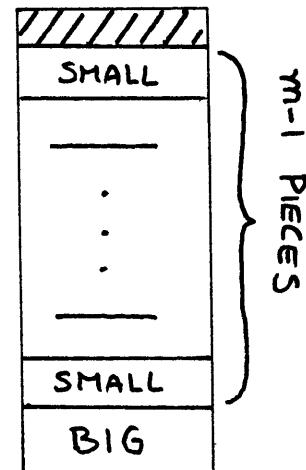
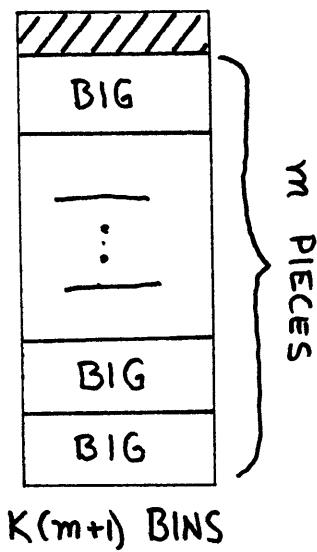


FIGURE 2.10. Lists L with $\text{Range}(\text{size}_L) \subseteq \left[\frac{1}{m+1}, \frac{1}{m+1} + \epsilon \right]$, and

$$\frac{FF(L)}{L^*} = 1 + \frac{1}{(m+1) + 2/(m-1)}$$

SECTION 2.2 - Page 80

big-pieces, and no bin can contain more than $m+1$ pieces. Thus given a set of pieces, the "worst" way in which they can be ordered as far as FF is concerned is so that as many bins as possible are forced to contain only m pieces. If $B > S/(m-1)$, it is clear that we can order the list so that no bin in the FF-PACKING contains more than m pieces. We shall use this information in what follows.

The number of bins used in the optimal packing is clearly

$$\left\lfloor \frac{S}{m+1} \right\rfloor + \left\lceil \frac{S - (m+1)\lfloor S/(m+1) \rfloor + B}{m} \right\rceil$$

If we increase the size of the $S - (m+1)\lfloor S/(m+1) \rfloor$ leftover small-pieces, we will not change the number of bins in the optimal packing, and can only increase the number of bins used in the FF-packing of the worst case ordering of the pieces, so we can assume S is divisible by $m+1$ and

$$L^* = S/(m+1) + \lceil B/m \rceil.$$

By adding $\lceil B/m \rceil(m)-B$ big-pieces, we will again not increase the optimal packing, and can only increase the number of bins used in the FF-packing of the worst case ordering of the list, so let L' be the list in worst case order after these pieces are added to L , and let B' be the new number of big-pieces, $n' = S+B'$ the

new total. we now have that m divides B' , $n' \leq n+m-1$, and

$$\begin{aligned} L'^* &= S/(m+1) + B'/m = [Sm+B'(m+1)]/[m(m+1)] \\ &\geq (S+B')/(m+1). \end{aligned}$$

Clearly $FF(L') \leq \lceil (S+B')/m \rceil \leq (S+B')/m + 1$, so

$$\begin{aligned} \frac{FF(L')}{L'^*} &\leq \frac{S+B'}{m} \frac{Sm+B'm+B'}{m(m+1)} + \frac{m+1}{S+B'} \\ &\leq \frac{(S+B')(m+1)}{(S+B')(m+1)-S} + \frac{m+1}{n'} \\ &\leq 1 + \frac{S}{(m+1)(S+B')-S} + \frac{m+1}{n'} \\ &\leq 1 + \frac{1}{(m+1)(1+B'/S)-1} + \frac{m+1}{n'} \end{aligned}$$

Now suppose $B' \leq S/(m-1) - 3$. Then $B' < \lfloor S/(m-1) \rfloor - 2$, so $\lfloor S/(m-1) \rfloor - B' > 2$, and there are at least $3(m-1)$ small-pieces which cannot be put in an m -piece bin with a big-piece (there are not enough big-pieces), and since $m \geq 2$, there are certainly at least $m+1$ of them. Therefore, by removing $m+1$ small-pieces from our set of n' pieces, we get a set of $n' - (m+1) \leq (m+n) - (m+1) = n-1$ pieces, for which the ratio of the number of bins in the FF-packing of the worst case permutation of the list

to the optimal number of bins would be

$$\frac{FF(L') - 1}{L'^* - 1} > \frac{FF(L')}{L'^*} \geq \frac{FF(L)}{L^*},$$

contradicting our hypothesis that L yielded the maximum ratio over all lists with n or fewer pieces (unless $FF(L)/L^* \leq 1$, in which case there would be nothing to prove anyway). Thus we may assume that $B' > S/(m-1) - 3$, and hence $B'/S > 1/(m-1) - 3/S$, so that

$$\begin{aligned} \frac{FF(L')}{L'^*} &\leq 1 + \frac{1}{(m+1)(1+1/(m-1)-3/S) - 1} + \frac{m+1}{n'} \\ &\leq 1 + \frac{1}{\frac{2}{m+1} + \frac{3(m+1)}{S}} + \frac{m+1}{n'} \end{aligned}$$

Now $B' < S/(m-1) - 3$, so that $n' = S+B' < S(1+1/(m-1)) - 3$, and hence $S > [(m-1)/m] \cdot [n'+3] > n'/2$. Therefore,

$$\begin{aligned} \frac{FF(L')}{L'^*} &\leq 1 + \frac{1}{\frac{2}{m+1} + \frac{6(m+1)}{n'}} + \frac{m+1}{n'} \\ &\leq 1 + \frac{1}{\frac{m+1 + 2}{m+1 + 2/(m-1)}} + \epsilon \end{aligned}$$

for any $\epsilon > 0$, given sufficiently large n , and so the upper bound holds for $R[FF, X]$, and Case 1B is proved. \square

Case 1IA. This case follows from the fact that for all L with $\text{Range}(\text{size}_L) \subseteq [1/2, 1]$, $FF(L) = L^*$. If L contains no pieces of size $1/2$ this is obvious, since FF will pack the pieces one per bin and an optimal packing can do no better. Furthermore, since there can be no more than one bin in a FF-packing with level $\leq 1/2$, pieces of size $1/2$ will be packed two to a bin under FF, and again the optimal packing can do no better. \square

Case 1IB. In this case the interval X must contain numbers both larger and smaller than $1/2$, but none smaller than $1/4$. We wish to prove that $R[FF, X] = 3/2$.

For the lower bound on $R[FF, X]$, simply consider a list L of N pieces of size $1/2 + \epsilon$, and N pieces of size $1/2 - \epsilon$, with all the smaller pieces coming first, and ϵ small enough so that all the sizes are in the given interval X . Then $FF(L) = 3N/2$, $L^* = N$, and $R[FF, X] \geq 3/2$ follows by Lemma 1.1.

For the upper bound, we will perform a discrete version of the upper bound proof for Theorem 2.6. Define a weighting function f as follows:

$$f(a) = \begin{cases} 1, & \text{if } 1/2 < \text{size}(a) \leq 1, \\ 1/2, & \text{if } 1/4 < \text{size}(a) \leq 1/2, \\ 1/4, & \text{if } \text{size}(a) = 1/4. \end{cases}$$

Now let L be a list with $\text{Range}(\text{size}_L) \subseteq X$, P^* be an optimal packing of L , and PF be the FF-packing of L . We again proceed by a series of Claims.

Claim 2.8.1. If $\sum_{i=1}^n \text{size}(a_i) \leq 1$, then $\sum_{i=1}^n f(a_i) \leq 3/2$.

Proof of Claim. Assume the a_i 's are indexed according to decreasing size. There are four cases depending on the value of n :

If $n = 1$, then $f(a_1) \leq 1$ by definition of f .

If $n = 2$, then $\text{size}(a_2)$ cannot exceed $1/2$, so $f(a_1) + f(a_2) \leq 1 + 1/2 = 3/2$.

If $n = 3$, then $\text{size}(a_1) \leq 1/2$, else there would not be room for two there pieces, so $\sum_{i=1}^3 f(a_i) \leq (3)(1/2) = 3/2$.

If $n = 4$, then all four pieces must have size = $1/4$, and $\sum_{i=1}^4 f(a_i) = (4)(1/4) = 1$. \square

Now let the coarseness of BIN_j in PF_b defined, as in Theorem 2.6, as $\text{MAX}\{\text{gap}_{\text{PF}}(j'): 1 \leq j' < j\}$.

CLAIM 2.8.2. If $\text{cont}_{\text{PF}}(j) = \{a_1, \dots, a_n\}$, $\text{level}_{\text{PF}}(j) > 3/4$, and BIN_j has coarseness $d < 1/4$, then $\sum_{i=1}^n f(a_i) \geq 1$.

Proof of Claim. Assume again that the pieces are indexed in order of decreasing size. If $\text{size}(a_1) > 1/2$, we are done, since $f(a_1) = 1$. If $\text{size}(a_1) \leq 1/2$, there must be an a_2 , and if $\text{size}(a_2) > 1/4$ we are done, since $f(a_1) + f(a_2) = (2)(1/2) = 1$. If $\text{size}(a_1) \leq 1/2$ and $\text{size}(a_2) = 1/4$, there must be an a_3 with $\text{size} = 1/4$. If then $\text{size}(a_1) > 1/4$, we are done since $\sum_{i=1}^3 f(a_i) = 1/2 + (2)(1/4) = 1$. If $\text{size}(a_1) = 1/4$, then there must be an a_4 and so $\sum_{i=1}^4 f(a_i) = (4)(1/4) = 1$. This exhausts the possibilities. \square

CLAIM 2.8.3. If $\text{cont}_{\text{PF}}(j) = \{a_1, \dots, a_n\}$, $\text{level}_{\text{PF}}(j) > 1/2$, and BIN_j has coarseness d , $1/4 \leq d < 1/2$, then $\sum_{i=1}^n f(a_i) \geq 1$.

Proof of Claim. Assume again that the a_i 's are indexed in order of decreasing size. By the first fit rule, all the a_i 's must have $\text{size} > d > 1/4$. If $\text{size}(a_1) > 1/2$, we are done since $f(a_1) = 1$. If not, then there must be an a_2 , and $f(a_1) + f(a_2)$

$$= (2)(1/2) = 1. \quad \square$$

CLAIM 2.8.4. $\sum_{a \in \text{PIECES}(L)} f(a) \geq \text{FF}(L) - 2.$

Proof of Claim. The coarsenesses of the bins of PF form an increasing sequence, and if $\text{level}_{PF}(j) < 1-d$, then the coarseness of BIN_{j+1} is at least d . Thus there can be at most one bin with coarseness $< 1/4$ and level $\leq 3/4$, and at most one bin with coarseness $< 1/2$ and level $\leq 1/2$. Moreover, any non-empty bin with coarseness $\geq 1/2$ must contain a piece a_1 of size $> 1/2$ and so have $f(a_1) = 1$. Thus, Claims 2.8.2 and 2.8.3 imply that all but at most two bins have $\sum_{a \in \text{cont}_{PF}(j)} f(a) \geq 1$. The Claim follows. \square

To complete the upper bound proof for Case IIB, we simply note that by Claim 2.8.1, $\sum_{a \in \text{PIECES}(L)} f(a) \leq (3/2)L^*$, so that by Claim 2.8.4, $\text{FF}(L) \leq (3/2)L^* + 2$. Lemma 1.1 then applies. \square

Case IIc. In this case the interval X must contain numbers both larger than $1/2$ and smaller than $1/4$, but none smaller than $1/6$. We wish to prove $R[\text{FF}, X] = 5/3$.

The lower bound examples are generated using the same method used in Theorems 1.5 and 2.6, so we will just describe the lists and leave the reader to work out the details. There

will be three types of pieces, one of pieces of size slightly less than $1/4$, one of pieces of size slightly more than $1/4$, and one of pieces all slightly bigger than $1/2$. Again we will be grouping the pieces into blocks, with N blocks, each containing 6 pieces of each type. Choose a d , $0 < d < \min(1/4-p, 1/20)$, and let $d(i) = d \cdot (5^{-i})$ for $1 \leq i \leq N$. The pieces of the three types corresponding to block i will be $a_{j,i}$, $b_{j,i}$, and $c_{j,i}$, $1 \leq j \leq 6$, with sizes

$$\begin{aligned}a(1,i) &= a(2,i) = \dots = a(6,i) = 1/4 - d(i), \\b(1,i) &= b(2,i) = \dots = b(6,i) = 1/4 + 4d(i), \\c(1,i) &= c(2,i) = \dots = c(6,i) = 1/2 + d(i).\end{aligned}$$

In sequence notation, the list will be

```
<b(1,1), b(2,1), b(3,1), b(4,1), a(1,1), a(2,1),
b(5,1), a(3,1), a(4,1), b(6,1), a(5,1), a(6,1)
      . . .
b(1,N), b(2,N), b(3,N), b(4,N), c(1,1), a(2,1),
b(5,N), a(3,N), a(4,N), b(6,N), a(5,N), a(6,N),
c(1,1), ..., c(6,N)>.
```

The reader may verify that this insures that under FF the $a_{j,i}$'s and $b_{j,i}$'s will go three to a bin, for a total of $4N$ bins, and the $c_{j,i}$'s will take up another $6N$ bins for a total of $FF(L) = 10N$, whereas there is a packing of the pieces into $6N + 1$ bins, using our usual tricks, and so by Lemma 1.1 $R[FF, X] \geq 10/6 = 5/3$, as desired.

For the upper bound, we will perform another discrete version of the upper bound proof for Theorem 2.6. Since there is nothing particularly new about the details, we shall simply define the weighting function and state the claims, leaving the rest to the reader. The weighting function is given by:

$$f(a) = \begin{cases} 1, & \text{if } 1/2 < \text{size}(a) \leq 1, \\ 2/3, & \text{if } 1/3 < \text{size}(a) \leq 1/2, \\ 1/3, & \text{if } 1/6 < \text{size}(a) \leq 1/3, \\ 1/6, & \text{if } \text{size}(a) = 1/6. \end{cases}$$

Now let L be a list with $\text{Range}(\text{size}_L) \subseteq X$, P^* an optimal packing of L , PF the FF-packing of L , and define coarseness as in Case IIB. The claims used to prove the upper bound are then simple analogues of those used in Case IIB:

Claim 2.8.5. If $\sum_{i=1}^n \text{size}(a'_i) \leq 1$, then
 $\sum_{i=1}^n f(a'_i) \leq 5/3$.

Claim 2.8.6. If $\text{cont}_{PF}(j) = \{a_1, \dots, a_n\}$, $\text{level}_{PF}(j) > 5/6$
and BIN_j has coarseness $d < 1/6$, then $\sum_{i=1}^n f(a'_i) \geq 1$.

CLAIM 2.8.7. If $\text{cont}_{PF}(j) = \{a_1, \dots, a_n\}$, $\text{level}_{PF}(j) > 2/3$,
and BIN_j has coarseness d , $1/6 \leq d < 1/3$, then $\sum_{i=1}^n f(a'_i) \geq 1$.

Claim 2.8.8. If $\text{cont}_{PF}(j) = \{a_1, \dots, a_n\}$, $\text{level}_{PF}(j) > 1/2$,
and BIN_j has coarseness d , $1/3 \leq d < 1/2$, then $\sum_{i=1}^n f(a'_i) \geq 1$.

Claim 2.8.9. $\sum_{a \in \text{PIECES}(L)} f(a) \geq \text{FF}(L) - 3$.

The upper bound for Case 11C then follows from Claims 2.8.5 and
2.8.8 via Lemma 1.1. \square

Thus we have proved all the cases of Theorem 2.8 and so the
whole Theorem is proved. \square

The type of analysis done in this section could be extended
to the problem of finding the worst case behavior when the
pieces of L are restricted to a finite union of intervals, for
instance, finding the value of $R[\text{FF}, X]$ when $X = (0, 1/5) \cup$

SECTION 2.2 - Page 90

[1/2,2/3). In this case the value is 8/5, and similar results could be obtained for other combinations of intervals. However, if Theorem 2.8 is any indication, the general solution would just be an incredibly long list of special cases, and we shall leave the complete solution to even the problem of the union of two intervals to later researchers.

SECTION 2.3. Implementation of ANY FIT Algorithms

In this section we begin by considering the problem of implementing the ANY FIT algorithms, with an eye toward determining how much time is required to generate the S-packing of a list L as a function of list length n. We prove that under certain reasonable assumptions about the implementation, all AF algorithms require time proportional to $n \log n$, and show that the four AF algorithms (FF, BF, WF, and AWF) that we have explicitly defined all can be implemented in $O(n \log n)$. We then consider ways of modifying the algorithms so that they take only linear time, and yet do not suffer from a degradation of worst case behavior.

We must talk about the "implementation" of our 2-part algorithms because they have only been specified by rules, not explicit programs, and there may be more than one way to accomplish the desired effect of a given rule. That is, we have so far left open the problem of how to permute a given list so that it satisfies a preprocessing rule, or how to determine, given a set of bin levels and a piece size, which bin satisfies the requirements of a packing rule.

Although all the algorithms we have studied so far have been assumed to operate on-line, with the pieces packed one at a time as they are input, we shall for the sake of the generality

of our lower bound proof consider implementations which need not output any information about the packing until they have seen the entire list, but then must output a complete specification of the packing (or set of packings in the case of incompletely determined packing rules) that could result if the packing rule was applied as specified in our PART 2 program. The implementations to which we shall restrict our attention will be branching implementations, ones which select the packing(s) by a sequence of yes-no tests, the result of each test determining either what test to make next or what packing(s) to output.

To be specific, given a list of length n , the S-packing (for S a 2-part algorithm) is a sequence on n bins (some possibly empty). There are thus only a finite number of possible packings of lists of length n , each corresponding to a sequence of n disjoint sets whose overall union is $\{1, 2, \dots, n\}$, where the j 'th set consists of the ranks of the pieces in BIN of the packing. A branching implementation of a bin packing algorithm S for lists of length n will be a binary tree, whose internal nodes correspond to yes-no tests and whose leaves correspond to sets of possible packings of lists of length n . The tree must obey the property that any input list L of length n determines a path from the root to a leaf node by determining the answers to the tests at the internal nodes, and the leaf node at the end on the path corresponds to the set of packings

generable by S applied to L . The cost for L of the implementation will be the number of internal nodes in the path.

This is the same type of theoretical framework within which the problem of sorting has been extensively analyzed, and all practical implementations of the algorithms would seem to fit into it, although of course the underlying tree structure will be implicit. Moreover, it is not unreasonable to expect that actual running times of such implementations would be proportional to the cost, as defined here.

We now briefly describe implementations of FF, BF, WF, and AWF, all of which do operate on-line, packing one piece at a time, and still have worst-case cost bounded by a constant times $n \log n$.

For FIRST FIT, we initially construct a binary tree of depth $\lceil \log_2 n \rceil$ with n leaves corresponding to the n initially empty bins, in sequence from left to right. The nodes of the tree are all labeled 1. As the packing progresses the labels will be updated, so that when a new piece is to be assigned, the label at each leaf will be the gap of the corresponding bin, and the label at each internal node will be the max of the labels of its two offspring, and hence the size of the largest gap over all the bins which descend from that node. See Figure 2.11 for an example of the state of the tree when a piece is about to be

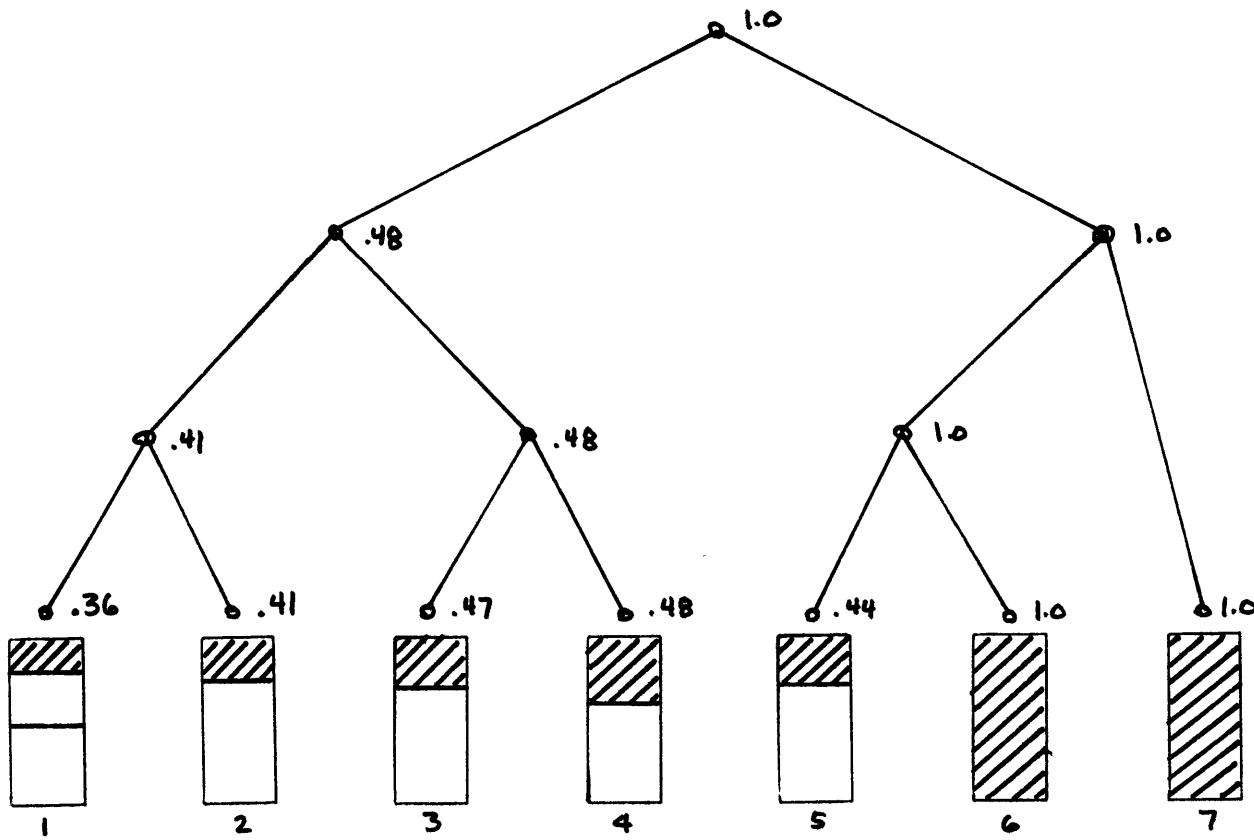


FIGURE 2.11. Tree Directory for the implementation of FIRST FIT, $n = 7$.

assigned.

To assign a piece, we merely start at the root of the tree and traverse to a leaf by always taking the leftmost branch which is labeled with a number larger than the piece size. This will require at most $\lceil \log_2 n \rceil$ comparisons (yes-no tests). After assigning the piece we must update the tree, and this will require an additional $\lceil \log_2 n \rceil$ comparisons. We first update the label of the leaf node to reflect the updated gap of the corresponding bin, and then proceed back up the path, relabeling each internal node with the max of its two (one possibly updated) offspring. Thus the total cost of the implementation for any list L of length n is at most $2n\lceil \log_2 n \rceil$.

Note that the tree structured directory in the above implementation is not to be confused with the tree structure of the branching implementation itself, which of course would be much more immense if it were all explicitly drawn out. The above implementation is not completely on-line, since we must know the length of the list in advance. For the other three algorithms, this piece of information is unnecessary (except insofar as that in a practical situation we are subject to storage constraints).

For BF, WF, and AWF, which unlike FF do not depend predominantly on the order of the bins in the original left to right sequence in making their assignments, we may use

structures such as AVL trees [Ad1] or trees of bounded balance [Ni1] for our bin directories. These can keep the bins sorted by gap, and can be searched (to find the appropriate fit) and updated, both in time $O(\log n)$. The details are left to the reader.

Summarizing, we have

THEOREM 2.9. For S any one of FF, BF, WF, and AWF there exists an implementation of S and a constant k such that if L is any list and its length is n , the cost for L of the implementation is at most $kn\log_2 n$.

In the light of the following, we cannot expect to do much better:

THEOREM 2.10. There exists a k such that for all $n \geq 4$, and any implementation of any AF algorithm for lists of length n , there exists a list L for which the cost is at least $kn\log_2 n$.

Proof. Let $j = \lfloor n/2 \rfloor$. We shall assume n is even and $j = n/2$, but if n is odd, merely add a piece of size 1 to the lists we present. Consider the set of lists represented by

$\langle x_1, \dots, x_j, 1/2-d, 1/2-2d, \dots, 1/2-jd \rangle,$

where $\langle x_1, \dots, x_j \rangle$ is any permutation of $\langle 1/2+d, 1/2+2d, \dots, 1/2+jd \rangle$, and $0 < d < 1/2j$. Note that there are $j!$ such lists, each is packed the same way by all AF algorithms, and each yields a packing different.

For consider the operation of S on one of the lists. The first j pieces are all larger than $1/2$, so they go into BIN_1 thru BIN_j in the order they appear in the list. The next piece has size $1/2-d$, and the only non-empty bin it will fit in will be the one containing the piece of size $1/2+d$, which depends on the particular permutation, and it must go there by the AF constraint. Then the next piece will only fit in the bin with the piece of size $1/2+2d$, and so on. Thus the bins which the last j pieces go into are completely determined by the permutation of the first j .

Therefore the tree representing the branching implementation must have at least $j! = \lfloor n/2 \rfloor!$ distinct leaf nodes, and hence the path to at least one of them must contain at least $\lceil \log_2(\lfloor n/2 \rfloor!) \rceil$ internal nodes. The list whose S-packing corresponds to that leaf thus costs at least that much. By Stirling's formula there exists a k such that for all n , this exceeds $kn\log_2 n$. \blacksquare

The intuition behind the above result is that all AF algorithms implicitly require some form of sorting in their implementation. If we wish to develop linear time algorithms, we must somehow avoid this problem.

We in fact already have discussed a number of linear time algorithms: NEXT- k FIT for $k \geq 1$. These avoid the growth in the cost of sorting by putting a fixed bound on the number of bins that need be considered, and consequently NkF has an upper bound of kn on the number of comparisons required for a list of length n . Moreover, by Theorems 1.5 and 2.5, for $k \geq 2$, NkF has the same worst case behavior as FF, at least when the maximum piece size is $\leq 1/2$.

The remainder of this section will be devoted to another way of avoiding the problems with sorting, while still not suffering from a degradation of worst case behavior. In addition, as we shall see in Chapter 7, this method can be adapted to yield very good average case results, while still taking only linear time.

The basic idea is to replace the direct comparisons of piece size versus bin gap or bin gap versus bin gap by comparisons against some fixed standards which serve to classify the pieces and gaps into groups all of whose members have similar sizes.

Let a schedule of intervals be a set $X = \{x_1, \dots, x_k\}$, where $x_1 = 0$, $x_k = 1$, and $x_i < x_{i+1}$, $1 \leq i < k$. X will be thought of as a partition of the unit interval $[0,1]$ into the subintervals

$$[x_1, x_2), [x_2, x_3), \dots, [x_{k-1}, x_k), \{1\}.$$

$[x_i, x_{i+1})$ will be called interval X_i , with interval X_k being $\{1\}$.

Given a schedule of intervals X , we have the following packing rule:

GROUP-X FIT (GXF): To assign piece b , let $i' = \min\{i: x_i \geq \text{size}(b) \text{ and for some } j, 1 \leq j \leq n, g_{app}(j) \in X_i\}$. Choose BIN_j such that $g_{app}(j) \in X_{i'}$ subject to the constraint that if BIN_j is empty, $j = \min\{j': \text{level}_P(j') = 0\}$.

The final constraint is present only to insure that no piece is assigned to a bin which is to the right of an empty bin, a fact that does not in this case influence the number of bins used by the packing, but which will make our proofs more straightforward. Note that i' is chosen so that b will fit in any BIN_j such that $g_{app}(j) \in X_{i'}$.

If Schedule X has k subintervals, GXF can be implemented

SECTION 2.3 - Page 100

using k stacks, $\text{STACK}_1, \dots, \text{STACK}_k$, one for each interval, and storing BIN_j (more precisely a pointer to a representation of BIN_j) on STACK_i if $\text{gap}_p(j) \in X_i$.

Initially, all bins are on STACK_k , ordered by index with BIN_1 on top. When b is to be assigned, we first find

$$i'' = \{\min i : x_i \geq \text{size}(b)\},$$

by a binary search using no more than $\lceil \log_2 k \rceil$ comparisons. We then search for

$$i' = \min\{i \geq i'': \text{STACK}_i \text{ is not empty}\},$$

pop the top bin on $\text{STACK}_{i'}$, and put b in the required position of that bin. Lastly, we find the proper stack on which to place the updated bin by a second binary search, again using no more than $\lceil \log_2 k \rceil$ comparisons.

Although the algorithm is clearly $O(n)$ for fixed k , we would like an implementation which is $O(n \log k)$, so that even for large k the constant of proportionality would not be unmanageable. Thus we cannot find i' given i'' by testing each stack in turn for non-emptiness, as this can require a number of tests linear in k .

Instead, we consider the stacks to be laid out in a line,

and construct a binary tree over them with the stacks as leaf nodes, much as we did in our implementation of FF, only now the leaves correspond to stacks rather than individual bins, and each internal node of the tree need only store the information as to whether there is a non-empty stack beneath it in the tree. It is a simple matter to update the tree in $O(\log k)$ when a stack is changed, and also to use the tree as a directory and find i' given i'' using only $O(\log k)$ tests at internal nodes.

Thus the total cost of each piece assignment is $O(\log k)$ and the whole packing costs $O(n \log k)$.

Note that GXF operates by leaving a margin of safety. If we define mesh(X) to be $\max\{x_{i+1} - x_i : 1 \leq i < |X|\}$, then if $gap(j) > size(b) + mesh(X)$ we know that b can be assigned to BIN_j under GXF. On the other hand, if $size(b) > gap(j) - mesh(X)$, it may be impossible to assign b to BIN_j even though it does fit. However, subject to the margin of error, GXF does try to put b in a bin so as to minimize the gap left. Thus as $mesh(X)$ approaches 0, GXF becomes more and more like BF, and we would expect its average case behavior to improve. (Note the similarity to the situation with NkF, which as k increases becomes more and more like FF and is expected to have better and better average case behavior).

When we turn to the question of worst case behavior, we

find that GXF is as good as BF for a surprisingly small schedule of intervals X (just as NkF was as good as FF from a worst case point of view as soon as $k = 2$). We shall reach this conclusion after presenting a series of lemmas. Our first Lemma formalizes the most important fact about the way the algorithm operates:

LEMMA 2.11. Suppose $y, z \in X$, $y < z$, and b is a piece in a list being packed by GXF, with $\text{size}(b) \leq y$. If when b was assigned there was a j' such that $\text{gapp}(j') \in [y, z]$, then b was not assigned to any BIN_j with $\text{gapp}(j) \geq z$.

Proof. Suppose $y = x_{i1}$, $z = x_{i2}$. Then when b was assigned, $\text{BIN}_{j'}$ was on some $\text{STACK}_{i'}$, $i1 \leq i' < i2$, and BIN_j was on some STACK_i , $i \geq i2$. Since $\text{size}(b) \leq x_{i1}$, STACK_i could have been chosen only if all $\text{STACK}_{i''}$, $i1 \leq i'' < i$, were empty, and this is not the case. \square

Using Lemma 2.11, we can prove analogues of Lemmas 2.3 and 2.4 for AAF algorithms, under a weak restriction as to the nature of X :

LEMMA 2.12. Suppose for $m \geq 2$ that $\{1/m, 1/(m+1)\} \subseteq X$, L is a list with $\text{Range}(\text{size}_L) \subseteq (0, 1/m]$, and PX is a GXF-packing of L . If $b = \text{piece}_{PX}(j, 1)$ for $j \leq \#PX$, then when b was assigned we had for all j' , $1 \leq j' < j$,

- (a) $\text{level}_P(j') > (m-1)/m$ and
- (b) $\text{height}_P(j') \geq m$.

Proof of Lemma. For (a), suppose for some $j' < j$ $\text{level}_P(j') \leq (m-1)/m$ when b was assigned. $\text{BIN}_{j'}$ can not have been empty at the time, since $j' < j$ and BIN_j is at the top of the empty bin stack. Thus $\text{gap}_P(j') \in [1/m, 1)$, $\text{gap}_P(j) \in \{1\}$, and $\text{size}(b) \leq 1/m$, so Lemma 2.11 would tell us that b did not go in BIN_j , a contradiction. (b) follows from (a) since at least m pieces of size $\leq 1/m$ are required to yield a level $> (m-1)/m$. |

LEMMA 2.13. Suppose L , X , and PX are as above and in addition $1 \leq j < \#PX$ and for some j' , $1 \leq j' < j$, $\text{gap}_{PX}(j') > 1/(m+1)$. Then BIN_j contains m pieces larger than $1/(m+1)$ in PX .

Proof of Lemma. Since $j+1 \leq \#PX$, Lemma 2.12 tells us that BIN_j must contain m pieces. Let $b = \text{piece}_{PX}(j, h)$, $1 \leq h \leq m$.

When b was assigned, BIN_j contained no more than $m-1$ pieces, and hence had $\text{gap}_P(j) > 1/m$. But again by Lemma 2.12, we know that $\text{gap}_P(j') < 1/m$, and hence by assumption ϵ

$[1/(m+1), 1/m]$. Thus if $\text{size}(b) \leq 1/(m+1)$, Lemma 2.11 would not allow b to go in BIN_j , and hence in fact $\text{size}(b) > 1/(m+1)$. \square

THEOREM 2.14. For $m = \lfloor 1/t \rfloor \geq 2$ and $\{1/(m+1), 1/m\} \subseteq X$

$$R[GXF, t] = 1 + 1/m.$$

Proof. The lists used to prove the same lower bound for NkF , $k \geq 2$, in Theorem 1.5 are such that for any $X \supseteq \{1/(m+1), 1/m\}$, GXF will generate the same packing as NkF (and for no X can GXF do any better than NkF), and so $R[GXF, t] \geq R[NkF, t] \geq 1 + 1/m$. The upper bound when X is as stated follows from Lemma 2.13 just as the upper bound in Theorem 2.5 followed from Lemma 2.4. \square

Thus for $t \leq 1/2$, and X the corresponding two-element set, we have $R[GXF, t] = R[BF, t]$. Although we cannot prove it, we conjecture that if $t > 1/2$ and $\{1/6, 1/3, 1/2\} \subseteq X$, then we also have $R[GXF, t] = R[FF, t] = 17/10$.

CHAPTER 3. ANY FIT DECREASING ALGORITHMS COMPARED

SECTION 3.1. Preprocessing Rules

In this and the next three chapters we consider 2-part algorithms in which the preprocessing rule is no longer vacuous, and which thus can no longer be considered to be "on-line" algorithms. To simplify our proofs about such algorithms, let us first state a rather trivial extension of Lemma 1.1:

LEMMA 3.1. Suppose SP is a 2-part algorithm with packing rule S and preprocessing rule P, and $X \subseteq (0,1]$.

(A) If there exists a K such that for all lists L with $\text{Range}(\text{size}_L) \subseteq X$ and rank_L obeying P, we have $S(L) \leq rL^* + K$, then $R[SP, X] \leq r$.

(B) If there exists a K' and a sequence of lists L_n with limit $L_n^* = \infty$, $\text{Range}(\text{size}_{L_n}) \subseteq X$, rank_{L_n} obeys P, and $S(L_n) \geq rL_n^* - K'$, then $R[SP, X] \geq r$.

The preprocessing rule to which we shall devote most of our attention is the following:

DECREASING RULE: L is in decreasing order, that is,

$$\text{size}(a) > \text{size}(b) \implies \text{rank}_L(a) < \text{rank}_L(b).$$

A list $L = \langle b_1, b_2, \dots, b_n \rangle$ which obeys this rule will have $b_1 \geq b_2 \geq \dots \geq b_n$.

If S is a packing rule, the algorithm specified by S and the DECREASING RULE will be called S DECREASING or simply SD. In Chapters 3, 4, and 5 we will be concerned with the case of SD where $S \in AF$, and we shall call the class of such algorithms AFD. The use of the DECREASING rule for preprocessing tends to lessen the difference between such algorithms. Recall that for Theorem 2.7 we exhibited lists L with $\text{Range}(\text{size}_L) \subseteq (1/4, 1]$ such that $\text{FF}(L)/\text{BF}(L) = 3/2$, and other lists such that $\text{BF}(L)/\text{FF}(L) = 4/3$. In [Ga1] it was shown that if L is a list in decreasing order, with $\text{Range}(\text{size}_L) \subseteq [1/5, 1]$, then $\text{FF}(L) = \text{BF}(L)$. But then, FF and BF do have the same worst case behavior. Perhaps even more surprising, in light of Theorem 2.1, is the fact that if $\text{Range}(\text{size}_L) \subseteq (1/4, 1]$, then for all $S \in AF$

$$\text{FF}(L) \leq S(L) \leq \text{FF}(L) + 1.$$

SECTION 3.1 - Page 107

This chapter will be devoted to proving these results and others of the same type, which in addition to any interest they might inspire on their own, have applications in the next two chapters when we prove upper bounds on $R[SD, t]$.

SECTION 3.2. A Proof Method

The results we wish to prove in this chapter are all of the form $S(L) \leq K$ where S is an AF algorithm and L a list in decreasing order. In the current section we develop a method for proving such results by induction. Informally, in our proofs we shall show that at each step of the generation of the S -packing of L , there is a way of extending the current packing to a packing of the whole list which uses no more than K bins.

More formally, if $L = \langle a_1, \dots, a_n \rangle$ in sequence notation, for each i , $1 \leq i \leq n+1$ let

$$\begin{aligned} L_i &= \langle a_1, \dots, a_{i-1} \rangle \\ L'_i &= \langle a_i, \dots, a_n \rangle. \end{aligned}$$

Then we have $L = L_i \cdot L'_i$, $1 \leq i \leq n+1$.

In accord with our description of PART 2 of a 2-part algorithm in Chapter 1, we may think of the generation of an S -packing of L as actually the generation of a sequence of packings P_1, \dots, P_{n+1} , where P_1 is a sequence of n empty bins making up a packing of $L_1 = \emptyset$, and in general each P_i is a sequence of n bins constituting a packing of L_i , with P_{n+1} being the final S -packing of L .

A property of this sequence is that for $1 \leq i \leq n$, P_i is a

subpacking of P_{i+1} , that is, for all $(j, h) \in \text{Domain}(\text{piece}_{P_i})$, $\text{piece}_{P_{i+1}}(j, h) = \text{piece}_{P_i}(j, h)$. What we shall show by induction is that for all i , $1 \leq i \leq n+1$, there is a packing Q_i of L with $\#Q_i \leq K$ and such that P_i is a subpacking of Q_i . The basic mechanism behind the induction will be to compare where the piece with rank 1 in L_i goes in P_{i+1} with its position in Q_i , and if they are not the same position, to switch pieces around in Q_i to obtain a packing Q_{i+1} of L of which P_{i+1} is a subpacking.

It will be easier to talk about our induction, however, if, instead of explicitly spelling out the entire packing Q_i , we merely tell how to construct it, given P_i . To this end we define the concept of an assignment function or a.f. Suppose P is a packing of list L_1 , and L_2 is a second list. Let $\text{EMPTYPOS}(P) = \text{POS}(P) - \text{Domain}(\text{piece}_P)$. An assignment function of L_2 into P is any 1-1 map

$f: \text{PIECES}(L_2) \dashrightarrow \text{EMPTYPOS}(P)$,

satisfying

(A) For all BIN_j in P ,

$$\text{gap}_P(j) \geq \sum_{\substack{b \text{ s.t. for some } h \\ f(b) = (j, h)}} \text{size}(b),$$

Thus if f is an a.f., all the pieces that map to a given bin of P will fit together in the gap of that bin. Let $\#f = \text{MAX}\{j : \text{there exist } h \text{ and } b \text{ such that } f(b) = (j, h)\}$. Note that if we were to place each piece b of L_2 into position $f(b)$ in P , we would obtain a (not necessarily ordered) packing of $L_1 \bullet L_2$, using no more than $\text{MAX}\{\#P, \#f\}$ bins.

As an instructive example, consider for each i , $1 \leq i \leq n$, the straightforward map

$$f_i : \text{PIECES}(L_i^*) \rightarrow \text{EMPTYPOS}(P_i^*),$$

given by

$$f_i^{-1}(b) = \text{piece}_{P_{n+1}^*}(b) = \text{the position } b \text{ fills in the final packing},$$

where the P_i 's are as defined for the generation of an S-packing of L. Each f_i so defined is an a.f. from L'_i into P_i , with $\#f_i \leq S(L)$. The basic lemma we shall use in our proofs will be the converse of this example:

LEMMA 3.2. Suppose $S \in AF$, L is a list with L_i, L'_i defined as above. If for all i , $1 \leq i \leq |PIECES(L)|$, and any sequence of P_i 's involved in the generation of an S-packing of L there exists an a.f. f_i from L'_i into P_i such that $\#f_i \leq K$, then $S(L) \leq K$.

Proof. Let $n = |PIECES(L)|$. Suppose an S-packing uses more than K bins. Then at some time during its generation a piece b must have been put in position $(K+1,1)$. Let $\text{rank}(b) = i$. Then by assumption there is an a.f. f_i of L'_i into P_i , where P_i is the packing into which b was placed by the algorithm. But by definition of a.f., $f_i(b) = (j,h)$ is an unfilled position of P_i , and b will fit in the gap in BIN_j . Since $\#f_i \leq K$, $j < K+1$, so BIN_j is to the left of BIN_{K+1} . Thus b cannot have been placed in position $(K+1,1)$ without violating the AF constraint, and the Lemma is proved by contradiction. \square

Thus our proof method, restated in terms of a.f.'s, will be to inductively define maps f_i with $\#f_i \leq K$, and then prove by induction that they are indeed a. f.'s, so that Lemma 3.2 will apply with the desired result.

f_1 , since it will initialize the induction, is of special significance. Since it is an a.f. of L into an empty packing, it will be piece_{BP}^{-1} for some base packing BP of L. The properties of this base packing will have a direct bearing on the induction as it proceeds.

For instance, we shall always have $\text{Range}(f_1) \subseteq \text{Range}(f_i) = \text{Domain}(\text{piece}_{BP})$. Moreover, BP will be what we shall call a semi-ordered packing, which means that it obeys packing properties (1) and (2) but not necessarily (3), and hence, although only the bottom-most positions in each bin are filled, they are not necessarily filled in order of increasing rank. Some additional concepts we shall need:

If P is a semi-ordered packing, let

$$\begin{aligned} \text{TOP}(P) &= \{(j, h) \in \text{Domain}(\text{piece}_P) : \\ &\quad (j, h+1) \notin \text{Domain}(\text{piece}_P)\}, \\ \text{NONTOP}(P) &= \text{Domain}(\text{piece}_P) - \text{TOP}(P). \end{aligned}$$

We shall omit the argument (P) whenever no confusion will result. By packing property (2) each non-empty bin of P has exactly one TOP position.

If p is a 1-1 partial map from $N \times N$ to $\text{PIECES}(L)$, a capacity map for p is a map

$$\text{CAP}: N \times N \rightarrow \mathbb{Q},$$

such that

(1) If $(j, h) \notin \text{Domain}(p)$, $\text{CAP}(j, h) = 0$,

(2) If $(j, h) \in \text{Domain}(p)$,

$$\text{CAP}(j, h) \geq \text{size}(p(j, h)),$$

(3) For all $j \geq 1$, $\sum_{h=1}^{\infty} \text{CAP}(j, h) \leq 1$.

CAP defines a capacity for each position. Note that if p has a capacity map, then by capacity map properties (2) and (3), $p = \text{piece}_P$ for a packing P of L .

We shall initialize our inductions by defining both a base packing BP and a capacity map CAP , and the above observation will allow us to save some time in showing that both constructions satisfy their respective definitions. The following series of lemmas then become applicable.

LEMMA 3.3. Suppose BP is a semi-ordered packing of L with $\#BP \leq K \leq n = |\text{PIECES}(L)|$, and CAP is a capacity map for piece_{BP} . Then the map f_1 defined for all $b \in \text{PIECES}(L)$ by $f_1(b) = \text{piece}_{BP}^{-1}(b)$, is an a.f. from $L = L'$ into P_1 , with $\#f_1 \leq K$.

Proof. f_1 is 1-1 since piece_{BP} is. Since P_1 is by definition the empty packing with n bins,

$$\text{EMPTYPOS}(P_1) = \{(j, h) : 1 \leq j \leq n, h \geq 1\}.$$

Thus, since $K \leq n$,

$$f_1 : \text{PIECES}(L) \longrightarrow \text{EMPTYPOS}(P_1),$$

and all we need verify is that assignment function property (A) holds. But this follows immediately from the fact that CAP obeys capacity map properties (2) and (3). \square

LEMMA 3.4. Suppose there exists a semi-ordered packing BP of L with $\#BP \leq K$, CAP is a capacity map for piece_{BP} , and L is in decreasing order, then for each i , $1 \leq i \leq |\text{PIECES}(L)|$, there exists a 1-1 map

$f_i : \text{PIECES}(L'_i) \rightarrow \text{EMPTYPOS}(P'_i)$,

with $\#f_i \leq K$ and satisfying

(P3.4) For each $b \in \text{PIECES}(L'_i)$, $\text{size}(b) \leq \text{CAP}(f_i(b))$.

Proof. f_1 is the map defined in Lemma 3.3, and obeys (P3.4) since by definition of CAP and f_1 , $\text{CAP}(f_1(b)) \geq \text{size}(\text{piece}_{\text{BP}}(f_1(b))) = \text{size}(b)$. We define f_i , $1 < i \leq |\text{PIECES}(L)|$, and prove it also satisfies the Lemma, by the following inductive procedure:

Suppose f_i is defined and satisfies the Lemma, and that the piece b with rank 1 in $\text{PIECES}(L'_i)$ goes into position (j_1, h_1) in P'_{i+1} . Then for each $c \in \text{PIECES}(L'_{i+1})$ we define

$$f'_{i+1}(c) = \begin{cases} f_i(b), & \text{if } f_i(c) = (j_1, h_1), \\ f_i(c), & \text{otherwise.} \end{cases}$$

Since f_i is 1-1 and b is not in $\text{PIECES}(L'_{i+1})$, f'_{i+1} will be 1-1. Since $\text{Range}(f_i) \subseteq \text{EMPTYPOS}(P'_i)$, $\#f_i \leq K$, and the only position in $\text{EMPTYPOS}(P'_i) - \text{EMPTYPOS}(P'_{i+1})$ is (j_1, h_1) , we will have $\text{Range}(f'_{i+1}) \subseteq \text{EMPTYPOS}(P'_{i+1})$ and $\#f'_{i+1} \leq K$. Thus all we need show is that f'_{i+1} obeys (P3.4).

Since f_i obeyed (P3.4), only a $c \in \text{PIECES}(L'_{i+1})$ for which $f'_{i+1}(c) \neq f_i(c)$ could violate (P3.4) for f'_{i+1} . By the

definition of f_{i+1} from f_i , the only such c must have $f_{i+1}(c) = f_i(b)$, where b is the piece with rank 1 in L'_i . But then $\text{rank}(c) > \text{rank}(b)$ and since L is in decreasing order, we must have

$$\text{size}(c) \leq \text{size}(b) \leq \text{CAP}(f_i(b)),$$

by (P3.4) for f_i . Thus (P3.4) holds for f_{i+1} . \square

LEMMA 3.5. If BP is a semi-ordered packing of L , CAP is a capacity map for piece_{BP},

$$f: \text{PIECES}(L'_i) \dashrightarrow \text{EMPTYPOS}(P_i)$$

is a 1-1 map obeying (P3.4), and P_i obeys

(P3.5) If $(j, h) \in \text{NONTOP}(BP)$ is filled in P_i by a piece b , then $\text{size}(b) \leq \text{CAP}(j, h)$,

then f is an a.f. from L'_i into P_i .

Proof. All we must show is that f obeys assignment function property (A). Let BIN_j be a bin in P_i , some of whose positions are in the range of f . Let $R(j)$ be the set of these positions. We must show that

$$\text{gap}_{P_i}(j) \geq \sum_{f(b) \in R(j)} \text{size}(b),$$

But the fact that $R(j)$ is non-empty means that at least one position in BIN_j with non-zero capacity is unfilled in P_i , by (P3.4) for f . Thus the $\text{TOP}(\text{BP})$ position in BIN_j cannot be filled, and so, if $F(j)$ is the set of positions in BIN_j which are filled in P_i , we must have $F(j) \subseteq \text{NONTOP}(\text{BP})$. Thus we have

$$\begin{aligned} \text{gap}_{P_i}(j) &= 1 - \sum_{(j,h) \in F(j)} \text{size}(\text{piece}_{P_i}(j,h)), \\ &\geq 1 - \sum_{(j,h) \in F(j)} \text{CAP}(j,h), \quad [\text{by (P3.5)}], \\ &\geq \sum_{(j,h) \in R(j)} \text{CAP}(j,h), \quad [\text{by capacity map property (3)}] \\ &\geq \sum_{f(b) \in R(j)} \text{size}(b), \quad [\text{by (P3.4)}]. \quad \square \end{aligned}$$

Lemmas 3.2 thru 3.5 together reduce the problem of proving results of the form $S(L) \leq K$ to that of applying the following lemma:

LEMMA 3.6. Suppose BP is a semi-ordered packing of a decreasing list L with $\#BP \leq K$, CAP is a capacity map for piece_{BP}, and $S \in AF$. Then if for all i , $1 \leq i \leq |PIECES(L)|$, and any sequence of P_i 's involved in the generation of an S-packing of L, P_i obeys property (P3.5), we have $S(L) \leq K$.

Proof. Since the P_i 's obey (P3.5), by Lemmas 3.4 and 3.5, there is a sequence of a.f.'s f_i , $1 \leq i \leq |PIECES(L)|$, with $\#f_i \leq K$, so the result follows by Lemma 3.2. \square

SECTION 3.3. Results for Arbitrary ANY FIT algorithms

In this section we shall prove the result about decreasing lists L with $\text{Range}(\text{size}_L) \subseteq (1/4, 1]$ mentioned in Section 3.1, and a generalization thereof. Let us first look at some of the properties of an AF-packing of a decreasing list:

LEMMA 3.7. Suppose L is a list in decreasing order, $S \in \text{AF}$, and PS is an S -packing of L . If $\text{size}(\text{piece}_{PS}(j,1)) \leq 1/m$, and $d = \text{MAX}\{\text{gap}_{PS}(j'): 1 \leq j' < j\}$, then for every j'' , $j \leq j'' < \#PS$,

- (1) $\text{BIN}_{j''}$ contains at least m pieces of size $> d$, and
- (2) for $1 \leq h \leq m$,

$$\text{size}(\text{piece}_{PS}(j'',h)) \geq \text{size}(\text{piece}_{PS}(j''+1,1)).$$

Proof. Since $\text{BIN}_{j''}$ is not the last bin, we know there exists a piece $b = \text{piece}_{PS}(j''+1,1)$. Let $a = \text{piece}_{PS}(j'',1)$. By Lemma 2.2 and its Corollary, and the fact that L is in decreasing order,

$$d < \text{size}(b) \leq \text{size}(a) \leq 1/m.$$

By packing property (3) and the fact that L is in decreasing order, we know that all the pieces in $\text{BIN}_{j''}$ have size $\leq 1/m$. Thus if $\text{BIN}_{j''}$ contained fewer than m pieces with rank less than that of piece b , Lemma 2.2 would be violated. Hence by packing

property (3) the bottom m positions of BIN_j^n must be filled with pieces of smaller rank (and hence no smaller size) than piece b . Both (1) and (2) follow. \square

In particular, if $\text{Range}(\text{size}_L) \subseteq (1/4, 1]$, we can partition $\text{PIECES}(L)$ into the three sets:

$$\begin{aligned} A\text{-PIECES}(L) &= \{a \in \text{PIECES}(L) : \text{size}(a) \in (1/2, 1]\}, \\ B\text{-PIECES}(L) &= \{b \in \text{PIECES}(L) : \text{size}(b) \in (1/3, 1/2]\}, \\ C\text{-PIECES}(L) &= \{c \in \text{PIECES}(L) : \text{size}(c) \in (1/4, 1/3]\}. \end{aligned}$$

For $X \in \{A, B, C\}$, let us call an element of $X\text{-PIECES}(L)$ an X -piece, a bin whose bottom piece is an X -piece an X -bin, and let us drop the argument on $X\text{-PIECES}(L)$ whenever no confusion will arise. We now consider the construction of the S -packing of L (see Figure 3.1):

S first assigns the A -pieces to the first $|A\text{-PIECES}|$ bins, one per bin in order of decreasing size. This yields a sequence of A -bins, whose gaps form an increasing sequence, are less than $1/2$, and hence have room for only one more piece from L .

Next the B -pieces are assigned, and those which do not go into the gaps in A -bins fill up a series of B -bins, each B -bin receiving two B -pieces before the next is started. Hence the B -bins have gaps in an increasing sequence, and all except the

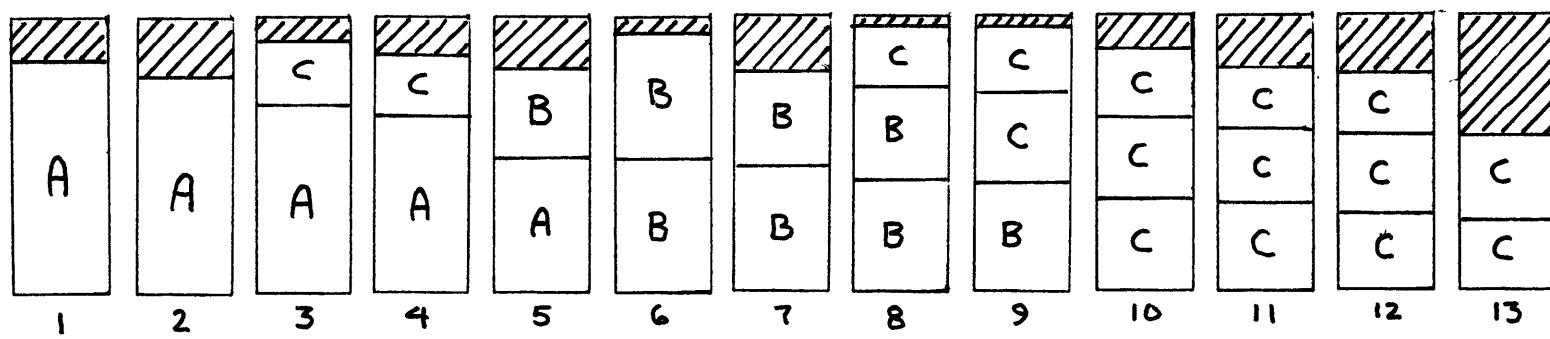


FIGURE 3.1. S-Packaging of decreasing list L with $\text{Range}(\text{size}_L) \in (\frac{1}{4}, 1]$

rightmost B-bin contain two B-pieces, have gaps less than $1/3$, and hence have room for at most one more piece from L.

Finally, the C-pieces are assigned. If the last B-bin had only one B-piece, its gap exceeded $1/2$, and by Lemma 3.5 no C-bin will be started until that B-bin's second position is filled. Then the C-pieces which do not go into the gaps of A- or B-bins will fill up successive C-bins, each receiving three C-pieces before the next is started.

Suppose we define an order relation on positions by

$$(j, h) \leq (j', h') \iff j < j' \text{ or } j = j' \text{ and } h \leq h',$$

we then have

LEMMA 3.8. Suppose $S \in AF$, L is a list in decreasing order with $\text{Range}(\text{size}_L) \subseteq (1/4, 1]$, and PS is an S-packing of L. If $(j, h) \in \text{NONTOP}(PS)$, $(j', h') \in \text{Domain}(\text{piece}_{PS})$, and $(j, h) \leq (j', h')$, then $\text{size}(\text{piece}_{PS}(j, h)) \geq \text{size}(\text{piece}_{PS}(j', h'))$.

Proof. If $j = j'$, the result follows from packing property (3) and the fact that L is decreasing. If $j < j'$, consider BIN_j . If it is an A-bin, then it can contain at most two pieces so we must have $h = 1$, and $\text{piece}_{PS}(j, h)$ is the A-piece. But as

we have seen that A-piece must have been assigned before any piece went into a bin to the right such as $\text{BIN}_{j'}$. If BIN_j is a B- or C-bin, then we must have $h \leq 2$, and BIN_j received its bottom two pieces before any piece went in a bin to the right. Thus in either case $\text{piece}_{\text{PS}}(j', h')$ has higher rank than $\text{piece}_{\text{PS}}(j, h)$ and hence can have no larger size. \square

We are now ready to prove our result:

THEOREM 3.9. If $S \in \text{AF}$ and L is a decreasing list with $\text{Range}(\text{size}_L) \subseteq (1/4, 1]$, then

$$\text{FF}(L) \leq S(L) \leq \text{FF}(L) + 1.$$

Lower Bound Proof. Let our base packing BP simply be an S -packing of L , so that clearly $\#BP \leq S(L)$. Then define a capacity map for piece_{BP} by simply letting $\text{CAP}(j, h) = \text{size}(\text{piece}_{\text{BP}}(j, h))$ for each $(j, h) \in \text{Domain}(\text{piece}_{\text{BP}})$, and $\text{CAP}(j, h) = 0$, otherwise. By Lemma 3.8, CAP obeys

(P3.9) If $(j, h) \in \text{NONTOP}(BP)$, $(j', h') \in \text{Domain}(\text{piece}_{\text{BP}})$, and $(j, h) \leq (j', h')$, then $\text{CAP}(j, h) \geq \text{CAP}(j', h')$.

(Property (P3.9) is the only place where the fact that $\text{Range}(\text{size}_L) \subseteq (1/4, 1]$ is used in this lower bound proof).

Let P_1, \dots, P_{n+1} be the sequence of packings involved in the generation of the FF-packing of L . By Lemma 3.6, all we must show is that, for $1 \leq i \leq n = |\text{PIECES}(L)|$, P_i obeys property (P3.5), which we recall says that if $(j, h) \in \text{NONTOP}(BP)$, and P_i has a piece b in position (j, h) , then $\text{size}(b) \leq \text{CAP}(j, h)$.

(P3.5) clearly holds for P_1 since no positions in P_1 are filled. Suppose it holds for some P_i , $1 \leq i < n$. Since the pieces already assigned positions in P_i retain the same positions in P_{i+1} , the only piece that could cause a violation of (P3.5) for P_{i+1} is the piece b with rank 1 in L'_i , and then only if in P_{i+1} that piece goes into a position $(j_1, h_1) \in \text{NONTOP}(BP)$, and $\text{size}(b) > \text{CAP}(j_1, h_1)$.

Now since (P3.5) holds for P_i , by Lemmas 3.3 and 3.4 there exists an a.f. f_i from L'_i into P_i obeying property (P3.4). Let $(j', h') = f_i(b)$. There are two cases:

Case 1. $(j', h') \geq (j_1, h_1)$. Then by (P3.4) for f and (P3.9), $\text{size}(b) \leq \text{CAP}(j', h') \leq \text{CAP}(j_1, h_1)$, so (P3.5) is not violated for $i+1$.

Case 2. $(j', h') < (j_1, h_1)$. (j_1, h_1) must be the bottom-most unfilled position in BIN_{j_1} in P_i . (j', h') must also be unfilled since it is in the range of an a.f. Thus we cannot have $j' = j_1$, $h' < h_1$, and so we must have $j' < j_1$. By a.f. property (A), b would have fit in $\text{BIN}_{j'}$ in P_i , and hence could not have gone to the right to BIN_{j_1} without violating the FF packing rule, so this case is impossible.

Thus by induction (P3.5) is satisfied for all i , $1 \leq i \leq n$, and the desired lower bound follows via Lemma 3.6. \square

Upper Bound Proof. For this proof we reverse the roles of S and FF. We cannot, however, use the exact same arguments as in the lower bound proof. For instance, in Case 2 above, S, as opposed to FF, might well place a piece b to the right of a bin into which it would fit, as long as b does not go into a bottom position. Thus, although we will use the FF-packing PF of L as a basis for our construction of our base packing BP, the construction will be a bit more complicated than before, as will be our definition of CAP.

To save time, we shall define piece_{BP} and $\text{CAP}: N \times N \dashrightarrow \mathbb{Q}$ in parallel. First let us divide PF into three (possibly vacuous) segments. Let PFA be the A-bins of PF, PFB

the B-bins, and PFC the C-bins. Let $j_3 = 1 + \#PFA + \#PFB$. See Figure 3.2.

piece_{BP} and CAP are defined as follows:

If BIN_j is in segment PFA, and b fills position (j,h) in PF, set $\text{piece}_{BP}(j,h) = b$, $CAP(j,h) = \text{size}(b)$.

If BIN_j is in segment PFB, and b fills position (j,h) in PF, and

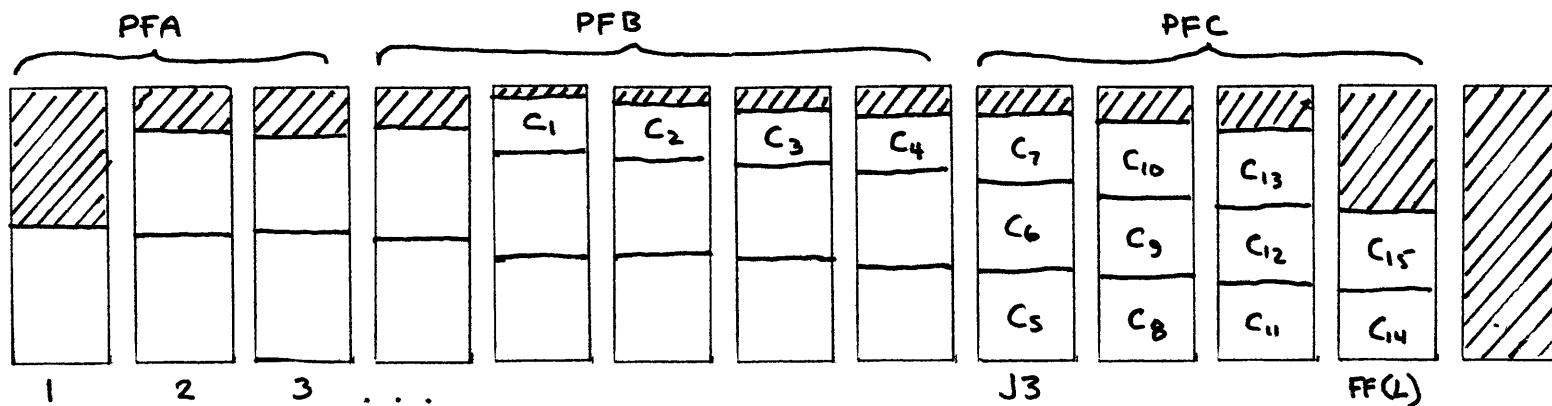
$h = 1$, set $\text{piece}_{BP}(j,1) = b$, $CAP(j,1) = \text{size}(b)$,
 $h = 2$, set $\text{piece}_{BP}(j,2) = b$, $CAP(j,2) = CAP(j,1)$
 $h = 3$, set $\text{piece}_{BP}(j+1,h') = b$, $CAP(j+1,h') = \text{size}(b)$, where
if $j+1 = j_3$, $h' = 1$,
if $j+1 < j_3$, $h' = \text{MIN}\{3, 1 + \text{height}_{PF}(j+1)\}$

If BIN_j is in segment PFC, and b fills position (j,h) in PF, and

$h = 1$, set $\text{piece}_{BP}(j,1) = b$, $CAP(j,1) = \text{size}(b)$,
 $h > 1$, set $\text{piece}_{BP}(j,h) = b$, $CAP(j,h) = CAP(j,1)$.

For all other positions, $CAP(j,h) = 0$, and piece_{BP} is undefined.

FIRST FIT PACKING PF of L:



BASE-PACKING BP of L:

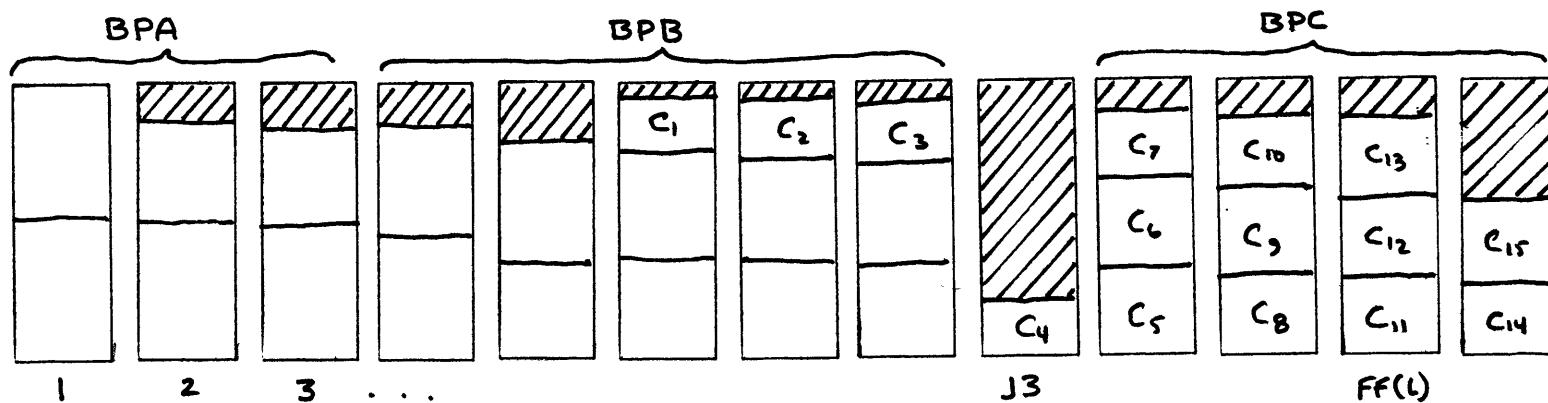


FIGURE 3.2. FIRST FIT Packing and BASE Packing of L.

(Pieces with changed positions are labeled c_i).

The next three claims will establish that piece_{BP} defines a semi-ordered packing BP, with $\text{POS}(BP) = \{(j, h) : 1 \leq j \leq \text{FF}(L)+1, h \geq 1\}$, and that CAP is a capacity map for piece_{BP}, thus setting us up for an application of Lemma 3.6.

CLAIM 3.9.1. piece_{BP} is a 1-1 partial map from $N \times N$ to PIECES(L) satisfying packing property (2).

Proof of Claim. Range(piece_{BP}) = PIECES(L) since no bin in PF could have contained more than three pieces of size exceeding $1/4$. That piece_{BP} is 1-1 and that packing property (2) is obeyed will be evident when we examine what has been done to PF to get BP (See Figure 3.2):

The bins of PFA have remained unchanged and make up segment BPA of BP. The bins of PFB have become the bins of BPB in BP, and remain unchanged except that all third pieces in these bins in PF have been shifted one bin to the right in BP. Finally, each bin of PFC has had its entire contents shifted one bin to the right, thus making up segment BPC. This in essence creates a new bin between BPB and BPC, BIN_{j3}, which will either be empty or contain a single piece in position (j3,1). The complicated business in the definition of piece_{BP} for the PFB, h = 3 case is to insure that a 3rd piece in BIN_{j3-1} in PF will go into the bottom position in BIN_{j3} in BP, or that if BIN_{j3-1} contains only

1 piece in PF (by Lemma 3.7 this latter case can only occur if PFC is vacuous) a third piece in BIN_{j-2} in PF will go in position $(j-1, 2)$ in BP, thus insuring that piece_{BP} is 1-1 and obeys packing property (2). \square

CLAIM 3.9.2. CAP is a capacity map for piece_{BP}.

Proof of Claim. That CAP obeys capacity map property (1) is immediate from its definition. For (2), note that the only time $\text{CAP}(j, h) \neq \text{size}(\text{piece}_{BP}(j, h))$ is when $\text{CAP}(j, h) = \text{size}(b)$ for $b = \text{piece}_{BP}(j, 1)$ and $c = \text{piece}_{BP}(j, h)$ was above b in the same bin in PF. By packing property (3) for PF and the fact that L is in decreasing order, we have that $\text{size}(c) \leq \text{size}(b) = \text{CAP}(j, h)$, and so capacity map property (2) is satisfied. All that remains is property (3):

If BIN_j is in BPA,

$$\sum_{h=1}^{\infty} \text{CAP}(j, h) = \sum_{b \in \text{cont}_{PF}(j)} \text{size}(b) \leq 1.$$

If BIN_j is in BPB, then since $\text{CAP}(j, 2) + \text{CAP}(j, 1) \leq 2(1/2)$, the only problem that could arise would be if $\text{CAP}(j, 3)$ were non-zero, that is, if position $(j-1, 3)$ were filled by some piece d in PF and BIN_j contained at least two pieces in PF. See

Figure 3.3. Let $a = \text{piece}_{PF}(j-1, 1)$, $b = \text{piece}_{PF}(j-1, 2)$, and $c =$

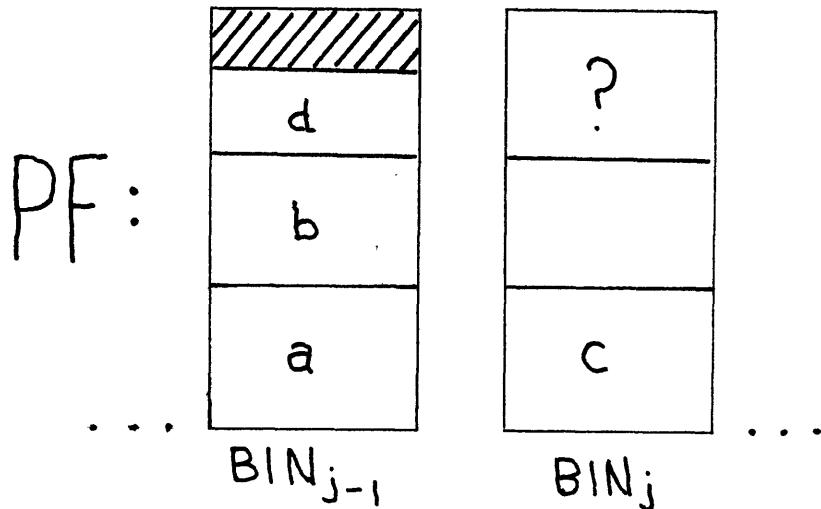


FIGURE 3.3

$\text{piece}_{\text{PF}}(j,1)$. Then $\text{size}(a) \geq \text{size}(b)$ by packing property (3), and $\text{size}(b) \geq \text{size}(c)$ by Lemma 3.7 applied to PF. Thus $1 \geq \text{size}(a) + \text{size}(b) + \text{size}(d) \geq 2\text{size}(c) + \text{size}(d) = \sum_{h=1}^{\infty} \text{CAP}(j,h)$.

If $j = j_3$, the only position in BIN_j with possibly non-zero capacity is $(j,1)$, and $\text{CAP}(j,1) \leq 1/3$, since the third piece in BIN_{j-1} in PF must be a C-piece.

If BIN_j is in BPC, then $\text{CAP}(j,1) \leq 1/3$, and

$$\sum_{h=1}^{\infty} \text{CAP}(j,h) \leq 3\text{CAP}(j,1) \leq 1.$$

For $j > \#BP$, $\sum_{h=1}^{\infty} CAP(j, h) = 0$, and so capacity map property (3) is satisfied and Claim 3.9.2 is proven. \blacksquare

CLAIM 3.9.3. BP is a semi-ordered packing
with $\#BP \leq FF(L)+1$.

Proof of Claim. Capacity map properties (2) and (3) for CAP imply that BP obeys packing property (1), so Claims 3.9.1 and 3.9.2 together imply that BP is a semi-ordered packing. Since piece_{BP} is undefined for all (j, h) with $j > \#PF+1 = FF(L)+1$, $\#BP \leq FF(L)+1$. \blacksquare

Claims 3.9.2 and 3.9.3 set us up for an application of Lemma 3.6 to derive our desired result. All that remains is an induction on property (P3.5). The next two claims give us the information about CAP and BP which will enable us to perform the induction:

CLAIM 3.9.4. If $(j, h) \in \text{NONTOP}(BP)$, $CAP(j, h) = CAP(j, 1)$.

Proof of Claim. If BIN_j is in BPA, then it has room for at most one piece besides its A-piece, hence position $(j, 1)$ is the only possible member of NONTOP(BP) and the Claim is satisfied trivially.

If BIN_j is in BPB and $(j,2) \in \text{NONTOP}$, then the bin contains three pieces in BP, and so must have contained at least two pieces in PF, and so $\text{CAP}(j,2)$ was defined as $\text{CAP}(j,1)$ in the $h = 2$ case for PFB.

BIN_{j_3} can contain at most one piece in BP and so has no NONTOP positions.

If BIN_j is in BPC, then by definition $\text{CAP}(j,h) = \text{CAP}(j,1)$ for all (j,h) which are filled in BP. \square

CLAIM 3.9.5. If $(j,h) \in \text{NONTOP}(\text{BP})$, $(j',h') \in \text{Domain}(\text{piece}_{\text{BP}})$, and $(j,h) \leq (j',h')$, then $\text{CAP}(j,h) \geq \text{CAP}(j',h')$. [This is the same as (P3.9) in the lower bound proof.]

Proof of Claim. We first consider the case $j = j'$, $h \leq h'$:

If BIN_j is in BPA, then $\text{CAP}(j,h') = \text{size}(\text{piece}_{\text{PF}}(j,h'))$ and $\text{CAP}(j,h) = \text{size}(\text{piece}_{\text{PF}}(j,h))$ and the result follows from packing property (3) and the fact that L is in decreasing order.

If BIN_j is in BPB, all positions that are filled by the same pieces in both PF and BP have capacities equal to $\text{CAP}(j,1)$. Thus the only problem that could arise would be if there were a piece c in position $(j-1,3)$ in PF, and $\text{size}(c) > \text{size}(\text{piece}_{\text{PF}}(j,1)) > 1/3$. But this is impossible since such a

piece c would have to be a C-piece by size constraints.

j cannot be j_3 , since BIN_{j_3} contains no NONTOP positions.

If BIN_j is in BPC, all nonzero capacities are equal.

Thus the Claim holds if $j = j'$. The only other possibility is $j < j'$, in which case by Claim 3.9.4, $\text{CAP}(j, h) = \text{CAP}(j, 1)$, and by the above, $\text{CAP}(j', h') \leq \text{CAP}(j', 1)$. So by definition of CAP all we must show is that if $j < j'$, and $j \neq j_3$,
 $\text{size}(\text{piece}_{BP}(j, 1)) \geq \text{size}(\text{piece}_{BP}(j', 1))$.

If $j' = j_3$, then BIN_j is either an A-bin or B-bin in BP, and since $\text{piece}_{BP}(j_3, 1)$ must be a C-piece, the conclusion is immediate.

If $j' \neq j_3$, then $d(j) < d(j')$, where $d(i) = i$, if $i < j_3$, and $i-1$ if $i > j_3$. Since we have $\text{piece}_{BP}(i, 1) = \text{piece}_{PF}(d(i), 1)$ for $i \in \{j, j'\}$, the result follows from Lemma 1.2 and the fact that L is in decreasing order. \square

We are now ready to complete the upper bound proof for Theorem 3.9. Let P_1, \dots, P_{n+1} be any sequence of packings involved in the generation of an S-packing of L. By Lemma 3.6 all we need shows that for all i , $1 \leq i \leq |\text{PIECES}(L)| = n$, (P3.5) holds for P_i , that is, no NONTOP(BP) position in P_i is filled by a piece larger than the position's capacity. (P3.5) holds for P_1 since no positions in P_1 are filled. Suppose it

holds for P_i with $i < n$. Then by Lemmas 3.4 and 3.5 there exists an a.f. f_i' from L'_i to P_i satisfying (P3.4).

Since (P3.5) holds for P_i , the only way (P3.5) could be violated for P_{i+1} would be if the piece b with rank 1 in L'_i went in P_{i+1} into a NONTOP(BP) position (j_1, h_1) and $\text{size}(b) > \text{CAP}(j_1, h_1)$. Let $(j', h') = f_i(b)$. Again there are two cases:

Case 1. $(j', h') \geq (j_1, h_1)$. Then by (P3.4) for i and Claim 3.9.5, $\text{size}(b) \leq \text{CAP}(j', h') \leq \text{CAP}(j_1, h_1)$, so (P3.5) is not violated for P_{i+1} .

Case 2. $(j', h') < (j_1, h_1)$. Since (j_1, h_1) must be the bottom-most unfilled position in BIN_{j_1} in P_i , and (j', h') must also be unfilled, we must have $j' < j_1$. See Figure 3.4. Since b would fit in $\text{BIN}_{j'}$ by a.f. property (A) for f_i' , b cannot go in position $(j_1, 1)$ without violating the AF constraint, so $h_1 > 1$. But then position $(j_1, 1)$ must be filled in P_i by some piece $c \neq b$, and by (P3.5) for i , $\text{size}(c) \leq \text{CAP}(j_1, 1)$. Furthermore, Claim 3.9.4 tells us that $\text{CAP}(j_1, h_1) = \text{CAP}(j_1, 1)$, and the fact that L is decreasing tells us that $\text{size}(b) \leq \text{size}(c)$. Therefore $\text{size}(b) \leq \text{CAP}(j_1, h_1)$ and (P3.5) is not violated in this case either.

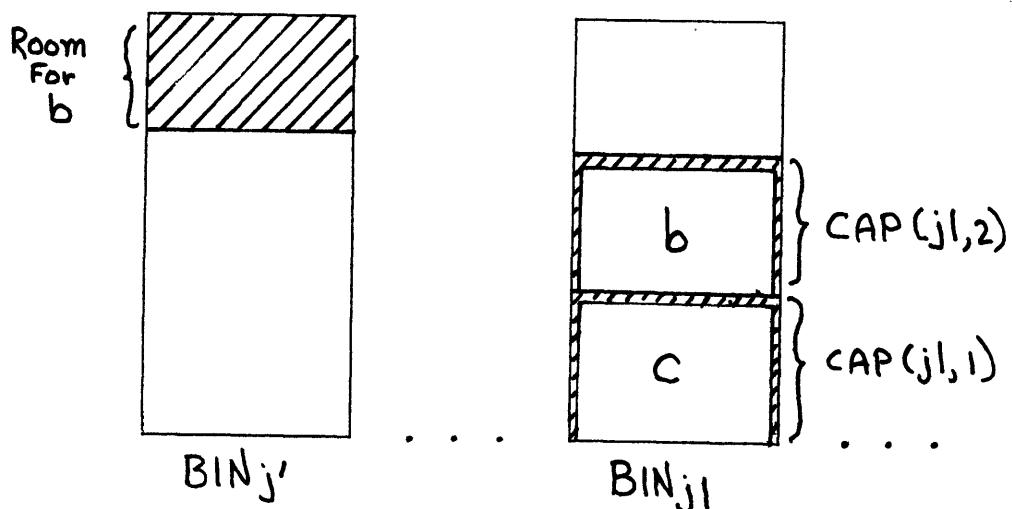


FIGURE 3.4

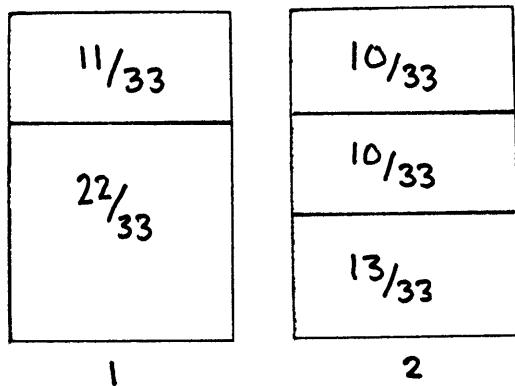
Thus by induction (P3.5) is satisfied for all i , $1 \leq i \leq n$, and the upper bound of Theorem 3.9 follows via Lemma 3.6. \square

Figure 3.5 gives an example of a decreasing list L with $\text{Range}(\text{size}_L) \subseteq (1/4, 1]$ for which $\text{WF}(L) = \text{FF}(L)+1$, so Theorem 3.9 gives the best bounds possible.

The significance of the fact that $\text{Range}(\text{size}_L) \subseteq (1/4, 1]$ in Theorem 3.9 was that this insured that Lemma 3.8 would hold and so we could prove that property (P3.9) held for CAP.

Generalizing Lemma 3.8, we have the following:

FF - PACKING



WF - PACKING

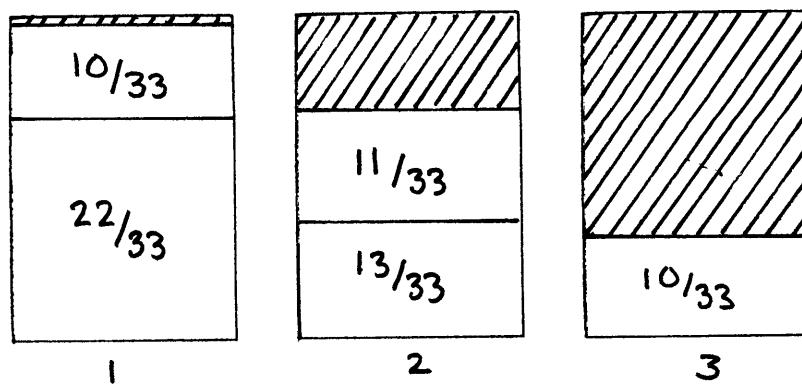


FIGURE 3.5. Decreasing list L with $\text{Range}(\text{size}_L) \subseteq (-\frac{1}{4}, 1]$

and $\text{WF}(L) = \text{FF}(L) + 1$.

LEMMA 3.10. Suppose L is a list in decreasing order with $\text{Range}(\text{size}_L) \subseteq (1/(2m+2), 1/m]$, $S \in \text{AF}$, and PS is an S -packing of L . Then if $(j, h) \in \text{NONTOP}(PS)$, $(j', h') \in \text{Domain}(\text{piece}_{PS})$, and $(j, h) \leq (j', h')$, we have $\text{size}(\text{piece}_{PS}(j, h)) \geq \text{size}(\text{piece}_{PS}(j', h'))$.

Proof. If $j = j'$ the result is immediate by packing property (3), so we may assume $j < j'$. We then have $\text{size}(\text{piece}_{PS}(j+1, 1)) \geq \text{size}(\text{piece}_{PS}(j', 1)) \geq \text{size}(\text{piece}_{PS}(j', h'))$, by Corollary 2.2.1 and packing property (3), so it is enough to show that $\text{size}(\text{piece}_{PS}(j, h)) \geq \text{size}(\text{piece}_{PS}(j+1, 1))$. See Figure 3.6. Let b and c be the respective pieces, and let k be such that $\text{size}(c) \in (1/(k+1), 1/k]$. Then by Lemma 3.7, $\text{size}(b) \geq \text{size}(c)$ unless $h > k$.

But if $h > k$, then $\sum_{i=1}^k \text{size}(\text{piece}_{PS}(j, i)) \geq k/(k+1)$, leaving a gap $\leq 1/(k+1) \leq 1/(m+1)$, and hence room for at most one piece with size $> 1/(2m+2)$. So h must be $k+1$, and $(j, h) \in \text{TOP}(PS)$, contrary to hypothesis. So we must have $h \leq k$ and $\text{size}(b) \geq \text{size}(c)$ as required. \square

We shall prove a generalization of Theorem 3.9 using Lemma 3.10 and the following observation:

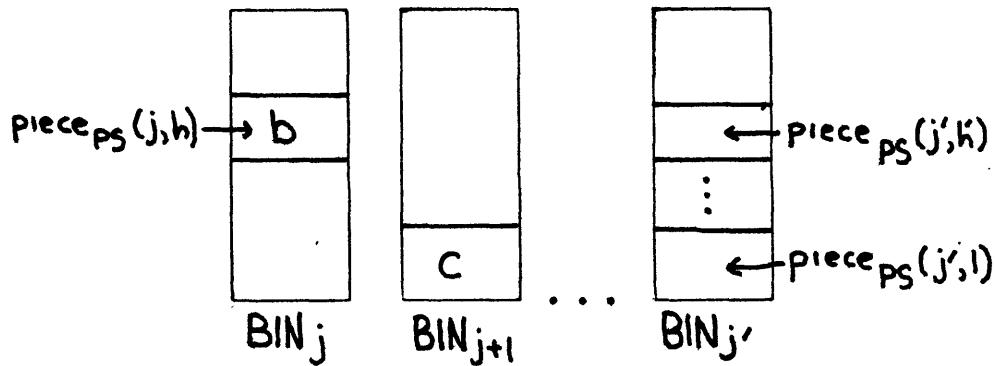


FIGURE 3.6

LEMMA 3.11. If L is a list in decreasing order with $\text{Range}(\text{size}_L) \subseteq (1/(m+1), 1/m]$, then for any $S \in \text{AF}$, S and FF yield the same packing of L .

Proof. By Lemma 2.6, BIN_2 cannot be started until BIN_1 contains m pieces, but by size constraints BIN_1 can contain no more than m pieces. Thus S will proceed by putting the first m pieces in BIN_1 , then the next m in BIN_2 , etc., until all the pieces are used up. This will be true for all $S \in \text{AF}$, and in particular for $\text{FF} \in \text{AF}$. \square

THEOREM 3.12. If L is a list in decreasing order with
 $\text{Range}(\text{size}_L) \subseteq (1/(2m+2), 1/m]$ for $m \geq 1$, and $S \in \text{AF}$, then

$$\text{FF}(L) \leq S(L) \leq \text{FF}(L) + m.$$

Lower Bound Proof. Let BP be an S -packing of L and define
 $\text{CAP}(j,h) = \text{size}(\text{piece}_{BP}(j,h))$ for $(j,h) \in \text{Domain}(\text{piece}_{BP})$, 0
otherwise. By Lemma 3.10 CAP obeys (P3.9) and we can proceed
exactly as in the lower bound proof for Theorem 3.9. \square

Upper Bound Proof. Again this is the more complicated
proof, and we must construct a more complicated BP and CAP .
First, let

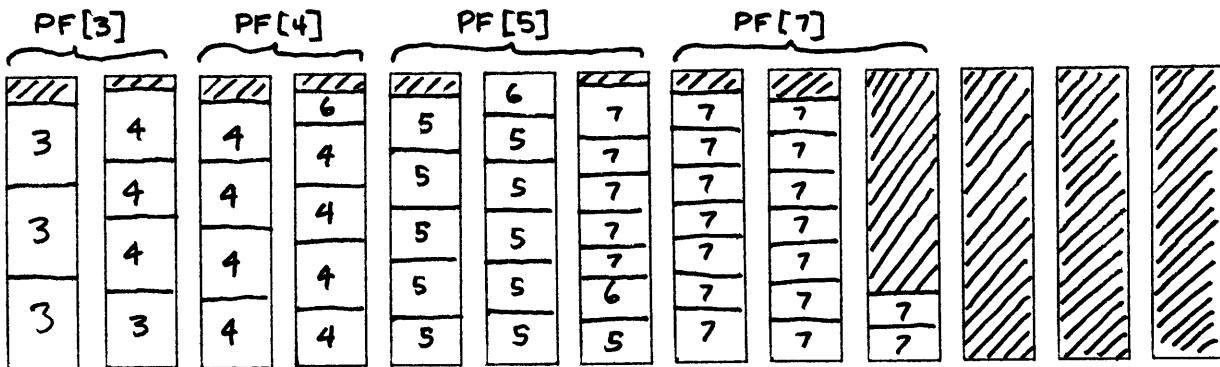
$$k\text{-PIECES}(L) = \left\{ x \in \text{PIECES}(L) : \text{size}(x) \in (1/(k+1), 1/k] \right\},$$

and define k -piece and k -bin as we previously defined A -piece
and A -bin, etc.

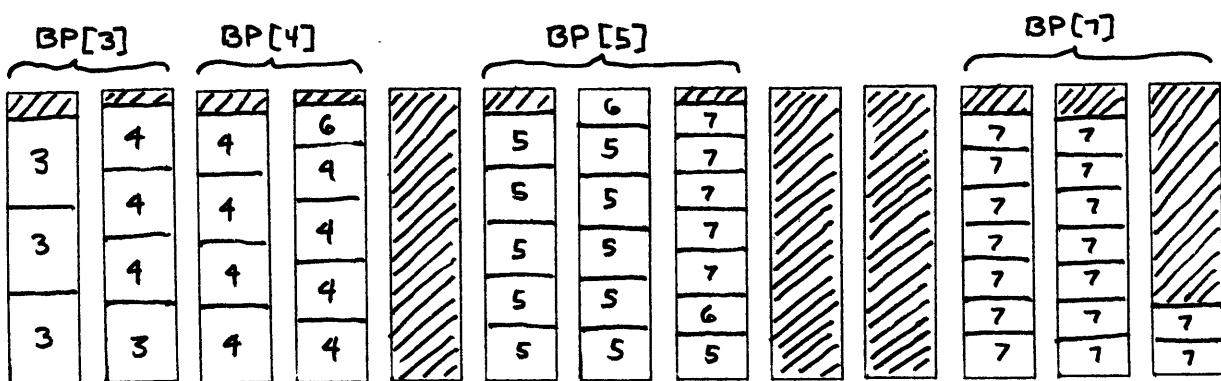
Now, let PF be the FF -packing of L , and divide it into
segments $PF|k|$, $m \leq k \leq 2m+1$, where $PF|k|$ is made up of all the
 k -bins of PF . (See Figure 3.7). We once more define piece_{BP}
and CAP in parallel, after first setting $\text{POS}(BP) = \{(j,h) : 1 \leq j$
 $\leq \#PF+m, h \geq 1\}$.

SECTION 3.3 - Page 140

FF-PACKING PF of L (PF[6] is vacuous):



(Imagined INTERMEDIARY PACKING):



BASE PACKING BP of L (Positions with same capacity are outlined):

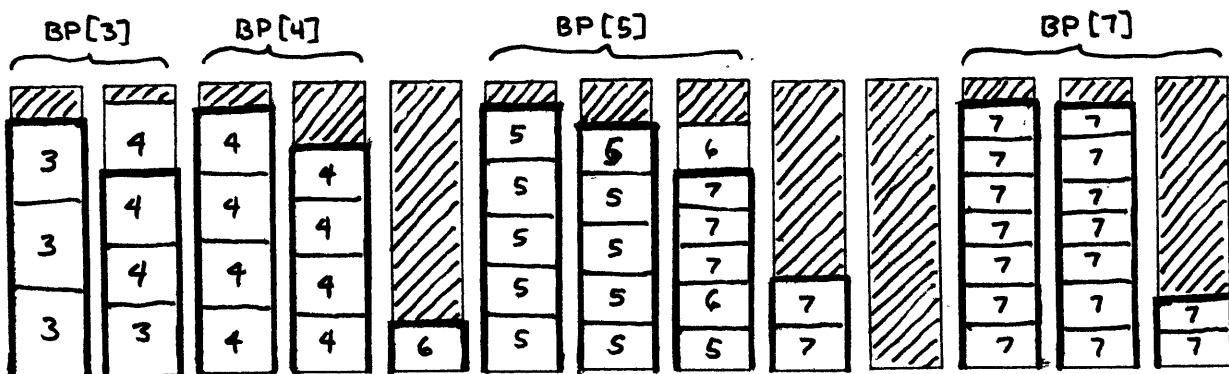


FIGURE 3.7. FF and BASE PACKINGS of L, with imagined intermediary.

If BIN_j is in $\text{PF}[m]$, and $b = \text{piece}_{\text{PF}}(j, h)$, set $\text{piece}_{\text{BP}}(j, h) = b$, $\text{CAP}(j, h) = \text{size}(b)$.

If BIN_j is in $\text{PF}[k]$ for $k > m$, $b = \text{piece}_{\text{PF}}(j, h)$, and if $h \leq k$, set

$$\begin{aligned} \text{piece}_{\text{BP}}(j+k-m-1, h) &= b, \\ \text{CAP}(j+k-m-1, h) &= \text{size}(\text{piece}_{\text{PF}}(j, 1)). \end{aligned}$$

If $h > k$ and BIN_{j+1} is a k -bin in PF , then by Lemma 2.6 the bottom k positions in BIN_j are filled with k -pieces in PF , leaving room for only one more piece, so $h = k+1$. Set

$$\begin{aligned} \text{piece}_{\text{BP}}(j+k-m, h') &= b, \\ \text{CAP}(j+k-m, h') &= \text{size}(b), \end{aligned}$$

where $h' = \min\{k+1, 1 + \text{height}_{\text{PF}}(j+1)\}$.

If $h > k$ and BIN_{j+1} is not a k -bin in PF , then by size constraints $h-k \leq 2m+1-k \leq m+1$, and also by size constraints, $\text{size}(b) \leq 1/(k+1) \leq 1/(m+1)$, and so $(h-k)(\text{size}(b)) \leq 1$. Set

$$\begin{aligned} \text{piece}_{\text{BP}}(j+k-m, h-k) &= b, \\ \text{CAP}(j+k-m, h-k) &= \text{size}(\text{piece}_{\text{PF}}(j, k+1)). \end{aligned}$$

Otherwise, $\text{CAP}(j, h) = 0$ and $\text{piece}_{\text{BP}}(j, h)$ is undefined.

CLAIM 3.12.1. piece_{BP} is a 1-1 partial map from $\mathbb{N} \times \mathbb{N}$ to $\text{PIECES}(\mathcal{L})$ satisfying packing property (2).

Proof of Claim. We shall explain with the aid of Figure 3.7 just what this construction does, and leave the rest to the reader. Basically, we can imagine the construction as a two step transformation on PF.

In the first step, each segment $PF[k]$, $m+1 \leq k \leq 2m+1$, is shifted $k-m-1$ bins to the right, becoming $BP[k]$. This introduces a "new" bin, $BIN_{j[k]}$, between each pair of segments $BP[k]$ and $BP[k+1]$, $m+1 \leq k < 2m+1$.

In the imagined second step, for each k , $m+1 \leq k < 2m+1$, and each BIN_j in segment $BP[k]$ the pieces in positions above the k 'th (bins in segment $BP[2m+1]$ can have no pieces above the $2m+1$ 'st due to size constraints) are all transferred to the bottom-most non-empty positions in BIN_{j+1} , with the excess pieces from the last bin in $BP[k]$ going into $BIN_{j[k]}$.

The reader may verify that this is the net effect of our construction and hence piece_{BP} is a 1-1 map with Range = $\text{PIECES}(L)$ and obeying packing property (2). \square

CLAIM 3.12.2. CAP is a capacity map for piece_{BP} .

Proof of Claim. Capacity map properties (1) and (2) are trivial consequences of the definition and packing property (3) for PF. So all we must show is that capacity map property (3) holds, i.e., that for all $j \geq 1$ $\sum_{h=1}^{\infty} \text{CAP}(j,h) \leq 1$.

In our definition of CAP, we have already shown that this is true if $j = j[k]$ for some k , $m+1 \leq k < 2m+1$.

If $j > \#PF+m$, then by definition $\sum_{h=1}^{\infty} CAP(j, h) = 0$.

The only other possibility is if BIN_j is in segment $BP[k]$ for some k . If $k = m$, then

$$\sum_{h=1}^{\infty} CAP(j, h) = \sum_{b \in cont_{PF}(j)} \text{size}(b) \leq 1,$$

by definition of CAP and packing property (1) for PF.

If $k > m$, the only problem that could arise would be if position $(j, k+1)$ were filled in BP, say by piece b , since all the k lower positions have $CAP \leq 1/k$. But in this case we have

$$CAP(j, h) = k \cdot \text{size(piece}_{PF}(j-k+m+1, 1)) + \text{size}(c)$$

$$\leq \sum_{h=1}^k \text{size(piece}_{PF}(j-k+m, h)) + \text{size}(c) \leq 1,$$

by Lemma 2.6 and packing property (1) for PF. \square

CLAIM 3.12.3. BP is a semi-ordered packing
with $\#BP \leq FF(L)+m$.

Proof of Claim. That BP is a semi-ordered packing follows from Claims 3.12.1 and 3.12.2. Since in our construction we created "new" bins $\text{BIN}_j[k]$ only for $m+1 \leq k \leq 2m$, there were only m such bins, so $\#BP \leq \#PF+m = \text{FF}(L)+m$. \square

CLAIM 3.12.4. If $(j,h) \in \text{NONTOP}(BP)$ and BIN_j is not in $BP[m]$, $\text{CAP}(j,h) = \text{CAP}(j,1)$.

Proof of Claim. By inspection. \square

CLAIM 3.12.5. (P3.9) If $(j,h) \notin \text{NONTOP}(BP)$, $(j',h') \in \text{Domain}(\text{piece}_{BP})$, and $(j,h) \leq (j',h')$, then $\text{CAP}(j,h) \geq \text{CAP}(j',h')$.

Proof of Claim. If $j = j'$ and $j = j|k|$ for some k , or BIN_j is in $BP[2m+1]$, the result is immediate since all nonempty positions in these bins in BP have the same capacity by definition.

If BIN_j is in $BP[m]$, then the result follows from packing property (3) for PF.

If BIN_j is in $BP[k]$, $m < k < 2m+1$, the only problem would be if $h < h'$ and $c = \text{piece}_{BP}(j,h')$ were from the bin in PF to the left of the bin that $\text{piece}_{BP}(j,1)$ came from, in which case $\text{CAP}(j,h') = \text{size}(c)$. However, in that case c was the $k+1^{\text{st}}$

piece in its original bin and hence had size $\leq 1/(k+1) < \text{size}(\text{piece}_{BP}(j,1)) = \text{CAP}(j,h)$, since $b = \text{piece}_{BP}(j,1)$ is a k -piece.

So the Claim holds if $j = j'$. If $j < j'$, by the above and Claim 3.12.4 it will be sufficient to show that $\text{CAP}(j,1) = \text{size}(\text{piece}_{BP}(j,1)) \geq \text{size}(\text{piece}_{BP}(j',1)) = \text{CAP}(j',1)$. If BIN_j is in $BP[k]$ and $\text{BIN}_{j'}$ is in $BP[k']$ for $k' > k$, then the result is immediate since $\text{piece}_{BP}(j,1)$ is a k -piece and $\text{piece}_{BP}(j',1)$ is a k' -piece. If $k' = k$, the result follows from Corollary 2.2.1.

If BIN_j is in segment $BP[k]$ and $j' = j[k']$ for some $k' \geq k$, then by packing property (3) for PF $\text{size}(\text{piece}_{BP}(j',1)) \leq \text{size}(\text{piece}_{BP}(j'-1,1))$ which is in turn $\leq \text{size}(\text{piece}_{BP}(j,1))$ by the previous argument since $\text{BIN}_{j'-1}$ is in segment $BP[k']$.

If on the other hand $j = j[k]$ for some k , then we must have $j' = j[k']$ or $\text{BIN}_{j'}$ in $BP[k']$ for some $k' > k$. Then since $(j,h) \in \text{NONTOP}(BP)$, $(j,1) \notin \text{NONTOP}(BP)$ and hence $b = \text{piece}_{BP}(j,1)$ was in a $\text{NONTOP}(PF)$ position in BIN_{j-k+m} in PF. Moreover, $c = \text{piece}_{BP}(j',1)$ cannot have come from a bin with index $\leq j-k+m$ in PF and still gone to $\text{BIN}_{j'}, j' > j$, in BP. Thus $\text{piece}_{PF}^{-1}(b) \geq \text{piece}_{PF}^{-1}(c)$ and the result follows by Lemma 3.10 for PF.

This exhausts the possibilities, so Claim 3.12.5 is proven.

□

To complete the upper bound proof, suppose P_1, \dots, P_{n+1} is any sequence of packings involved in the generation of an S-packing of L. By Lemma 3.6 all we need do is show that (P3.5) holds for all i , $1 \leq i \leq n = |\text{PIECES}(L)|$.

Let $i_1 = \min\{\text{rank}_L(x) : \text{size}(x) \leq 1/(m+1)\}$. (If there are no such pieces the upper bound holds trivially by Lemma 3.11.) Now note that for all BIN_j in segment $\text{BP}[m]$ and $(j, h) \in \text{Domain}(\text{piece}_{PF})$, $\text{piece}_{BP}(j, h) = \text{piece}_{PF}(j, h)$. Thus (P3.5) holds for P_i for all $i \leq i_1$, since by Lemma 3.11 each piece x with $\text{rank} < i_1$ goes in position $\text{piece}_{PF}^{-1}(x)$ under packing rule S, and by capacity map property (3) $\text{CAP}(\text{piece}_{BP}^{-1}(x)) \geq \text{size}(x)$.

The induction for $i_1 \leq i \leq n$ now proceeds precisely as did the induction for the upper bound proof of Theorem 3.9, since Claims 3.12.4 and 3.12.5 are identical to Claims 3.9.3 and 3.9.4. Thus (P3.5) holds for all i , $1 \leq i \leq n$, and the upper bound follows via Lemma 3.6. \square

We know of no examples of lists which realize the upper bound of Theorem 3.12. We conjecture that that the actual upper bound is $\text{FF}(L) + 1$ for $m \leq 3$, and $\text{FF}(L) + 2$ for $m \geq 4$. To prove such a strong result, one must construct a much more compact base packing than the one constructed in the Theorem 3.12 upper bound proof, but this would seem possible granted all the open spaces in the one we did construct. In fact the construction is

easy; proving that the base packing constructed has the proper number of bins is the hard part, and we have yet to verify all the details. Fortunately, in applications of Theorem 3.12 we are only interested in the fact that $S(L)$ can only exceed $FF(L)$ by a constant number of bins independent of the value of L^* , and so we do not need to have the best possible bound.

However, as it turns out, in our applications the lists L with which we shall be dealing all have $\text{Range}(\text{size}_L) \subseteq (1/(m+2), 1/m]$ for some $m \geq 2$, and for such restricted lists we do have an easy improvement on Theorem 3.12:

COROLLARY 3.12.1. If L is a list in decreasing order with $\text{Range}(\text{size}_L) \subseteq (1/(m+2), 1/m]$ for $m \geq 2$, and $S \in AF$, then

$$FF(L) \leq S(L) \leq FF(L) + 1.$$

Proof. If L is as hypothesized, the base packing BP constructed for it in the proof of Theorem 3.12 will have $BP[k]$ vacuous for all $k \geq m+2$, and the only "new" bin that can be non-empty is $BIN_{j[m+1]}$. Thus $\#BP \leq \#PF + 1 = FF(L) + 1$. Since the remainder of the proof shows that $S(L) \leq \#BP$, the result follows. \blacksquare

A second corollary shows that the AAF algorithms still maintain a slight advantage over the worst of the AF algorithms:

Corollary 3.12.1. If L is a list in decreasing order with $\text{Range}(\text{size}_L) \subseteq (1/(2m+2), 1/m]$ for $m \geq 1$, and $S \in \text{AAF}$, then $S(L) = \text{FF}(L)$.

Proof. All that is needed is a slight modification (whose details we omit) on the lower bound proof for Theorem 3.12 (actually Theorem 3.9) to show that it will work even if FF is replaced by an arbitrary AAF algorithm. Thus we have for $S \in \text{AAF}$ and $S' \in \text{AF}$, $S(L) \leq S'(L)$. Setting $S' = \text{FF}$ gives the desired result. \square

SECTION 3.4. Results for FF, BF, and Less Restricted Lists

The near-equality concluded in Theorems 3.9 and 3.12 need not hold if the pieces in L are not restricted to the required range. However, there is still no evidence of the superiority or inferiority of any particular AFD algorithm. For instance, recall that in a sense WORST FIT was the worst AF algorithm, considerably worse than FF. However, WORST FIT DECREASING seems to be on equal footing with FFD. For every list we have tried we have found that

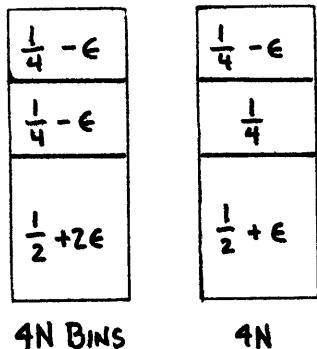
$$\frac{8}{9} \leq \text{FFD}(L)/\text{WFD}(L) \leq \frac{9}{8},$$

and both the bounds can be attained, as seen in Figures 3.8 and 3.9. Note that in these examples we still have $\text{Range}(\text{size}_L) \subseteq [1/5, 1]$. In [Ga1] Graham shows that for such lists $\text{FFD}(L) = \text{BFD}(L)$. Using Lemma 3.6 we can prove both this result and an extension, mainly that if $\text{Range}(\text{size}_L) \subseteq [1/6, 1]$, $\text{BFD}(L) \leq \text{FFD}(L)$.

First we prove a fact analogous to Lemma 3.11:

WFD-PACKING

$$WFD(L) = 8N$$



FFD-PACKING

$$FFD(L) = 9N$$

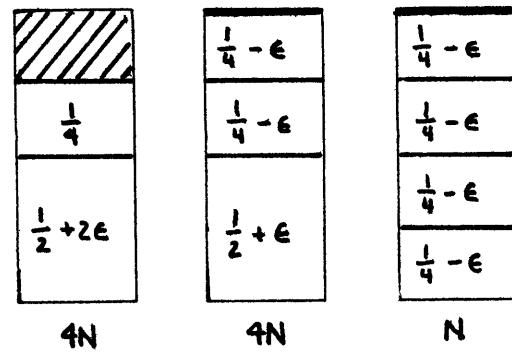
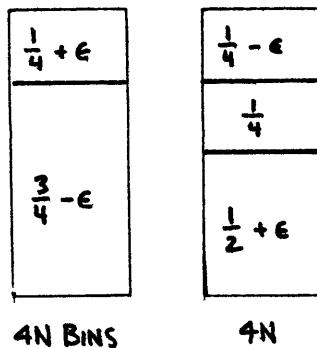


FIGURE 3.8. Lists L with $FFD(L)/WFD(L) = 9/8$.

FFD-PACKING

$$FFD(L) = 8N$$



WFD-PACKING

$$WFD(L) = 9N$$

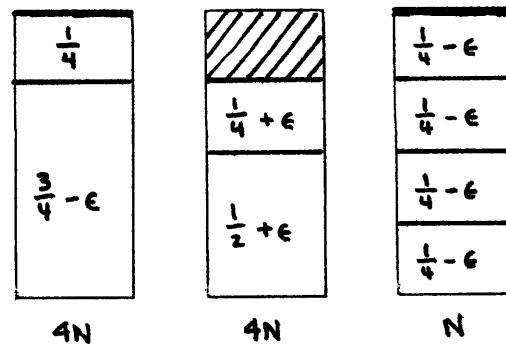


FIGURE 3.9. Lists L with $FFD(L)/WFD(L) = 8/9$.

LEMMA 3.13. Suppose L is in decreasing order and $\text{Range}(\text{size}_L) \subseteq (1/3, 1]$. Then FF and BF yield identical packings.

Proof. Under the hypothesis, L is made up entirely of A- and B-pieces, as defined in Section 3.3. By Lemma 3.11, all the A-pieces of L go in the same positions under both FF and BF. Suppose all the pieces in L with rank \leq rank (b) go in the same positions under both rules, and let P be the mutual packing of these pieces. See Figure 3.10. Let BIN_j be the bin that the FF

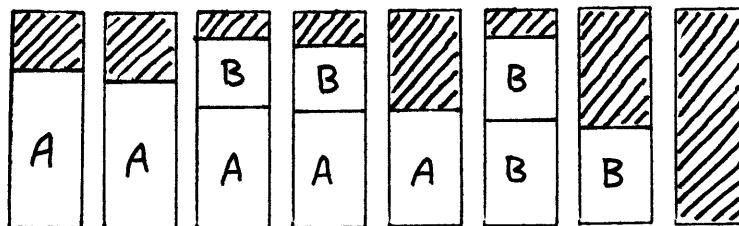


FIGURE 3.10. Mutual Packing P

rule would choose for b . If BIN_j is empty in P , then b could not have fit in any bin to the left, and so BF will have to place b in BIN_j also. If BIN_j contains one piece in P , then all non-empty $\text{BIN}_{j'}$ with $j' > j$ either contain two pieces and hence have level exceeding $2/3$ and no room for b , or by Corollary 2.2.1 contain one piece of size no larger than that of $\text{piece}_P(j, 1)$, hence have gap at least as large as BIN_j . Thus since b will not fit in any bin to the left of BIN_j , BIN_j is again the BF as well as the FF. Since BIN_j itself cannot contain two pieces and still have room for b , this exhausts all possibilities, so b will be assigned to the same bin by both BF and FF. The Lemma follows by induction. \square

We shall first show how, with the aid of Lemma 3.13, to adapt our by now standard method to the proof of Graham's result, by giving the proof of the half of the result that cannot be improved upon, and follow with our improvement on the other half.

THEOREM 3.14. If L is a list in decreasing order with $\text{Range}(\text{size}_L) \subseteq [1/5, 1]$, then $\text{FF}(L) \leq \text{BF}(L)$.

Proof. Let our base packing BP be the BF-packing of L, so that $\#BP = BF(L)$, and define $CAP(j,h) = \text{size}(\text{piece}_{BP}(j,h))$ for $(j,h) \in \text{Domain}(\text{piece}_{BP})$, 0 otherwise. By Lemma 3.6 we must show that property (P3.5) holds for all P_i , $1 \leq i \leq n = |\text{PIECES}(L)|$, where the P_i are the packings involved in the generation of the FF-packing.

Let $i_1 = \min\{\text{rank}_L(x) : \text{size}(x) \leq 1/3\}$ (If there are no such pieces the Theorem is immediate via Lemma 3.13). By Lemma 3.13, (P3.5) will hold for all P_i , $1 \leq i \leq i_1$.

CLAIM 3.14.1. If $(j,h), (j',h') \in \text{EMPTYPOS}(P_{i_1})$, $(j,h) \in \text{NONTOP}(BP)$, $(j',h') \in \text{Domain}(\text{piece}_{BP})$, and $(j,h) \leq (j',h')$, then $CAP(j,h) \geq CAP(j',h')$.

Proof of Claim. If $j = j'$ the result follows from packing property (3). Suppose $j < j'$ and the Claim fails. Let $b = \text{piece}_{BP}(j,h)$, $b' = \text{piece}_{BP}(j',h')$. Then we have by definition of CAP that $\text{size}(b') > \text{size}(b)$ and hence $\text{rank}(b') < \text{rank}(b)$. And note that neither piece is an A- or B-piece since their positions are empty in the packing P_{i_1} of all A- and B-pieces.

Now let $b_1 = \text{piece}_{BP}(j',1)$ and $c = \text{piece}_{BP}(j,h+1)$. We know that the latter exists because $(j,h) \in \text{NONTOP}(BP)$. At this point we are not sure whether the former is distinct from b' , but by packing property (3) and our hypothesis we know $\text{rank}(b_1)$

$\leq \text{rank}(b') < \text{rank}(b) < \text{rank}(c)$. Thus by Lemma 1.2, $\text{size}(b_1) > \text{size}(b) + \text{size}(c) \geq 2/5$, and b_1 is a B- or A-piece, and hence distinct from b' .

Since $j < j'$, this means that BIN_j is an A-bin or a B-bin with two B-pieces. It cannot be the latter since b and c are both not A- or B-pieces, and have total size at least $2/5$ which is too much for the gap of less than $1/3$ above two B-pieces. So BIN_j is an A-bin. Let a be its A-piece. Since no A-bin can contain more than two additional pieces of size $\geq 1/5$, we thus

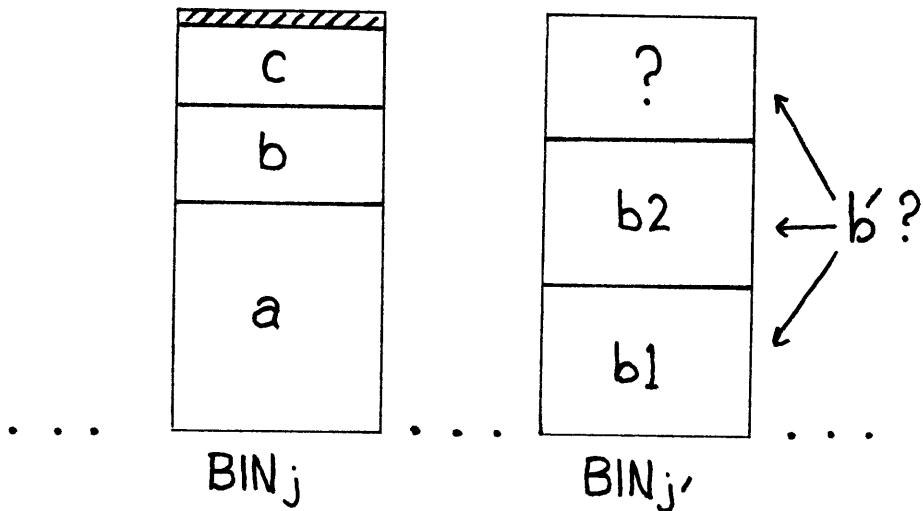


FIGURE 3.11

have $h = 2$. See Figure 3.11. Moreover, since both b and c do fit, we must have $\text{size}(a) \leq 3/5$.

Consider piece $\text{BP}(j', 2) = b_2$, which again may or may not be

b' , but must have $\text{rank}(b_2) \leq \text{rank}(b')$. When b_2 was assigned by BF, neither b nor c had been assigned, and so $\text{gap}_P(j) = 1 - \text{size}(a) \leq 1 - \text{size}(b_1) = \text{gap}_P(j')$. Thus the only way $\text{BIN}_{j'}$ could have been chosen by BF was if b_2 did not fit in $\text{BIN}_{j'}$, and so $\text{size}(b_2) > 1 - \text{size}(a) > 2/5$.

But then $\text{size}(b_1) + \text{size}(b_2) > 4/5$ and there is no room left in $\text{BIN}_{j'}$ for b' , a contradiction. So the Claim did not fail. \square

We can now proceed exactly as in the lower bound proof for Theorem 3.9, with Claim 3.14.1 substituting for (P3.9), and show that (P3.5) holds for $i_1 \leq i \leq n$. Theorem 3.14 follows via Lemma 3.6. \square

THEOREM 3.15. If L is a list in decreasing order with $\text{Range}(\text{size}_L) \subseteq [1/6, 1]$, then $\text{BF}(L) \leq \text{FF}(L)$.

Proof. Let BP be the FF-packing of L , so that $\#BP = \text{FF}(L)$, and define CAP in the usual way. If there are no pieces in L other than A- or B-pieces we are done by Lemma 3.13, so assume there are such pieces and let $i_1 = \text{MIN}\{\text{rank}_L(x) : \text{size}(x) \leq 1/3\}$. Let P_1, \dots, P_{n+1} be the sequence of packings involved in the generation of the BF-packing of L , where $n = |\text{PIECE}(L)|$. By Lemma 3.13, property (P3.5) holds for all P_i , $1 \leq i \leq i_1$.

CLAIM 3.15.1. If $(j, h), (j', h') \in \text{EMPTYPOS}(P_{i1})$, $(j, h) \text{ NONTOP}(BP)$, $(j', h') \in \text{Domain}(\text{piece}_{BP})$, and $(j, h) \leq (j', h')$, then $\text{CAP}(j, h) \geq \text{CAP}(j', h')$.

Proof of Claim 3.15.1. If $j = j'$ the Claim follows from packing property (3). Suppose $j < j'$ and the Claim fails. Letting $b = \text{piece}_{BP}(j, h)$ and $b' = \text{piece}_{BP}(j', h')$, we then have $\text{size}(b') > \text{size}(b)$ and hence $\text{rank}(b') < \text{rank}(b)$. Let $c = \text{piece}_{BP}(j, h+1)$ (we know such a piece exists because $(j, h) \text{ NONTOP}(BP)$). See Figure 3.12.

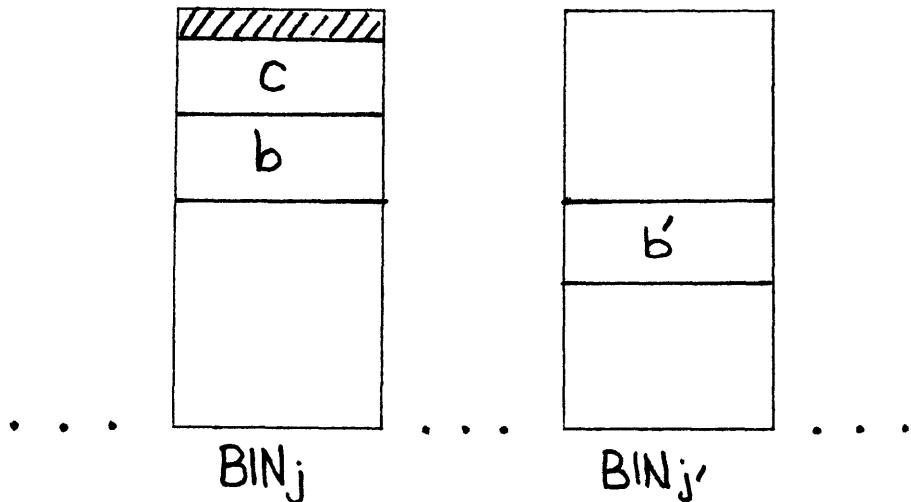


FIGURE 3.12

When b' was assigned we must have had $\text{gap}_{P_i}(j) \geq \text{size}(b) + \text{size}(c) \geq 2/6 = 1/3$, so the fact that b' went to the right to $\text{BIN}_{j'}$ under FF means that $\text{size}(b') > 1/3$. But this means b' is an A- or B-piece, which is impossible since its position is empty in P_{i1} , the packing of all A- and B-pieces. So the Claim in fact could not have failed. \square

We now can proceed with the induction on property (P3.5). Suppose (P3.5) holds for some i , $i1 \leq i < n$. Then as usual by Lemmas 3.4 and 3.5, there is an a.f. f_i from L'_i to P_i obeying (P3.4), and the only way that (P3.5) can be violated for P_{i+1} is if the piece b with rank 1 in L'_i goes into a NONTOP(BP) position (j_1, h_1) in P_{i+1} with $\text{CAP}(j_1, h_1) < \text{size}(b)$. Let $(j', h') = f_i(b)$. There are two cases:

Case 1. $(j_1, h_1) \leq (j', h')$. Then since both positions are unfilled in P_i and hence P_{i1} , Claim 3.15.1 applies and by (P3.4) for i we have $\text{size}(b) \leq \text{CAP}(j', h') \leq \text{CAP}(j_1, h_1)$, so (P3.5) is not violated.

Case 2. $(j_1, h_1) > (j', h')$. Then by our standard argument we must have $j_1 > j'$, and by a.f. property (3), b would have fit in $\text{BIN}_{j'}$. The fact that it went to the right to BIN_{j_1} under BF means that $\text{gap}_{P_i}(j_1) < \text{gap}_{P_i}(j')$.

We cannot have $h_1 = 1$ for then we would have $\text{gap}_{P_i}(j_1) = 1$, which is the largest possible gap. We also cannot have $h_1 = 2$, since in that case $\text{gap}_{P_i}(j_1) = 1 - \text{size}(\text{piece}_{P_i}(j_1, 1)) \geq 1 - \text{size}(\text{piece}_{P_i}(j'_1, 1)) \geq \text{gap}_{P_i}(j'_1)$, by Lemma 1.2. Thus $h_1 \geq 3$, and BIN_{j_1} must already contain at least two pieces in P_i . See Figure 3.13.

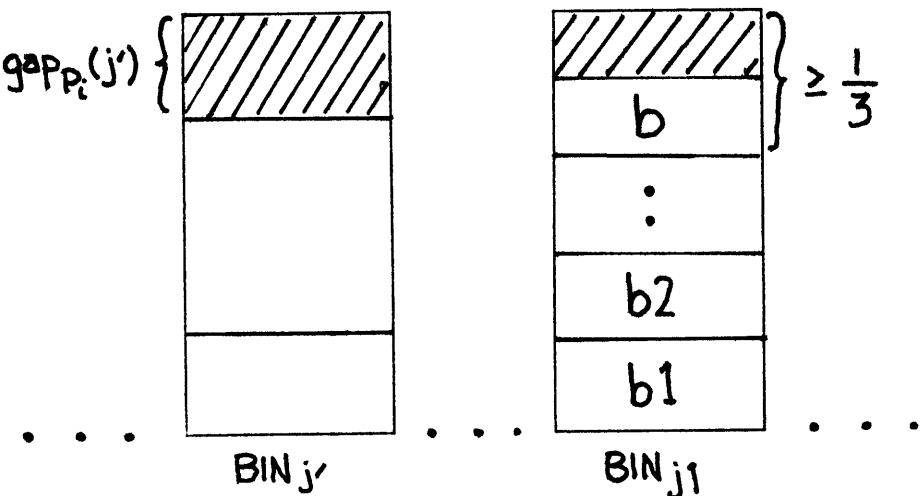


FIGURE 3.13

If $(j_1, h_1) \in \text{NONTOP}(\text{BP})$, then (j_1, h) for all $1 \leq h < h_1$ must also be in NONTOP , and so must contain in P_i pieces no larger than their capacities by (P3.5) for P_i . Thus $\text{gap}_{P_i}(j_1) \geq$ the sum of the capacities of the unfilled positions, by capacity map property (3). Since $(j_1, h_1) \in \text{NONTOP}$ and is unfilled in P_i , there must be at least two such. Thus $\text{gap}_{P_i}(j_1) \geq 1/3$.

Furthermore, if the bottom two pieces in BIN_{j1} were A- or B-pieces, the bin could not have so large a gap. Thus $b_2 = \text{piece}_{p_i}(j_1, 2)$ must have $\text{size}(b_2) \leq 1/3$. Since b_2 was placed by the FF rule in BIN_{j1} to the right of BIN_{j1} , this means we must have $\text{gap}_{p_i}(j') < 1/3$.

But now we have a contradiction, for we have

$$\text{gap}_{p_i}(j') < 1/3 \leq \text{gap}_{p_i}(j_1),$$

and so b , which would have fit in $\text{BIN}_{j'}$, would violate the BF rule if it were assigned to BIN_{j1} . So this case is impossible if $(j_1, h_1) \notin \text{NONTOP}(\text{BP})$, no matter what the size of b .

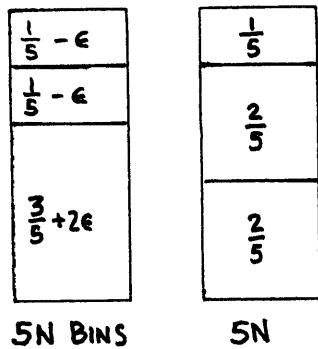
Hence (P3.5) holds for P_{i+1} , and by induction for all i , $i_1 \leq i \leq n$. Since we already knew it held for $1 \leq i \leq i_1$, the theorem follows via Lemma 3.6. \square

Theorems 3.14 and 3.15 are the best results possible as shown by the examples given in Figures 3.14 and 3.15 (from [Ga1]), the first giving lists L with $\text{Range}(\text{size}_L) \subseteq [1/5 - \epsilon, 1]$ and $\text{FFD}(L) = (11/10) \text{BFD}(L)$, the second giving lists L with $\text{Range}(\text{size}_L) \subseteq [1/6 - \epsilon, 1]$ and $\text{BFD}(L) = (10/9) \cdot \text{FFD}(L)$. We shall have an important application of Theorem 3.15 in Chapter 5.

SECTION 3.4 - Page 160

BFD-PACKING

$$BFD(L) = 10N$$



FFD-PACKING

$$FFD(L) = 11N$$

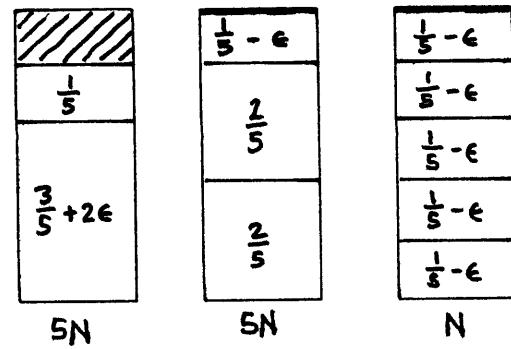
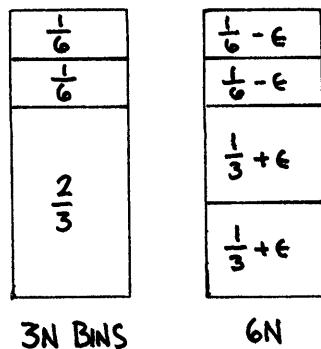


FIGURE 3.14. Lists L for which $FFD(L)/BFD(L) = 11/10$.

FFD-PACKING

$$FFD(L) = 9N$$



BFD-PACKING

$$BFD(L) = 10N$$

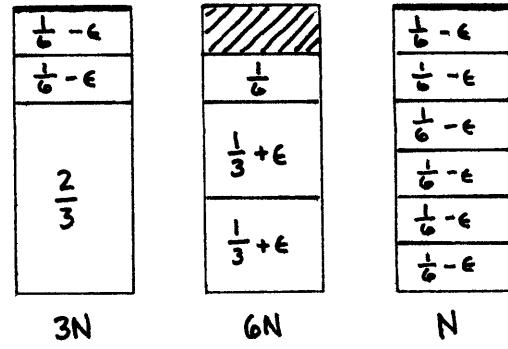


FIGURE 3.15. Lists L for which $BFD(L)/FFD(L) = 10/9$.

One minor difference between our results about FF and BF in this section and those about FF and an arbitrary $S \in AF$ in the last is the type of interval to which piece sizes are restricted: half open for the latter, and closed for the former - $(1/4, 1]$ vs $[1/5, 1]$ for instance. As a matter of fact, one may replace the "(" by a "[" in both Theorems 3.9 and 3.12. However the details are slightly messy and the extension unnecessary for our later applications, so we settled for the weaker result.

CHAPTER 4. WORST CASE BEHAVIOR OF AFD ALGORITHMS

SECTION 4.1 Lower Bounds

The previous chapter was devoted to comparisons between the packings that various AF algorithms might generate when applied to a given decreasing list. In this and the next chapter we return to our main type of analysis, comparing the packings the algorithms generate with optimal packings. Our goal is to find the values of $R[SD, t]$ for $0 < t \leq 1$ and $S \in AF$, as we have already done in Chapter 1 for $R[S, t]$.

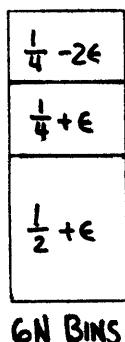
For all $t \in (0,1]$, the best lower bounds on worst case behavior known are given by examples which yield the same packings under all $S \in AF$. These examples are much less involved than the ones we encountered in the on-line situation, and can be presented without much further explanation.

Figure 4.1 shows decreasing lists L with arbitrarily large L^* such that if $S \in AF$, $S(L)/L^* = 11/9$. This example originally appeared in [Gal]. Note that the value is considerably lower than the $17/10$ (or worse for $S \notin AF$ - AAF) encountered in the on-line case.

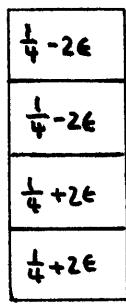
When we restrict ourselves to decreasing lists with $\text{Range}(\text{size}_L) \subseteq (0, t]$ for $0 < t \leq 1/2$, we can exhibit lists for which

OPTIMAL PACKING

$$L^* = 9N$$



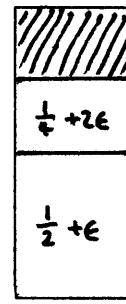
6N BINS



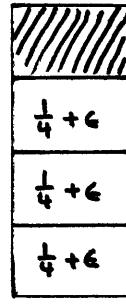
3N

SD-PACKING

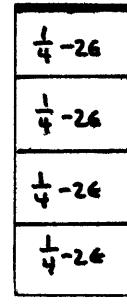
$$SD(L) = 11N$$



6N



2N



3N

FIGURE 4,1. Lists L for which $SD(L)/L^* = 11/9$, for all $\epsilon \in AF$.

$$\frac{S(L)}{L^*} = \frac{m+2m-1}{m(m+1)} = \frac{m+3}{m+2} - \frac{2}{m(m+1)(m+2)}$$

where $m = \lfloor 1/t \rfloor$, again a distinct improvement over the on-line case. See Figure 4.2.

For $m = 2, 3, 4, 5, 6$, these numbers are $7/6, 7/6, 23/20, 34/30$, and $47/42$. We had originally conjectured that these were the actual values of $R[SD, t]$, until in the course of trying to prove this bound for $t = 1/2$, we found the example shown in Figure 4.3, for which, although no piece exceeds even $1/3$ in size, $SD(L)/L^* = 71/60 > 7/6$.

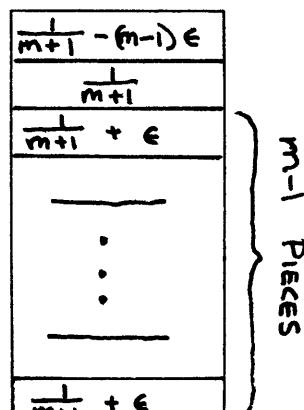
We can combine the information given in these figures into the following:

THEOREM 4.1: If $S \in AF$, then

$$\begin{aligned} R[SD, t] &\geq 11/9, \text{ for } t > 1/2, \\ &\geq 71/60, \text{ for } 8/29 < t \leq 1/2, \\ &\geq 7/6, \text{ for } 1/4 < t \leq 8/29, \\ &\geq \frac{m+3}{m+2} - \frac{2}{m(m+1)(m+2)}, \text{ for } m = \lfloor 1/t \rfloor \geq 4. \end{aligned}$$

OPTIMAL PACKING

$$L^* = Nm(m+1)$$



$Nm(m+1)$ BINS

SD - PACKING

$$SD(L) = m^2 + 2m - 1$$

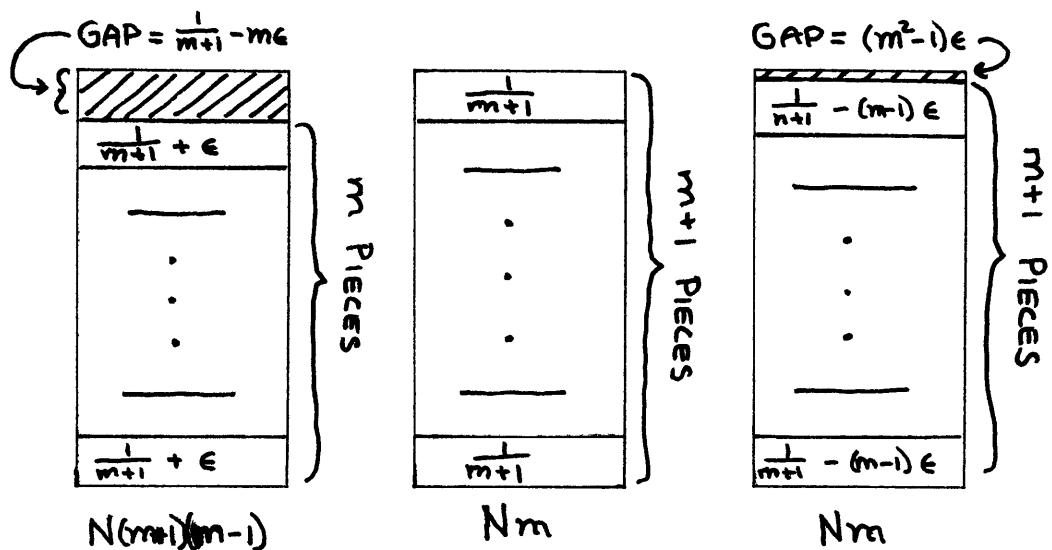


FIGURE 4.2. Lists L with $\text{Range}(\text{size}_L) \subseteq (0, t]$, $m = \lfloor 1/t \rfloor$,

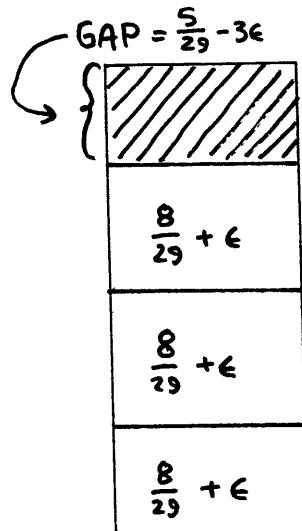
$$\text{for which } \frac{SD(L)}{L^*} = \frac{m^2 + 2m - 1}{m(m+1)}$$

OPTIMAL PACKING

$$L^* = 60N$$

$\frac{5}{29} - \epsilon$
$\frac{5}{29} - \epsilon$
$\frac{5}{29} - \epsilon$
$\frac{6}{29} + 2\epsilon$
$\frac{8}{29} + \epsilon$

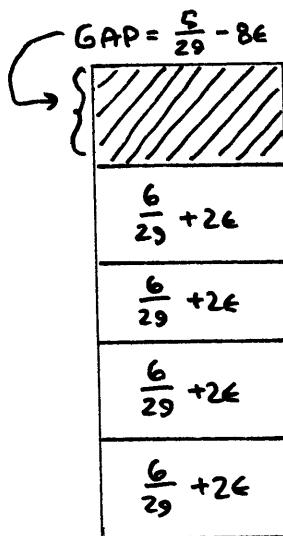
60N



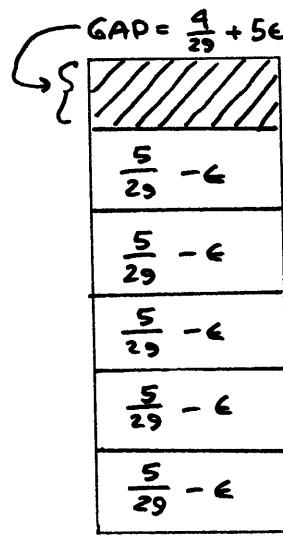
20N

SD - PACKING

$$SD(L) = 71N$$



15N



36N

FIGURE 4.3. Lists L with $\text{Range}(\text{size}_L) \subseteq (0, \dots, \frac{8}{29} + \epsilon]$,
for which $SD(L)/L^* = 71/60$,
for all $s \in AF$.

SECTION 4.2. Weak Upper Bounds for all $S \in AF$

We now turn to the problem of obtaining upper bounds on the worst case ratios. We conjecture that the the upper bounds are in fact the lower bounds given above, and that, unlike the on-line case, the results hold for all $S \in AF$, not just $S \in AAF$. This is in part suggested by our results from Chapter 3, which showed how use of the DECREASING rule to pre-order the list has an equalizing effect on the AF algorithms. Unfortunately, we have only been able to prove that the upper bound equals the lower for certain particular algorithms, such as FIRST FIT, and for certain ranges of t . These proofs will be presented later on in Chapter 5. In the current chapter we will prove that, for all t , $0 < t \leq 1$, upper bounds which are very close to the above lower bounds do hold for all AF algorithms.

First we shall prove some very useful Lemmas about packings of decreasing lists by AF algorithms, which will have applications in both this and the next chapter. The first originally appeared in [Ga1]:

LEMMA 4.2. Suppose $S \in AF$, $1 \leq r \leq 2$, and $K \geq 1$. Then if there exists a decreasing list L with $\text{Range}(\text{size}_L) \subseteq (0, t]$ and $S(L) > rL^* + K$, there exists a decreasing list L' with $S(L') > rL^* + K$ and $\text{Range}(\text{size}_{L'}) \subseteq ((r-1)/r, t]$, and moreover, such a list can be obtained from L by deleting all pieces of size $\leq (r-1)/r$.

Proof. Divide L into two sublists $L = L_1 \bullet L_2$, where $\text{Range}(\text{size}_{L_1}) \subseteq ((r-1)/r, t]$ and $\text{Range}(\text{size}_{L_2}) \subseteq (0, (r-1)/r]$. Let PS be an S -packing of L using $S(L)$ bins, and divide it into two segments, PS_1 the set of the first $S(L)-1$ bins, and PS_2 , $\text{BIN}_{S(L)}$. See Figure 4.4. If the bottom piece in $\text{BIN}_{S(L)}$ is from L_2 , then $\text{size}(\text{piece}_{PS}(S(L), 1)) \leq (r-1)/r$, and so by Lemma 2.1, for all $j < S(L)$, $\text{level}_{PS}(j) > 1 - [(r-1)/r] = 1/r$. Thus $L^* \geq W(L) > w(PS_1) > (1/r)(S(L)-1)$ and so $S(L) < rL^* + 1$, contrary to hypothesis. Thus the bottom piece in $\text{BIN}_{S(L)}$ is from L_1 , so $S(L_1) = S(L)$ and $L' = L_1$ is our desired list. \square

LEMMA 4.3. Suppose $S \in AF$, and $L = L_1 \bullet L_2$ is a decreasing list with $\text{Range}(\text{size}_{L_2}) \subseteq (0, 1/n]$ for $n > 1$. Then $S(L_1) \leq L^* + k \implies S(L) \leq [(n+1)/n]L^* + (k+1)$.

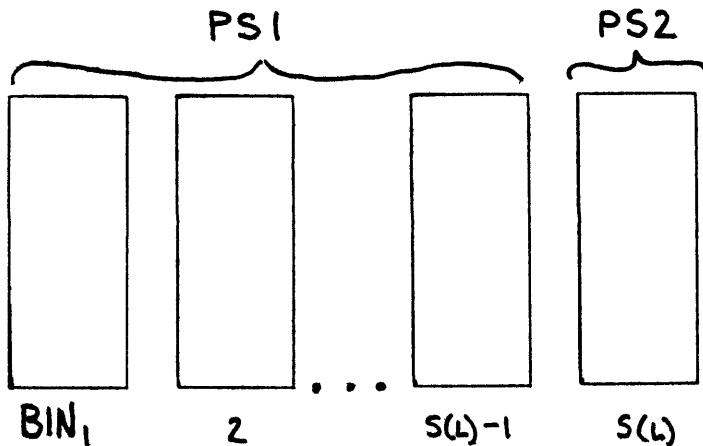


FIGURE 4.2. Diagram for Lemma 4.2.

Proof. Let PS be an S -packing of L using $S(L)$ bins.

Divide it into three segments as shown in Figure 4.5. PS1 is the set those BIN_j with $j \leq \min\{L^*, S(L)-1\}$, PS2 those BIN_j , $L^* < j \leq \min\{L^*+k, S(L)-1\}$, PS3 is those BIN_j with $L^*+k < j \leq S(L)-1$, and PS4 is $\text{BIN}_{S(L)}$.

If PS3 is vacuous we are done, for we would have $S(L) = \#\text{PS1} + \#\text{PS2} + \#\text{PS4} = L^* + k + 1$. So we may assume PS3 , and hence PS2 , is not vacuous, and hence has $\#\text{PS3} = S(L) - L^* - k - 1$. Let $d = \max\{\text{gapp}_S(j) : \text{BIN}_j \text{ in } \text{PS1}\}$. Thus $w(\text{PS1}) \geq (1-d)(\#\text{PS1}) = (1-d)L^*$. Since $S(L1) \leq L^* + k$, no bottom piece in any bin in PS3 can be from L1 . Thus by Lemma 3.6, each bin in PS3 must contain at least n pieces of size $> d$, and so $w(\text{PS3}) >$

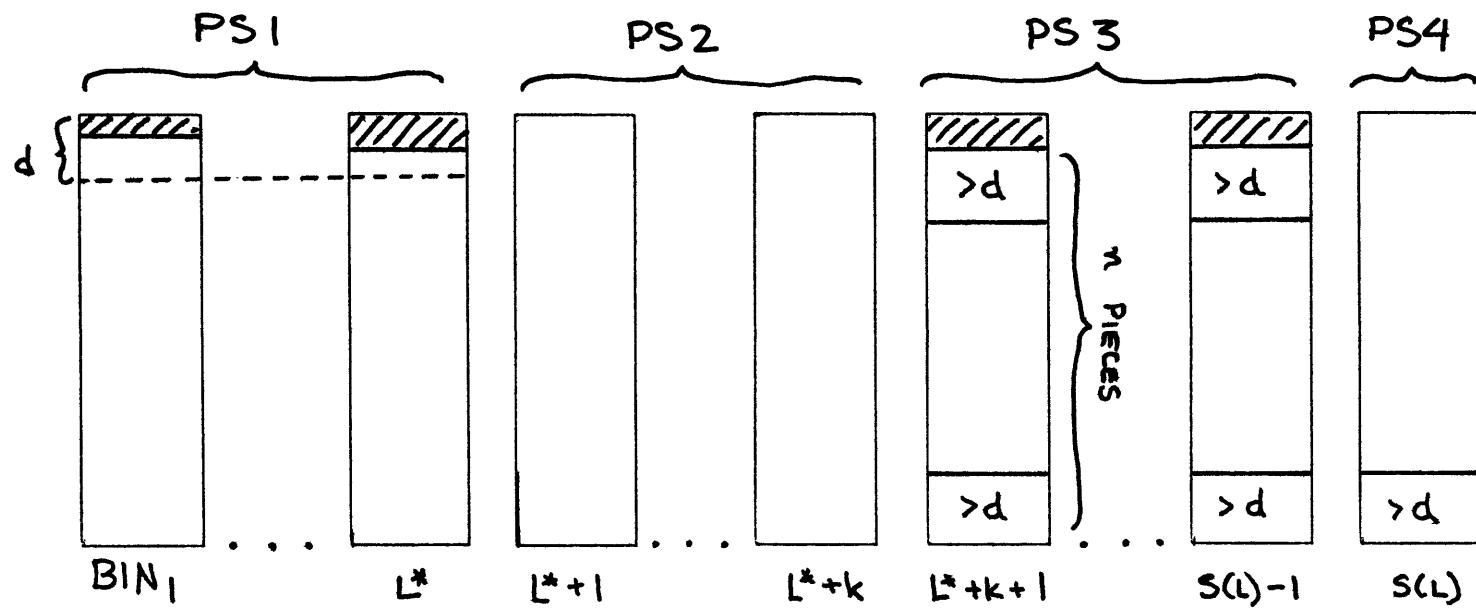


FIGURE 4.5. Diagram for Lemma 4.3.

$(nd)(\#PS3) = (nd)(S(L)-L*-k-1)$. We thus conclude

$$\begin{aligned} L^* &\geq W(L) \geq w(PS1) + w(PS2) \\ &\geq (1-d)L^* + (nd)(S(L)-L*-k-1) \\ &\geq L^* - (n+1)(d)L^* + (nd)(S(L)-k-1) \end{aligned}$$

and so $S(L) \leq [(n+1)/n]L^* + (k+1)$. \square

Lemmas 4.2 and 4.3 allow us to restrict our attention to lists all of whose pieces are bigger than a given constant - just the type of lists to which our results of Chapter 3 apply. We will thus be able to use those results to show that an upper bound on the worst case behavior of one algorithm extends to an upper bound for another. Thus, in fact, all the major work in this section will concern the FF algorithm, with the results being extended to arbitrary $S \in AF$ via Theorems 3.9 and 3.12.

The next two lemmas are about FF and are of technical importance in both this and the next chapter. First recall our definitions of A- and B-pieces and A- and B-bins in Section 3.3. An A-piece has size in the range $(1/2, 1]$, and a B-piece in $(1/3, 1/2]$, and an A-bin is a bin whose bottom piece is an A-piece, similarly for B-bin. We then have:

LEMMA 4.4. Suppose L is a list in decreasing order and PF is the FF-packing of L . If piece $PF(j,1)$ is an A-piece a , piece b is in a bin to the right of BIN_j in PF , and $\text{size}(a) + \text{size}(b) \leq 1$, then there is a $b' = \text{piece}_{PF}(j,2)$ with $\text{rank}(b') < \text{rank}(b)$ and hence $\text{size}(b') \geq \text{size}(b)$.

Proof. When b was assigned by FF, it went to the right of BIN_j . Since it would have fit in BIN_j had that bin contained only piece a at the time, the FF rule would have been violated unless position $(j,2)$ were already filled by some piece $b' = \text{piece}_{PF}(j,2)$. But then we must have $\text{rank}(b') < \text{rank}(b)$. \square

LEMMA 4.5. Suppose L is a list in decreasing order with $\text{Range}(\text{size}_L) \subseteq (1/3, 1]$. Then $\text{FF}(L) = L^*$.

Proof. Let P^* be an optimal packing of L , and PF the FF-packing. Note that the only pieces in L are A- and B-pieces. Let

$$YF = \{b \in B\text{-PIECES}(L) : b \text{ is in an A-bin in } PF\},$$

$$Y^* = \{b \in B\text{-PIECES}(L) : b \text{ is in an A-bin in } P^*\},$$

and label the elements of Y^* b_1, b_2, \dots, b_k in order of increasing rank. For each b_j let $\text{BIN}_{I(j)}$ be the A-bin containing it in P^* , and a_j the A-piece in $\text{BIN}_{I(j)}$. Since no A-bin can contain more than one B-piece, the $I(j)$, $1 \leq j \leq k$, are all distinct.

Now define sets $B(j)$ as follows: $B(0) = \emptyset$, and for $1 \leq j \leq k$,

$$B(j) = \{b \in YF : \text{rank}(b) \leq \text{rank}(b_j)\}.$$

Note that for $0 \leq j < k$, $B(j) \subseteq B(j+1)$. We shall prove the Lemma using an induction on the following hypothesis:

$$(H) |B(j)| \geq j.$$

The hypothesis holds for $j = 0$, since $|B(0)| = 0$ by definition. Suppose it holds for $j-1$. Now since b_j is no larger than any of the first $j-1$ pieces in Y^* , $\text{size}(b_j) + \text{size}(a_i) \leq 1$ for $1 \leq i \leq j$. Thus if $b_j \notin YF$ and hence in a B-bin to the right of all A-bins in PF , by Lemma 4.4, $\text{piece}_{PF}(I(i), 2)$ is a B-piece with $\text{rank} < \text{rank}(b_j)$ for all i , $1 \leq i \leq j$, and hence $|B(j)| \geq j$. On the other hand, if $b_j \in YF$, then b_j is a B-piece with $\text{rank} \leq \text{rank}(b_j)$ which is in $B(j)$ but not in $B(j-1)$, so $|B(j)| \geq |B(j-1)| + 1 \geq j$.

Thus by induction (H) holds for $j = k$, and so $|YF| \geq |Y^*|$, and at least as many of the B-pieces are in A-bins in PF as are in A-bins in P^* . Thus there are no more B-pieces in B-bins in PF than there are in P^* . Since under FF these are packed 2 to a bin by Lemma 3.7, and this is the most efficient way possible, we thus conclude that PF has no more B-bins than P^* . And since both packings have the same number of A-bins, we can only conclude that $FF(L) = \#PF \leq \#P^* = L^*$. The lemma follows. \blacksquare

We are now prepared to prove the major result of this chapter, that if $S \notin AF$, $R[SD] \leq 5/4$, an upper bound only slightly bigger than the lower bound of $11/9$ given in Theorem 4.1. The following two lemmas use Lemmas 4.2 and 4.3 and Theorem 3.9 to reduce the problem to one about FF:

LEMMA 4.6. Suppose $L = L_1 \bullet L_2 \bullet L_3$ is a list in decreasing order with $\text{Range}(\text{size}_{L_1}) \subseteq (1/4, 1]$, $\text{Range}(\text{size}_{L_2}) \subseteq (1/5, 1/4]$, and $\text{Range}(\text{size}_{L_3}) \subseteq (0, 1/5]$, and that $S \notin AF$. If $FF(L_1) \leq (L_1 \bullet L_2)^*$, then $S(L) \leq (5/4)L^* + 2$.

Proof. By Theorem 3.9, $S(L_1) \leq FF(L_1) + 1 \leq (L_1 \bullet L_2)^* + 1$. Thus by Lemma 4.3, since $\text{Range}(\text{size}_{L_2 \bullet L_3}) \subseteq (0, 1/4]$, $S(L) \leq (5/4)L^* + 2$. \blacksquare

LEMMA 4.7. Suppose L and S are as above. If $\text{FF}(L_1) + \text{FF}(L_2) \leq (5/4)L^* + 2$, then $S(L) \leq (5/4)L^* + 3$.

Proof. By Lemma 4.2 it is sufficient to show that $S(L_1 \cup L_2) \leq (5/4)L^* + 3$. Let PS be an S -packing of $L_1 \cup L_2$ using $S(L_1 \cup L_2)$ bins, and divide it into segments PS_1 and PS_2 as shown in Figure 4.6. PS_1 is made up of the bins containing pieces from L_1 , and

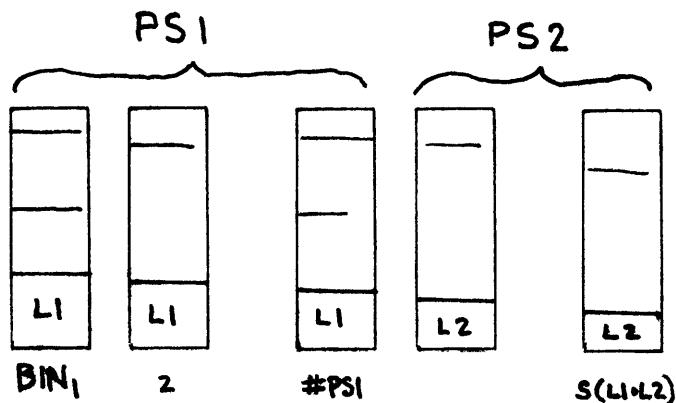


FIGURE 4.6. S-PACKING PS of L.

PS2 the remaining non-empty bins.

By Theorem 3.9, $\#PS_1 \leq \text{FF}(L_1) + 1$. By Lemma 3.7, each of the bins of PS_2 except possibly the last must contain 4 pieces from L_2 . Since no packing of pieces with size $\in (1/5, 1/4]$ can be more efficient than that, we must have $\#PS_2 \leq \text{FF}(L_2)$. Thus

$$\begin{aligned} S(L_1 \bullet L_2) &= \#PS_1 + \#PS_2 \leq FF(L_1) + FF(L_2) + 1 \\ &\leq (5/4)L^* + 3. \end{aligned}$$

□

In light of Lemmas 4.6 and 4.7, the desired upper bound will follow from the next Theorem:

THEOREM 4.8. Suppose $L = L_1 \bullet L_2$ is a list in decreasing order with $\text{Range}(\text{size}_{L_1}) \subseteq (1/4, 1]$ and $\text{Range}(\text{size}_{L_2}) \subseteq (1/5, 1/4]$. Then

$$FF(L_1) > L^* \implies FF(L_1) + FF(L_2) \leq (5/4)L^* + 2.$$

Proof. The basic strategy of this proof is to show that if $FF(L_1) > L^*$, then any optimal packing of L is too crowded with pieces from L_1 for there to be very many pieces from L_2 around. So assume $FF(L_1) > L^*$, and let $L = L_A \bullet L_B \bullet L_C \bullet L_D$, where

$$\begin{aligned} \text{Range}(\text{size}_{L_A}) &\subseteq (1/2, 1], \quad \text{PIECES}(L_A) = A\text{-PIECES}(L), \\ \text{Range}(\text{size}_{L_B}) &\subseteq (1/3, 1/2], \quad \text{PIECES}(L_B) = B\text{-PIECES}(L), \\ \text{Range}(\text{size}_{L_C}) &\subseteq (1/4, 1/3], \quad \text{PIECES}(L_C) = C\text{-PIECES}(L), \\ \text{Range}(\text{size}_{L_D}) &\subseteq (1/5, 1/4], \quad \text{PIECES}(L_D) = D\text{-PIECES}(L). \end{aligned}$$

Thus we have $L_1 = L_A \cdot L_B \cdot L_C$, and $L_2 = L_D$. A piece from $X\text{-PIECES}(L)$ will be called an X -piece, and a bin in a packing whose bottom piece is an X -piece will be called an X -bin, for each $X \in \{A, B, C, D\}$. Since each A -bin can contain at most one A -piece a , we can identify it with that A -piece and call it the a -bin. This will allow us to compare the contents of particular a -bins between two different packings involving L_A . A non-empty bin other than an A -bin will sometimes be referred to as a non- A -bin.

Now let P be the FF-packing of L_1 , and choose Q_{MAX} and Q so that $Q_{MAX} < \#P$, and Q is an ordered packing of L_1 with $\text{MAX}\{j : \text{level}_Q(j) > 0\} \leq Q_{MAX}$, and such that if in P BIN_j is the a -bin for $a \in A\text{-PIECES}$, then BIN_j is also the a -bin in Q . There must be such packings since by assumption $L_1^* \leq L^* < \text{FF}(L)$, and any packing can have its bins rearranged so that its A -bins appear as required. For instance (and this trick will be applied later in the proof), we could get such a Q by taking an optimal packing of L and removing all the D -pieces.

Now divide P into segments PA , PB , and PX , and Q into segments QA and QB as follows (see Figure 4.7):

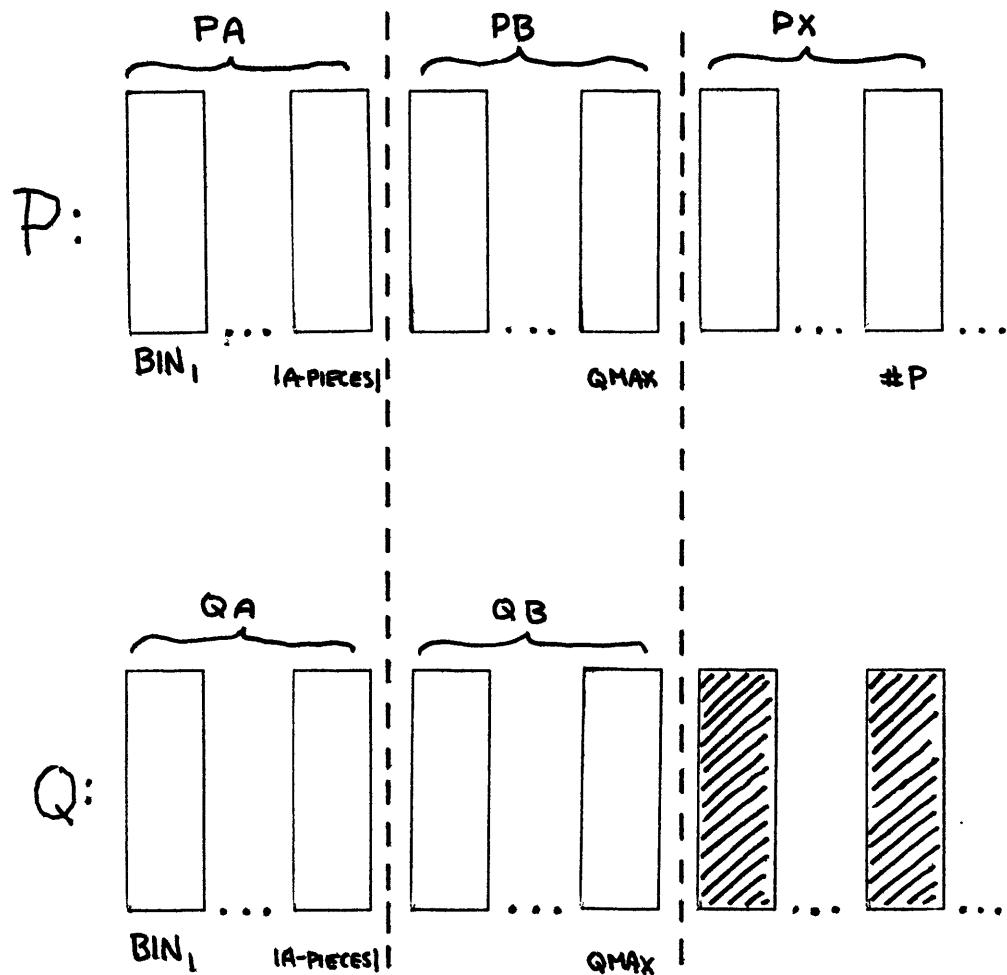


FIGURE 4.7. Packings P and Q of L1.

$$PA = A\text{-bins of } P = \{BIN_j : 1 \leq j \leq |A\text{-PIECES}|\},$$

$$PB = \{BIN_j : |A\text{-PIECES}| < j \leq QMAX\},$$

$$PX = \{BIN_j : \text{cont}_P(j) \neq \emptyset \text{ and } j > QMAX\},$$

$$QA = A\text{-bins of } Q = \{BIN_j : 1 \leq j \leq |A\text{-PIECES}|\},$$

$$QB = \{BIN_j : |A\text{-PIECES}| < j \leq QMAX\}.$$

Note that some of the bins in QB may be empty, since we can have $QMAX > \#Q$; but since $\#P > QMAX$, segment PX is not vacuous. Moreover, PX cannot contain any B-pieces, since if it did we would have $FF(LA \cdot LB) > QMAX \geq L^* \geq (LA \cdot LB)^*$, in violation of Lemma 4.5. Thus we have

CLAIM 4.8.1. $\emptyset \neq \text{cont}(PX) \subseteq C\text{-PIECES}$.

Now let $B = B\text{-PIECES}(L) \cup C\text{-PIECES}(L)$, and define the following subsets of B:

$$Q = B - \text{cont}(QA),$$

$$P = B - \text{cont}(PA),$$

$$\alpha = \{x \in B \cap \text{cont}(PA) : \text{the } A\text{-bin containing } x \text{ in } P \text{ has an } A\text{-piece with size } < 3/5, \text{ and contains no element of } B \text{ in } Q\}.$$

Letting $e = \text{piece}_P(Q\text{MAX}+1, 1)$, we have $\text{size}(e) \leq 1/3 < 2/5$ by Claim 4.8.1. Thus e would have fit as the second piece in any-A-bin whose A-piece had size $< 3/5$. Consequently, by Lemma 4.4, each such A-bin must contain a β -piece with $\text{rank} < \text{rank}(e)$.

There can be no more than one by size constraints, so we have

CLAIM 4.8.2. If $a \in A\text{-PIECES}$, $\text{size}(a) < 3/5$, and the a-bin contains no β -pieces in Q , then it contains exactly one element of α in P .

We wish to show that the crowding of Q , due to the fact that it uses fewer bins than B and that the A-bins which contain elements of α in P contain no β -pieces in Q , is all concentrated in segment QB . To this end we shall define a 1-1 map f from $P \cup \alpha$ to Q .

The definition proceeds as follows: We say that a piece x points to a piece y if $x = \text{piece}_Q(j, 2)$ and $y = \text{piece}_P(j, 2)$ for BIN_j an A-bin. A chain of distinct pieces is a sequence $\langle x_1, \dots, x_k \rangle$ such that for $1 \leq i < k$, x_i points to x_{i+1} . A loop is a chain $\langle x_1, \dots, x_k \rangle$ in which x_k points to x_1 . A maximal chain is a chain which is not a loop and not a proper subsequence of any other chain. If $\langle x_1, \dots, x_k \rangle$ is a maximal chain, x_1 is its head, and x_k is its tail.

For each $x \in P \cup \alpha$, define

$f(x) = \text{tail of maximal chain headed by } x.$

CLAIM 4.8.3. f is a 1-1 map from $P \cup Q$ to Q satisfying

- (A) $x \in P \implies \text{size}(f(x)) \geq \text{size}(x),$
- (B) $x \in Q \implies \text{size}(f(x)) \geq \text{size}(e).$

Proof of Claim. If x is not pointed to by any piece, then x must be the head of some maximal chain, even if the chain is only $\langle x \rangle$, which would occur if x itself did not point to anything. Now if $x \in P$, x is not in an A-bin in P , so cannot be pointed to. Similarly, if $y \in Q$, $\text{piece}_Q(j, 2)$ is not defined for BIN_j the A-bin containing y in P , and so y is not pointed to. Thus f is well-defined. Since no piece can be pointed to by more than one other piece, f is clearly 1-1.

That $\text{Range}(f) \subseteq Q$ and properties (A) and (B) hold follow from a simple induction. Let $\langle x_1, \dots, x_k \rangle$ be a maximal chain headed by $x_1 \in P \cup Q$. Our induction hypothesis is

- (H) If $x_i \in P$, $\text{size}(x_i) \geq \text{size}(x_1).$
- If $x_i \in Q$, $\text{size}(x_i) \geq \text{size}(e).$

For $i = 1$, (H) holds trivially if $x_1 \in P$, and by Claim 4.8.2 if $x_1 \notin Q$. Suppose it holds for x_i and $x_i \notin Q$. Then $x_i = \text{piece}_Q(j, 2)$ for BIN_j the a-bin for some $a \in A\text{-PIECES}$, so

$$\begin{aligned} \text{size}(x_i) + \text{size}(a) &\leq \text{size}(x'_i) + \text{size}(a) \leq 1, \\ [\text{size}(e) + \text{size}(a)] &\leq \text{size}(x'_i) + \text{size}(a) \leq 1. \end{aligned}$$

Since $x_1 [e]$ is not in cont(PFA) , by lemma 4.4 there is a piece $b = \text{piece}_P(j, 2)$ with $\text{size}(b) \geq \text{size}(x_1)$ [$\text{size}(b) \geq \text{size}(e)$]. Hence x_i points to b. Since $\langle x_1, \dots, x_k \rangle$ is a maximal chain, x_k cannot point to anything, so $i < k$, $b = x_{i+1}$, and (H) holds for x_{i+1} .

Thus by induction $x_k \in Q$, and $f(x_i) = x_k$ obeys (A) [(B)], and Claim 4.8.3 is proved. \square

From this point on we shall ignore any pieces in Q -Range(f). The bins of QB will be crowded enough without them. In order to further emphasize this crowdedness, we shall use f to show that QB must contain a number of "big" pieces. For the moment let us return our attention to segment PB of P. Let

$$\begin{aligned} r &= \text{MAX}\{j: \text{BIN}_j \text{ in PB and } \text{height}_P(j) = 2\}, \\ \text{BOT} &= \{\text{piece}_P(j, h): |A\text{-PIECES}(L)| < j \leq r, h \leq 2\} \\ &\quad - \{\text{piece}_P(r, 2)\}, \end{aligned}$$

$$\text{TOP} = \{\text{piece}_P(j, 3) : |\text{A-PIECES}(L)| < j \leq r\}.$$

See Figure 4.8. Now since the sums of the sizes of the bottom two pieces in the bins of PB must form a non-increasing sequence by Lemma 3.7, each $b \in \text{TOP}$ would fit in $\text{gap}_P(r)$. Thus every piece in a bin to the right of BIN_r in P must have rank $<$ $\text{rank}(b)$ for all $b \in \text{TOP}$, and so the pieces in TOP must all be comparatively small. Now let

$$\text{BIG} = f(\text{BOT}),$$

$$\text{SMALL} = \{z \in \text{Range}(f) : \text{there exist } x \neq y \in \text{BIG} \text{ with } \text{size}(x) + \text{size}(y) + \text{size}(z) \leq 1\}.$$

These two names will take on significance in the light of the next two claims:

CLAIM 4.8.4. $\text{BIG} \subseteq \text{B-PIECES}(L)$.

Proof of Claim. If $\text{piece}(r, 1)$ were a C-piece, then by packing property (3) so would be $\text{piece}_P(r, 2)$ and we would have $\text{level}_P(r) \leq 2/3$. Thus piece e would have fit in BIN_r and could not have gone on to $\text{BIN}_{Q_{\text{MAX}}+1}$, contrary to the definition of e. Thus $\text{piece}_P(r, 1)$ is a B-piece, and hence by Lemma 3.7, so are all the other elements of BOT. The Claim follows by Claim

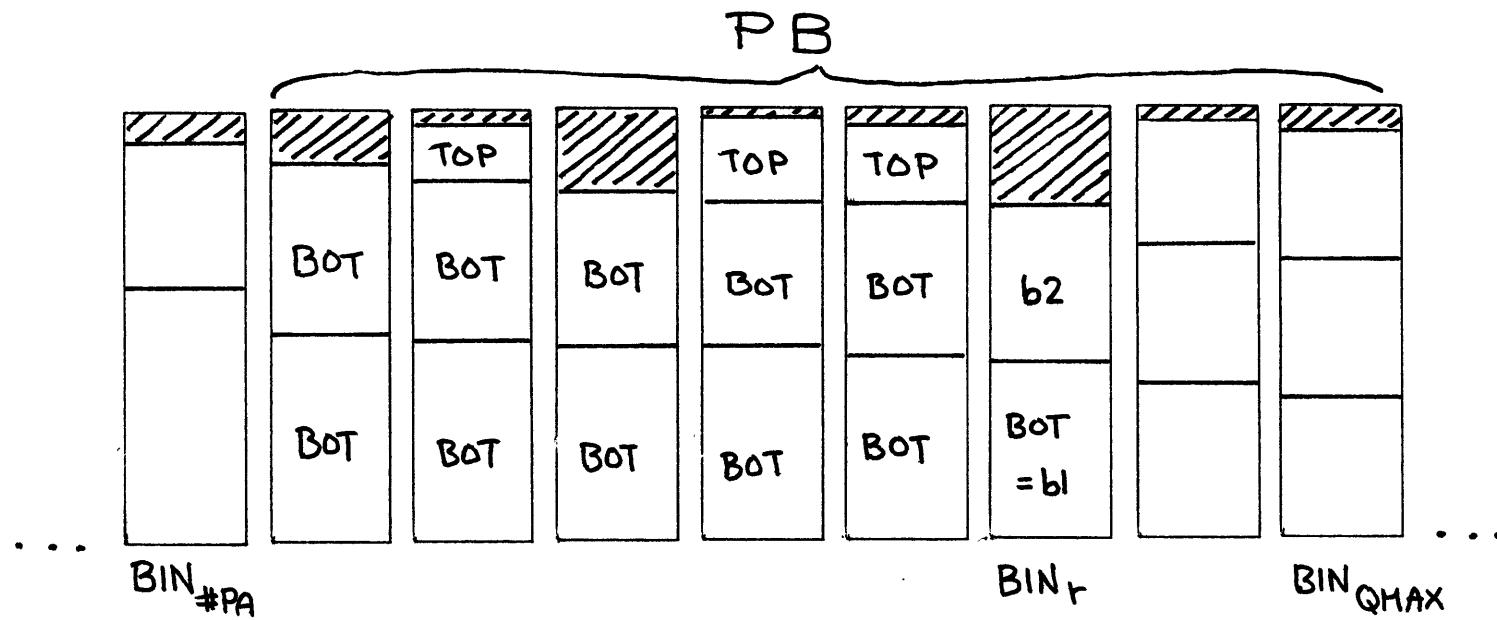


FIGURE 4.8. P with elements of BOT and TOP labeled.

4.8.3A. \square CLAIM 4.8.5. $\text{SMALL} \subseteq f(\text{TOP})$.

Proof of Claim. Let $b_1 = \text{piece}_P(r, 1)$, $b_2 = \text{piece}_P(r, 2)$.
 $\text{size}(b_1) \leq \min\{\text{size}(x) : x \in \text{BIG}\}$ by Lemma 3.7 and Claim 4.8.3A,
and $\text{size}(b_2) \leq \text{size}(b_1)$ by packing property (3). Therefore, if
 $x \in \text{SMALL}$, $\text{size}(x) + \text{size}(b_1) + \text{size}(b_2) \leq 1$, and so

$$\text{size}(x) \leq 1 - \text{size}(b_1) - \text{size}(b_2) = \text{gap}_P(r).$$

Consider $y = f^{-1}(x)$. There are four possibilities:

- (1) $y \in \text{cont}_P(j)$ for some $j > r$,
- (2) $y \in \text{a}$,
- (3) $y \in \text{BOT} \cup \{b_2\}$,
- (4) $y \in \text{TOP}$.

If $y \in \text{cont}_P(j)$ for $j > r$, then by the FF rule, $\text{size}(y) > \text{gap}_P(r) \geq \text{size}(x)$, contradicting Claim 4.8.3A, so $y \notin f^{-1}(x)$.

[As a special case note that $\text{size}(e) > \text{gap}_P(r)$.]

If $y \in \text{a}$, by Claim 4.8.3B, $\text{size}(x) = \text{size}(f(y)) \geq \text{size}(e) > \text{gap}_P(r)$, so $y \notin f^{-1}(x)$.

If $y \in \text{BOT} \cup \{b_2\}$, by Lemma 3.7 $\text{size}(y) \geq \text{size}(e)$, so again

by Claim 4.8.3A, $y \neq f^{-1}(x)$.

Thus $y = f^{-1}(x) \implies y \in \text{TOP}$, and the Claim is proven. \square

We now do some explicit counting of the pieces in $\text{Range}(f)$.

Let

$$p_2 = |\{j : \text{BIN}_j \text{ in } \text{PB} \text{ and } \text{height}_{\text{P}}(j) = 2\}|,$$

$$p_3 = |\{j : \text{BIN}_j \text{ in } \text{PB} \text{ and } \text{height}_{\text{P}}(j) = 3\}|.$$

CLAIM 4.8.6.

$$(A) \quad p_2 + p_3 = \#\text{QB},$$

$$(B) \quad |\text{Range}(f)| = 2(p_2) + 3(p_3) + |\text{cont}(\text{PX})| + |\alpha|,$$

$$(C) \quad |\text{BIG}| = 2(p_2 + |\text{TOP}|) - 1,$$

$$(D) \quad |\text{Range}(f) - \text{BIG}| = 3(p_3) - 2|\text{TOP}| + 1 \\ + |\text{cont}(\text{PX})| + |\alpha|.$$

Proof of Claim. $\text{Range}(\text{size}_U) \subseteq (1/4, 1]$, so no bin can contain more than 3 pieces in P. $\text{BIN}_{Q_{\text{MAX}}+1}$ is not empty, so Lemma 3.7 says that every bin in PB must contain at least two pieces. Since $\#\text{PB} = \#\text{QB}$, (A) holds. (B) follows from Claim 4.8.3, (C) follows from the definitions of BIG and TOP, and (D) follows from (B) and (C). \square

We now return to the packing Q . Even though crowded, some of its bins may still have room left over for one or more D-pieces. Let D_{MAX} be the maximum number of pieces of size $> 1/5$ that could be placed in the gaps in the first Q_{MAX} bins of Q . To compute an upper bound on D_{MAX} , let us classify the bins of Q as to their contents. Let

$$A(1) = \{BIN_j : j \leq |A-PIECES|, \text{level}_Q(j) \geq 4/5\},$$

$$A(2) = \{BIN_j : j \leq |A-PIECES|, \text{level}_Q(j) \in [3/5, 4/5)\},$$

$$A(3) = \{BIN_j : j \leq |A-PIECES|, \text{level}_Q(j) < 3/5\},$$

$$B(i, k) = \{BIN_j \text{ in } QB : |cont_Q(j) \cap BIG| = k,$$

$$|cont_Q(j) \cap Range(f)| = i.$$

In addition, let $a(i) = |A(i)|$, $b(i, k) = |B(i, k)|$. Then we have

CLAIM 4.8.7.

$$(A) \ #QA = \sum_{i=1}^3 a(i),$$

$$(B) \ #QB = \sum_{k=0}^2 b(3, k) + \sum_{k=0}^2 b(2, k) + \sum_{k=0}^1 b(1, k) + b(0, 0).$$

$$\begin{aligned} (C) \ D_{MAX} \leq & [a(1) + b(3, 0) + b(2, 2)] \\ & + 2[a(2) + b(2, 1) + b(2, 0)] \\ & + 3[b(1, 1) + b(1, 0)] + 4[b(0, 0)]. \end{aligned}$$

Proof of Claim. (A) is immediate. Since no bin can contain more than 3 pieces from L1, the only way $b(i,k)$ can be non-zero is if $0 \leq k \leq i \leq 3$. $b(3,3) = 0$ since three BIG-pieces would have a total size exceeding $3(1/3)$ by Claim 4.8.5. Thus the right hand side of (B) contains all non-zero $b(i,k)$'s and so (B) holds.

(C) follows due to size constraints: In order for a bin to have room for k D-pieces, it must have a gap in Q of more than $k/5$. Let us consider the maximum possible gaps for each class of bins.

A(1): gap $\leq 1/5$, at most 0 D-pieces.

A(2): gap $\leq 2/5$, at most 1 D-piece.

A(3): gap $\leq 1/2$, at most 2 D-pieces.

B(3,2): gap $\leq 1 - 2/3 - 1/4 = 1/12$, at most 0.

B(3,1): gap $\leq 1 - 1/3 - 2/4 = 1/6$, at most 0.

B(3,0): gap $\leq 1 - 3/4 = 1/4$, at most 1.

B(2,2): gap $\leq 1 - 2/3 = 1/3$, at most 1.

B(2,1): gap $\leq 1 - 1/3 - 1/4 = 5/12$, at most 2.

B(2,0): gap $\leq 1 - 2/4 = 1/2$, at most 2.

B(1,1): gap $\leq 1 - 1/3 = 2/3$, at most 3.

B(1,0): gap $\leq 1 - 1/4 = 3/4$, at most 3.

B(0,0): gap ≤ 1 , at most 4.

(C) is simply a summary of this case analysis, since as argued above the sets listed form a partition of the bins of QA and QB.

□

In order for the above bound on DMAX to have much meaning, we must know something more about the values of the $a(i)$'s and $b(i,k)$'s. To do this we shall use the information we got about Range(f) in Claim 4.8.6. For convenience, let us use the following shorthand notation:

$$\begin{aligned} B(3) &= \bigcup_{k=0}^2 B(3,k), \quad b(3) = |B(3)|, \\ B(2) &= \bigcup_{k=0}^2 b(2,k), \quad b(2) = |B(2)|, \\ B(1) &= \bigcup_{k=0}^1 b(1,k), \quad b(1) = |B(1)|. \end{aligned}$$

We are especially interested in counting the 3-piece bins containing a BIG-piece, $B(3,1) \cup B(3,2)$, for as we saw in the proof of Claim 4.8.7, these are the bins that do not have room for any D-pieces. We shall show that there are many of them. To this end, let

$$IN = \{x \in BIG \wedge \text{cont}_Q(j) : BIN_j \in B(3)\},$$

By obtaining a value for $|IN|$, the number of BIG-pieces in bins of $B(3)$, we can then get a lower bound on the number of such bins containing BIG-pieces. An intuitive argument about the value of $|IN|$ goes as follows:

Recall that PB was made up entirely of 2- and 3-piece bins, the number of BIG-pieces in 3-piece bins in PB is simply $2|TOP|$, and each of the 2-piece bins (except for BIN_r) contains 2 BIG-pieces. Thus for each piece that QB contains in excess of the number PB contains, there must be an additional 3-piece bin, and hence two additional BIG-pieces in IN . This makes for a total contribution of $2(|cont(PX)| + |Q|)$. Another possible reason for an increase in the number of 3-piece bins in QB is that some bin which is not a 3-piece bin contains fewer than 2 pieces. Such deficient bins will thus cause two pieces to be added to IN for each piece they are deficient. And finally, if a bin which is not a 3-piece bin does not contain 2 BIG-pieces, an additional piece is added to IN for each BIG-piece the bin is deficient. Thus each of the classes $B(i,k)$, $i \leq 2$, of bins not containing three pieces from $\text{Range}(f)$ will contribute $[2(2-i) + (2-k)]b(i,k)$ to $|IN|$. Let

$$\begin{aligned} \text{EXTRA} &= \sum_{i=0}^2 \sum_{k=0}^i [2(2-i)+(2-k)]b(i,k) \\ &= 6b(0,0)+4b(1,0)+3b(1,1)+2b(2,0)+b(2,1). \end{aligned}$$

The next two claims summarize these arguments and give more formal proofs.

$$\text{CLAIM 4.8.8. } b(3) - p_3 = |\text{cont}(P_X)| + |\alpha| \\ + 2b(0,0) + b(1).$$

Proof of Claim. By claims 4.8.6A and 4.8.7B,

$$p_2 + p_3 = b(0,0) + b(1) + b(2) + b(3).$$

By Claim 4.8.6B and the definition of the $b(i,k)$'s,

$$b(1) + 2b(2) + 3b(3) = |\text{range}(f)| \\ = 2(p_2) + 3(p_3) + |\text{cont}(P_X)| + |\alpha|.$$

Claim 4.8.8 follows from combining these two equations. \square

$$\text{CLAIM 4.8.9. } |\text{IN}| = 2\{|\text{cont}(P_X)| + |\alpha| + |\text{TOP}| \} - 1 + \text{EXTRA}.$$

Proof of Claim. Let

$$\text{IN}' = \{x \in [\text{Range}(f) - \text{BIG}] \cap \text{cont}_Q(j) : \text{BIN}_j \in B(3)\}, \\ \text{OUT}' = \{x \in [\text{Range}(f) - \text{BIG}] \cap \text{cont}_Q(j) : \text{BIN}_j \notin B(3)\}.$$

As immediate consequences of the definitions we have

$$\begin{aligned} |\text{IN}| + |\text{IN}'| &= 3b(3), \\ |\text{IN}'| + |\text{OUT}'| &= |\text{Range}(f)-\text{BIG}|, \\ |\text{OUT}'| &= 2b(2,0) + b(2,1) + b(1,0) \end{aligned}$$

Combining we get

$$|\text{IN}| = 3b(3) + 2b(2,0) + b(2,1) + b(1,0) - |\text{Range}(f)-\text{BIG}|.$$

But by Claims 4.8.6B and 4.8.8,

$$\begin{aligned} |\text{Range}(f)-\text{BIG}| &= 3(p_3) - 2|\text{TOP}| + 1 + |\text{cont}(P_X)| + |\alpha|, \\ 3b(3) - 3(p_3) &= 3[|\text{cont}(P_X)| + |\alpha| + 2b(0,0) + b(1)]. \end{aligned}$$

Claim 4.8.9 follows by substitution. \square

We now can give a lower bound on the number of 3-piece bins containing BIG-pieces:

CLAIM 4.8.10.

$$b(3,1) + b(3,2) \geq 2[|\text{cont}(P_X)| + |\alpha|] + |\text{TOP}| + \text{EXTRA} - 1.$$

Proof of Claim. By the definitions, we have $|IN| = b(3,1) + 2b(3,2)$. Therefore,

$$b(3,1) + b(3,2) = |IN| - b(3,2).$$

But by Claim 4.8.5 and the definition of SMALL,

$$b(3,2) \leq |\text{SMALL}| \leq |\text{TOP}|.$$

Using this to substitute for $b(3,2)$ and Claim 4.8.9 to substitute for $|IN|$ yields the desired result. \square

We are now ready to get a more meaningful bound on DMAX.

CLAIM 4.8.11. $DMAX \leq QMAX - 2|\text{cont}(PX)| + 1$.

Proof of Claim. By Claim 4.8.7 we have

$$\begin{aligned} QMAX &= \#QA + \#QB = b(3,2) + b(3,1) + b(3,0) + \sum_{i=1}^3 a(i) \\ &\quad + \sum_{k=0}^2 b(2,k) + \sum_{k=0}^1 b(1,k) + b(0,0). \end{aligned}$$

Using Claim 4.8.10 to substitute for $b(3,2) + b(3,1)$ we get

$$\begin{aligned} Q_{\text{MAX}} &\geq 2|\text{cont}(P_X)| - 1 + |\text{TOP}| + |\alpha| + b(3,0) \\ &\quad + a(i) + 3b(2,0) + 2b(2,1) + b(2,2) \\ &\quad + 4b(1,1) + 5b(1,0) + 7b(0,0). \end{aligned}$$

$$\begin{aligned} &\geq 2|\text{cont}(P_X)| - 1 \\ &\quad + [a(1)+b(3,0)+b(2,2)] + 2[a(2)+b(2,1)+b(2,0)] \\ &\quad + 3[b(1,1)+b(1,0)] + 4[b(0,0)], \end{aligned}$$

since $|\alpha| = a(2)$ by Claim 4.8.2. But by Claim 4.8.7C, this yields $Q_{\text{MAX}} \geq 2|\text{cont}(P_X)| - 1 + D_{\text{MAX}}$, and Claim 4.8.11 follows.

□

The next claim will complete the proof of Theorem 4.8:

CLAIM 4.8.12. $\text{FF}(L_1) + \text{FF}(L_2) \leq (5/4)L^* + 2$.

Proof of Claim. Suppose Q_{MAX} and Q were obtained as follows: Let P^* be an optimal packing of $L = L_1 \cdot L_2$. Obtain Q from P^* by removing all the D -pieces (the pieces of L_2). Set $Q_{\text{MAX}} = L^* < \#P$. By the definition of D_{MAX} and Claim 4.8.11, we then have

$$|\text{PIECES}(L_2)| \leq D_{\text{MAX}} \leq L^* - 2|\text{cont}(P_X)| + 1.$$

Now since by Claim 4.8.1 all pieces in $\text{cont}(P_X)$ have size in $(1/4, 1/3]$, and by definition $\text{Range}(\text{size}_{L_2}) \subseteq (1/5, 1/4]$, Lemma 3.7 tells us that

$$\#P_X = \left\lceil \frac{|\text{cont}(P_X)|}{3} \right\rceil < \frac{|\text{cont}(P_X)|}{3} + 1,$$

$$FF(L_2) = \left\lceil \frac{|\text{PIECES}(L_2)|}{4} \right\rceil < \frac{L^* - 2|\text{cont}(P_X)| + 1}{4} + 1.$$

$$\text{Thus } FF(L_1) + FF(L_2) = \#P_A + \#P_B + \#P_X + FF(L_2)$$

$$= L^* + \#P_X + FF(L_2)$$

$$\leq (5/4)L^* + 2 - |\text{cont}(P_X)|/6 \leq (5/4)L^* + 2,$$

and both Claim 4.8.12 and Theorem 4.8 are proven. \square

The immediate application of Theorem 4.8 is to prove our claimed upper bound on $R[SD]$:

THEOREM 4.9. If $S \in AF$, $t \in (0,1]$, then

$$R[SD, t] \leq 5/4.$$

Proof. Let L be an arbitrary list in decreasing order, and divide it into segments $L_1 \cdot L_2 \cdot L_3$, where $\text{Range}(\text{size}_{L_1}) \subseteq (1/4, 1]$, $\text{Range}(\text{size}_{L_2}) \subseteq (1/5, 1/4]$, and $\text{Range}(\text{size}_{L_3}) \subseteq (0, 1/5]$. If $\text{FF}(L_1) \leq (L_1 \cdot L_2)^*$, then by Lemma 4.6, $S(L) \leq (5/4)L^* + 2$. If not, then by Theorem 4.8, $\text{FF}(L_1) + \text{FF}(L_2) \leq (5/4)L^* + 2$, and so by Lemma 4.7, $S(L) \leq (5/4)L^* + 3$. Thus the latter inequality holds in any case, and the Theorem follows via Lemma 3.1. \square

The bound given in Theorem 4.9 is slightly higher than our best lower bound of $11/9$, and even worse when compared to $7/6$, our best lower bound for $t \leq 1/2$. However, by a second application of Theorem 4.8 we have the following result for more restricted lists:

THEOREM 4.10. If $S \in AF$, $t \in (1/3, 1]$, then

$$R[SD, (1/4, t)] = 7/6.$$

Proof. For the lower bound, consider the lists given in Figure 4.2 with $m = 2$ and $0 < \epsilon < t - 1/3$. These lists can have arbitrarily large L^* , and yet for all,

$$\text{FF}(L)/L^* = (2^2 + 2 \cdot 2 - 1)/(2^2 + 2) = 7/6.$$

For the upper bound, by Lemma 3.1 all we need prove is that $S(L) \leq (7/6)L^* + 2$ for all decreasing lists L with $\text{Range}(\text{size}_L) \subseteq (1/4, 1]$. So let L be such a list. Since by Theorem 3.9 $S(L) \leq \text{FF}(L) + 1$, it is enough to prove that $\text{FF}(L) \leq (7/6)L^* + 1$. If $\text{SS}(L) \leq L^*$ we are done, so assume $\text{FF}(L) > L^*$.

Now, suppose that in the Theorem 4.8 construction we took $L_1 = L$, $Q_{\text{MAX}} = L^*$, and Q = an optimal packing of L. Then Claim 4.8.11 tells us that

$$|\text{cont}(P_X)| \leq (L^* + 1)/2.$$

It then follows from Claim 4.8.1 and Lemma 3.7 that

$$\begin{aligned} \text{FF}(L) &= L^* + \#P_X \leq L^* + \lceil |\text{cont}(P_X)|/3 \rceil \\ &\leq L^* + \lceil |(L^*+1)/2|/3 \rceil \\ &\leq (7/6)L^* + 1. \quad \square \end{aligned}$$

Now the major complication in the proof of Theorem 4.8 was the presence of the A-pieces in L and hence A-bins in P and Q. It was due to them that we had to define f and worry about the

set \mathcal{A} . If there had been no A-pieces, and hence $\text{Range}(\text{size}_L) \subseteq (1/5, 1/2]$, we could have simply used x in place of $f(x)$ in our arguments, since PA and QA would be vacuous and hence $\mathcal{B} = \mathcal{P} = \mathcal{Q}$. We now generalize this simpler situation and derive improved upper bounds on $R[SD, t]$ for all $t \in (0, 1/2]$.

First we need analogues of Lemmas 4.6 and 4.7. We omit the proofs except to note that they are analogous to those of the lemmas they imitate, with the reference to Theorem 3.9 replaced by a reference to Corollary 3.12.1:

LEMMA 4.11. Suppose $L = L_1 \bullet L_2 \bullet L_3$ is a list in decreasing order with $\text{Range}(\text{size}_{L_1}) \subseteq (1/(m+2), 1/m]$, $\text{Range}(\text{size}_{L_2}) \subseteq (1/(m+3), 1/(m+2)]$, and $\text{Range}(\text{size}_{L_3}) \subseteq (0, 1/(m+3)]$, and $S \in \text{AF}$. If $\text{FF}(L_1) \leq (L_1 \bullet L_2)^*$, then $S(L) \leq [(m+3)/(m+2)]L^* + 2$.

LEMMA 4.12. Suppose L and S are as above. If $\text{FF}(L_1) + \text{FF}(L_2) \leq [(m+3)/(m+2)]L^* + 2$, then $S(L) \leq [(m+3)/(m+2)]L^* + 3$.

Continuing, we next have an analogue of Theorem 4.8:

THEOREM 4.13. Suppose $L = L_1 \bullet L_2$ is a list in decreasing order with $\text{Range}(\text{size}_{L_1}) \subseteq (1/(m+2), 1/m]$ and $\text{Range}(\text{size}_{L_2}) \subseteq (1/(m+3)]$. Then

$$\text{FF}(L1) > L* \implies \text{FF}(L1) + \text{FF}(L2) \leq \left[\frac{(m+3)}{(m+2)}\right]L* + 2.$$

Proof. Our basic strategy is the same. Assume $\text{FF}(L1) > L*$, and let $L = L[m] L[m+1] L[m+2]$, where $L[m+2] = L2$, and $\text{Range}(\text{size}_{L[m]}) \subseteq (1/(m+1), 1/m]$, $\text{Range}(\text{size}_{L[m+1]}) \subseteq (1/(m+2), 1/(m+1)]$. Let $k\text{-pieces}(L) = \text{PIECES}(L[k])$, $m \leq k \leq m+2$. This is basically the notation used in the proof of Theorem 3.12.

Now let P be the FF-packing of $L1$, and choose Q_{MAX} and Q so that $Q_{\text{MAX}} < \#P$, and Q is an ordered packing of $L1$ with $\text{MAX}\{j : \text{level}_Q(j) > 0\} \leq Q_{\text{MAX}}$. By QB we shall mean the first Q_{MAX} bins of Q , but note that some of these may well be empty. Divide P into segments PB and PX as follows:

$$\begin{aligned} PB &= \{ \text{BIN}_j : 1 \leq j \leq Q_{\text{MAX}} \}, \\ PX &= \{ \text{BIN}_j : Q_{\text{MAX}} < j \leq \#P \}. \end{aligned}$$

CLAIM 4.13.1. $\emptyset \neq \text{cont}(PX) \subseteq (m+1)\text{-PIECES}$.

Proof of Claim. $\text{cont}(PX)$ cannot be empty since that would imply $L* \leq Q_{\text{MAX}} < \#P$, contrary to hypothesis. If PX contained an m -piece, then by Lemma 3.7 $|\text{PIECES}(L1)| > m \cdot Q_{\text{MAX}} \geq mL*$, which is impossible since no bin can contain more than m m -pieces. \square

We now shall proceed at a rather rapid rate when the claims are simple analogues of those in Theorem 4.8, not repeating the explanatory material or going into the details of the proofs. However, in certain cases the more general nature of the claims will require a different type of argument, and these we shall present. To begin, let

$$\begin{aligned}
 r &= \text{MAX}\{j: \text{BIN}_j \text{ in PB and } \text{height}_P(j) = m\}, \\
 \text{BIG} &= \{\text{piece}_P(j, h): 1 \leq j \leq r, h \leq m\} \\
 &\quad - \{\text{piece}_P(r, h): h > 1\}, \\
 \text{TOP} &= \{\text{piece}_P(j, m+1): 1 \leq j \leq r\}, \\
 \text{SMALL} &= \{z \in \text{PIECES}(L): \text{there exists } X \subseteq \text{BIG} \text{ with} \\
 &\quad |X| = m \text{ and } \sum_{x \in X} \text{size}(x) + \text{size}(z) \leq 1\}, \\
 p[m] &= |\{j: \text{BIN}_j \text{ in PB and } \text{height}_P(j) = m\}|, \\
 p[m+1] &= |\{j: \text{BIN}_j \text{ in PB and } \text{height}_P(j) = m+1\}|.
 \end{aligned}$$

CLAIM 4.13.2. $\text{BIG} \subseteq m\text{-PIECES}(L)$.

CLAIM 4.13.3. $\text{SMALL} \subseteq \text{TOP}$.

CLAIM 4.13.4.

- (A) $p[m] + p[m+1] = QMAX.$
- (B) $|PIECES(L1)| = m(p[m]) + (m+1)p[m+1] + |\text{cont}(P_X)|,$
- (C) $|\text{BIG}| = m(p[m] + |\text{TOP}|) - (m-1),$
- (D) $|PIECES(L1)-\text{BIG}| = (m+1)p[m+1] - m|\text{TOP}| + (m-1)$
 $+ |\text{cont}(P_X)|.$

Now partition the bins of QB as follows:

$$B(i, k) = \left\{ \text{BIN}_j \text{ in QB: } \text{height}_Q(j) = i, \text{ and} \right. \\ \left. |\text{cont}_Q(j) \cap \text{BIG}| = k \right\},$$

letting $b(i, k) = |B(i, k)|$, $B(i) = \bigcup_{k=0}^i B(i, k)$, and $b(i) = |B(i)|$.

Let DMAX be the maximum number of pieces of size $> 1/(m+3)$ that could be placed in the gaps of the first QMAX bins of Q.

CLAIM 4.13.5.

$$(A) QMAX = \sum_{k=0}^m b(m+1, k) + \sum_{i=0}^m \sum_{k=0}^i b(i, k),$$

$$(B) DMAX \leq \sum_{l=0}^{m+1} \left[\sum_{k=0}^{m+1-l} (m+2-i)b(i, k) + \sum_{k=m+2-i}^i (m+1-i)b(i, k) \right].$$

Proof of Claim. (A) is immediate since no bin can contain $m+1$ BIG-pieces. For (B), since no bin can contain more than $m+2$ pieces of size $> 1/(m+3)$, there is an easy upper bound on DMAX

of $\sum_{i=0}^{m+1} (m+2-i)b(i)$. However, note that an $(m+2)$ -piece and an m -piece must have total size exceeding

$$\frac{1}{m+3} + \frac{1}{m+1} = \frac{2m+4}{(m+3)(m+1)} > \frac{2}{m+2},$$

since $2m^2 + 8m + 8 > 2m^2 + 8m + 6$. Thus a bin with i pieces from L1, $m+2-i$ of which are BIG-pieces can only have room for $m+1-i$ $(m+3)$ -pieces, since if $m+2-i$ such pieces were added, it would yield a level exceeding

$$\frac{m+2-i}{m+3} + \frac{m+2-i}{m+1} + \frac{i-(m+2-i)}{m+2} > \frac{2(m+2-i)}{m+2} + \frac{i-(m+2-i)}{m+2} = 1.$$

The result follows from this observation. \square

The next two claims are exact generalizations of Claims 4.8.8 and 4.8.9, proved and intuitively explained in the same way:

CLAIM 4.13.6.

$$b(m+1) - p[m+1] = |\text{cont}(PX)| + \sum_{i=0}^m (m-i)b(i).$$

Define $IN = \{x \in BIG \cap \text{cont}_Q(j) : BIN_j \in B(m+1)\}$.

$$\begin{aligned} \text{CLAIM 4.13.7. } |IN| &= m(|\text{cont}(PX)| + |\text{TOP}|) - (m-1) \\ &+ \sum_{i=0}^m \sum_{k=0}^i |m(m-i)+(m-k)| b(i,k). \end{aligned}$$

CLAIM 4.13.8.

$$b(m+1, k) \geq \frac{|IN| - |\text{TOP}|}{m-1}.$$

Proof of Claim. By definition of IN,

$$\begin{aligned} |IN| &= \sum_{k=1}^m kb(m+1, k) \\ &= (m-1) \sum_{k=0}^m b(m+1, k) + b(m+1, m) - \sum_{k=0}^{m-2} (m-1-k)b(m+1, k). \end{aligned}$$

By Claim 4.13.3 and the definition of SMALL, $b(m+1, m) \leq |\text{TOP}|$.

The Claim follows. \square

$$\text{CLAIM 4.13.9. } DMAX \leq QMAX - [m/(m-1)] \cdot |\text{cont}(PX)| + 1.$$

Proof of Claim. By Claims 4.13.5, 4.13.7, and 4.13.8,

$$Q_{\text{MAX}} \geq b(m+1,0) + \sum_{i=0}^m \sum_{k=0}^i b(i,k)$$

$$+ \frac{m|\text{cont}(PX)| + (m-1)|\text{TOP}| - (m-1) + \sum_{i=0}^m \sum_{k=0}^i [m(m-i)+(m-k)]b(i,k)}{m-1}$$

$$\text{Hence, } Q_{\text{MAX}} = \left[\frac{m}{m-1} \right] |\text{cont}(PX)| + 1 \geq$$

$$\sum_{i=0}^m \sum_{k=0}^i \frac{m(m-i) + (m-k) + (m-1)}{m-1} b(i,k) + b(m+1,0),$$

The proof is thus reduced to comparing the coefficients of the $b(i,k)$'s given above (the QMAX coefficients) with those given for the upper bound on DMAX given in Claim 4.13.5B. There, the coefficient of $b(i,k)$ is

$$(m+2-i), \text{ if } 0 \leq k \leq \min\{i, m+1-i\},$$

$$(m+1-i), \text{ if } \min\{i, m+2-i\} \leq k \leq i.$$

If $i = m+1$, then the only way the DMAX-coefficient of $b(i,k)$ can be non-zero is if $k = 0$, in which case it is 1, which is the QMAX-coefficient.

If $i < m+1$, there are two cases:

Case 1. $m+2-i \leq k \leq i$. In this case the numerator of the QMAX coefficient is $m(m-i) + (m-k) + (m-1)$

$$> (m-1)(m-i) + (m-1) = (m-1)(m+1-i),$$

so that the QMAX coefficient exceeds $m+1-i = \text{DMAX coefficient}.$

Case 2. $0 \leq k \leq m+1-i.$ In this case the numerator is at least $m(m-i) + (m-(m+1-i)) + (m-1)$

$$\begin{aligned} &= (m-1)(m+1-i) + (m-i) + (i-1) \\ &= (m-1)(m+1-i) + m-1 = (m-1)(m+2-i), \end{aligned}$$

so that the QMAX coefficient is at least as large as the DMAX coefficient.

Since each of the $b(i,k)$'s consequently has at least as large a QMAX coefficient as a DMAX coefficient, the Claim follows. \square

The next Claim completes the proof of Theorem 2.13:

$$\text{CLAIM 4.13.10. } \text{FF}(L1) + \text{FF}(L2) \leq [(m+3)/(m+2)] L^* + 2.$$

Proof of Claim. Suppose QMAX and Q were obtained as follows: Let P^* be an optimal packing of $L = L1 \bullet L2.$ Obtain Q from P^* by removing all the $(m+2)$ -pieces (the pieces of $L2$). Set $\text{QMAX} = L^* < \#P.$ By definition of DMAX and Claim 4.13.9 we

then have

$$|\text{PIECES}(L2)| \leq D_{\text{MAX}} \leq L^* - \left[\frac{(m/(m-1))}{m+1} |\text{cont}(P_X)| + 1 \right].$$

Now since by Claim 4.13.1 all the pieces in $\text{cont}(P_X)$ are $(m+1)$ -pieces, and by definition all the pieces in L_2 are $(m+2)$ -pieces, Lemma 3.7 tells us that

$$\#P_X = \left\lceil \frac{|\text{cont}(P_X)|}{m+1} \right\rceil < \frac{|\text{cont}(P_X)|}{m+1} + 1,$$

$$FF(L_2) = \left\lceil \frac{|\text{PIECES}(L_2)|}{m+2} \right\rceil < \frac{L^* - \left[\frac{(m/(m-1))}{m+1} |\text{cont}(P_X)| + 1 \right]}{m+2} + \frac{1}{m+2}.$$

Thus $FF(L_1) + FF(L_2) = \#P_B + \#P_X + FF(L_2)$

$$= L^* + \#P_X + FF(L_2)$$

$$\leq \left[\frac{(m+3)/(m+2)}{m+2} L^* + 2 - \left(2 / \left[\frac{(m-1)(m+1)(m+2)}{m+2} \right] \right) |\text{cont}(P_X)| \right]$$

$$\leq \left[\frac{(m+3)/(m+2)}{m+2} L^* \right].$$

Thus Claim 4.13.10 and Theorem 4.13 are proven. \square

We thus can conclude via Lemmas 4.11, 4.12, and 3.1 that

THEOREM 4.14. If $S \in AF$ and $m = \lfloor 1/t \rfloor \geq 2$, then

$$R[SD, t] \leq 1 + 1/(m+2).$$

The upper bound given in Theorem 4.14 is slightly worse than the best lower bound known of $1 + 1/(m+2) - 2/[m(m+1)(m+2)]$. However, we can, as in Theorem 4.10, get an optimal result by further restricting the range of the piece sizes:

THEOREM 4.15. If $S \notin AF$ and $m = \lfloor 1/t \rfloor \geq 2$, then

$$R[SD, (1/(m+2), t)] = 1 + 1/(m+2) - 2/[m(m+1)(m+2)].$$

Proof. The upper bound follows from the generic example given in Figure 4.2, with $0 < \epsilon < t - 1/(m+1)$. For the upper bound, let L be a list in decreasing order with $\text{Range}(\text{size}_L) \subseteq (1/(m+2), 1/m]$. As in Theorem 4.10, by using L as L^* in the construction for Theorem 4.13, with $Q_{\text{MAX}} = L^*$ and Q an optimal packing of L , Claim 4.13.9 tells us that

$$|\text{cont}(P_X)| \leq [(m-1)/m](L^* + 1).$$

By Claim 4.13.1 all the pieces in $\text{cont}(P_X)$ are $(m+1)$ -pieces, so by Lemma 3.7 we then have

$$\begin{aligned} \text{FF}(L) &= L^* + \#P_X \leq L^* + \lceil |\text{cont}(P_X)|/(m+1) \rceil \\ &\leq L^* + \left[(m-1)/(m(m+1)) \right] (L^* + 1) \\ &\leq \left[1 + 1/(m+2) - 2/\left[m(m+1)(m+2)\right] \right] L^* + 1. \end{aligned}$$

Since we have by Corollary 3.12.1 that $S(L) \leq \text{FF}(L)+1$, the Theorem follows by Lemma 3.1. |

CHAPTER 5. EXACT UPPER BOUNDS

SECTION 5.1. A Weighting Function

In the previous Chapter, we proved upper bounds on $R[SD, t]$ for $S \in AF$ which were strictly larger than the best lower bounds known. In this Chapter we develop a method with which we will be able to determine the precise values in a number of cases.

Recall that the principle ingredient in the proofs of Theorem 2.6 and some of the subcases of Theorem 2.8 in Chapter 2 was a weighting function,

$$w: PIECES(L) \longrightarrow Q,$$

which assigned number values to pieces according to their size. The proofs consisted of showing that for all L with piece sizes restricted to the appropriate range, the following two inequalities held:

$$(5A) \quad w[PIECES(L)] \leq rL^*,$$

$$(5B) \quad w[PIECES(L)] \geq S(L) - K,$$

where $w[\text{PIECES}(L)] = \sum_{x \in \text{PIECES}(L)} w(x)$, r is the upper bound on $R[S, t]$ to be proved, and K is a constant depending on the allowable range of piece sizes, and independent of L^* .

In this Chapter we shall use the same approach; however, our weighting function W_d will be more complex than a simple function on pieces. Instead it will be defined for sets of pieces:

$$W_d: 2^{\text{PIECES}(L)} \dashrightarrow Q,$$

so that $W_d[\text{PIECES}(L)]$ will be a direct application of W_d , rather than shorthand for an implicit sum.

In addition, W_d will only be defined for lists L with $\text{Range}(\text{size}_L) \subseteq (0, 1/2]$. This seemingly restricts the use of W_d to proving results of the form $R[S, t] \leq r$ for $t \leq 1/2$, and such indeed will be our first application. However, in Section 5.2 we shall show how to adapt W_d to the more general case, where pieces of size larger than $1/2$ are allowed.

W_d will be defined in terms of two constituent functions:

$$w_1: \text{PIECES}(L) \dashrightarrow Q,$$

$$w_2: \text{PIECES}(L) \times \text{PIECES}(L) \dashrightarrow Q,$$

again with values depending only on piece size. Given w_1 and w_2 , and a set D of pieces, $W_d(D)$ is defined as follows:

For any partition Π of D into 1- and 2-element sets, let

$$\Pi_1 = \{x \in D : \{x\} \in \Pi\},$$

$$\Pi_2 = \{(x, y) : \{x, y\} \in \Pi, \text{rank}(x) < \text{rank}(y)\},$$

and

$$w_{12}(\Pi) = \sum_{(x,y) \in \Pi_2} w_2(x, y) + \sum_{x \in \Pi_1} w_1(x).$$

We then have

$$W_d(D) = \min \{w_{12}(\Pi) : \Pi \text{ is a partition of } D \text{ into 1- and 2-element sets}\}.$$

Without yet going into the details of the definitions of w_1 and w_2 , we can already see how we shall go about proving inequality (5A) for $w = W_d$:

LEMMA 5.1. Suppose $\text{Range}(\text{size}_L) \subseteq (0, 1/2]$, T is a finite collection of disjoint subsets D_i of $\text{PIECES}(L)$, and $U = \bigcup_i D_i$. Then $W_d(U) \leq \sum_i W_d(D_i)$.

Proof. Since each D_i is finite, for each there must be a partition Π_i of D_i into 1- and 2-element sets such that

$$\begin{aligned} w_{12}(\Pi_i) &= \min \left\{ w_{12}(\pi) : \pi \text{ is any partition of } D_i \right. \\ &\quad \left. \text{into 1- and 2-element sets} \right\} \\ &= Wd(D_i). \end{aligned}$$

Then, since $\bigcup_i \Pi_i$ is a partition of U into 1- and 2-element sets,

$$\begin{aligned} Wd(U) &\leq w_{12}\left(\bigcup_i \Pi_i\right) = \sum_i w_{12}(\Pi_i) \\ &= \sum_i Wd(D_i). \quad \square \end{aligned}$$

COROLLARY 5.1.1. If for any BIN_j of an optimal packing P^* of a list L with $\text{Range}(\text{size}_L) \subseteq (0, 1/2]$, we have $Wd[\text{cont}_{P^*}(j)] \leq r$, then $Wd[\text{PIECES}(L)] \leq rL^*$.

Thus we can prove that (5A) holds using the same type of case analysis that was used in Chapter 2, restricting our attention to the possible configurations of pieces in a single bin.

Inequality (5B) is independent of the bound we are trying to prove, so one proof of that inequality, with the constant K specified in terms of the allowable range of piece size, can be made to serve for all our various upper bound proofs. But first we must provide the details on the definitions of w_1 and w_2 .

Recall the definition of k-piece from Chapter 3: x is a k-piece if $\text{size}(x) \in (1/(k+1), 1/k]$. We define w_1 by $w_1(x) = 1/k$, where k is such that x is a k-piece. Thus w_1 is a step function on $\text{size}(x)$. w_2 is a bit more complicated.

If x and y are such that x is a k-piece and $k \cdot \text{size}(x) + \text{size}(y) \leq 1$, we say that (x, y) obeys Discounting Relation k, or simply Relation k. Note that y must be a k' -piece for some $k' > k$ for (x, y) to obey Relation k. w_2 is defined as follows:

If (x, y) obeys Relation k,
 $w_2(x, y) = w_1(x) + [(k-1)/k]w_1(y).$

If (x, y) obeys no Discounting Relation, then
 $w_2(x, y) = w_1(x) + w_1(y).$

Let $\text{DISCOUNT}(x, y) = w_2(x, y) - w_1(x) - w_1(y)$. Then if (x, y) obeys Relation k, $\text{DISCOUNT}(x, y) = [1/k]w_1(y)$ and we say that y has been discounted by $1/k$. If R is a set of pairs, let $\text{DISCOUNT}(R) = \sum_{(x,y) \in R} \text{DISCOUNT}(x, y)$.

The significance of all this will appear later on. For the moment we continue with our definitions. Suppose $S \in \text{AF}$, L is a decreasing list, and PS is an S-packing of L. Define a function $J: N \times N \rightarrow N$ as follows:

$J[k, 1]$ is the index of the rightmost k-bin in PS if there is one, otherwise undefined.

$J[k, i+1]$ is the index of the bin to the left of $\text{BIN}_{J[k, i]}$, if such a bin exists and is a k-bin, otherwise undefined.

See Figure 5.1, which in addition illustrates the following two definitions:

$\text{BASIC} = \{x \in \text{PIECES}(L) : x \text{ is a } k\text{-piece and in a } k\text{-bin,}$
 $\text{for some } k \geq 2\}$,

$\text{SURPLUS} = \text{PIECES}(L) - \text{BASIC}$.

BASIC = \boxed{k}

SURPLUS = \boxed{k}

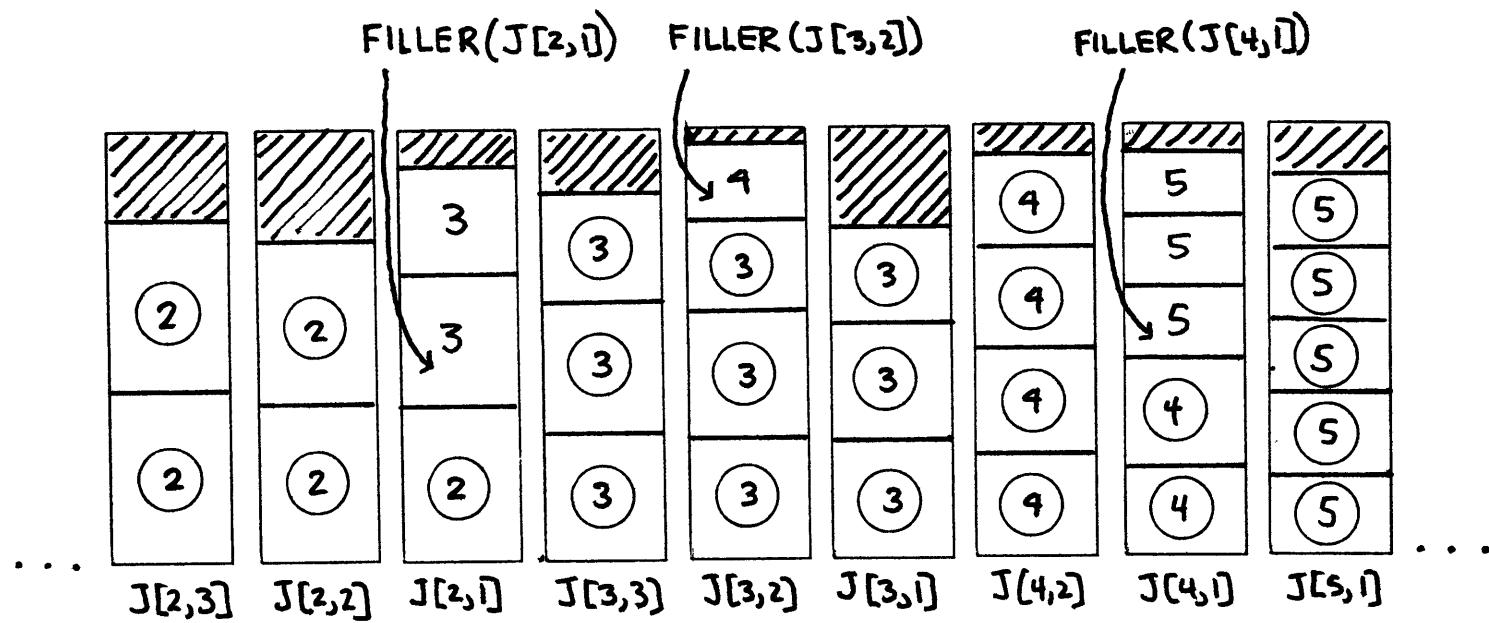


FIGURE 5.1. Packing PS illustrating definitions.

Relative to these sets, For each BIN_j in PS, let $Bcont(j) = \text{cont}_{PS}(j) \cap \text{BASIC}$, $Blevel(j) = \sum_{x \in Bcont(j)} \text{size}(x)$, and $Bgap(j) = 1 - Blevel(j)$.

LEMMA 5.2. Suppose L , S , and PS are as above, and $R = \{(x_1, y_1), \dots, (x_n, y_n)\}$ is a set of disjoint pairs obeying Relation k , such that $\text{rank}(y_1) < \dots < \text{rank}(y_n)$ and each x_i is in BASIC. Then for all i , $1 \leq i \leq \lfloor n/k \rfloor$,

$$\text{size}(y_{k_i}) \leq Bgap(J[k, i]).$$

Proof. Let $X_i = \{x_h : h \leq k_i\}$,

$$K_i = \{x \in Bcont(J[k, h]) : 1 \leq h \leq i\}.$$

Since the pairs in R are all disjoint and no k -bin can have more than k k -pieces,

$$|K_i| \leq k_i = |X_i|.$$

Moreover, by Lemma 3.6, the elements of K_i are the $|K_i|$ smallest k -pieces in BASIC. A simple induction establishes that

$$\begin{aligned} KMAX &= \text{MAX} \left\{ \sum_{x \in S} \text{size}(x) : S \subseteq K_i, |S| \leq k \right\} \\ &\leq XMAX = \text{MAX} \left\{ \sum_{x \in S} \text{size}(x) : S \subseteq X_i, |S| = k \right\}. \end{aligned}$$

But again by Lemma 3.6, $Blevel(j[k, i]) = KMAX$. On the other hand, since $\text{rank}(y_{k_i}) \geq \text{rank}(y_h)$ for all $h \leq i$, and each (x_h, y_h) obeys Relation k, we have

$$\text{size}(y_{k_i}) + k \cdot \text{size}(x_h) \leq 1, \text{ for all } h \leq k_i.$$

Hence by averaging, we have that for all $S \subseteq X$ with $|S| = k$,

$$\text{size}(y_{k_i}) \leq 1 - \sum_{x \in S} \text{size}(x),$$

and so

$$\text{size}(y_{k_i}) \leq 1 - XMAX \leq 1 - KMAX = Bgap([k, i]). \quad \square$$

Continuing with our definitions, let $\text{FILLER}(j)$ be the bottom-most non-BASIC piece in BIN_j of PS, if such a piece exists, otherwise undefined. Each BIN_j in Figure 5.1 has $\text{FILLER}(j)$ designated if it exists. We then have the following:

LEMMA 5.3. Suppose L , S , and PS are as above, and in addition $S \in AAF$. If $x \in BASIC$ is a k' -piece, BIN_j is a k -bin in PS for $k' > k$, and $\text{size}(x) \leq Bgap(j)$, then there exists a $z = \text{FILLER}(j)$ and $\text{rank}(z) < \text{rank}(x)$.

Proof. Since $x \in BASIC$, x is in a k' -bin $BIN_{j'}$ somewhere to the right of the k -bin BIN_j . When x was assigned by S , $BIN_{j'}$ can have contained at most $k'-1$ k' -pieces, and hence must have had $\text{level}_P(j') \leq (k'-1)/k'$.

But by Lemma 2.1 and the fact that $BIN_{j'}$'s bottom piece is a k' -piece, all previous bins must have had levels exceeding $1 - 1/k' = (k'-1)/k'$, and by Lemma 3.6, $BIN_{j'+1}$ could not yet have been started. Thus $BIN_{j'}$ was at the time the unique bin in packing P with minimum non-zero level. Thus, by the AAF Constraint, x could not have been assigned to $BIN_{j'}$ unless it would not have fit in any bin to the left.

If BIN_j had at that time contained only its k -pieces, we would have had $\text{gap}_P(j) \geq Bgap(j)$, and x would have fit. Thus BIN_j must already have received its first non- k -piece z , which by definition is $\text{FILLER}(j)$, and by the fact that it was assigned before x was must have $\text{rank}(z) < \text{rank}(x)$. \square

We are now ready to address ourselves to the problem of proving a version of Inequality (5B) which will suit the needs of our upper bound proofs. Recall that, by Lemma 4.2, in proving that $R[S, t] \leq r$ it is sufficient to consider lists all of whose pieces are bigger than some fixed minimal size. So in what follows we shall assume that $\text{Range}(\text{size}_L) \subseteq (1/N, 1/2]$, for some fixed N .

Lemma 5.4. If $S \in AF$, L a list in decreasing order with $\text{Range}(\text{size}_L) \subseteq (1/N, 1/2]$, and PS is an S -packing of L , then

$$\sum_{x \in \text{BASIC}} w_1(x) \geq \#\#PS - \sum_{k=2}^{N-1} [(k-1)/k].$$

Proof. If BIN_j is a k -bin and contains k k -pieces in PS , then $\sum_{x \in B_{\text{cont}}(j)} w_1(x) = k(1/k) = 1$, so the only non-empty bins in PS which do not contribute at least 1 to $\sum_{x \in \text{BASIC}} w_1(x)$ are those k -bins with fewer than k k -pieces. By Lemma 3.6 there can be at most one of these for each k , $2 \leq k \leq N-1$, since only the rightmost k -bin can fail to have k k -pieces. See Figure 5.2. Since such a bin must have at least one k -piece and hence have $\sum_{x \in B_{\text{cont}}(j)} w_1(x) \geq 1/k$, the result follows. \square

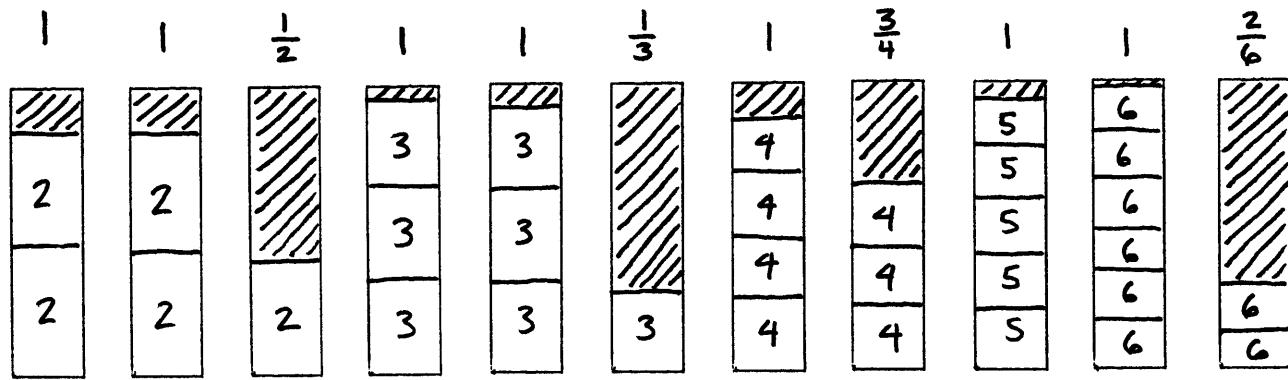
$w_1(B_{cont}(j)) =$


FIGURE 5.2. Packing PS with only BASIC pieces drawn in, $N = 7$.

Thus if we were to define $w_1[D] = \sum_{x \in D} w_1(x)$, w_1 itself would satisfy (5B) with $K = \sum_{k=2}^{N-1} [(k-1)/k]$. However,

$$\begin{aligned} w_1[\text{PIECES}(L)] &= w_1[\text{BASIC}] + w_1[\text{SURPLUS}] \\ &\geq \#PS - \sum_{k=2}^{N-1} [(k-1)/k] + w_1[\text{SURPLUS}], \end{aligned}$$

and so w_1 by itself might yield too large a value for (5A) to hold. This in fact turns out to be the case for the bounds we wish to prove, and it is for this reason that we have introduced w_2 and discounting.

LEMMA 5.5. Suppose S , L , and PS satisfy the hypotheses of the preceding Lemma and in addition $S \in \text{AAF}$. Then if Π is any partition of $\text{PIECES}(L)$ into 1- and 2-element sets,

$$w_{12}(\Pi) \geq \#PS - (N-2).$$

Proof. By Lemma 5.4 we have

$$\begin{aligned} w_{12}(\Pi) &= w_1[\text{PIECES}(L)] - \text{DISCOUNT}(\Pi_2) \\ &= w_1[\text{BASIC}] + w_1[\text{SURPLUS}] - \text{DISCOUNT}(\Pi_2) \\ &\geq \#PS - \sum_{k=2}^{N-1} [(k-1)/k] \\ &\quad + w_1[\text{SURPLUS}] - \text{DISCOUNT}(\Pi_2). \end{aligned}$$

Thus it will be sufficient to show that

$$(5.5A) \text{ DISCOUNT}(\Pi_2) \leq w_1[\text{SURPLUS}] + \sum_{k=2}^{N-1} [1/k].$$

Let $R_k \subseteq \Pi_2$ be the set of all pairs in Π_2 obeying Discounting Relation k , $2 \leq k \leq N-2$. (No pairs can obey Relation k for $k > N-2$, since the second component could be at most and N -piece, and by hypothesis L contains no pieces that small.) In addition, let $R_k\text{-PIECES} = \bigcup_{(x,y) \in R_k} \{x, y\}$. Then we have that $\{R_k\text{-PIECES}: 2 \leq k \leq N-2\}$ is a partition of $\Pi_2\text{-PIECES} = \bigcup_{(x,y) \in \Pi_2} \{x, y\}$.

For an intuition as to why (5.5A) should hold, Consider a particular R_k , and assume that all the first components are in BASIC, and that the pairs are labeled (x_1, y_1) thru (x_n, y_n) in order of increasing rank of their second components.

Then by Lemma 5.2 each y_{ki} , $1 \leq i \leq \lfloor n/k \rfloor$, has a unique k -bin into which it will fit, and by Lemma 5.3, either y_{ki} itself is in SURPLUS, or else the k -bin contains an element of SURPLUS at least as large as y_{ki} . If we can show that these elements of SURPLUS are unique, we would thus have a total contribution to $w_1[\text{SURPLUS}]$ of at least

$$\sum_{i=1}^{\lfloor n/k \rfloor} w_1(y_{ki}) \sim (1/k) \sum_{i=1}^n w_1(y_i) = \sum_{i=1}^n [1/k] w_1(y_i) = \text{DISCOUNT}(R_k).$$

The details and precise calculation of the edge effects will unfortunately be fairly complicated. What we will do is set up a billing function BILL which will assign the individual discounts to specific elements of SURPLUS. The difficulty will lie in proving that BILL is 1-1, and in adding in extra charges to take care of special cases without allowing any element of SURPLUS to be billed for more than its own weight under w_1 .

We shall define BILL inductively with the aid of an auxiliary function POINTER, starting first with the set R_2 and proceeding through R_{N-2} . In addition, each R_k will itself be processed inductively, after an initial bout of preprocessing, and only selected representatives of R_k will actually be added to Domain(BILL). We shall start off the induction with R_2 :

OVERALL INITIALIZATION:

Set $K = 2$, $I = 1$. For each non-empty BIN_j in PS, set $\text{POINTER}_{2,1}(j) = j$. Let $\text{Domain}(\text{BILL}_{2,1}) = \emptyset$.

R_2 -INITIALIZATION:

Label the pairs in R_2 $(x_1, y_1), \dots, (x_n, y_n)$ so that $\text{rank}(y_1) < \dots < \text{rank}(y_n)$.

R₂-LOOP:

If $I > \lfloor n/2 \rfloor$, go to R₂-FINALIZATION. Otherwise,
 for all $j \in \text{Domain}(\text{POINTER}_{2,I})$,
 set $\text{POINTER}_{2,I+1}(j) = \text{POINTER}_{2,I}(j)$, and
 for all $y \in \text{Domain}(\text{BILL}_{2,I})$,
 set $\text{BILL}_{2,I+1}(y) = \text{BILL}_{2,I}(y)$.

CASE A: If $y_{2I} \in \text{BASIC}$, set $\text{BILL}_{2,I+1}(y_{2I}) = \text{FILLER}(\text{POINTER}_{2,I}(J[2,I]))$ and go to ENDLOOP.

CASE B: If $y_{2I} \in \text{SURPLUS}$, set $\text{BILL}_{2,I+1}(y_{2I}) = y_{2I}$. If in addition $y_{2I} = \text{FILLER}(j')$ for some $j' \in \text{Range}(\text{POINTER}_{2,I})$ and $j'' = \text{POINTER}_{2,I}^{-1}(j')$, reset $\text{POINTER}_{2,I+1}(j'') = \text{POINTER}_{2,I}(J[2,I])$. Go to ENDLOOP.

ENDLOOP: Delete $J[2,I]$ from $\text{Domain}(\text{POINTER}_{2,I+1})$,
 set $I = I + 1$, and go to R₂-LOOP.

R₂-FINALIZATION:

For all $y \in \text{Domain}(\text{BILL}_{2,I})$,
 set $\text{BILL}_{3,I}(y) = \text{BILL}_{2,I}(y)$.
 For all $j \in \text{Domain}(\text{POINTER}_{2,I})$,
 set $\text{POINTER}_{3,I}(j) = \text{POINTER}_{2,I}(j)$.

For all y such that $(x, y) \in R_2$,

set $SUB_2(y) = y$.

For all $z \in \text{Range}(\text{BILL}_{3,1})$,

set $BALANCE_2(z) = w_1(z) - w_1(\text{BILL}_{3,1}^{-1}(z))$,

For all $z \in \text{SURPLUS} - \text{Range}(\text{BILL}_{3,1})$,

set $BALANCE_2(z) = w_1(z)$,

For all $z \in \text{BASIC}$, set $BALANCE_2(z) = 0$.

For all y such that $(x, y) \in R_2$,

set $YCHARGE_2(y) = 0$.

Set $\text{REP}_2 = \{y_{2i} : 1 \leq i \leq \lfloor n/2 \rfloor\}$,

$\text{NORM}_2 = R_2$,

$\text{SPEC}_2 = \text{Domain}(XCHARGE_2) = \emptyset$,

$I = 1$, $K = 3$,

And go to R_K -INITIALIZATION.

REP , NORM , SPEC , EXTRA , SUB , XCHARGE , YCHARGE , and BALANCE all are introduced at this point to fulfill the requirements of our later inductions. However, the only induction hypotheses we need initially are the following, which can clearly be seen to hold for $K = 2$, $I = 1$ (more hypotheses will be introduced later, but for now they could only prove distracting):

(B1) $\text{Domain}(\text{BILL}_{2,I}) = \{y_{zh} : 1 \leq h < I\}$, $\text{Range}(\text{BILL}_{2,I}) \subseteq \text{SURPLUS}$, and if $I > 1$ and $y \in \text{Domain}(\text{BILL}_{2,I-1})$, then $\text{BILL}_{2,I}(y) = \text{BILL}_{2,I-1}(y)$.

(B2) $\text{BILL}_{2,I}$ is 1-1.

(B3) If $y \in \text{Domain}(\text{BILL}_{2,I})$, $\text{rank}(\text{BILL}_{2,I}(y)) \leq \text{rank}(y)$.

(B4) If $y \in \text{Domain}(\text{BILL}_{2,I})$ and $\text{BILL}_{2,I}(y) \neq y$, then $y \in \text{BASIC}$.

(P1) $\{j[k,i] : k > K, \text{ or } k = K \text{ and } i \geq I\} \subseteq \text{Domain}(\text{POINTER}_{K,I})$.

(P2) $\text{POINTER}_{K,I}$ is 1-1.

(P3) If $j \in \text{Domain}(\text{POINTER}_{K,I})$ and $z = \text{FILLER}(\text{POINTER}_{K,I}(j))$, then $z \notin \text{Range}(\text{BILL}_{K,I})$.

(P4) If $j \in \text{Domain}(\text{POINTER}_{K,I})$, and BIN_j is a k -bin in PS, then for any k' -piece $x \in \text{BASIC}$, $k' > k$, $\text{size}(x) \leq \text{Bgap}(j) \implies \text{size}(x) \leq \text{Bgap}(\text{POINTER}_{K,I}(j))$.

POINTER is used to indicate where to look for a piece to serve as BILL(y) when y is not in SURPLUS and hence cannot itself be BILL(y). Hypotheses (P1) thru (P4) serve mainly to insure that this plan will work properly. (B1) thru (B4) help to show that BILL is actually doing its job, as we shall see later.

CLAIM 5.5.1. The above 8 hypotheses hold for $K = 2, I = 1$.

Proof. By inspection. \square

CLAIM 5.5.2. If the above 8 hypotheses hold for $K = 2, I = i \leq \lfloor n/2 \rfloor$, then they hold for $K = 2, I = i+1$.

Proof. (B1) held for $I = i$, so to say that it holds for $I = i+1$ is essentially to say that $BILL_{2,i+1}(y_{2i})$ is a well-defined element of SURPLUS. In CASE B this is obvious, since $BILL_{2,i+1}(y_{2i}) = y_{2i} \in SURPLUS$. Moreover, since $\text{rank}(y_{2i}) = \text{rank}(y_{2i})$, (B3) also will remain valid in this case. In CASE A we have that $\text{size}(y_{2i}) \leq \text{Bgap}(J[2,i])$ by Lemma 5.2, and moreover, y_{2i} is a k' -piece for some $k' > 2$ by the definition of Relation 2. Thus by (P4) for i , $\text{size}(y_{2i}) \leq \text{Bgap}(\text{POINTER}_{2,i}(J[2,i]))$, and so by Lemma 5.3 there exists a $z = \text{FILLER}(\text{POINTER}_{2,i}(J[2,i]))$ which is in SURPLUS and can be

assigned as $\text{BILL}_{z,i+1}(y_{zi})$, and moreover, $\text{rank}(z) < \text{rank}(y_{zi})$, so in this case (B3) will again remain valid. Thus both (B1) and (B3) hold for $I = i+1$.

(B2) held for $I = i$, so the only way it can fail for $i+1$ is if $\text{BILL}_{z,i+1}(y_{zi}) \in \text{Range}(\text{BILL}_{z,i})$. This cannot occur in CASE A by (P3) for i . It cannot occur in CASE B, since by (B1') and (B3') for i we would thus have $\text{rank}(y_{zi}) \leq \text{rank}(y_{zh})$ for some $h < i$, which is contrary to our original labeling of the y_i 's.

(B4) holds for all $y \in \text{Domain}(\text{BILL}_{z,i})$ by (B4) for i and (B1) for $i+1$. It holds for y_{zi} since $y_{zi} \in \text{SURPLUS} \implies \text{BILL}_{z,i+1}(y_{zi}) = y_{zi}$.

Since (P1) held for $I = i$, and $J[2,i]$ is deleted from the domain of $\text{POINTER}_{z,i+1}$, the only way (P1) could fail for $i+1$ would be if the resetting of $\text{POINTER}_{z,i+1}(j'')$ to $\text{POINTER}_{z,i}(J[2,i])$ in CASE B were an undefined process. But there is a $\text{BIN}_{J[2,i]}$ by Lemma 5.2, so by (P1) for i , $\text{POINTER}_{z,i}(J[2,i])$ is defined, and the resetting does work.

(P2) held for $I = i$, and so the only way it could be violated for $I = i+1$ would be by some j for which $\text{POINTER}_{z,i+1}(j) \neq \text{POINTER}_{z,i}(j)$. This only can occur in CASE B, where $\text{POINTER}_{z,i+1}(j'')$ is reset to $\text{POINTER}_{z,i}(J[2,i])$. However, since $J[2,i]$ is subsequently deleted from $\text{Domain}(\text{POINTER}_{z,i+1})$, $\text{POINTER}_{z,i+1}$ remains 1-1.

(P3) can only be violated if $z = \text{BILL}_{z,i+1}(y_{zi})$, since

$\text{Range}(\text{POINTER}_{z,i+1}) \subseteq \text{Range}(\text{POINTER}_{z,i})$. In CASE A, this would mean that $z = \text{FILLER}(\text{POINTER}_{z,i}(J[2,i]))$, but since in this case no POINTER values are reset and $J[2,i]$ is deleted from $\text{Range}(\text{POINTER}_{z,i+1})$, the fact that by (P1) $\text{POINTER}_{z,i+1}$ is 1-1 implies that $\text{FILLER}^{-1}(z)$ is no longer in $\text{Range}(\text{POINTER}_{z,i+1})$. In CASE B, we would have $z = y_{zi}$. But in this case $\text{FILLER}^{-1}(z)$ is replaced as $\text{POINTER}_{z,i+1}(j'')$ by $\text{POINTER}_{z,i}(J[2,i])$, and hence deleted from $\text{Range}(\text{POINTER}_{z,i+1})$.

(P4) can only be violated in CASE B, when $\text{POINTER}_{z,i+1}(j'') = j'$ is reset to $\text{POINTER}_{z,i}(J[2,i])$. So suppose $\text{BIN}_{j''}$ is a k'' -bin, and $x \in \text{BASIC}$ is a k' -piece for some $k' > k''$, with $\text{size}(x) \leq \text{Bgap}(j'')$. Then by (P4) for i , $\text{size}(x) \leq \text{Bgap}(j')$, and by Lemma 5.3, since $y_{zi} = \text{FILLER}(j')$, $\text{rank}(y_{zi}) < \text{rank}(x)$, and so $\text{size}(y_{zi}) \geq \text{size}(x)$. Moreover, by Lemma 5.3, $\text{size}(y_{zi}) \leq \text{BGAP}(J[2,i])$, and so $\text{size}(x) \leq \text{Bgap}(J[2,i])$. In addition, by definition of FILLER , y_{zi} is not a 2-piece and so neither is x . Thus, since $\text{BIN}_{J[2,i]}$ is a 2-bin, (P4) for i again applies, and $\text{size}(x) \leq \text{Bgap}(\text{POINTER}_{z,i}(J[2,i]))$. And since $\text{POINTER}_{z,i+1}(j'') = \text{POINTER}_{z,i}(J[2,i])$, (P4) continues to hold for $i+1$.

Thus all 8 hypotheses continue to hold for $i = i+1$ and Claim 5.5.2 is proven. \square

Thus by Claims 5.5.1 and 5.5.2 and induction, the 8 hypotheses hold for $K = 2$, $I = \lfloor n/2 \rfloor + 1$. Moreover, with REP_2 and BALANCE_2 defined as in R_2 -FINALIZATION, we have the following:

$$\begin{aligned}
 \text{CLAIM 5.5.3. } & \sum_{y \in \text{REP}_2} [w_1(\text{BILL}_{3,1}(y)) - \text{BALANCE}_2(\text{BILL}_{3,1}(y))] \\
 &= \sum_{y \in \text{REP}_2} w_1(y) \geq \sum_{l=1}^{\lfloor n/2 \rfloor} (1/2)[w_1(y_{2l}) + w_1(y_{2l+1})] \\
 &\geq \sum_{l=1}^n (1/2)w_1(y_l) - (1/2)w_1(y_1) \\
 &\geq \text{DISCOUNT}(R_2) - (1/2)(1/3).
 \end{aligned}$$

Our next claim will be that the following 15 hypotheses, which we shall use for our outer induction over K , hold for $K = 2$:

- (C1) For $2 \leq k' \leq K$,
- $\text{REP}_{k'} \subseteq \{y : (x, y) \in \text{NORM}_{k'}\}$.
- (C2) For $2 \leq k' \leq K$,
- $\text{SPEC}_{k'} \cup \text{NORM}_{k'} = R_{k'}$ is a disjoint union.

(C3) $\text{Domain}(\text{XCHARGE}_K) = \{x: (x, y) \in \text{SPEC}_k' \text{ for some } 2 \leq k' \leq K\} \subseteq \text{SURPLUS}$,
 and if $K > 2$ and $x \notin \text{Domain}(\text{XCHARGE}_{K-1})$, then
 $\text{XCHARGE}_K(x) = \text{XCHARGE}_{K-1}(x)$.

(C4) If $x \in \text{Domain}(\text{XCHARGE}_K)$ and x is a k'' -piece,
 then $\text{XCHARGE}_K(x) = 1/[(k'')(k''+1)]$.

(C5) $\text{DOMAIN}(\text{YCHARGE}_K) = \{y: (x, y) \in \text{NORM}_k' \text{ for some } 2 \leq k' \leq K\}$, and
 if $K > 2$ and $y \notin \text{Domain}(\text{YCHARGE}_{K-1})$, then
 $\text{YCHARGE}_K(y) = \text{YCHARGE}_{K-1}(y)$.

(C6) $\text{Domain}(\text{SUB}_K) = \{z \in \{x, y\}: (x, y) \in \text{NORM}_k' \text{ for some } 2 \leq k' \leq K\}$,
 and if $K > 2$ and $z \notin \text{Domain}(\text{SUB}_{K-1})$, then
 $\text{SUB}_K(z) = \text{SUB}_{K-1}(z)$.

(C7) $\text{Range}(\text{SUB}_K) \subseteq \bigcup_{k'=2}^K R_{k'} \text{-PIECES.}$

(C8) If $x \in \text{Domain}(\text{SUB}_K)$, then $\text{rank}(x) \leq \text{rank}(\text{SUB}_K(x))$.

(C9) $\text{BILL}_{k+1,1}$ is a 1-1 map from
 $\{y: (x,y) \in \text{REP}_k' \text{ for some } 2 \leq k' \leq K\}$ to SURPLUS
and if $K > 2$ and $y \notin \text{Domain}(\text{BILL}_{k,1})$, then
 $\text{BILL}_{k+1,1}(y) = \text{BILL}_{k,1}(y)$.

(C10) If $x \neq y \in \text{Domain}(\text{BILL}_{k+1,1})$ and
 $\text{BILL}_{k+1,1}(x), \text{BILL}_{k+1,1}(y) \in \bigcup_{k'=k+1}^{N-2} R_{k'}\text{-PIECES},$
then $\text{SUB}_k(x) \neq \text{SUB}_k(y)$.

(C11) If $x \in \text{Range}(\text{BILL}_{k+1,1})$, then
 $\text{BALANCE}_k(x) = w_1(x) - w_1(\text{SUB}_k^{-1}(\text{BILL}_{k+1,1}^{-1}(x)))$,
If $x \in \text{SURPLUS} - \text{Range}(\text{BILL}_{k+1,1})$,
then $\text{BALANCE}_k(x) = w_1(x)$,
If $x \in \text{BASIC}$, $\text{BALANCE}_k(x) = 0$.

(C12) If $x \in \text{Domain}(\text{XCHARGE}_k)$, then
 $\text{XCHARGE}_k(x) \leq \text{BALANCE}_k(x)$.

(C13) If $y \in \text{Domain}(\text{YCHARGE}_k)$ and $\text{YCHARGE}_k(y) > 0$,
then $y \in \text{Range}(\text{BILL}_{k+1,1})$ and $\text{YCHARGE}_k(y) = \text{BALANCE}_k(y)$.

(C14) For each k' , $2 \leq k' \leq K$,

$$\begin{aligned} & \sum_{y \in \text{REP}_{k'}} [w_1(\text{BILL}_{k+1,1}(y)) - \text{BALANCE}_k(\text{BILL}_{k+1,1}(y))] \\ & + (1/k') \sum_{(x,y) \in \text{NORM}_{k'}} \text{YCHARGE}_k(y) \\ & \geq \text{DISCOUNT}(\text{NORM}_{k'}) - 1/k'. \end{aligned}$$

(C15) For each k' , $2 \leq k' \leq K$,

$$\sum_{(x,y) \in \text{SPEC}_{k'}} \text{XCHARGE}_k(x) \geq \text{DISCOUNT}(\text{SPEC}_{k'}).$$

CLAIM 5.5.4. Hypotheses (C1) thru (C15) hold for $K = 2$.

Proof. All but (C14) are immediate consequences of the definitions of the sets and maps in question, which are for the most part trivial for $K = 2$. (C14) follows from Claim 5.5.3. \square

As we shall see later, if hypotheses (C1) thru (C3), (C5), (C9), and (C11) thru (C15) hold for $K = N-2$, Lemma 5.5 will follow. The remaining 5 hypotheses are needed for the induction over K to work. In addition, we will need to know that (P1) thru (P4), and the following revised versions of (B1) thru (B4) hold:

(B1') $\text{Domain}(\text{BILL}_{K,1}) \cap \{y: (x,y) \in R_K\} =$
 $\{y_{Kh}: 1 \leq h < I\}$, $\text{Range}(\text{BILL}_{K,I}) \subseteq \text{SURPLUS}$
 (where the labeling of the y_{Kh} 's is as specified in the
 R_K -INITIALIZATION), and if $I > 2$ and $y \in$
 $\text{Domain}(\text{BILL}_{K,I-1})$, then $\text{BILL}_{K,I}(y) = \text{BILL}_{K,I-1}(y)$.

(B2') $\text{BILL}_{K,I}$ is 1-1.

(B3') If $y \in \text{Domain}(\text{BILL}_{K,I})$, then
 $\text{rank}(\text{BILL}_{K,I}(y)) \leq \text{rank}(\text{SUB}_K(y))$.

(B4') If $y \in \text{Domain}(\text{BILL}_{K,I})$ and $\text{BILL}_{K,I}(y) \neq$
 $\text{SUB}_K(y)$, then $\text{SUB}_K(y) \in \text{BASIC}$.

CLAIM 5.5.5. Hypotheses (B1') thru (B4'), and (P1) thru
 (P4) all hold for $K = 3$, $I = 1$.

Proof. (B2') follows from (B2) for $K = 2$, $I = \lfloor n/2 \rfloor + 1$.
 (B1') follows from (B1) for $K = 2$, $I = \lfloor n/2 \rfloor + 1$, and the fact
 that R_2 -PIECES and R_3 -PIECES are disjoint. (B3') and (B4')
 follow from (B3), since for $y \in \text{Domain}(\text{BILL}_{3,1})$, $\text{SUB}_2(y) = y$.
 (P1) thru (P4) follow from the same properties for $K = 2$, $I =$
 $\lfloor n/2 \rfloor + 1$. \square

So suppose (C1) thru (C15) hold for $K-1$, $3 \leq K \leq N-2$, and (B1') thru (B4'), (P1) thru (P4) hold for K and $I = 1$. We shall now describe how to process the set of pairs R_K obeying Relation K , so as to insure that the hypotheses will again hold with K incremented by 1.

The general procedure for processing R_K , $K > 2$, is given by the following: First we must take care of the fact that, unlike the case with R_2 , some of the first components of pairs $(x,y) \in R_K$ may not be in BASIC, and so Lemma 5.2 need not apply. So we first preprocess R_K , and here is where the sets $SPEC_K$ and $NORM_K$, and the partial maps, $SUB_K: PIECES(L) \rightarrow PIECES(L)$ and $ZCHARGE_K: PIECES(L) \rightarrow Q$ for $Z \in \{X,Y\}$, are introduced.

R_K -INITIALIZATION:

For each element of the respective domains for $K-1$,

set $XCHARGE_K(x) = XCHARGE_{K-1}(x)$,
 $YCHARGE_K(y) = YCHARGE_{K-1}(y)$, and
 $SUB_K(z) = SUB_{K-1}(z)$.

Set $NORM_K = SPEC_K = \emptyset$.

For each $(x,y) \in R_K$,

If $x \in SURPLUS$ and

If $x \notin Range(BILL_{K,1})$ or

$x \in \text{Range}(\text{BILL}_{K,1})$ and $\text{BALANCE}_{K-1}(x) > 0$,

set $X\text{CHARGE}_K(x) = (1/K)[1/(K+1)]$ and

add (x,y) to SPEC_K .

If $x \in \text{Range}(\text{BILL}_{K,1})$ and $\text{BALANCE}_{K-1}(x) = 0$,

set $\text{SUB}_K(x) = \text{SUB}_{K-1}(\text{BILL}_{K,1}^{-1}(x))$ and

add (x,y) to NORM_K .

If $x \in \text{BASIC}$,

set $\text{SUB}_K(x) = x$ and add (x,y) to NORM_K .

If $(x,y) \in \text{NORM}_K$ and

If $y \in [\text{SURPLUS} - \text{Range}(\text{BILL}_{K,1})] \cup \text{BASIC}$,

set $\text{SUB}_K(y) = y$, $Y\text{CHARGE}_K(y) = 0$.

If $y \in \text{SURPLUS} \cap \text{Range}(\text{BILL}_{K,1})$,

set $\text{SUB}_K(y) = \text{SUB}_{K-1}(\text{BILL}_{K,1}^{-1}(y))$,

$Y\text{CHARGE}_K(y) = \text{BALANCE}_{K-1}(y)$.

Label the pairs in $\text{NORM}_K(x_1, y_1), \dots, (x_m, y_m)$ so

that $\text{rank}(\text{SUB}_K(y_1)) < \dots < \text{rank}(\text{SUB}_K(y_m))$.

Set $\text{REP}_K = \{y_{Ki}: 1 \leq i \leq \lfloor m/K \rfloor\}$.

Note that the definition of $\text{SUB}_K(z)$ as $\text{SUB}_{K-1}(\text{BILL}_{K,1}^{-1}(z))$

works because (C9), (C2), and (C6) hold for $K-1$. When x itself

is not in BASIC , $\text{SUB}_K(x)$ will serve as a stand-in for x in the

NORM_K -PROCESSING, a procedure which BILLS the discounts of pairs

in NORM_K to pieces in SURPLUS as was done in the R_k -PROCESSING above. $\text{SUB}_k(y)$ serves as a stand-in for y in the procedure when y is already in $\text{Range}(\text{BILL})$. Those pairs for which a proper stand-in for x cannot be found are added to SPEC_k , and XCHARGE takes over the job of assigning their discount to an individual piece in SURPLUS. YCHARGE is used when the stand-in for y is not big enough, and assures that the deficiency will itself be assigned to a piece in SURPLUS.

Since REP_k , SUB_k , XCHARGE_k , YCHARGE_k , SPEC_k , and NORM_k will not be tampered with in the remaining processing of R_k , we already have the following trivial consequences of the R_k -INITIALIZATION:

CLAIM 5.5.6. If (C1) thru (C16) hold for $K = k-1$, $3 \leq k \leq N-2$, and (P1) thru (P4), (B1') thru (B3') hold for $K = k$, $I = 1$, then (C1) thru (C8) hold for $K = k$.

Proof. (C1) thru (C6) follow from the specifications of REP_k , NORM_k , SPEC_k , XCHARGE_k , and YCHARGE_k in the R_k -INITIALIZATION, and the facts that they held for $K = k-1$ and all the pieces for which new values were defined were R -pieces, and hence not in their domain for $K = k-1$. (C7) follows since the only pieces added to $\text{Range}(\text{SUB}_k)$ beyond those already in $\text{Range}(\text{SUB}_{k-1})$ are members of pairs from R_k . (C8) is immediate

if $\text{SUB}_k(x) = x$, and if $\text{SUB}_k(x) = \text{SUB}_{k-1}(\text{BILL}_{k,1}^{-1}(x))$, then it follows from (C8) for $k-1$ and (B3') for k , which tell us that $\text{rank}(x) \leq \text{rank}(\text{BILL}_{k,1}^{-1}(x)) \leq \text{rank}(\text{SUB}_{k-1}(\text{BILL}_{k,1}^{-1}(x)))$. \square

The remainder of the processing of R_k is very similar to the processing of R_2 , except that NORM_k is used instead of R_k , and $\text{SUB}_k(y)$ is used as an intermediary between y and $\text{BILL}(y)$:

NORM_k -LOOP:

If $I > \lfloor m/K \rfloor$, go to R_k -FINALIZATION. Otherwise,
 for all $j \in \text{Domain}(\text{POINTER}_{k,I})$,
 set $\text{POINTER}_{k,I+1}(j) = \text{POINTER}_{k,I}(j)$, and
 for all $y \in \text{Domain}(\text{BILL}_{k,I})$,
 set $\text{BILL}_{k,I+1}(y) = \text{BILL}_{k,I}(y)$.

CASE A: If $\text{SUB}_k(y_{kI}) \in \text{BASIC}$, set
 $\text{BILL}_{k,I+1}(y_{kI}) = \text{FILLER}(\text{POINTER}_{k,I}(J[K,I]))$,
 and go to ENDLOOP.

CASE B: If $\text{SUB}_k(y_{kI}) \in \text{SURPLUS}$, set
 $\text{BILL}_{k,I+1}(y_{kI}) = \text{SUB}_k(y_{kI})$. If, in addition,
 $\text{SUB}_k(y_{kI}) = \text{FILLER}(j')$ for some $j' \in$
 $\text{Range}(\text{POINTER}_{k,I})$ and $j'' = \text{POINTER}_{k,I}^{-1}(j')$, reset
 $\text{POINTER}_{k,I+1}(j'') = \text{POINTER}_{k,I}(J[K,I])$. Go to ENDLOOP.

ENDLOOP: Delete $j[k, I]$ from Domain(PINTER $_{k, I+1}$),
 set $I = I+1$, and go to NORM $_k$ -LOOP.

 R_k -FINALIZATION:

For all $y \in$ Domain(BILL $_{k, I}$),
 set BILL $_{k+1, I}(y) =$ BILL $_{k, I}(y)$.

For all $j \in$ Domain(PINTER $_{k, I}$),
 set PINTER $_{k+1, I}(j) =$ PINTER $_{k, I}(j)$.

Define BALANCE $_k$: PIECES(L) ---> \mathbb{Q} , so that (C11) holds for K.

Set $I = 1$, $K = K+1$.

If $K \geq N-1$, halt, else go to R_k -INITIALIZATION.

Our first observation will allow us to use Lemma 5.2 in our upcoming inner induction over I for fixed K, in a fashion analogous to its use in the proof of Claim 5.5.2:

CLAIM 5.5.7. If hypotheses (C1) thru (C15) hold for $K = k-1$, $3 \leq k \leq N-2$, and (P1) thru (P4), (B1') thru (B4') hold for $K = k$ and $I = 1$, then $\{(SUB_k(x_i), SUB_k(y_i)): (x_i, y_i) \in NORM_k\}$ as labeled in the R_k -INITIALIZATION, is a set of disjoint pairs obeying Relation k , such that $\text{rank}(SUB_k(y_1)) < \dots < \text{rank}(SUB_k(y_m))$, each $SUB_k(x_i) \in \text{BASIC}$, and no $SUB_k(y_i) \in \text{Range}(\text{BILL}_{k, I})$.

Proof. To show disjointness, let $\text{SUB}_k(z_i)$ and $\text{SUB}_k(z_j)$ be two distinct members of pairs from the set. If they equal z_i and z_j respectively, then they differ because the pairs in R_k and hence in NORM_k are all disjoint. If they are $\text{SUB}_{k-1}(\text{BILL}_{k,1}^{-1}(z_i))$ and $\text{SUB}_{k-1}(\text{BILL}_{k,1}^{-1}(z_j))$ respectively, then they differ by (C10) since $\text{BILL}_{k,1}$ is 1-1 by (C9). If one is z_i and the other $\text{SUB}_{k-1}(\text{BILL}_{k,1}^{-1}(z_j))$ or vice versa, then they differ because no element of R_k -PIECES is in $\text{Range}(\text{SUB}_{k-1})$ by (C7).

Now $\text{size}(\text{SUB}_k(z_i)) \leq \text{size}(z_i)$, since by Claim 5.5.6, (C8) holds for k . Thus $k \cdot \text{size}(\text{SUB}_k(x_i)) + \text{size}(\text{SUB}_k(y_i)) \leq 1$, by the fact that Relation k held for (x_i, y_i) . To show that $(\text{SUB}_k(x_i), \text{SUB}_k(y_i))$ obeys Relation k , all that remains is to show that $\text{SUB}_k(x_i)$ is a k -piece. There are two cases, depending on how (x_i, y_i) was assigned to NORM_k :

Case X1. $x_i \in \text{BASIC}$. Then $\text{SUB}_k(x_i) = x_i$ and is a k -piece because x_i is. Moreover, note that we also have $\text{SUB}_k(x_i) \in \text{BASIC}$.

Case X2. $x_i \in \text{Range}(\text{BILL}_{k,1})$ and $\text{BALANCE}_{k-1}(x_i) = 0$. Thus, by (C11) for $k-1$ and the definition of w_1 we have that $\text{SUB}_k(x_i) = \text{SUB}_{k-1}(\text{BILL}_{k,1}^{-1}(x_i))$ is the same type of piece as x_i ,

and hence a k -piece. Moreover, since $\text{BILL}_{k,1}^{-1}(x_i) \in \text{Domain}(\text{SUB}_{k-1}) \cap \text{Domain}(\text{BILL}_{k,1})$, and since $\text{BILL}_{k,1}^{-1}(\text{BILL}_{k,1}^{-1}(x_i)) = x_i \in R_k$ -PIECES, and $\text{SUB}_{k-1}^{-1}(\text{BILL}_{k,1}^{-1}(x_i)) \in \bigcup_{k'=2}^{k-1} R_{k'}\text{-PIECES}$, by (C7) these two must differ, and so by (C10) the latter, which is also $\text{SUB}_k(x_i)$, is in BASIC.

Thus each $(\text{SUB}_k(x_i), \text{SUB}_k(y_i))$ obeys Relation k and has its first component in BASIC. All that remains is to show that the second component is not in $\text{Range}(\text{BILL}_{k,1})$. Again there are two cases, depending on how $\text{SUB}_k(y_i)$ was defined:

Case Y1. If $y_i \in \text{BASIC}$ or $\text{SURPLUS} - \text{Range}(\text{BILL}_{k,1})$, $\text{SUB}_k(y_i) = y_i$ and the result is immediate, since $\text{Range}(\text{BILL}_{k,1}) \cap \text{BASIC} = \emptyset$ by (C1) for $K = k-1$.

Case Y2. If $y_i \in \text{SURPLUS} \cap \text{Range}(\text{BILL}_{k,1})$, then $\text{SUB}_k(y_i) \in \text{BASIC}$ by the same argument as in Case X2, and again cannot be in $\text{Range}(\text{BILL}_{k,1})$.

Thus all the conclusions of Claim 5.5.7 are now verified. \square

CLAIM 5.5.8. Suppose (C1) thru (C15) hold for $K = k-1$, $3 \leq k \leq N-2$, and (B1') thru (B4'), (P1) thru (P4) hold for $K = k$, $I = i$, $1 \leq i \leq \lfloor m/k \rfloor$. Then the latter 8 hypotheses also hold for

$K = k, I = i+1.$

Proof. This proof is very similar to that of Claim 5.5.2.

(B1') held for $I = i$, so to say that it holds for $I = i+1$ is essentially to say that $BILL_{k,i+1}(y_{ki})$ is a well-defined element of SURPLUS. [It is clear that y_{ki} is not in $\text{Domain}(BILL_{k,i})$.] In CASE B this is obvious, since $BILL_{k,i+1}(y_{ki}) = \text{SUB}_k(y_{ki}) \in$ SURPLUS. Moreover, since $\text{rank}(\text{SUB}_k(y_{ki}))$ equals itself, (B3') also will remain valid in this case. In CASE A we have that $\text{size}(\text{SUB}_k(y_{ki})) \leq \text{Bgap}(J[k,i])$ by Lemma 5.2, and moreover, $\text{SUB}_k(y_{ki})$ is a k' -piece for some $k' > k$ by the definition of Relation k . Thus by (P4) for i , $\text{size}(\text{SUB}_k(y_{ki})) \leq \text{Bgap}(\text{POINTER}_{k,i}(J[k,i]))$, and so by Lemma 5.3 there exists a $z = \text{FILLER}(\text{POINTER}_{k,i}(J[k,i]))$ which is an element of SURPLUS and can be assigned as $BILL_{k,i+1}(y_{ki})$. Moreover, $\text{rank}(z) < \text{rank}(\text{SUB}_k(y_{ki}))$, so in this case (B3') will again remain valid. Thus both (B1') and (B3') hold for $I = i+1$.

(B2') held for $I = i$, so the only way it can fail for $i+1$ is if $BILL_{k,i+1}(y_{ki}) \in \text{Range}(BILL_{k,i})$. This cannot occur in CASE A by (P3) for i . It cannot occur in CASE B, since by Claim 5.5.7 $\text{SUB}_k(y_{ki})$ is not in $\text{Range}(BILL_{k,1})$, and so by (B1') it cannot be in $\text{Range}(BILL_{k,i})$ unless it is $BILL_{k,i}(y_{kh})$ for some h , $1 \leq h < i$. This is impossible since by (B3') for i we would have to have $\text{rank}(\text{SUB}_k(y_{ki})) < \text{rank}(\text{SUB}_k(y_{kh}))$, which would be

contrary to our original labeling of the y_i 's.

(B4') holds for all $y \in \text{Domain}(\text{BILL}_{k,i})$ by (B4') for i and (B1') for $i+1$. It holds for y_{ki} since $\text{SUB}_k(y_{ki}) \in \text{SURPLUS} \implies \text{BILL}_{k,i+1}(y_{ki}) = \text{SUB}_k(y_{ki})$.

(P1) thru (P4) are proved exactly as in Claim 5.5.2, except that "2" is replaced by "k" throughout. \square

CLAIM 5.5.9. If hypotheses (C1) thru (C15) hold for $K = k-1$, $3 \leq k \leq N-2$, and (B1') thru (B4'), (P1) thru (P4) hold for $K = k$, $I = 1$, then the latter 8 hypotheses hold for $K = k+1$, $I = 1$.

Proof. By Claim 5.5.8 and induction, the latter 8 hold for $K = k$, $I = \lfloor m/k \rfloor + 1$. They thus hold for $K = k+1$, $I = 1$, since both POINTER and BILL remain the same functions despite the change of indices, and $R_{k+1}-\text{PIECES} \cap \text{Domain}(\text{BILL}_{k+1,1}) = \emptyset$ by (C9) for $K = k-1$ and (B1') for $K = k$, $I = \lfloor m/k \rfloor + 1$.

CLAIM 5.5.10. If hypotheses (C1) thru (C15) hold for $K = k-1$, $3 \leq k \leq N-2$, and (B1') thru (B4'), (P1) thru (P4) hold for $K = k$, $I = 1$, then the first 15 hold for $K = k$.

Proof. We shall proceed down the list, recalling that we already know that (C1) thru (C8) hold by Claim 5.5.6.

(C9). $\text{BILL}_{k+1,1}$ is 1-1 and has the proper range by (B1') for $K = k+1$, $I = 1$. If $y \in \text{Domain}(\text{BILL}_{k,1})$, then by a simple induction on (B1') for $I = 1$ thru $\lfloor m/k \rfloor + 1$, $\text{BILL}_{k+1,1}(y) = \text{BILL}_{k,1}(y)$. Hence by (C9) for $K = k-1$ and (B1') for $K = k$, $I = \lfloor m/k \rfloor + 1$, $\text{Domain}(\text{BILL}_{k+1,1}) = \{y : (x,y) \in \text{REP}_k \text{ for some } 2 \leq k' \leq k\}$, so (C9) holds for k .

(C10). Suppose $x \neq y \in \text{Domain}(\text{BILL}_{k+1,1})$ and $\text{BILL}_{k+1,1}(x), \text{BILL}_{k+1,1}(y) \in \bigcup_{k'=k+1}^{N-2} R_{k'}\text{-PIECES}$. We must show that $\text{SUB}_k(x) \neq \text{SUB}_k(y)$. If both x and $y \in \text{Domain}(\text{BILL}_{k,1})$ then by (C10) for $K = k-1$, and (C6) and (C9) for $K = k$, the result is immediate. If both are in $\text{Domain}(\text{BILL}_{k+1,1}) - \text{Domain}(\text{BILL}_{k,1}) = \text{REP}_k$, then $\text{SUB}_k(x) \neq \text{SUB}_k(y)$ by Claim 5.5.7.

If $x \in \text{Domain}(\text{BILL}_{k,1})$ and $y \in \text{REP}_k$ (or vice versa), then $\text{SUB}_k(y) = y$ would imply $\text{SUB}_k(y) \neq \text{SUB}_k(x)$, since the latter cannot be an R_k -piece by (C7) for $K = k$, and the former must be one by (C1) and (C2) for $K = k$. Thus the only problem would be if $\text{SUB}_k(y) = \text{SUB}_{k-1}(\text{BILL}_{k,1}^{-1}(y))$. But then $\text{BILL}_{k,1}(\text{BILL}_{k,1}^{-1}(y)) = y \in R_k\text{-PIECES}$, so by (C10) for $K = k-1$, $\text{SUB}_k(y) = \text{SUB}_{k-1}(\text{BILL}_{k,1}^{-1}(y)) = \text{SUB}_k(x) = \text{SUB}_{k-1}(x)$ would imply that $x = \text{BILL}_{k,1}^{-1}(y)$. Hence $\text{BILL}_{k+1,1}(x) = y \notin \bigcup_{k'=k+1}^{N-2} R_{k'}\text{-PIECES}$, a

contradiction of our hypothesis. So $\text{SUB}_k(x) \neq \text{SUB}_k(y)$.

(C11). This holds immediately by the definition of BALANCE_k in the R_k -FINALIZATION.

(C12). Suppose $x \in \text{Domain}(\text{XCHARGE}_k)$. We must show that $\text{XCHARGE}_k(x) \leq \text{BALANCE}_k(x)$. If $\text{BALANCE}_k(x) = \text{BALANCE}_{k-1}(x)$, there are two cases: If $x \in \text{Domain}(\text{XCHARGE}_{k-1})$, the result follows by (C12) for $K = k-1$, and (C3) for $K = k$. If on the other hand $x \in \text{Domain}(\text{XCHARGE}_k) - \text{Domain}(\text{XCHARGE}_{k-1})$, then by (C3) for $K = k$ and $k-1$, x is a k -piece, and by (C4) for $K = k$, $\text{XCHARGE}_k(x) = 1/[(k)(k+1)]$. Since $x \in \text{SURPLUS}$, the only problem would be if $\text{BALANCE}(x) < 1/[(k)(k+1)]$, in which case $\text{BALANCE}_k(x) = 0$, since no intermediary values are possible. ($1/k - 1/k'$ for $k' \geq k$ is 0 for $k = k'$, and at least $1/[(k)(k+1)]$ for all $k' > k$.) But since $x \in \text{Domain}(\text{XCHARGE}_k) - \text{Domain}(\text{XCHARGE}_{k-1})$, by (C3) for $K = k$, there is a y such that $(x, y) \in \text{SPEC}_k$, which implies by the R_k -INITIALIZATION that $0 < \text{BALANCE}_{k-1}(x) = \text{BALANCE}_k(x)$. Thus in this case too we have $\text{XCHARGE}_k(x) \leq \text{BALANCE}_k(x)$.

If $\text{BALANCE}_k(x) \neq \text{BALANCE}_{k-1}(x)$, then $x \in \text{Range}(\text{BILL}_{k+1,1}) - \text{Range}(\text{BILL}_{k,1})$ by (C11) for $K = k$ and $k-1$, and so $x = \text{BILL}_{k+1,1}(y)$ for some $y \in \text{REP}_k$. By Claim 5.5.7 $\text{SUB}_k(y)$ is thus the second component of a pair obeying Relation k and hence a

k' -piece for some $k' > k$. But x is a k'' -piece for some $k'' \leq k$ by (C3) and (C2) for $K = k$, so by (C11) $\text{BALANCE}_k(x) = 1/k'' - 1/k' \geq 1/[(k'')(k''+1)] = \text{YCHARGE}_k(x)$ by (C4). This exhausts the possibilities.

(C13). Suppose $y \in \text{Domain}(\text{YCHARGE}_k)$ and $\text{YCHARGE}_k(y) > 0$. We wish to show $\text{YCHARGE}_k(y) = \text{BALANCE}_k(y)$. If $\text{BALANCE}_k(y) \neq \text{BALANCE}_{k-1}(y)$, then by (C11) for $K = k$ and $k-1$, $y \notin \text{Range}(\text{BILL}_{k,1})$. But by the R_k -INITIALIZATION this would mean that $\text{YCHARGE}_k(y) = 0$, contrary to hypothesis. If $\text{BALANCE}_k(y) = \text{BALANCE}_{k-1}(y)$, there are again two cases: If $y \in \text{Domain}(\text{YCHARGE}_{k-1})$, the result follows by (C13) for $K = k-1$ and (C5) for $K = k$. If $y \in \text{Domain}(\text{YCHARGE}_k) - \text{Domain}(\text{YCHARGE}_{k-1})$, then $\text{YCHARGE}_k(y) > 0$ would imply by the R_k -INITIALIZATION that $\text{YCHARGE}_k(y) = \text{BALANCE}_{k-1}(y) = \text{BALANCE}_k(y)$.

(C14). The stated inequality holds for all k' , $2 \leq k' \leq k-1$, by (C14) for $K = k-1$, (C5) and (C9) for $K = k$, and the fact that by (C11) for $K = k$ and $k-1$, $z \in \text{Range}(\text{BILL}_{k,1}) \implies \text{BALANCE}_k(z) = \text{BALANCE}_{k-1}(z)$. Thus we may restrict our attention to the case $k' = k$.

By (C11), for all $y \in \text{REP}_k$, $w_1(\text{BILL}_{k+1,1}(y)) - \text{BALANCE}_k(y) = w_1(\text{SUB}_k(y))$. Thus

$$\begin{aligned}
\sum_{y \in \text{REP}_k} [w_1(\text{BILL}_{k+1,1}(y)) - \text{BALANCE}_k(y)] &= \sum_{y \in \text{REP}_k} w_1[\text{SUB}_k(y)] \\
&\geq \sum_{l=1}^{\lfloor m/k \rfloor} w_1[\text{SUB}_k(y_{ki})] \geq \sum_{l=1}^{\lfloor m/k \rfloor} (1/k) \sum_{h=0}^{k-1} w_1[\text{SUB}_k(y_{ki+h})] \\
&\geq \sum_{l=1}^m (1/k) w_1[\text{SUB}_k(y_i)] - \sum_{l=1}^{k-1} (1/k) w_1[\text{SUB}_k(y_i)] \\
&\geq \sum_{l=1}^m (1/k) w_1[\text{SUB}_k(y_i)] - 1/k,
\end{aligned}$$

since all the $\text{SUB}_k(y_i)$ are second components of pairs obeying Relation k by Claim 5.5.7, and hence have $w_1[\text{SUB}_k(y_i)] \leq 1/(k+1)$, so that $\sum_{l=1}^{k-1} (1/k) w_1[\text{SUB}_k(y_i)] \leq (k-1)(1/k)(1/(k+1)) < 1/k$.

Now let us consider $w_1[\text{SUB}_k(y_i)] + \text{YCHARGE}_k(y_i)$: If $\text{SUB}_k(y_i) = y_i$, then $\text{YCHARGE}_k(y_i) = 0$, and the sum is $w_1(y_i)$. If on the other hand $\text{SUB}_k(y_i) \neq y_i$, then $y_i \in \text{Range}(\text{BILL}_{k,1}) \cap \text{SURPLUS}$, and $\text{YCHARGE}_k(y_i) = \text{BALANCE}_{k-1}(y_i) = w_1(y_i) - w_1(\text{SUB}_{k-1}(\text{BILL}_{k,1}^{-1}(y_i))) = w_1(y_i) - w_1(\text{SUB}_k(y_i))$, so again the sum is $w_1(y_i)$. Thus

$$\begin{aligned}
\sum_{y \in \text{REP}_k} [w_1(\text{BILL}_{k+1,1}(y)) - \text{BALANCE}_k(y)] + \sum_{(x,y) \in \text{NORM}_k} (1/k) \text{YCHARGE}_k(y) \\
&\geq \sum_{i=1}^m (1/k) w_1(\text{SUB}_k(y_i)) + \sum_{l=1}^m (1/k) \text{YCHARGE}_k(y_i) - 1/k
\end{aligned}$$

$$\geq \sum_{k=1}^m (1/k) w_1(y_k) - 1/k = \text{DISCOUNT}(\text{NORM}_k) - 1/k.$$

(C15). The stated inequality holds for all k' , $2 \leq k' \leq k-1$ by (C15) for $K = k-1$ and (C3) for $K = k$. If $(x, y) \in \text{SPEC}_k$, then by definition of Relation k , x is a k -piece, so by (C4) for $K = k$, $\text{XCHARGE}_k(x) = 1/[(k)(k+1)]$. Since this is the maximum possible value for $\text{DISCOUNT}(x, y)$ if (x, y) obey Relation k , we have that $\text{XCHARGE}_k(x) \geq \text{DISCOUNT}(x, y)$, and the desired inequality follows by summation over all (x, y) in SPEC_k .

Thus all 15 hypotheses hold for $K = k$ and Claim 5.5.10 is proven. \square

CLAIM 5.5.11. If (C1) thru (C3), (C5), (C9), and (C11) thru (C15) all hold for $K = N-2$, then

$$(5.5A) \quad \text{DISCOUNT}(\Pi_2) \leq w_1[\text{SURPLUS}] + \sum_{k=2}^{N-1} (1/k).$$

Proof. In the following we shall for convenience drop the subscripts from $\text{BILL}_{N-1,1}$, XCHARGE_{N-2} , YCHARGE_{N-2} , and BALANCE_{N-2} .

Now $\text{DISCOUNT}(\Pi_2) = \sum_{k=2}^{N-2} \text{DISCOUNT}(R_k)$ by the definition of R_k and the fact that if $(x, y) \in \Pi_2$ but obeys no Discounting Relations, $\text{DISCOUNT}(x, y) = 0$. Thus by (C2), (C14), and (C15),

$$\text{DISCOUNT}(\Pi_2) = \sum_{k=2}^{N-2} [\text{DISCOUNT}(\text{NORM}_k) + \text{DISCOUNT}(\text{SPEC}_k)]$$

$$\leq \sum_{k=2}^{N-2} \left[\sum_{y \in \text{REP}_k} [w_1(\text{BILL}(y)) - \text{BALANCE}(\text{BILL}(y))] \right. \\ \left. + (1/k) \sum_{(x,y) \in \text{NORM}_k} \text{YCHARGE}(y) + 1/k \right. \\ \left. + \sum_{(x,y) \in \text{SPEC}_k} \text{XCHARGE}(x) \right]$$

$$\leq \sum_{y \in \text{Range}(\text{BILL})} [w_1(y) - \text{BALANCE}(y)] + \sum_{k=2}^{N-2} (1/k) \\ + \sum_{y \in \text{Domain}(\text{YCHARGE})} \text{YCHARGE}(y) + \sum_{x \in \text{Domain}(\text{XCHARGE})} \text{XCHARGE}(x),$$

since the REP_k 's are all disjoint by (C1) and (C2), similarly for the NORM_k 's and SPEC_k 's by (C2), and because BILL is 1-1 by (C9).

Note also that $\text{Range}(\text{BILL}) \subseteq \text{SURPLUS}$ by (C9), $\text{Domain}(\text{XCHARGE}) \subseteq \text{SURPLUS}$ by (C3), and $\text{Domain}(\text{YCHARGE}) \subseteq \text{Range}(\text{BILL}) \subseteq \text{SURPLUS}$ by (C13). Thus we will be done if we can show that no $z \in \text{SURPLUS}$ contributes more than $w_1(z)$ to the right-hand side of the above inequality, which we shall abbreviate by RHS.

We first observe that $\text{Domain}(\text{YCHARGE}) \cap \text{Domain}(\text{XCHARGE}) = \emptyset$ by (C2), (C3), and (C5). Thus there are only a few cases to

consider. Let $\text{Contrib}(z)$ be the total amount contributed by $z \in \text{SURPLUS}$ to RHS.

If $z \in \text{Domain(YCHARGE)} - \text{Range(BILL)}$, then by (C11) $\text{BALANCE}(z) = w_1(z)$, and since $\text{YCHARGE}(z)$ is either 0 or $\text{BALANCE}(z)$ by (C13), we have $\text{Contrib}(z) = \text{YCHARGE}(z) \leq w_1(z)$.

If $z \in \text{Domain(YCHARGE)} \cap \text{Range(BILL)}$ and if $\text{YCHARGE}(z) = 0$, then $\text{Contrib}(z) = w_1(z) - \text{BALANCE}(z) + 0 \leq w_1(z)$. If on the other hand $\text{YCHARGE}(z) > 0$, then by (C13) it equals $\text{BALANCE}(z)$, so $\text{Contrib}(z) = w_1(z) - \text{BALANCE}(z) + \text{BALANCE}(z) = w_1(z)$.

If $z \in \text{Domain(XCHARGE)} - \text{Range(BILL)}$, then by (C11) and (C12), $\text{Contrib}(z) = \text{XCHARGE}(z) \leq \text{BALANCE}(z) \leq w_1(z)$.

If $z \in \text{Domain(XCHARGE)} \cap \text{Range(BILL)}$, then by (C12) $\text{Contrib}(z) \leq w_1(z) - \text{BALANCE}(z) + \text{BALANCE}(z) = w_1(z)$.

If $z \in \text{Range(BILL)} - \text{Domain(YCHARGE)} - \text{Domain(XCHARGE)}$, then $\text{Contrib}(z) = w_1(z) - \text{BALANCE}(z) \leq w_1(z)$.

Finally, if $z \in \text{SURPLUS} - \text{Range(BILL)} - \text{Domain(YCHARGE)} - \text{Domain(XCHARGE)}$, then $\text{Contrib}(z) = 0$.

Thus no element z of SURPLUS contributes more than $w_1(z)$ to RHS, so $\text{RHS} - \sum_{k=2}^{N-2} (1/k) \leq w_1[\text{SURPLUS}]$, and so

$$\text{DISCOUNT}(\pi_2) \leq w_1[\text{SURPLUS}] - \sum_{k=2}^{N-1} (1/k),$$

and Claim 5.5.11 is proven. \square

Thus by Claims 5.5.4, 5.5.5, 5.5.9, and 5.5.10, induction, and Claim 5.5.11, inequality 5.5A does indeed hold, and so Lemma 5.5 is proven. \square

THEOREM 5.6. If L is in decreasing order with $\text{Range}(\text{size}_L) \subseteq (1/N, 1/2]$ and $S \in \text{AAF}$, then

$$\text{Wd}(L) \geq S(L) - (N-2).$$

Proof. Let PS be an S -packing of L with $\#PS = S(L)$. Then for all partitions Π of L into 1- and 2-element sets, we have by Lemma 5.5 that

$$w_{12}(\Pi) \geq \#PS - (N-2) = S(L) - (N-2).$$

Since $\text{Wd}(L) = \min\{w_{12}(\Pi) : \Pi \text{ is a partition of } L \text{ into 1- and 2-element sets}\}$, the Theorem follows. \square

Remark. Lemma 5.5 and Theorem 5.6 only apply if $S \in \text{AAF}$. We conjecture that by changing the constant from $N-2$ to something like N^2 we can get a similar result for arbitrary $S \in \text{AF}$. The details would probably be mind-boggling, but the basic idea is this: if $\text{size}(y) \leq \text{Bgap}(j)$, for y a k' -piece, BIN_j a k -bin, $k < k'$, then although Lemma 5.3 does not apply and so y could be in BASIC while $\text{FILLER}(j)$ had higher rank, if indeed it

existed at all, still not all of the N pieces in $\{x: \text{rank}(y) \leq \text{rank}(x) \leq \text{rank}(y) + N-1\}$ can be in BASIC unless FILLER(j) does exist and has rank $< \text{rank}(y) + N$, since one of the pieces in the set would have had to go into a bottom position, and hence would have violated the AF Constraint by going to the right of BIN_j , which at the time would have had $\text{gap}_p(j) = \text{Bgap}(j)$. Thus we can perhaps base our BILLing on something like $\text{BILL}(y_{k_i+N}) = \text{FILLER}(\text{POINTER}(J[k,i]))$. The difficulty is in deciding what to do when the FILLER isn't there or has too high rank, so that we must choose one of the N elements in the set, and yet still keep BILL 1-1. Once that is solved, the proof might be much like that of Lemma 5.5.

SECTION 5.2. The 71/60 Theorem

We shall now show how to combine Corollary 5.1.1 and Theorem 5.6 into an upper bound proof.

THEOREM 5.7: If $S \in AAF$, then

$$R[SD, t] = 71/60, \text{ for } 8/29 < t \leq 1/2,$$

$$7/6, \text{ for } 1/4 < t \leq 8/29.$$

Proof. The lower bounds follow from Theorem 4.1. By Lemma 4.2, since the upper bounds we wish to prove are either $7/6$ or larger, these upper bounds will follow if we can show that for every list L with $\text{Range}(\text{size}_L) \subseteq (1/7, t]$,

$$(5.7A) \quad S(L) \leq (7/6)L^* + K \quad [S(L) \leq (71/60)L^* + K],$$

for some K independent of L^* . In fact, we can choose $K = 5$, for by Therem 5.6, we know that

$$(5.7B) \quad Wd(L) \geq S(L) - 5,$$

and we shall now show that for every possible configuration of pieces in a bin of an optimal packing P^* of L , $Wd(\text{Cont}_{P^*}(j)) \leq (7/6)$, or $Wd(\text{Cont}_{P^*}(j)) \leq (71/60)$, as the case may be, so that

by Corollary 5.1.1,

$$(5.7C) \quad Wd(L) \leq (7/6)L^* \quad [Wd(L) \leq (71/60)L^*],$$

and (5.7A) will follow from (5.7B) and (5.7C).

For ease of reference, let us use the notation:

- B-piece = 2-piece = piece with size in $(1/3, 1/2]$,
- C-piece = 3-piece = piece with size in $(1/4, 1/3]$,
- D-piece = 4-piece = piece with size in $(1/5, 1/4]$,
- E-piece = 5-piece = piece with size in $(1/6, 1/5]$, and
- F-piece = 6-piece = piece with size in $(1/7, 1/6]$.

We must consider all possible configurations of B's, C's, D's, etc. whose sizes total no more than 1, for all such could occur as the set of pieces in a bin in P^* . Formally, a configuration $[x_1, x_2, \dots, x_n]$ where $x_i \in \{B, C, D, E, F\}$, $1 \leq i \leq n$, will be the set of all sets $\{x_1, \dots, x_n\} \subseteq \text{PIECES}(L)$, such that x_i is an x_i -piece, $1 \leq i \leq n$, $\text{rank}(x_1) < \dots < \text{rank}(x_n)$, and $\sum_{i=1}^n \text{size}(x_i) \leq 1$. An example would be $[B, B, C]$.

Every possible set $\text{Cont}_{P^*}(j)$ must be in some configuration. Moreover, we can restrict our attention to a relatively small class of configurations, the legal ones. If we let u be a

function which gives the lower bound on the possible size for each type of piece, with $u(B) = 1/3$, $u(C) = 1/4$, etc., then the criterion for a given configuration $[x_1, \dots, x_n]$ to be illegal is that

$$u[x_1, \dots, x_n] \equiv \sum_{i=1}^n u(x_i) \geq 1.$$

A configuration which is illegal must be empty, since if x is an X -piece, $\text{size}(x) > u(X)$. Thus we may restrict our attention to the legal configurations. We extend our definitions of W_d and w_l to configurations as follows:

$$\begin{aligned} W_d[x_1, \dots, x_n] &= \text{MAX}\{W_d(D) : D \in [x_1, \dots, x_n]\}, \\ w_l[x_1, \dots, x_n] &= \text{MAX}\{w_l(D) : D \in [x_1, \dots, x_n]\}. \end{aligned}$$

Note that since $w_l(D)$ only depends on the types of pieces in D , for all $D \in [x_1, \dots, x_n]$, $w_l(D) = w_l[x_1, \dots, x_n]$, but this need not be the case if w_l is replaced by W_d .

Our job will be to prove that

$$W_d[x_1, \dots, x_n] \leq 7/6 \quad [\text{or } \leq 71/60].$$

We can prove the inequality either by showing that $w_1[x_1, \dots, x_n] \leq 7/6$, or that if $D \in [x_1, \dots, x_n]$, there must be a partition Π of D such that $w_1(\Pi) \leq 7/6$.

An additional shortcut is supplied by the fact that certain configurations are clearly worse than others. If $X, Y \in \{B, C, D, E, F\}$, we shall say that X "w₁-dominates" Y if $w_1(x) \geq w_1(y)$ for any X -piece x and Y -piece y . Thus B w₁-dominates B, C, D, E , and F . Extending this to configurations, we will say that $[x_1, \dots, x_n]$ w₁-dominates $[y_1, \dots, y_n]$ if x_i w₁-dominates y_i , for $1 \leq i \leq n$. In this case $w_1[y_1, \dots, y_n] \leq w_1[x_1, \dots, x_n]$, so if $w_1[x_1, \dots, x_n] \leq 7/6$, it clearly would be redundant to check $[y_1, \dots, y_n]$ separately.

With these introductory comments out of the way, let us begin the case analysis. For ease of reference, we shall refer to piece x_i in $\{x_1, \dots, x_n\} \in [x_1, \dots, x_n]$ as piece x_i . For instance piece B_2 would be a B -piece and would be the piece with second lowest rank in its set. For added convenience, we shall identify each piece x_i with its size, and refer to the configuration $[x_1, \dots, x_n]$ as $|x_1, \dots, x_n|$, for instance $[B_1, B_2, C_3]$. For conciseness, we shall write simply W for $w_d[x_1, \dots, x_n]$, and similarly for w_1 and u .

There will only be two cases, $[B_1, B_2, E_3, F_4]$ and $[C_1, D_2, E_3, E_4, E_5]$, in which we cannot prove that $W \leq 7/6$, and we shall in each case show that, if all pieces are $\leq 8/29$, then

either the configuration becomes empty or we suddenly can prove the $7/6$ bound.

On a first reading of this proof, it might be wise to look just at these and the first few cases in detail, and to assure oneself by a quick run through that all legal configurations are covered by some argument. Our organization of cases is such as to make this last task fairly straightforward.

One-Piece Bins:

The class of such bins is clearly w_1 -dominated by $[B_1]$, and $w_1(B) = 1/2 \leq 7/6$.

Two-Piece Bins:

This class is clearly w_1 -dominated by $[B_1, B_2]$, and $2w_1(B) = 1 \leq 7/6$.

Three-Piece Bins:

1) Two or more B's:

$[B_1, B_2, B_3]$. This is impossible since $u = 3/3 = 1$.

$[B_1, B_2, X_3]$ for $X = C, D, E, \text{ or } F$.

No matter what X_3 is, we must have $2(B_2) + X_3 \leq 1$,
 and so (B_2, X_3) obeys relation 2, and so
 $w \leq w_1(B_1) + w_2(B_2, X_3) \leq 1/2 + 1/2 + 1/6 = 7/6$.

2) One or fewer B's: This, the class of all remaining
 three-piece configurations, is w_1 -dominated by $[B_1, C_2, C_3]$,
 and $w_1(B) + 2w_1(C) = 1/2 + 2/3 = 7/6$.

Four-Piece Bins:

1) Three or more B's: Impossible, as seen above.

2) Two B's and a C or D: Impossible, since the fourth
 piece must be at least an F, and
 $u = 2/3 + 1/5 + 1/7 = 106/105 > 1$.

3) Two B's and two E's: Impossible, since $u = 2/3 + 2/6 = 1$.

4) Legal configurations with two B's:

$[B_1, B_2, E_3, F_4]$. $w_1 = (2/2) + (1/5) + (1/6) = 41/30 > 7/6$.

However, $B_1+E_3 \leq 1-u(B_2)-u(F_4) = 1-(1/3)-(1/7) = 11/21$, and This implies that $2(B_1)+E_3 \leq (22/21)-(1/6) < 1$, so (B_1, E_3) obeys Relation 2 and we get a discount of $(1/2)(1/5) = 1/10$. Similarly, $B_2+F_4 \leq 1-(1/3)-(1/6) = 1/2$, which implies $2(B_2)+F_4 \leq 1-(1/7) < 1$, so (B_2, F_4) obeys Relation 2, and we get an additional discount of $1/12$. Thus $W \leq (41/30)-(11/60) = 71/60$. This is one of the two cases in which the $71/60$ bound cannot be improved upon using our methods, although there are other ways of showing that no $71/60$ examples can be constructed using this type of bin. However, for our purposes all we need note is that if all pieces must be $\leq 8/29$, there can be no B-pieces, so this configuration could not occur.

$[B_1, B_2, F_3, F_4]$. $w_1 = (2/2) + (2/6) = 4/3$.

$B_1+F_3 \leq 1-(1/3)-(1/7) = 11/21$, so that $2(B_1)+F_3 \leq (22/21)-(1/7) < 1$, and (B_1, F_3) obeys Relation 2, and the same goes for (B_2, F_4) . Thus $W \leq (4/3)-2(1/2)(1/6) = 7/6$.

5) One B, two or more C's:

$[B_1, C_2, C_3, X_4]$, for $X = C, D, \text{ or } E$. Impossible, since $u \geq 1/3 + 2/4 + 1/6 = 1$.

$[B_1, C_2, C_3, F_4]$. $w_1 = (1/2) + (2/3) + (1/6) = 4/3 > 7/6$.
 $B_1 + C_2 \leq 1 - (1/4) - (1/7) = 17/28$, so $2(B_1) + C_2 \leq (17/14) - (1/4) = 27/28 < 1$, and (B_1, C_2) obeys Relation 2.
 Thus $W \leq (4/3) - (1/6) = 7/6$.

6) One B, one C:

$[B_1, C_2, D_3, D_4]$. $w_1 = (1/2) + (1/3) + (2/4) = 4/3$.
 $B_1 + C_2 \leq 1 - (2/5) = 3/5 \implies 2(B_1) + C_2 \leq (6/5) - (1/4) < 1$,
 so (B_1, C_2) obeys Relation 2 and $W \leq (4/3) - (1/6) = 7/6$.

$[B_1, C_2, D_3, E_4]$. $w_1 = (5/6) + (9/20) = 77/60 > 7/6$.
 $B_1 + D_3 \leq 1 - (1/4) - (1/6) = 7/12 \implies 2(B_1) + D_3 \leq (7/6) - (1/5) < 1$, so (B_1, D_3) obeys Relation 2 and
 $W \leq (77/60) - (1/8) = 139/120 < 7/6$.

$[B_1, C_2, D_3, F_4]$. $w_1 = (5/6) + (5/12) = 5/4 > 7/6$.
 $B_1 + F_4 \leq 1 - (1/4) - (1/5) = 11/20 \implies 2(B_1) + F_4 \leq (11/10) - (1/7) < 1$, so (B_1, F_4) obeys Relation 2 and

$$W \leq (5/4) - (1/12) = 14/12 = 7/6.$$

$[B_1, C_2, E_3, E_4]$. $w_1 = (5/6) + (2/5) = 37/30 > 7/6.$
 $B_1 + E_3 \leq 1 - (1/4) - (1/6) = 7/12 \implies 2(B_1) + E_3 \leq (7/6) - (1/6) \leq 1$, so (B_1, E_3) obeys Relation 2 and
 $W \leq (37/30) - (1/10) = 34/30 < 7/6.$

$[B_1, C_2, E_3, F_4]$. $w_1 = (5/6) + (11/30) = 36/30 > 7/6.$
If $B_1 + F_4 \leq 4/7$, then $2(B_1) + F_4 \leq (8/7) - (1/7) = 1$, so
 (B_1, F_4) would obey Relation 2 and we would have $W \leq (36/30) - (1/12) = 67/60 < 7/6$. If not, then $C_2 + E_3$ must be
 $\leq 1 - (4/7) = 3/7$, and so $3(C_2) + E_3 \leq (9/7) - 2(1/6) < 1$,
and so (C_2, E_3) would obey Relation 3 and we would still
have $W \leq (36/30) - (1/3)(1/5) = 34/30 < 7/6$.

$$[B_1, C_2, F_3, F_4]. w_1 = (5/6) + (2/6) = 7/6.$$

7) One B, no C's, two or more D's:

$[B_1, D_2, D_3, D_4]$. $w_1 = (1/2) + (3/4) = 5/4 > 7/6.$
 $B_1 + D_4 \leq 1 - (2/5) = 3/5 \implies 2(B_1) + D_4 \leq (6/5) - (1/5) = 1$,
so (B_1, D_4) obeys Relation 2 and $W \leq (5/4) - (1/8) = 9/8 < 7/6$.

$[B_1, D_2, D_3, E_4]$. $w_1 = (1/2) + (2/4) + (1/5) = 6/5 > 7/6$.

If $4(D_3) + E_4 \leq 1$, then (D_3, E_4) obeys Relation 4 and so
 $w \leq (6/5) - (1/4)(1/5) = 23/20 < 7/6$. If not, then we must
have $D_2 \geq D_3 > (1-E_4)/4$, so that $B_1+E_4 \leq 1 - 2(1-E_4)/4$
and consequently $2(B_1)+E_4 \leq 2 - (1-E_4) - E_4 = 1$, and so (B_1, E_4)
would obey Relation 2 and we would still have
 $w \leq (6/5) - (1/10) = 11/10 < 7/6$.

$[B_1, D_2, D_3, F_4]$. $w_1 = 1 + (1/6) = 7/6$.

8) One B, no C's, no more than one D: This class is clearly
 w_1 -dominated by $[B_1, D_2, E_3, E_4]$, for which
 $w_1 = (3/4) + (2/5) = 23/20 < 7/6$.

9) No B's, three or more C's:

$[C_1, C_2, C_3, C_4]$. This is impossible since $u = 4/4 = 1$.

$[C_1, C_2, C_3, X_4]$, for $X = D, E$, or F .

No matter what X is, (C_3, X_4) must obey Relation 3, so
 $w \leq (3/3) + (2/3)(1/4) = 7/6$.

10) No B's, no more than 2 C's: This, the class of all remaining four-piece configurations, is clearly w1-dominated by $[C_1, C_2, D_3, D_4]$, for which $w_1 = (2/3)+(2/4) = 7/6$.

Five-Piece Bins:

1) Two or more B's: Impossible since we must have $u \geq 2/3 + 3/7 = 23/21 > 1$.

2) One B and one or more C's: Impossible since then we must have $u \geq 1/3 + 1/4 + 3/7 = 85/84 > 1$.

3) One B, no C's, and two or more D's: Impossible, since then we must have $u \geq 1/3 + 2/5 + 2/7 = 107/105 > 1$.

4) One B, no C's, one D, and two or more E's: Impossible, since then $u \geq 1/3 + 1/5 + 2/6 + 1/7 = 106/105 > 1$.

5) Legal configurations with one B:

$$[B_1, D_2, E_3, F_4, F_5]. \quad w_1 = (3/4)+(1/5)+(2/6) = 77/60 > 7/6.$$

$B_1+D_2 \leq 1-(1/6)-(2/7) = 23/42 \implies 2(B_1)+D_2 \leq (23/21)-(1/5) < 1$, so (B_1, D_2) obeys Relation 2 and

$$W \leq (77/60) - (1/8) = 139/120 < 7/6.$$

$[B_1, D_2, F_3, F_4, F_5]$. $w_1 = (3/4) + (3/6) = 5/4 > 7/6.$

$$B_1 + D_2 \leq 1 - (3/7) = 4/7 \implies 2(B_1) + D_2 \leq (8/7) - (1/5) > 1,$$

$$\text{so } (B_1, D_2) \text{ obeys Relation 2 and } W \leq (5/4) - (1/8) = 9/8 < 7/6.$$

$[B_1, E_2, E_3, E_4, E_5]$. Impossible since $u = 1/3 + 4/6 = 1$.

$[B_1, E_2, E_3, E_4, F_5]$. $w_1 = (1/2) + (3/5) + (1/6) = 76/60 > 7/6.$

$$B_1 + E_4 \leq 1 - (2/6) - (1/7) = 11/21 \implies 2(B_1) + E_4 \leq$$

$(22/21) - (1/6) < 1$, so that (B_1, E_4) obeys Relation 2 and

$$W \leq (76/60) - (1/10) = 70/60 = 7/6.$$

$[B_1, E_2, E_3, F_4, F_5]$. $w_1 = (1/2) + (2/5) + (2/6) = 74/60 > 7/6.$

$$B_1 + F_4 \leq 1 - (2/6) - (1/7) = 11/21 \implies 2(B_1) + F_4 \leq$$

$(22/21) - (1/7) < 1$, so that (B_1, F_4) obeys Relation 2 and

$$W \leq (74/60) - (1/12) = 69/60 < 7/6.$$

$[B_1, E_2, F_3, F_4, F_5]$. $w_1 = (7/10) + (3/6) = 6/5 > 7/6.$

$$B_1 + F_5 \leq 1 - (1/6) - (2/7) = 23/42 \implies 2(B_1) + F_5 \leq$$

$(23/21) - (1/7) < 1$, so that (B_1, F_5) obeys Relation 2 and

$$W \leq (6/5) - (1/12) = 67/60 < 7/6.$$

$[B_1, F_2, F_3, F_4, F_5]$. $w_1 = (1/2) + (4/6) = 7/6.$

6) No B's, three or more C's: Impossible since we must then have $u \geq 3/4 + 2/7 = 29/28 > 1$.

7) No B's, two C's, two or more D's: Impossible, since then $u \geq 2/4 + 2/5 + 1/7 = 73/70 > 1$.

8) No B's, two C's, one D, and one or more E's: Impossible since then $u \geq 2/4 + 1/5 + 1/6 + 1/7 = 424/420 > 1$.

9) Legal configurations with no B's, two C's:

$[C_1, C_2, D_3, F_4, F_5]$. $w_1 = (2/3) + (1/4) + (2/6) = 5/4 > 7/6$.
 $C_1 + D_3 \leq 1 - (1/4) - (2/7) = 13/28 \implies 3(C_1) + D_3 \leq (39/28) - (2/5) = 139/140 < 1$, and so (C_1, D_3) obeys Relation 3
and $W \leq (5/4) - (1/12) = 14/12 = 7/6$.

$[C_1, C_2, E_3, E_4, E_5]$. Impossible, since $u = 2/4 + 3/6 = 1$.

$[C_1, C_2, E_3, E_4, F_5]$. $w_1 = (2/3) + (2/5) + (1/6) = 74/60 > 7/6$.
 $C_1 + E_3 \leq 1 - (1/4) - (1/6) - (1/7) = 37/84 \implies 3(C_1) + E_3 \leq (111/84) - (2/6) = 83/84 < 1$, so (C_1, E_3) obeys Relation 3 and
 $W \leq (74/60) - (1/15) = 70/60 = 7/6$.

$[C_1, C_2, E_3, F_4, F_5]$. $w_1 = (2/3) + (1/5) + (2/6) = 6/5 > 7/6$.
 If $C_1 + F_4 \leq 3/7$ or $C_2 + F_5 \leq 3/7$, then for that pair we would have $3(C) + F \leq (9/7) - (2/7) = 1$, so the pair would obey Relation 3 and we would have $W \leq (6/5) - (1/18) = 103/90 < 7/6$. And one of the two inequalities must hold, since otherwise the total of the pieces would exceed $2(3/7) + (1/6) > 1$.

$[C_1, C_2, F_3, F_4, F_5]$. $w_1 = (2/3) + (3/6) = 7/6$.

10) No B's, one C, one or more D's:

$[C_1, D_2, D_3, D_4, X_5]$, for $X = D$ or E . Impossible since $u \geq 1/4 + 3/5 + 1/6 = 61/60 > 1$.

$[C_1, D_2, D_3, D_4, F_5]$. $w_1 = (1/3) + (3/4) + (1/6) = 5/4 > 7/6$.
 $C_1 + D_2 \leq 1 - (2/5) - (1/7) = 16/35 \implies 3(C_1) + D_2 \leq (48/35) - (2/5) = 34/35 < 1$, so that (C_1, D_2) obeys Relation 3 and $W \leq (5/4) - (1/12) = 14/12 = 7/6$.

$[C_1, D_2, D_3, E_4, E_5]$. $w_1 = (1/3) + (2/4) + (2/5) = 74/60 > 7/6$.
 $C_1 + D_2 \leq 1 - (1/5) - (2/6) = 7/15 \implies 3(C_1) + D_2 \leq (21/15) - (2/5) = 1$, so that (C_1, D_2) obeys Relation 3 and $W \leq (74/60) - (1/12) = 69/60 < 7/6$.

$[C_1, D_2, D_3, E_4, F_5]$. $w_1 = (5/6) + (1/5) + (1/6) = 6/5 > 7/6$.

If $4(D_3) + E_4 \leq 1$, then (D_3, E_4) would obey relation 4, and W would be $\leq (6/5) - (1/20) = 23/20 < 7/6$. If not, then $D_2 \geq D_3 > (1-E_4)/4$, and so $C_1+E_4 \leq 1 - (2(1-E_4)/4) - (1/7) = (5/14) + (E_4/2)$. This means that $3(C_1) + E_4 \leq (15/14) - (E_4/2) \leq (15/14) - (1/12) < 1$, so that (C_1, E_4) would obey Relation 3 and we would still have $W \leq (6/5) - (1/15) = 17/15 < 7/6$.

$[C_1, D_2, D_3, F_4, F_5]$. $w_1 = (5/6) + (2/6) = 7/6$.

$[C_1, D_2, E_3, E_4, E_5]$. $w_1 = (7/12) + (3/5) = 71/60$.

This is the second and last configuration for which we have not been able to prove that any pairs obey any of the Relations, and this is not surprising since we have already seen an example where this configuration yields a list for which $SD(L)/L^* = 71/60$ (Figure 4.3). However, if all the pieces must be $\leq 8/29$, then, in particular, we must have $C_1 \leq 8/29$. If $3(C_1) + E_5 \leq 1$, then (C_1, E_5) would obey Relation 3 and we would have $W \leq (71/60) - (1/15) < 7/6$. If not, then we must have $E_3 \geq E_4 \geq E_5 > 1 - (24/29) = 5/29$. So let $E_5 - (5/29) = x > 0$. We then have $C_1 > (8/29) - (x/3)$. And so $D_2 < 1 - [(8/29) - (x/3)] - 3[(5/29) + x] = (6/29) - (8x/3)$, and therefore $4(D_2) + E_5 \leq (29/29) - (32x/3) + x \leq 1$, and

(D_2, E_5) obeys Relation 4, so that we still would have $W \leq (71/60) - (1/20) < 7/6$. So the problems with this configuration go away if all pieces in L are $\leq 8/29$.

$[C_1, D_2, E_3, E_4, F_5]$, $[C_1, D_2, E_3, F_4, F_5]$, and $[C_1, D_2, F_3, F_4, F_5]$.

These are all w_1 -dominated by the first, for which $w_1 = (7/12) + (2/5) + (1/6) = 23/20 < 7/6$.

11) No B's, one C, and no D's: This class is clearly w_1 -dominated by $[C_1, E_2, E_3, E_4, E_5]$ for which $w_1 = (1/3) + (4/5) = 17/15 < 7/6$.

12) No B's, No C's, and four or more D's:

$[D_1, D_2, D_3, D_4, D_5]$. Impossible, since $u = 5/5 = 1$.

$[D_1, D_2, D_3, D_4, X_5]$, for $X = E$ or F .

No matter what X is, (D_4, X_5) will obey Relation 4, so that $W \leq (4/4) + (3/4)(1/5) = 23/20 < 7/6$.

13) No B's, no C's, no more than three D's: This, the class of all remaining five-piece configurations, is clearly w_1 -dominated by $[D_1, D_2, D_3, E_4, E_5]$, for which $w_1 = (3/4) + (2/5) = 23/20 < 7/6$.

Six-Piece Bins:

1) One or more B's: Impossible, since we would have

$$u \geq 1/3 + 5/7 = 22/21 > 1.$$

2) No B's, two or more C's: Impossible, since we would have

$$u \geq 2/4 + 4/7 = 15/14 > 1.$$

3) No B's, one C, one or more D's: Impossible, since we

$$\text{would have } u \geq 1/4 + 1/5 + 4/7 = 143/140 > 1.$$

4) No B's, one C, no D's, two or more E's: Impossible, since

$$\text{we would have } u \geq 1/4 + 2/6 + 3/7 = 85/84 > 1.$$

5) Legal configurations with one C:

$$[C_1, E_2, F_3, F_4, F_5, F_6]. \quad w_1 = (8/15)+(4/6) = 18/15 > 7/6.$$

$$C_1+E_2 \leq 1-(4/7) = 3/7 \implies 3(C_1)+E_2 \leq (9/7)-(2/6) < 1, \text{ so}$$

$$(C_1, E_2) \text{ obeys Relation 3 and } W \leq (6/5)-(1/15) = 17/15 < 7/6.$$

$$[C_1, F_2, F_3, F_4, F_5, F_6]. \quad w_1 = (1/3)+(5/6) = 7/6.$$

6) No B's, no C's, three or more D's: Impossible, since then we would have $u \geq 3/5 + 3/7 = 36/35 > 1$.

7) No B's, No C's, two D's, and two or more E's: Impossible, since we would have $u \geq 2/5 + 2/6 + 2/7 = 214/210 > 1$.

8) Legal configurations with no C's, two D's:

$[D_1, D_2, E_3, F_4, F_5, F_6]$. $w_1 = (2/4) + (1/5) + (3/6) = 6/5 > 7/6$.
 $D_1 + F_4 \leq 1 - (1/5) - (1/6) - (2/7) = 73/210 \implies 4(D_1) + F_4 \leq (146/105) - (3/7) = 101/105 < 1$, so (D_1, F_4) obeys Relation 4,
and $W \leq (6/5) - (1/24) = 139/120 < 7/6$.

9) No B's, no C's, one D, four or more E's: Impossible, since we would have $u \geq 1/5 + 4/6 + 1/7 = 212/210 > 1$.

10) Legal configurations with no C's, one D:

$[D_1, E_2, E_3, E_4, F_5, F_6]$. $w_1 = (1/4) + (3/5) + (2/6) = 71/60$.
 $D_1 + F_5 \leq 1 - (3/6) - (1/7) = 5/14 \implies 4(D_1) + F_5 \leq (10/7) - (3/7) = 1$, so (D_1, F_5) obeys Relation 4 and
 $W \leq (71/60) - (1/24) < 7/6$.

$[D_1, E_2, E_3, F_4, F_5, F_5]$, $[D_1, E_2, F_3, F_4, F_5, F_6]$, and
 $[D_1, F_2, F_3, F_4, F_5, F_6]$.

These three are all w_1 -dominated by the first, for which
 $w_1 \leq (1/4) + (2/5) + (3/6) = 23/20 < 7/6$.

- 11) No B's, no C's, no D's, and six E's: Impossible, since we would have $u = 6/6 = 1$.
- 12) No B's, no C's, no D's, no more than five E's: This, the class of all remaining six-piece configurations, is clearly w_1 -dominated by $[E_1, E_2, E_3, E_4, E_5, F_6]$, for which $w_1 = 7/6$.

Since there can be no seven-piece bins if all the pieces exceed $1/7$, this completes our case analysis, and the Theorem is proved. \square

A similar case analysis can be constructed to show

THEOREM 5.8: If $S \in AAF$, and $1/5 < t \leq 1/4$, then

$$R[S_D, t] = 23/20.$$

We will generously omit the details, only mentioning that for this proof we must allow pieces as small as $3/23 < 1/7$, in order to apply Lemma 4.2, and the fractional inequalities become

even more difficult to follow than in the above.

Remark. We conjecture that both Theorems 5.7 and 5.8 extend to arbitrary $S \in AF$, basing this on our conjecture that Theorem 5.6 similarly extends with only a change in the constant.

We also conjecture that if we had the time and the patience, we could use this method to establish $R[SD, t]$ for any $t \in (0, 1/2]$, and that the results would be the lower bounds given by Theorem 4.1. However, this approach is not really practical, since it appears that we would need a separate case analysis for each $n > 0$ to cover the case when $\lfloor 1/t \rfloor = n$.

SECTION 5.3. The 11/9 Theorem

In this section we use the weighting function W_d to derive an exact upper bound on $R[FFD, t]$ for all $t \in (1/2, 1]$. To do this, we must consider lists with pieces of size greater than $1/2$. In Chapter 4 we saw that the presence of such pieces made for a considerable increase in complexity over the case when they were not allowed. And indeed, W_d was defined under the assumption that they were absent. Nevertheless, as we shall see in the proof of the following Theorem, there are ways to surmount these difficulties, at least for the algorithm FFD.

THEOREM 5.9. For all decreasing lists L ,

$$FF(L) \leq (11/9)L^* + 4.$$

We shall divide the proof of this Theorem into sections. The first will reduce the general statement to be proved to a simpler, more manageable situation, and present an outline of how we can adapt our weighting function to these circumstances. The second will develop the machinery for this adaptation, and prove that it works. The final section will be, of course, the case analyses.

PROOF OF THEOREM 5.9

I. Definitions, Reductions, and the Idea of the Proof:

Our first major reduction is

REDUCTION 1: $\text{Range}(\text{size}_L) \subseteq (2/11, 1]$.

If there is a list violating the Theorem, there must be one with piece sizes in this range by Lemma 4.2, with $K = 4$, $r = 11/9$, since $(11/9 - 1)/(11/9) = 2/11$.

Granted this reduction, let us divide L into sublists, $L = LA \cdot LB \cdot LC \cdot LD \cdot LE$, where

$\text{Range}(\text{size}_{LA}) \subseteq (1/2, 1]$, $\text{PIECES}(LA) = A\text{-PIECES}(L)$,
 $\text{Range}(\text{size}_{LB}) \subseteq (1/3, 1/2]$, $\text{PIECES}(LB) = B\text{-PIECES}(L)$,
 $\text{Range}(\text{size}_{LC}) \subseteq (1/4, 1/3]$, $\text{PIECES}(LC) = C\text{-PIECES}(L)$,
 $\text{Range}(\text{size}_{LD}) \subseteq (1/5, 1/4]$, $\text{PIECES}(LD) = D\text{-PIECES}(L)$,
 $\text{Range}(\text{size}_{LE}) \subseteq (2/11, 1/5]$, $\text{PIECES}(LE) = E\text{-PIECES}(L)$.

We shall usually drop the argument (L) for $X\text{-PIECES}(L)$. As usual, elements of $X\text{-PIECES}$ will be called X -pieces. A bin in a packing of L which contains an A -piece will be called an A -bin and the a -bin. All other non-empty bins will simply be

called non-A-bins.

Now let PF be the FF-packing of L, and let P* be an optimal packing of L with its bins so ordered that if in PF BIN_j is the a-bin for $a \in A\text{-PIECES}(L)$, then BIN_j is the a-bin in P* also.

Divide the two packings into segments as shown in Figure 5.3.

PA^* is the A-bins of P*, PB^* the non-A-bins, PFA the A-bins of PF, and PFB the non-A-bins of PF. Let us further define

$$\mathcal{B} = \text{PIECES}(LB \cdot LC \cdot LD \cdot LE) = \text{PIECES}(L) - \text{PIECES}(LA),$$

$$\Theta = \text{cont}(PB^*) = \{b \in \mathcal{B} : b \text{ is not in an A-bin in } P^*\},$$

$$\mathcal{J} = \text{cont}(PFB) = \{b \in \mathcal{B} : b \text{ is not in an A-bin in } PF\}.$$

The following two reductions are of a more technical nature than the previous one, and their importance will not be immediately apparent, but they are presented now for convenience.

REDUCTION 2: \mathcal{J} contains the last element e' in the decreasing list L, and hence no pieces were placed by FF in A-bins after the last piece had been placed in a non-A-bin.

If any pieces were, they certainly would not affect $FF(L)$, and their omission could only decrease L^* , and so if there is a decreasing list L for which $FF(L) > (11/9)L^* + 4$, then there

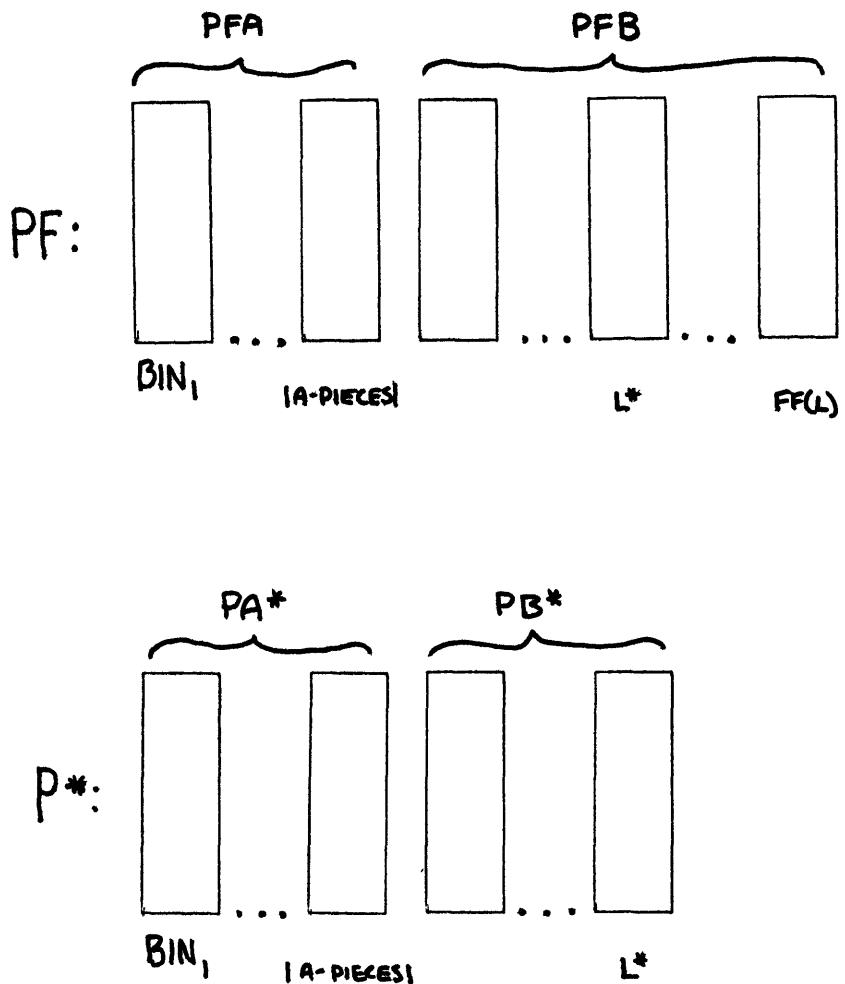


FIGURE 5.3. PF and P*.

must be one obeying the Reduction.

REDUCTION 3: $\exists \gamma \wedge \text{PIECES}(LC \cdot LD) \neq \emptyset$.

By Lemma 4.4, $\text{FF}(LA \cdot LB) \leq L^*$. If no C- or D-piece went in a non-A-bin in PF, we thus would have $\text{FF}(LA \cdot LB \cdot LC \cdot LD) \leq L^*$, and could apply Lemma 4.3 with $n = 5$ to conclude that $\text{FF}(L) \leq (6/5)L^* + 1 < (11/9) + 4$. So the only lists L which possibly could violate our Theorem are ones obeying Reduction 3.

We are now prepared to show how we adapt our weighting function W_d to the presence of A-pieces. The key idea is the observation that we could have generated the packing of segment PFB, the non-A-bins of PF, by simply considering γ as a list, ranked so that

$$\text{rank}_{\gamma}(b) < \text{rank}_{\gamma}(c) \iff \text{rank}_L(b) < \text{rank}_L(c),$$

and applying FF to γ . And since γ does not contain any A-pieces, all the facts that we proved about W_d will hold for that packing. In particular, we know from Theorem 5.6 with $N = 6$ that

CLAIM 5.9.1: $Wd(\mathcal{B}) \geq \#PFB - 4$
 $\geq FF(L) - |A\text{-PIECES}| - 4.$

However, we still face the problem of identifying which pieces in L actually go into \mathcal{B} . It would be convenient if these were precisely the pieces in Θ , but this need not be the case. Some pieces that are not in A-bins in one packing may be in A-bins in the other. In addition, there is the fact that the number of A-bins, which is the same in both packings, but not counted by $\#PFB$, must somehow be counted when we try to put a bound on $FF(L)$. To take care of these two problems, we shall introduce two functions:

$$f: PIECES(L) \longrightarrow 2^{\mathcal{B}}$$

will be a "responsibility assigning" function which assigns to each piece in L the set of the zero or more pieces in \mathcal{B} for which it is, in a sense to be explained in more detail later, uniquely "responsible." It will obey

$$\text{PROPERTY 5.9A: } \mathcal{B} = \bigcup_{b \in \text{PIECES}(L)} f(b).$$

The second function

$g: \text{PIECES}(L) \dashrightarrow Q$

will serve to count those A-bins which, in a sense also to be explained later, "collaborated" in the above "responsibility." It will obey

$$\text{PROPERTY 5.9B: } |\text{A-PIECES}| \geq \sum_{b \in \text{PIECES}(L)} g(b).$$

We can then easily extend f and g to functions on sets by

$$f(X) = \bigcup_{b \in X} f(b), \quad g(X) = \sum_{b \in X} g(b).$$

Our case analysis will be devoted to showing that for each BIN_j in P*, the following property holds:

PROPERTY 5.9C: For $X = \text{cont}_{P^*}(j)$,

$$Wd(f(X)) + g(X) \leq (11/9)(y(X) + g(X)),$$

where $y(X) = \begin{cases} 1, & \text{if } X \text{ contains no A-pieces,} \\ 0 & \text{otherwise.} \end{cases}$

We can say that the left-hand side of the inequality counts bins in PF , and the right-hand side does the same for P^* (and multiplies the result by $11/9$), since Wd counts non-A-bins in PF , y counts them in P^* , and g counts the A-bins in both packings. This intuitive way of looking at the Property hopefully will help us understand what is going on when we observe that, given that f and g obey Properties 5.9A and 5.9B respectively, and that Property 5.9C holds for all BIN_j in P^* , the Theorem follows by summation. For we would have

$$\begin{aligned} & \sum_{j=1}^{L^*} [Wd[f(\text{cont}_{P^*}(j))] + g[\text{cont}_{P^*}(j)]] \\ & \leq \sum_{j=1}^{L^*} (11/9)(y[\text{cont}_{P^*}(j)] + g[\text{cont}_{P^*}(j)]). \end{aligned}$$

Thus, since $Wd(\text{PIECES}(L)) \leq \sum_{j=1}^{L^*} Wd(f[\text{cont}_{P^*}(j)])$ by Property 5.9A and Lemma 5.1, we have

$$\begin{aligned} & Wd(\text{PIECES}(L)) + g[\text{PIECES}(L)] \\ & \leq (11/9) \cdot (L^* - |\text{A-PIECES}| + g[\text{PIECES}(L)]), \end{aligned}$$

and so by Claim 5.9.1,

$$\begin{aligned} \text{FF}(L) - |\text{A-PIECES}| - 4 + g[\text{PIECES}(L)] \\ \leq (11/9) \cdot (L^* - |\text{A-PIECES}| + g[\text{PIECES}(L)]). \end{aligned}$$

$$\text{FF}(L) \leq (11/9)L^* - (2/9)(|\text{A-PIECES}| - g[\text{PIECES}(L)]) + 4,$$

and so by Property 5.9B,

$$\text{FF}(L) \leq (11/9)L^* + 4.$$

Summarizing, we have

LEMMA 5.10. If for any decreasing list L obeying
 Reductions 1, 2, and 3, there exist maps
 $f: \text{PIECES}(L) \rightarrow \mathbb{Z}_{\geq 0}$ and
 $g: \text{PIECES}(L) \rightarrow \mathbb{Q}$
 obeying Properties 5.9A and 5.9B, respectively, and such that
 Property 5.9C holds for all BIN_j in an optimal packing of L,
 then for all possible decreasing lists L,

$$\text{FF}(L) \leq (11/9)L^* + 4.$$

Thus our theorem will be proven once we have produced f and g and shown that the case analysis can be done. First, f and g:

PROOF OF THEOREM 5.9

II. Responsibility and Collaboration

First let us make some observations about segments PA^* and PFA .

CLAIM 5.9.2. No bin in PA^* or PFA contains more than three pieces.

This is because an A-piece and three additional pieces would have a total size of more than $1/2 + 6/11 > 1$. However, note that two non-A-pieces are possible, as long as neither is a B-piece and at least one is a D-piece or smaller, since an A-piece, a C-piece, and a D-piece could have total size as small as $1/2 + 1/4 + 1/5 + \epsilon = 19/20 + \epsilon \leq 1$, for small enough ϵ .

CLAIM 5.9.3. If BIN_j is an A-bin and $height_{P^*}(j) \geq 2$, then $height_{PF}(j) \geq 2$.

Proof of Claim. Let $b = piece_{P^*}(j, 2)$. Since $size(b) \geq size(e')$, where e' is the last piece in list L, and e' goes in a non-A-bin under FF by Reduction 2 and hence to the right of BIN_j , by Lemma 4.4 there is a $c = piece_{PF}(j, 2)$. \square

Claims 5.9.2 and 5.9.3 imply that the only A-bins which contain more pieces in P^* than in PF are those which contain three pieces in P^* and two in PF . We shall call such bins Deficit Bins, and the set of all Deficit Bins will be called DEFICIT.

One way of showing that a given A-bin is not a Deficit Bin is the following:

CLAIM 5.9.4. (FITTING LEMMA). If BIN_j is an A-bin, $a = piece_{PF}(j,1)$, $b = piece_{PF}(j,2)$, and $c \neq b$ is such that $\text{size}(a) + \text{size}(b) + \text{size}(c) \leq 1$, then $\text{height}_{PF}(j) = 3$ and BIN_j is not a Deficit Bin.

Proof of Claim. Since e' does not go in an A-bin in PF , when e' was assigned we must have had $\text{level}_P(j) > 1 - \text{size}(e') \geq 1 - \text{size}(c) \geq \text{size}(a) + \text{size}(b)$. Thus by definition of level, BIN_j must have contained some additional piece. \square

We are now ready to begin the construction which will eventually lead to the definition of both f and g , a construction which will be analogous to that used in the definition of the function "f" used in the proof of Theorem 4.8. This time we define the relation points to as follows:

If BIN_j is an A-bin, and for $h \in \{2,3\}$,

$x = \text{piece}_{P^*}(j, h)$,

$y = \text{piece}_{P^*}(j, h)$,

then x points to y .

(note that for $h = 2$ this is the reverse of the definition of points to in Theorem 4.8. There we would have said that y points to x).

A chain of pieces is a sequence $\langle b_1, \dots, b_n \rangle$ such that b_i points to b_{i+1} , $1 \leq i < n$. A loop is a chain $\langle b_1, \dots, b_n \rangle$ where b_n points to b_1 . A maximal chain is a chain which is not a loop, and not a proper subsequence of any other chain. If $\langle b_1, \dots, b_n \rangle$ is a maximal chain, we say b_1 is its head and b_n its tail.

CLAIM 5.9.5. $\{b : b \text{ is the head of a maximal chain}\} = \Theta \cup \{x : x = \text{piece}_{P^*}(j, 3) \text{ for } \text{BIN}_j \in \text{DEFICIT}\}$.

Proof of Claim. The set of heads of maximal chains is precisely $\{x \in \mathcal{B} : x \text{ is not pointed to}\}$. No element of Θ is in an A-bin in P^* , so none are pointed to, and of the pieces which are in A-bins in P^* , all except the top pieces in Deficit Bins are pointed to, by Claims 5.9.2, 5.9.3 and the definition of

Deficit Bin. \emptyset

We can illustrate some of these concepts by drawing a diagram of the optimal packing P^* , and if piece x points to piece y , drawing an arrow from x to y in the diagram. See Figure 5.4. Note that a Deficit Bin has two arrows leaving and only one entering.

Continuing our definitions, if b is the tail of a maximal chain and in addition $b \in \mathcal{G}$, then we shall call b a terminator. The tail of a maximal chain need not be a terminator, for it could be in an A-bin in PF, for instance piece $_{PF}(j,3)$ for an A-bin BIN_j with height $p^*(j) = 2$. However, since no element of \mathcal{G} is in an A-bin in PF and hence no element points to anything, each element of \mathcal{G} must be the tail of some maximal chain and so we have

CLAIM 5.9.6. $\{b : b \text{ is a terminator}\} = \mathcal{G}$.

A very important fact about terminators which is really the cornerstone of this proof is the following:

LOOP: $\langle D_3, D_1 \rangle$

MAXIMAL CHAINS: $\langle B_3, B_1, C_2, C_1 \rangle$, $\langle E_1, D_4 \rangle$

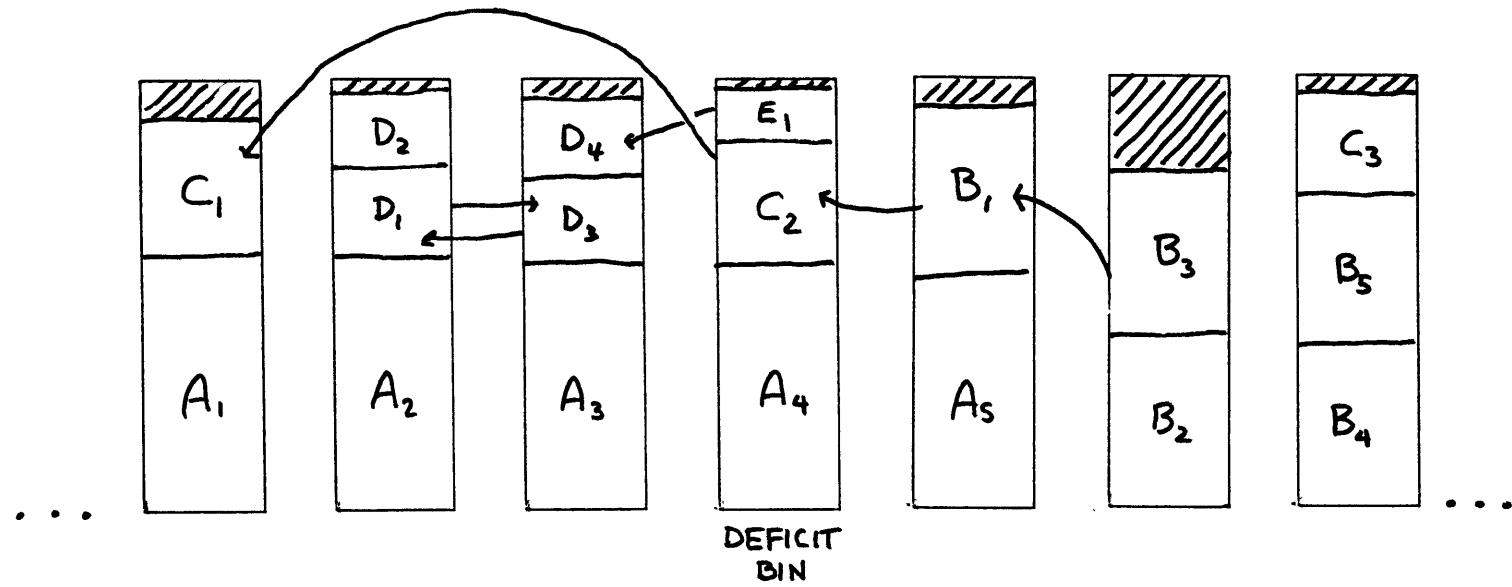


FIGURE 5.4. P^* with arrows.

CLAIM 5.9.7. (SIZE LEMMA). Suppose $\langle b_1, \dots, b_n \rangle$ is a maximal chain, b_n is a terminator, and $i^* = \max\{i : b_i = \text{piece}_{P^*}(j, 3)$ for some j or $i = 1\}$. Then $\text{rank}(b_n) > \text{rank}(b_i)$ for all $i^* \leq i < n$.

Proof of Claim. For each b_i , $i^* \leq i \leq n$, let $j[i]$ be the index of the A-bin containing b_i in P^* . Let b_h be the piece with maximal index in $\{b_i : i^* \leq i \leq n\}$. If $h = n$ we are done, so assume $h < n$. Then there exists a $b_{h+1} = \text{piece}_{P^*}(j[h+1], 2)$, and we have $b_h = \text{piece}_{P^*}(j[h+1], 2)$.

See Figure 5.5. In terms of this figure, we will show that

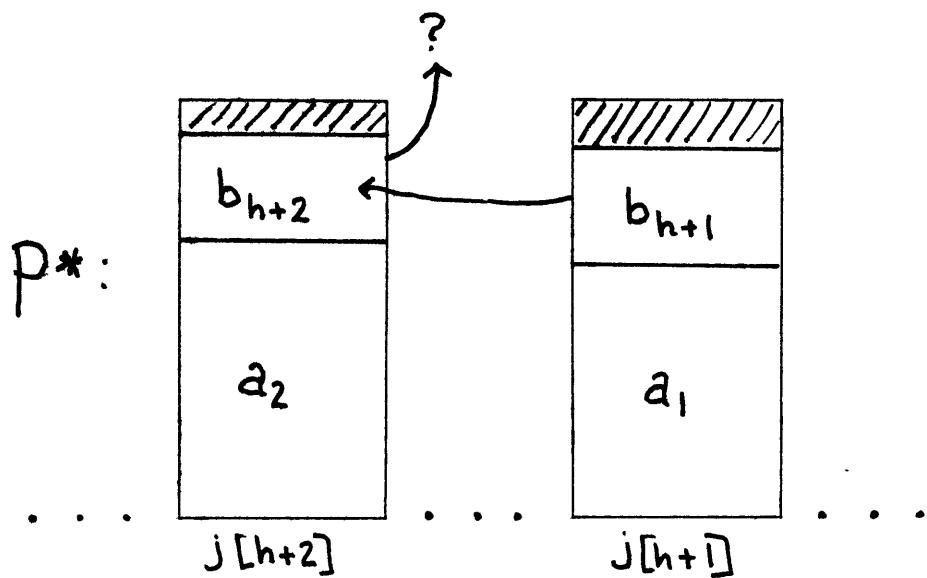


FIGURE 5.5

the arrow from piece b_{h+1} must point somewhere to the right of $\text{BIN}_{j[h+1]}$. The idea of the proof is then that no piece to the left of $\text{BIN}_{j[h+1]}$ in the diagram can be a terminator, and that no such piece can point back to the right of $\text{BIN}_{j[h+1]}$ either.

Thus the chain cannot have a terminator, contrary to assumption.

The details:

Let a_1 be the A-piece in $\text{BIN}_{j[h+1]}$. Then since $\text{size}(a_1) + \text{size}(b_{h+1}) \leq 1$, b_{h+1} cannot in PF go in any bin to the right of $\text{BIN}_{j[h+1]}$, since by Lemma 4.4 this would imply that $\text{rank}(\text{piece}_{PF}(j[h+1], 2)) = \text{rank}(b_h) < \text{rank}(b_{h+1})$, contradicting the definition of b_h . b_{h+1} cannot go in $\text{BIN}_{j[h+1]}$ in PF because it would have to go in position $(j[h+1], 3)$ since b_h is in position $(j[h+1], 2)$, and hence would be above a higher ranked piece, thus violating packing property (3). Thus b_{h+1} goes in a bin to the left of $\text{BIN}_{j[h+1]}$ in PF, and hence does not go in a non-A-bin and hence is not a terminator.

So there exists a b_{h+2} in the chain, with $b_{h+2} = \text{piece}_{P^*}(j[h+2], 2)$ and $b_{h+1} = \text{piece}_{PF}(j[h+2], 2)$. Let a_2 be the A-piece in $\text{BIN}_{j[h+2]}$. Since $j[h+2] < j[h+1]$, we must have $\text{size}(a_2) \geq \text{size}(a_1)$ by Lemma 1.2. Thus $\text{size}(a_1) + \text{size}(b_{h+2}) \leq \text{size}(a_2) + \text{size}(b_{h+2}) \leq 1$, and so by the same argument used above, b_{h+2} must in PF go in a bin to the left of $\text{BIN}_{j[h+1]}$ and hence cannot be a terminator, so there exists a b_{h+3} .

A simple induction based on this argument will thus show

that for no $k > 0$ is b_{h+k} a terminator. Since the chain is finite by Claim 5.9.5, and does have a terminator by hypothesis, this is a contradiction. Thus we must have $h = n$, and the Claim is proved. \square

We call Claim 5.9.7 the SIZE LEMMA since because L is decreasing, the terminator must have the minimum size of all the pieces in $\{b_i : i' \leq i \leq n\}$.

Continuing, we now extend the definition of points to to bins. We shall say a piece b points to BIN_j if BIN_j is an A-bin and there is a piece $c \in \text{cont}_{P^*}(j)$ which b points to. In terms of our diagram with arrows, Figure 5.3, piece b points to BIN_j if the arrow from b goes to BIN_j . If BIN_j and BIN_i are A-bins, we say BIN_j points to BIN_i if there is a $b \in \text{cont}_{P^*}(j)$ which points to BIN_i , that is, if an arrow goes from one bin to the other in our diagram. Note that by Claim 5.9.3, every A-bin which contains a non-A-piece in P^* is pointed to.

The following Claim tells us about what happens when Deficit Bins point to each other:

CLAIM 5.9.8. If BIN_j and BIN_i are Deficit Bins, and BIN_j points to BIN_i , then BIN_j is pointed to by a B-piece.

Proof of Claim. Since the two bins are Deficit Bins, each must contain three pieces in P^* . For $h \in \{j, i\}$, let

$$a_h = \text{piece}_{PF}(h, 1),$$

$$b_h = \text{piece}_{PF}(h, 2),$$

$$c_h = \text{piece}_{PF}(h, 3).$$

Let $x \in \{b_j, c_j\}$ be the piece in BIN_j which points to BIN_i and hence $= \text{piece}_{PF}(i, 2)$, y be the other piece, and $z = \text{piece}_{PF}(j, 2)$ be the piece that points to BIN_j (z must exist by Claim 5.9.3).

Now observe that in PF , the bottom two pieces in BIN_j are a_j and z , and the bottom two pieces in BIN_i are a_i and x . Thus there can be no piece $u \neq z$ such that

$$\text{size}(a_j) + \text{size}(z) + \text{size}(u) \leq 1,$$

and no piece $v \neq x$ such that

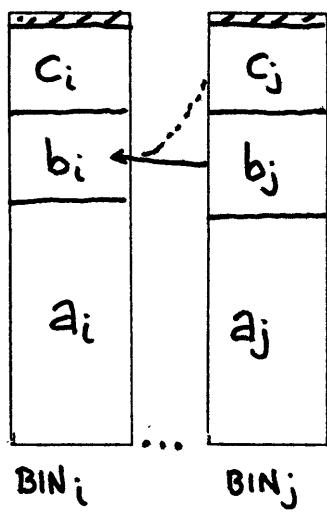
$$\text{size}(a_i) + \text{size}(x) + \text{size}(v) \leq 1,$$

else one of the A-bins would not be a Deficit Bin, by the FITTING LEMMA (Claim 5.9.4). There are three cases, depending on the relation of i to j (see Figure 5.6):

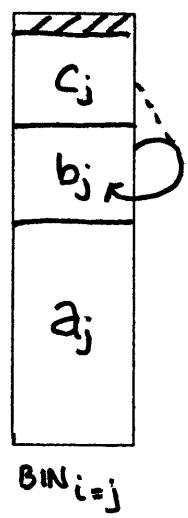
Case 1. $i < j$.

If z is not a B-piece, then $\text{size}(z) \leq 1/3 < \text{size}(b_j) + \text{size}(c_i)$, so since z goes in BIN_j in FF and is to the right of BIN_i , by Lemma 4.4 we have $\text{rank}(\text{piece}_{PF}(i, 2)) = \text{rank}(x) < \text{rank}(z)$. Thus $\text{size}(z) \leq \text{size}(x)$. If $z = y$, then z and x are distinct pieces and we can take $u = x$, and have $\text{size}(a_j) +$

CASE 1:



CASE 2:



CASE 3:

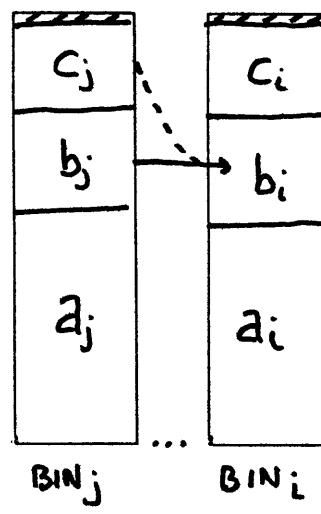


FIGURE 5.6. CASEs for CLAIM 5.9.8.

$\text{size}(z) + \text{size}(x) = \text{size}(a_j) + \text{size}(y) + \text{size}(x) \leq 1$, a contradiction. If $z \neq y$, then we can take $u = y$ and have $\text{size}(a_j) + \text{size}(z) + \text{size}(y) \leq \text{size}(a_j) + \text{size}(x) + \text{size}(y) \leq 1$, again a contradiction. Therefore z must be a B-piece.

Case 2. $i = j$.

Then $\text{piece}_{\text{PF}}(j, 2) = x$, and we can take $u = y$, since $\text{size}(a_j) + \text{size}(x) + \text{size}(y) \leq 1$. So this case is impossible.

Case 3. $i > j$.

Then by Lemma 1.2, $\text{size}(a_i) \leq \text{size}(a_j)$, so that we can take $v = y$ and have $\text{size}(a_i) + \text{size}(x) + \text{size}(y) \leq \text{size}(a_j) + \text{size}(x) + \text{size}(y) \leq 1$. So this case too is impossible.

So the only possible case is Case 1, and there we have z is a B-piece. \blacksquare

As an immediate corollary of Claim 5.9.8, we have

CLAIM 5.9.9. There cannot be a sequence of three Deficit Bins, each pointing to the next.

This is because the first in the series would have to contain a B-piece in P^* , and due to size constraints no bin containing both an A-piece and a B-piece has room for any more pieces with size exceeding $2/11$. Thus the first bin could not contain 3 pieces in P^* , and hence could not be a Deficit Bin. Note that this also means that there cannot be a loop, each piece of which is in a Deficit Bin in P^* .

We are now ready to assign responsibility for each Deficit Bin. Let

$$\mathcal{A} = \{a \in A\text{-PIECES: the } a\text{-bin is } \underline{\text{not}} \text{ a Deficit Bin}\}.$$

For each $a \in \mathcal{A}$, let $\text{RESPONSIBILITY}(a)$ be the smallest set of Deficit Bins containing

- (1) All Deficit Bins pointed to by the a -bin,
- (2) All Deficit Bins pointed to by members of $\text{RESPONSIBILITY}(a)$.

Similarly, for each piece $b \in \mathcal{B}$, let $\text{RESPONSIBILITY}(b)$ be the smallest set of Deficit Bins containing

- (1) The Deficit Bin pointed to by b , if there is one,
- (2) All Deficit Bins pointed to by members
of $\text{RESPONSIBILITY}(b)$.

CLAIM 5.9.10. Every Deficit Bin is in $\text{RESPONSIBILITY}(x)$
for some unique $x \in \Theta \cup \mathcal{A}$.

Proof of Claim. Every Deficit Bin is pointed to by some piece, and if we trace back the arrows in a diagram of P^* , we cannot by Claim 5.9.9 always find more Deficit Bins. Thus we either must come to a non-Deficit A-bin, or a piece which is not in an A-bin in P^* and hence is in Θ . \square

CLAIM 5.9.11.

If $a \in \mathcal{A}$ and the a-bin contains no B-piece in P^* , then

$$|\text{RESPONSIBILITY}(a)| \leq 2.$$

If $a \in \mathcal{A}$ and the a-bin contains a B-piece in P^* ,

$$|\text{RESPONSIBILITY}(a)| \leq 3.$$

If $b \in \Theta \cap \text{B-PIECES}$, $|\text{RESPONSIBILITY}(b)| \leq 1$.

If $b \in \Theta \cap \text{B-PIECES}$, $|\text{RESPONSIBILITY}(b)| \leq 3$.

Proof of Claim. If the a-bin contains no B-piece in P^* , then by Claim 5.9.8, any Deficit Bins it points to cannot themselves point to Deficit Bins. By Claim 5.9.2, the a-bin can

point to at most 2 Deficit Bins itself. If the a-bin did contain a B-piece, then it has room for no more pieces in P^* and so can point to at most one Deficit bin itself, however, that bin could point to two more, for a total of 3. These 2 could not point to any further ones by Claim 5.9.9.

If $b \in \emptyset$ is not a B-piece, then no Deficit Bin it might point to can point to any further Deficit Bins, by Claim 5.9.8. If b is a B-piece, then it can point to a Deficit Bin which points to two more, but by Claim 5.9.9 that is the end of the line. \square

We are now ready to define f and g. Actually, we will define four special purpose functions, f_1 , f_2 , g_1 , and g_2 , and will then set

$$f(x) = f_1(x) \cup f_2(x),$$

$$g(x) = g_1(x) + g_2(x).$$

The specific special purposes will be explained in a moment. First of all, certain elements of PIECES(L) can in essence be deleted from the domains of the functions. If x is in A-PIECES - α or $\beta - \theta$, we set

$$f(x) = f_1(x) = f_2(x) = \emptyset,$$

$$g(x) = g_1(x) = g_2(x) = 0.$$

In addition, the function f_1 will be null on elements of \mathcal{Q} . Its job is to map each $b \in \Theta$ to the piece in \mathcal{J} (if there is one) for which b is "directly responsible." By Claim 5.9.5, $b \in \Theta$ is the head of a maximal chain, unique since no piece can point to more than one other piece. $f_1(b)$ is defined to be zero- or one-element set

$$f_1(b) = \{\text{terminator of maximal chain headed by } b\}.$$

f_2 is used to account for the pieces in \mathcal{J} for which $x \in \mathcal{Q} \cup \mathcal{B}$ is "indirectly responsible," due to the Deficit Bins it is responsible for:

$$f_2(x) = \{\text{terminators of maximal chains headed by top pieces in Deficit Bins belonging to } \text{RESPONSIBILITY}(x)\}.$$

We thus have, by Claims 5.9.5, 5.9.6, and 5.9.10:

CLAIM 5.9.12. $f = f_1 \cup f_2$ obeys

$$(5.9A) \quad \bigcup_{x \in \text{PIECES}(L)} f(x) = \mathcal{J}.$$

The functions g_1 and g_2 count the A-bins that in a sense "collaborate" in the productions of pieces in $f(x)$ from pieces $x \in \Theta$ via f_1 and f_2 . g_2 is defined uniformly for all such x by

$$g_2(x) = (2/3) |\{ \text{maximal chains } \langle b_1, \dots, b_n \rangle \text{ with terminators} \\ \text{in } f(x), \text{ such that for some } i, 1 < i \leq n, \\ \text{and } j \leq A\text{-PIECES}, b_i = \text{piece}_{P^*}(j, 3) \}|.$$

The function g_1 is a little more ad hoc, but in all cases takes into account the number of pieces for which x is indirectly responsible: For $b \in \Theta$

$$g_1(b) = |f_2(b)|.$$

For $a \in \alpha$ when the a-bin does not contain a B-piece in P^* ,

$$g_1(a) = |f_2(a)| + 1/3.$$

For $a \in \alpha$ when the a-bin does contain a B-piece in P^* ,

$$g_1(a) = |f_2(a)| + 1.$$

This concludes the definition of g_1 and g_2 . To illustrate the definitions of both f and g , we present in Figures 5.7 thru 5.9 a series of possible packings P^* , with arrows drawn in as above, and the values of f_1 , f_2 , g_1 , and g_2 for the various pieces involved. Figures 5.7 and 5.8 give very simple examples focussing on the generation of $f_2(x)$ from x , while 5.9 gives a more complicated example where processes interact.

CLAIM 5.9.13. $g = g_1 + g_2$ obeys

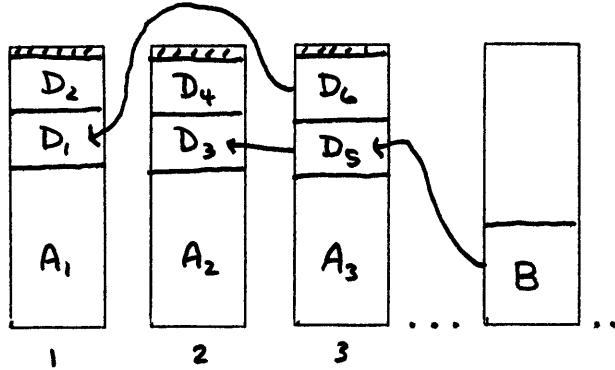
$$(5.9B) \quad \sum_{x \in \text{PIECES}(L)} g(x) \leq |\text{A-PIECES}|.$$

Proof of Claim. We may think of g_1 and g_2 as counting A-bins. Thus it will be sufficient to show that no A-bin is counted for more than a total of 1. g_1 counts both Deficit Bins and a-bins for $a \in \alpha$. The $|f_2(x)|$ part can be thought of as counting 1 for each Deficit Bin in $\text{RESPONSIBILITY}(x)$ whose 3rd piece heads a maximal chain with a terminator. The added $1/3$ or 1 if $x \in \alpha$ can be thought of as counting that much for the x -bin itself. g_2 only counts A-bins whose 3rd piece in P^* is pointed to. These, by definition of points to, must also contain a 3rd piece in PF and so are not Deficit Bins, so g_2 does not count Deficit Bins.

Thus Deficit Bins are counted only by g_1 , and since none is in $\text{RESPONSIBILITY}(x)$ for more than one x by Claim 5.9.10, no

P*:

Deficit Bins: $\text{BIN}_1, \text{BIN}_2, \text{BIN}_3$



$$f_1(B) = \{D_3\}$$

$$f_2(B) = \{D_1, D_2, D_4\}$$

$$g_1(B) = 3$$

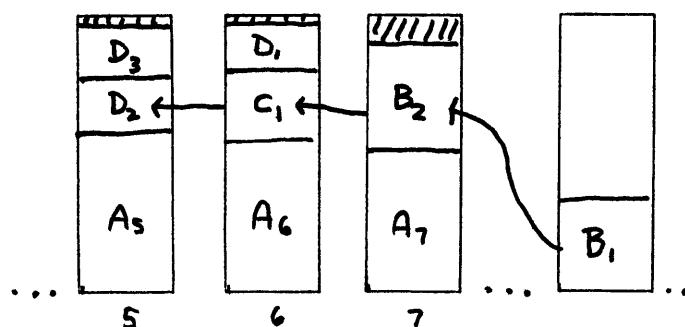
$$g_2(B) = 0$$

(assuming $D_1, D_2, D_3, D_4 \in \mathcal{D}_1$)

FIGURE 5.7. Illustration of definitions of f_1, f_2, g_1, g_2 .

P*:

Deficit Bins: $\text{BIN}_5, \text{BIN}_6$



$$f_1(A_7) = \emptyset$$

$$f_2(A_7) = \{D_1, D_3\}$$

$$g_1(A_7) = 3$$

$$g_2(A_7) = 0$$

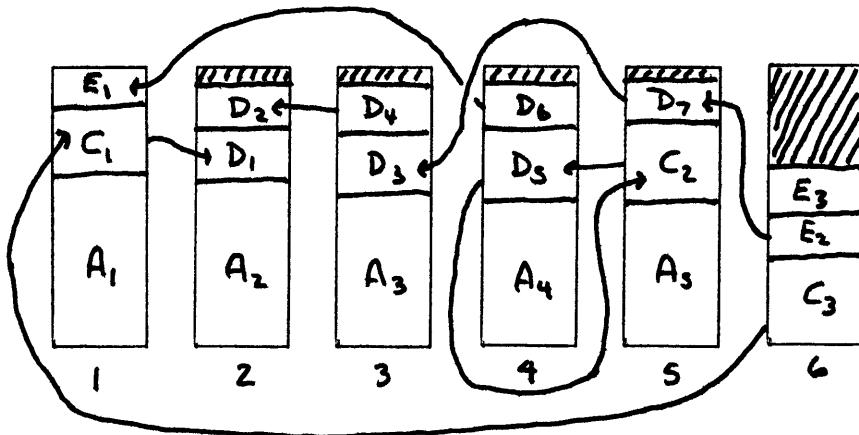
$$f_1(B_1) = \{D_2\}$$

$$f_2(B_1) = \emptyset$$

$$g_1(B_1) = g_2(B_1) = 0$$

(assuming $D_1, D_2, D_3 \in \mathcal{D}_1$)

FIGURE 5.8. Illustration of definitions.



Deficit Bins: $\text{BIN}_3, \text{BIN}_4$. $\mathcal{A} = \{A_1, A_2, A_3\}$

(Assume $D_1, D_2, D_3, E_1, E_3 \in \mathcal{D}$)

$$\begin{aligned} f_1(A_1) &= f_2(A_1) = \emptyset \\ g_1(A_1) &= 1/3 \\ g_2(A_1) &= 0 \end{aligned}$$

$$\begin{aligned} f_1(C_3) &= \{D_1\} \\ f_2(C_3) &= \emptyset \\ g_1(C_3) &= g_2(C_3) = 0 \end{aligned}$$

$$\begin{aligned} f_1(A_2) &= f_2(A_2) = \emptyset \\ g_1(A_2) &= 1/3 \\ g_2(A_2) &= 0 \end{aligned}$$

$$\begin{aligned} f_1(E_2) &= \{D_3\} \\ f_2(E_2) &= \emptyset \\ g_1(E_2) &= 0 \\ g_2(E_2) &= 2/3 \end{aligned}$$

$$\begin{aligned} f_1(A_5) &= \emptyset \\ f_2(A_5) &= \{D_2, E_1\} \\ g_1(A_5) &= 7/3 \\ g_2(A_5) &= 4/3 \end{aligned}$$

$$\begin{aligned} f_1(E_3) &= \{E_3\} \\ f_2(E_3) &= \emptyset \\ g_1(E_3) &= g_2(E_3) = 0 \end{aligned}$$

FIGURE 5.9. Illustration of definitions of f_1, f_2, g_1, g_2 .

Deficit Bin is counted for more than 1. If a non-Deficit A-bin is not counted by $g_2(x)$ for any x , then it is not counted for more than 1 by definition of g_1 . If it is counted by g_2 , then it must contain a 3rd piece in P^* , hence cannot contain a B-piece, so g_1 counts it for $1/3$. Since its 3rd piece can be in only one maximal chain, the bin can be counted by $g_2(x)$ for at most one x , and thus has a total count of at most $1/3 + 2/3 = 1$.

□

Claims 5.9.12 and 5.9.13 tell us that f and g obey Properties 5.9A and 5.9B, as required by Lemma 5.10. Thus all that remains for the proof of Theorem 5.9 is to show that Property 5.9C holds for all BIN in P^* . But first we shall prove a few more properties of f and g which will be of use in the case analysis to come.

CLAIM 5.9.14.

- (A) $b \in \Theta - B\text{-PIECES} \implies g_1(b) \leq 1$.
- (B) $b \in \Theta \cap B\text{-piece} \implies g_1(b) \leq 3$.
- (C) $a \in \mathcal{Q}$ and the a-bin contains no B-pieces in P^*
 $\implies g_1(a) \leq 7/3$.
- (D) $a \in \mathcal{Q}$ and the a-bin contains a B-piece in P^*
 $\implies g_1(a) \leq 4$.

Proof of Claim. Follows immediately from Claim 5.9.11 and the definition of g_1 . \square

CLAIM 5.9.15. If $\Theta \cap C\text{-PIECES} \neq \emptyset$, then

$$b \in \Theta \cap B\text{-pieces} \implies g_1(b) \leq 2.$$

Proof of Claim. The fact that there is a C-piece in a non-A-bin of PF means by Lemma 4.4 that for all A-bins BIN_j with room enough for two pieces besides the A-piece and hence $\text{size}(\text{piece}_{PF}(j,1)) < 4/11 < 2/3$, we must have $\text{size}(\text{piece}_{PF}(j,2)) > 1/4$. In particular, all Deficit Bins must thus contain pieces of this size in PF.

Now for a piece to point to a deficit bin, it must be in that bin in PF, by definition of points to. Thus no A-bin can contain in P^* two pieces which point to Deficit Bins, since both these pieces would have to have size exceeding $1/4$, which is impossible by size constraints.

Thus, if our B-piece b points to a Deficit Bin, under the Claim's hypothesis that bin can only point to one more, limiting $|IRESPONSIBILITY(b)|$ and hence $|f_2(b)|$ and $g_1(b)$ to at most 2. \square

The remaining Claims about f and g are all consequences of the SIZE LEMMA (Claim 5.9.7). The first follows immediately from that Lemma and the definition of g_2 :

CLAIM 5.9.16. For all $x \in \Theta \cup A$,

$g_2(x) \geq (2/3) |\{$ terminators in $f(x)$ which are larger than
than the heads of the maximal chains they
terminate $\}|.$

CLAIM 5.9.17.

(A) If $b \in \Theta \cap (B\text{-PIECES} \cup C\text{-PIECES})$, then

$y \in f_1(b) \implies \text{size}(y) \leq \text{size}(b).$

(B) If $b \in \Theta \cap (D\text{-PIECES} \cup E\text{-PIECES})$, then

$y \in f_1(b) \implies \text{size}(y) \leq 1/4.$

Proof of Claim. If no piece in the maximal chain headed by b is $\text{piece}_P^*(j, 3)$ for $j \leq |A\text{-PIECES}|$, the results are automatic by the SIZE LEMMA. If some piece in the chain is $\text{piece}_P^*(j, 3)$, we still must have $\text{size}(y) \leq$ the size of the last such piece, call it c . By ordered packing property (3), we must have $\text{size}(\text{piece}_P^*(j, 2)) \geq \text{size}(c)$. Since the total of these two size cannot be as great as $1/2$, we thus have $\text{size}(y) \leq \text{size}(c) < 1/4$. Thus (B) holds. (A) holds because under its hypothesis $\text{size}(b) > 1/4$. \square

CLAIM 5.9.18. If $b \in \mathcal{B} \cap B\text{-PIECES}$, $y \in f_1(b)$, and $g_1(b) \geq 1$, then $\text{size}(y) \leq 1/3$.

Proof of Claim. By definition of g_1 , b must point to a Deficit Bin, and in particular to $c = \text{piece}_{P^*}(j, 2)$ for that bin. Since the Deficit Bin contains three pieces in P^* , by size constraints we must have $\text{size}(c) < 1 - 1/2 - 2/11 = 7/22 < 1/3$. The argument for the previous Claim can then be used to show that either $\text{size}(y) \leq \text{size}(c)$ or $\text{size}(y) \leq 1/4$, either of which is sufficient. \square

CLAIM 5.9.19. If $y \in f_2(x)$ for any x , $\text{size}(y) \leq 1/4$.

Proof of Claim. The head of the chain of which y is the terminator is by definition of f_2 the 3rd piece in a Deficit Bin in P^* , and hence by size constraints has size $\leq 1/4$. The result follows as in Claim 5.9.17. \square

CLAIM 5.19.20. If \mathcal{B} contains a B-piece b' with $\text{size}(b') \leq 2/5$, then for all $x \in \mathcal{B} - B\text{-PIECES}$,

$$y \in f_2(x) \iff y \in E\text{-PIECES}.$$

Proof of Claim. By Lemma 4.4, any a-bin BIN_j with $\text{size}(a) \leq 3/5$ has piece $\text{piece}_{\text{PF}}(j,2) \in \text{B-PIECES}$. Since every a-bin with room for a C- and a D-piece, or two D-pieces, has $\text{size}(a) < 3/5$, the only A-bins which contain 3 pieces in PF, but do not contain a B-piece in PF, must have E-pieces for their 3rd piece in P*.

If $y \in f_2(x)$, then y must be the terminator of a maximum chain headed by the 3rd piece in a Deficit Bin pointed to by x , since by Claim 5.9.14 and the fact that x is not a B-piece, $|f_2(x)| \leq 1$. By the above the head of that chain must be an E-piece. If no subsequent piece in the chain is $\text{piece}_{\text{P}^*}(j,3)$ for any BIN_j , we are done by the SIZE LEMMA. If some such piece is in the chain, let c' be the last such, and BIN_j , the a' -bin, be the bin that contains c' in P^* . The piece that points to c' must be $\text{piece}_{\text{PF}}(j,3)$. We must have $\text{size}(a') > 3/5$, else $\text{piece}_{\text{PF}}(j,2)$ would be a B-piece and there wouldn't be room for a piece $\text{piece}_{\text{PF}}(j,3)$. But if $\text{size}(a') > 3/5$, we must have $\text{size}(c') \leq 1/5$, and so $\text{size}(y) \leq 1/5$ by the SIZE LEMMA and y is an E-piece.

□

With these properties of f_1 , f_2 , g_1 , and g_2 under our belts, we are now ready to face the case analysis.

PROOF OF THEOREM 5.9

III. Case Analysis

To complete our proof of Theorem 5.9, we must, by Lemma 5.10, show that for all non-empty BIN_j in the optimal packing P^* of L , the following property holds:

(5.9C) For $X = \text{cont}_{P^*}(j)$,

$$Wd[f(X)] + g(X) \leq (11/9)[y(X) + g(X)],$$

where y is 0 or 1 depending on whether BIN_j is an A-bin or not.

The case when BIN_j is an A-bin can be handled without further ado, and will serve as an introduction to the more complicated arguments required for the non-A-bins.

CLAIM 5.9.21. (5.9C) holds for all A-bins in the optimal packing P^* .

Proof of Claim. For BIN_j the a-bin, (5.9C) reduces to

$$Wd[f_2(a)] \leq (2/9)[g_1(a) + g_2(a)],$$

since the only pieces in A-bins in P^* for which $f(x) \neq \emptyset$ or $g(x) \neq 0$ are the A-pieces, and since $f_1(a)$ is by definition empty for $a \notin A\text{-PIECES}$.

There are basically only 4 cases to consider, depending on the value of g_1 . Let W stand for $W_d[f_2(a)]$:

1. $0 \leq g_1(a) < 4/3$. By definition of g_1 , this means we have $f_2(a) = \emptyset$, so the inequality is immediate.

2. $4/3 \leq g_1(a) < 7/3$. By definition of g_1 , this means that $f_2(a)$ contains one element, which by Claim 5.9.19 must be a D- or E-piece. But then $w_1[f_2(a)] \leq 1/4$, so $W \leq 1/4 < (2/9)(4/3) = 8/27$.

3. $7/3 \leq g_1(a) < 4$. In this case $f_2(a)$ contains two elements, which can be at most D-pieces, so $W \leq 2/4 < (2/9)(7/3) = 14/27$.

4. $g_1(a) = 4$. In this case $f_2(a)$ contains 3 pieces, all at most D-pieces, so $W \leq 3/4 < (2/9)(4) = 8/9$.

Note that if $a \notin A$, the first case automatically holds. Since if $a \notin A$, $g_1(a) \leq 4$ by Claim 5.9.14, this exhausts the possibilities. \square

The situation gets considerably more complicated if BIN_j is not an A-bin. First of all, there are more than one pieces in the bin for which f and g are non-vacuous. Second, in the light of Claims 5.9.14 and 5.9.15, it makes a difference whether \mathcal{G} contains C-pieces or not. So much a difference in fact, that we find it easier to treat each possibility as a separate case, and analyze each of the possible non-A-bins of P^* twice, once under each assumption. We shall denote the two possibilities as CASE I and CASE II:

CASE I. $\mathcal{G} \cap C\text{-PIECES} \neq \emptyset$. In this case all A-bins which contain two non-A-pieces in P^* will contain either a B- or C-piece as their second piece in PF by Lemma 4.4. Consequently, no D- or E-piece can point to a Deficit Bin, and Claim 5.9.15 applies.

CASE II. $\mathcal{G} \cap C\text{-PIECES} = \emptyset$. Then by Reduction 3 we know that $\mathcal{G} \cap D\text{-PIECES} \neq \emptyset$. Thus all A-bins which contain two non-A-pieces in P^* will contain a B-, C-, or D-piece as their second piece in PF, and no E-piece can point to a Deficit Bin.

We shall treat these two CASES separately, giving for each, as in the proof of Theorem 5.7, a complete enumeration of the legal configurations. If $\text{cont}_{P^*}(j) = \{x_1, \dots, x_n\}$ is in configuration $[x_1, \dots, x_n]$, we may rewrite (5.9C) as

$$(5.9C') Wd[f\{x_1, \dots, x_n\}] - (2/9)[g\{x_1, \dots, x_n\}] \leq 11/9,$$

and this is the inequality that we shall prove. Unfortunately, for each configuration there are still myriad possibilities to consider, depending on the values of the variables $f_1(x_i)$, $f_2(x_i)$, $g_1(x_i)$, and $g_2(x_i)$, $1 \leq i \leq n$. If we are ever to finish writing down this proof, we must find a way to avoid covering each of these possibilities separately. We do this by introducing the concept of the worst case settings of the variables.

The simplest example of this approach involves the weighting function w_1 . Suppose for $X \in \{B, C, D, E\}$ and $K \in \{I, II\}$ we had a bound $VAL_1(X, K)$ such that

$$(5.9D) \quad VAL_1(X, K) \geq \max \left\{ w_1[f(x)] - (2/9)g(x) : x \text{ is an } X\text{-piece and } f_1, f_2, g_1, \text{ and } g_2 \text{ obey the constraints imposed by their definitions, CASE } K, \text{ and Claims 5.9.14 through 5.9.20} \right\}.$$

Then we could conclude

CLAIM 5.9.22. Suppose $\text{cont}_p^*(j) \in [x_1, \dots, x_n]$, VAL1 obeys (5.9D), and $\sum_{l=1}^n \text{VAL1}(x_l, K) \leq 11/9$. Then property (5.9C') holds for BIN_j in CASE K.

Proof of Claim. By definition of Wd, f, g,

$$\begin{aligned} Wd[f\{x_1, \dots, x_n\}] &= (2/9)[g\{x_1, \dots, x_n\}] \\ &\leq \sum_{l=1}^n w1(f(x_l)) - (2/9) \sum_{l=1}^n g(x_l) \\ &\leq \sum_{l=1}^n [w1(f(x_l)) - (2/9)g(x_l)] \end{aligned}$$

and by (5.9D) for VAL1, this must be

$$\leq \sum_{l=1}^n \text{VAL1}(x_l) \leq 11/9. \quad \square$$

In order to calculate values for $\text{VAL1}(X, K)$ which will satisfy (5.9D), we use arguments about the possible settings of f_1, f_2, g_1 , and g_2 and their effect on $w1(f(x)) - (2/9)g(x)$ for x an X-piece, in order to derive the w1-worst case settings. These settings then yield $\text{VAL1}(X, K) = w1(f(x)) - (2/9)g(x)$. The w1-worst case settings for all $X \in \{B, C, D, E\}$ and $K \in \{1, 11\}$ are given in Table 5.1. The entries for f_1 and f_2 are streamlined by merely indicating the type and quantity of pieces in the

given set. For instance $f_2 = 2D$ means that $f_2(x)$ is a set consisting of two D-pieces. WT_1 stands for $w_1[f(x)]$ under the given settings, and $COST$ stands for $(2/9)g(x)$, so that $VAL_1 = WT_1 - COST$.

<u>CASE</u>	<u>X</u>	<u>g1</u>	<u>g2</u>	<u>f1</u>	<u>f2</u>	<u>WT1</u>	<u>COST</u>	<u>VAL1</u>
I	B	0	0	B	\emptyset	$1/2$	0	$1/2$
	C	1	0	C	D	$7/12$	$2/9$	$13/36$
	D	0	0	D	\emptyset	$1/4$	0	$1/4$
	E	0	0	E	\emptyset	$1/5$	0	$1/5$
II	B	0	0	B	\emptyset	$1/2$	0	$1/2$
	C	1	0	D	D	$1/2$	$2/9$	$5/18$
	D	1	0	D	D	$1/2$	$2/9$	$5/18$
	E	0	0	E	\emptyset	$1/5$	0	$1/5$

Table 5.1: The w_1 -worst case settings.

CLAIM 5.9.23. For each $X \in \{B, C, D, E\}$, $K \in \{I, II\}$, the entries for $\text{VAL}_1(X, K)$ in Table 5.1 obey property (5.9D).

Proof of Claim. Note that we do not claim that the settings given in the table can actually be realized by any piece in our list L , just that they yield a value $w_1(f(x)) - (2/9)g(x)$ obeying the desired property. The settings are derived using a relatively simple set of principles. Let X stand for the X -piece in question:

(P1.1) $g_2(X)$ should be 0.

For if $g_2(X) > 0$, it would contribute at least $(2/3)(2/9) = 4/27$ to COST, and can only contribute to WT1 by allowing $f_1(x)$ to be a bigger piece than X , (where we identify the singleton set $f_1(X)$ with its contents, an abuse of notation that will occur frequently in what follows). By Claim 5.9.17 this can only effect WT1 if $X = E$ and $f_1(E)$ a D-piece, in which case the gain to WT1 is only $1/4 - 1/5 = 1/20 < 4/27$, so setting $g_2(X) > 0$ can only decrease $WT1 - COST = VAL1$.

(P1.2) $f_2(X)$ should contain $g_1(X)$ D-pieces.

By definition $|f_2(X)| = g_1(X)$, and $f_2(X)$ can only contain D- or E-pieces by Claim 5.9.19. D-pieces add the most to WT1, at no charge to COST.

(P1.3) If $X \in \{C, D\}$, $g1(X)$ should take on its maximum value. If $X = B$ and $f1(B)$ is not a B-piece, $g1(B)$ should take on its maximum value.

For each addition of 1 to $g1(X)$, we add $2/9$ to COST, but we also can add a D-piece to $f2(X)$, for an increase in WT1 of $1/4$ and hence a net increase in $WT1 - COST$ of $1/36$. This argument breaks down for $X = B$ and $g1(B) = 0$, for if we added 1 to $g1$ in this situation, we would by Claim 5.9.18 have to demote $f1(B)$ from a B-piece to a C-piece, for an unbalanced loss to WT1 of $1/2 - 1/3 = 1/6$, a possibility we avoid if we assume that $f1(B)$ is not a B-piece to begin with.

(P1.4) Given the values of $g1(X)$ and $g2(X)$, $f1(X)$ should be the largest piece allowable by the constraints. This principle is automatic, since given $g1$ and $g2$, the COST is fixed.

Granted these principles, the table entries follow quickly:

$X = B$. By (P1.1), $g2(B) = 0$. If $f1(B)$ is a B-piece, then by Claim 5.9.18, $g1(B) = 0$ and $f2(B) = \emptyset$, and so

$WT1 = 1/2$, $COST = 0$, and $VAL1 = 1/2$,

as given in the table for both CASES. If on the other hand $f1(B)$ is not a B-piece, then in CASE I by (P1.3) and Claim 5.9.15 we should have $g1(B) = 2$, and so by (P1.2), $f2(B) =$ two D-pieces. By (P1.4) we should have $f1(B)$ a C-piece. In CASE II, we should have $f2(B) =$ three D-pieces by (P1.3) and Claim 5.9.14, but can only have $f1(B)$ a D-piece, since in CASE II there are no C-pieces in \mathcal{F} . This would thus give us

- (I) $WT1 = 1/3 + 2/4 = 5/6$, $COST = 4/9$, $VAL1 = 14/36 < 1/2$,
- (II) $WT1 = 1/4 + 3/4 = 1$, $COST = 6/9$, $VAL1 = 1/3 < 1/2$,

so the table entries for B in CASES I and II are correct.

$X \in \{C, D, E\}$. The entries for these X in both CASES are all derivable by straightforward applications of the principles, the definitions of the CASES, and Claim 5.9.14, and so the details will be omitted, thus concluding the proof of Claim 5.9.23. I

In addition to using the bounds provided by $VAL1$ in applying Claim 5.9.22, we can also use them to combine the treatment of a number of similar configurations. Recalling the definition of w1-domination in the proof of Theorem 4.7, we say

that configuration $[x_1, \dots, x_n]$ w1-dominates configuration $[y_1, \dots, y_n]$ if for each i , $1 \leq i \leq n$, $\text{VAL1}(x_i, K) \geq \text{VAL1}(y_i, K)$. Note that this does not depend on the value of K , since in both CASES,

$$\text{VAL1}(B) \geq \text{VAL1}(C) \geq \text{VAL1}(D) \geq \text{VAL1}(E).$$

We then immediately have

CLAIM 5.9.24. Suppose $[x_1, \dots, x_n]$ w1-dominates $[y_1, \dots, y_n]$ and $\sum_{i=1}^n \text{VAL1}(x_i, K) \leq 11/9$. Then if $\text{cont}_{p^*}(j) \in [y_1, \dots, y_n]$, (5.9C') holds for BIN_j in CASE K.

Unfortunately, $\sum_{i=1}^n \text{VAL1}(x_i, K)$ often exceeds $11/9$. As we saw in Theorem 5.7, it was for just such cases that w2 and Discounting were introduced into our definition of Wd. We shall now turn to the task of extending the idea of worst case settings to handle the possibility of discounting. But first, let us note that we can use principles (P1.1 thru P1.4) from the proof of Claim 5.9.22 to derive a number of conditional worst case settings. That is, granted a restriction R on the values for f_1 , f_2 , g_1 , and g_2 , we can still calculate an upper bound $\text{VAL}[R](X, K)$, satisfying

(5.9E) $\text{VAL}[R](X, K) \geq$

$\text{MAX}\left\{w_1(f(x)) - (2/9)g(x) : x \text{ is an } X\text{-piece, } f_1, f_2, g_1, g_2 \text{ obey } R \text{ and the constraints imposed by their definitions, CASE K, and Claims 5.9.14 thru 5.9.20}\right\}.$

As a corollary to the arguments in the proof of Claim 5.9.22, we have

CLAIM 5.9.25. For $K \in \{I, II\}$, the following conditional worst case settings and bounds obey (5.9E):

$$\text{VAL}[f_1(B) \in B\text{-PIECES}](B, K) = 1/2,$$

$$[f_1=B, f_2=\emptyset, g_1=g_2=0],$$

$$\text{VAL}[f_1(B) \notin B\text{-PIECES}](B, I) = 14/36,$$

$$[f_1=C, f_2=2D, g_1=2, g_2=0],$$

$$\text{VAL}[f_1(B) \notin B\text{-PIECES}](B, II) = 1/3,$$

$$[f_1=D, f_2=3D, g_1=3, g_2=0],$$

$$\text{VAL}[f_1(C) \in C\text{-PIECES}](C, I) = 13/36,$$

$$[f_1=C, f_2=D, g_1=1, g_2=0],$$

$$\text{VAL}[f_1(C) \notin C\text{-PIECES}](C, I) = 10/36,$$

$$[f_1=D, f_2=D, g_1=1, g_2=0],$$

$$\text{VAL}[f_1(D) \in D\text{-PIECES}](D, I) = 1/4,$$

$$[f_1=D, f_2=\emptyset, g_1=0, g_2=0],$$

$\text{VAL}[\{f_1(D) \in D\text{-PIECES}\}](D, 11) = 5/18,$
 $\quad [f_1=D, f_2=D, g_1=1, g_2=0],$
 $\text{VAL}[\{f_1(D) \notin D\text{-PIECES}\}](D, 1) = 1/5,$
 $\quad [f_1=E, f_2=\emptyset, g_1=g_2=0],$
 $\text{VAL}[\{f_1(D) \notin D\text{-PIECES}\}](D, 11) = 41/180,$
 $\quad [f_1=E, f_2=D, g_1=1, g_2=0],$
 $\text{VAL}[\{g_2(D) \geq 2/3\}](D, 1) = 1/4 - 4/27 = 11/108,$
 $\quad [f_1=D, f_2=\emptyset, g_1=0, g_2=2/3],$
 $\text{VAL}[\{g_2(D) \geq 2/3\}](D, 11) = 5/18 - 4/27 = 7/54,$
 $\quad [f_1=D, f_2=\emptyset, g_1=0, g_2=2/3],$
 $\text{VAL}[\{g_2(E) \geq 2/3\}](E, K) = 1/4 - 4/27 = 11/108,$
 $\quad [f_1=D, f_2=\emptyset, g_1=0, g_2=2/3].$

Returning to the question of how to take advantage of the possibility of discounting, recall that there are partitions of $f\{x_1, \dots, x_n\}$ other than the simple partition into one-element sets implicit in our use of w_1 . It is to our advantage to find pairs in $f\{x_1, \dots, x_n\}$ which obey Discounting Relations (the only Relations possible when all pieces have size $> 2/11$ are Relations 2, 3, and 4). In particular, if (x_i, x_j) obeys Relation k, it is possible that $(f_1(x_i), f_1(x_j))$ obeys the same Relation. Thus, in analogy with our definition of VAL_1 , we might wish to determine an upper bound $\text{VAL}_2(X, Y, K)$ satisfying

(5.9F) $VAL2(X, Y, K) \geq$

$$\text{MAX}\{w_1[f_2(x) \cup f_2(y)] + Wd\{f_1(x), f_1(y)\} - (2/9)g\{x, y\} :$$

x is an X-piece, y is a Y-piece, (x, y) obeys a
 Discounting Relation, and f_1, f_2, g_1, g_2 for x and y
 obey the constraints imposed by their definition,
 CASE K, and Claims 5.9.14 thru 5.9.20},

where $Wd\{f_1(x), f_1(y)\}$ will simply be $w_2(f_1(x), f_1(y))$, unless one of the two sets is empty (we are again identifying the singleton set with its contents).

$VAL2(X, Y, K)$ is only defined when it is possible that a pair made up of an X-piece and a Y-piece could obey a Discounting Relation. In that case our w_1 -worst case settings no longer need apply, since certain settings may now have the side-effect of either allowing, or not allowing, the discount to take place.

It is now that the conditional worst case settings can come into play. There are only a few possible dispositions of the discount, and each corresponds to a restriction R on f_1, f_2, g_1 , or g_2 for x or y. We can thus determine an upper bound for each possibility, and choose the largest for $VAL2$. This is what has been done in the construction of Table 5.2. There are no entries for $X = C$ in CASE II, since in that CASE $f_1(x)$ must be a D-piece, so that $(f_1(x), f_1(y))$ could never obey the same Relation as (x, y) . In the Table, WT2 stands for

<u>CASE</u>	<u>(X_i, X_j)</u>	<u>g₁</u>	<u>g₂</u>	<u>f₁</u>	<u>f₂</u>	<u>WT₂</u>	<u>COST</u>	<u>VAL₂</u>
I	(B, C) (2, 1)	(0, 0)	(C, C)	(2D, D)	17 -- 12	6 - 9	3 - 4	~ .75
	(B, D) (2, 0)	(0, 0)	(C, D)	(2D, Ø)	13 -- 12	4 - 9	23 -- 36	~ .64
	(B, E) (0, 0)	$\frac{2}{3}$	(B, D)	(Ø, Ø)	3 - 4	4 -- 27	65 --- 108	~ .60
	(C, D) (1, 0)	(0, 0)	$\frac{2}{3}$ (C, -D)	(D, Ø)	3 - 4	2 - 9	19 -- 36	~ .53
	(C, E) (1, 0)	(0, 0)	$\frac{2}{3}$ (C, -E)	(D, Ø)	43 -- 60	2 - 9	89 --- 180	~ .49
	(D, E) (0, 0)	(0, 0)	$\frac{3}{4}$ (D, -E)	(Ø, Ø)	2 - 5	0	2 - 5	~ .40
II	(B, C) (0, 1)	(0, 0)	$\frac{1}{2}$ (B, -D)	(Ø, D)	7 - 8	2 - 9	47 -- 72	~ .65
	(B, D) (0, 1)	(0, 0)	$\frac{1}{2}$ (B, -D)	(Ø, D)	7 - 8	2 - 9	47 -- 72	~ .65
	(B, E) (0, 0)	$\frac{2}{3}$	(B, D)	(Ø, Ø)	3 - 4	4 -- 27	65 --- 108	~ .60
	(D, E) (1, 0)	(0, 0)	$\frac{3}{4}$ (D, -E)	(D, Ø)	13 -- 20	2 - 9	77 --- 180	~ .43

Table 5.2. The w2-worst case settings.

$$w_1[f_2(x) \cup f_2(y)] + Wd\{f_1(x), f_1(y)\}, \text{ and } COST = g\{x, y\} \cdot (2/3)$$

An entry of $(2/3)D$ in the f_2 column indicates that under the w2-worst case settings, the pair $(f_1(x), f_1(y))$ obeys Relation 3, and hence the discount can be taken. If the $f_1(y)$ entry is not fractional, then under the given settings the discount has been cancelled, that is, $(f_1(x), f_1(y))$ is not a pair obeying the same Relation as did (x, y) .

CLAIM 5.9.26. The entries for VAL2 in Table 5.2 all satisfy property (5.9E).

Proof of Claim. In what follows we let X and Y stand for the X - and Y -pieces in question. For each line of the table, we must examine a maximum of four possibilities:

- (1) $(f_1(X), f_1(Y))$ obeys the same Discounting Relation as (X, Y) .
- (2) The discount was cancelled because $f_1(X)$ is not an X -piece, but is either empty or a piece of smaller size (the demotion of X).
- (3) The discount was cancelled because $\text{size}(f_1(X)) > \text{size}(X)$ (the promotion of X)
- (4) The discount was cancelled because $\text{size}(f_1(Y)) > \text{size}(Y)$ (the promotion of Y).

If none of (2), (3), (4) hold, and (X, Y) obeyed Relation k, then we must have $f_1(X)$ a k-piece, and $k \cdot \text{size}(f_1(X)) + \text{size}(f_1(Y)) \leq k \cdot \text{size}(X) + \text{size}(Y) \leq 1$, and so (1) must hold, so this is a complete list.

However, note that by Claim 5.9.17, promotions can only occur if X (or Y) is a D- or E-piece, and by Claim 5.9.16 a promotion of Z entails that $g_2(z) \geq 2/3$. This limits the applicability of (3) and (4) and in addition provides our needed restriction.

We shall now verify the entries in the Table, line by line:

$X = B, Y = C, K = 1$. Alternatives (3) and (4) are impossible. If, as in the table, (2) holds and the discount is cancelled by demoting B , we have that $f_1(B) \notin B\text{-PIECES}$. Thus the maximum contribution of B to VAL_2 is $\text{VAL}[f_1(B) \notin B\text{-PIECES}](B, 1) = 14/36$, and the maximum contribution of C is $\text{VAL}_1(C, 1) = 13/36$, for a total of $27/36 = 3/4$, as given in the Table.

This is the maximum possible and hence satisfies (5.9F), for suppose (1) holds. Then $f_1(B)$ must be a B -piece, so B 's maximum contribution is $\text{VAL}[f_1(B) \in B\text{-PIECES}](B, 1)$. Since $f_1(C)$ is discounted, its contribution will only be $(1/2)\text{w}_1(f_1(C))$, but this will still be maximized by taking the largest piece

allowable as in (P1.4), so the maximum contribution due to C will be $\text{VAL1}(C,1) - (1/2)(1/3)$ for a maximum total of $1/2 + 13/36 - 1/6 = 25/36 < 27/36 = 3/4$.

$X = B, Y = C, K = II$. Again only (1) and (2) are possible. If, as in the Table, (1) holds, the maximum total contribution is

$$\begin{aligned} \text{VAL}[f_1(B) \in B\text{-PIECES}](B,II) + \text{VAL1}(C,II) - 1/8 \\ = 1/2 + 5/18 - 1/8 = 56/72 - 9/72 = 47/72, \end{aligned}$$

as given in the Table. (The discount is $1/8$ since $f_1(C)$ can be at most a D-piece in CASE II.)

This is the maximum possible, since if (2) holds, the maximum total is

$$\begin{aligned} \text{VAL}[f_1(B) \notin B\text{-PIECES}](B,II) + \text{VAL1}(C,II) \\ = 1/3 + 5/18 = 11/18 = 44/72 < 47/72. \end{aligned}$$

$X = B, Y = D, K = I$. Cases (1), (2), and (4) are possible. The table entry corresponds to (2), in which case we have a bound of

$$\begin{aligned} \text{VAL}[f_1(B) \notin B\text{-PIECES}](B,I) + \text{VAL1}(D,I) \\ = 14/36 + 1/4 = 23/36, \text{ as given.} \end{aligned}$$

This is the worst possible since if (1) held we would have a bound of

$$\text{VAL}[f_1(B) \in B\text{-PIECES}](B,I) + \text{VAL1}(D) - 1/8$$

SECTION 5.3 - Page 323

$$= 1/2 + 1/4 - 1/8 = 5/8 = 45/72 < 46/72 = 23/36,$$

and if (4) held we would have a bound of

$$\begin{aligned} \text{VAL1}(B,1) + \text{VAL}[g_2(D) \geq 2/3](D,1) \\ = 1/2 + 11/108 = 65/108 < 69/108 = 23/36. \end{aligned}$$

$X = B$, $Y = D$, $K = II$. Since the w1-worst case settings for C and D in CASE II are the same, the w2-worst case settings here will be the same as for B, C, and II. Table 5.2 agrees.

$X = B$, $Y = E$, $K = I$. Again only (1), (2), and (4) are possible. The table entry is derived under the assumption that (4) holds, in which case we have a bound of

$$\begin{aligned} \text{VAL1}(B,1) + \text{VAL}[g_2(E) \geq 2/3](E,1) \\ = 1/2 + 11/108 = 65/108, \text{ as given.} \end{aligned}$$

This is the worst bound possible since if (1) held we would have a bound of

$$\begin{aligned} \text{VAL}[f_1(B) \in B\text{-PIECES}](B,1) + \text{VAL1}(E,1) - 1/10 \\ = 1/2 + 1/5 - 1/10 = 6/10 = 324/540 < 325/540 = 65/108, \end{aligned}$$

and if (2) held we would have a bound of

$$\begin{aligned} \text{VAL}[f_1(B) \notin B\text{-PIECES}](B,1) + \text{VAL1}(E,1) \\ = 14/36 + 1/5 = 53/90 = 318/540 < 325/540. \end{aligned}$$

$X = B, Y = E, K = 11$. Again only (1), (2), and (4) are possible. The Table entry is again under the assumption that (4) holds, in which case the bound is

$$\begin{aligned} \text{VAL1}(B) + \text{VAL}[g_2(E, 11) \geq 2/3](E, 11) \\ = 1/2 + 11/108 = 65/108, \text{ as given.} \end{aligned}$$

This is the worst possible, since if (1) holds the bound is

$$\begin{aligned} \text{VAL}[f_1(B) \in B\text{-PIECES}](B, 11) + \text{VAL1}(E, 11) \\ = 1/2 + 1/5 - 1/10 = 6/10 < 65/108, \end{aligned}$$

and if (2) holds the bound is

$$\begin{aligned} \text{VAL}[f_1(B) \notin B\text{-PIECES}](B, 11) + \text{VAL1}(E, 11) \\ = 1/3 + 1/5 = 8/15 = 288/540 < 325/540 = 65/108. \end{aligned}$$

$X = C, Y = D, K = 1$. Again only (1), (2), and (4) are possible. The Table entry assumes (1) holds, in which case the bound is

$$\begin{aligned} \text{VAL}[f_1(C) \in C\text{-PIECES}](C, 1) + \text{VAL1}(D) - (1/3)(1/4) \\ = 13/36 + 1/4 - 1/12 = 19/36, \text{ as given.} \end{aligned}$$

This is the worst possible, since if (2) holds the bound is

$$\begin{aligned} \text{VAL}[f_1(C) \notin C\text{-PIECES}](C, 1) + \text{VAL1}(D) \\ = 10/36 + 1/4 = 19/36, \end{aligned}$$

and if (4) holds the bound is

$$\begin{aligned} \text{VAL1}(C, 1) + \text{VAL}[g_2(D) \geq 2/3](D, 1) \\ = 13/36 + 11/108 = 50/108 < 57/108 = 19/36. \end{aligned}$$

$X = C, Y = E, K = I$. Again the only possibilities are (1), (2), and (3) and the Table assumes (1), in which case the bound is

$$\begin{aligned} \text{VAL}[f_1(C) \in C\text{-PIECES}](C, I) + \text{VAL}_1(E, I) &= (1/3)(1/5) \\ &= 13/36 + 1/5 - 1/15 = 89/180, \text{ as given.} \end{aligned}$$

This is the worst possible, since if (2) holds the bound is

$$\begin{aligned} \text{VAL}[f_1(C) \notin C\text{-PIECES}](C, I) + \text{VAL}_1(E, I) \\ &= 10/36 + 1/5 = 86/180 < 89/180, \end{aligned}$$

and if (4) holds the bound is

$$\begin{aligned} \text{VAL}_1(C, I) + \text{VAL}[g_2(E) \geq 2/3](E, I) \\ &= 13/36 + 11/108 = 50/108 = 250/540 < 267/540 = 89/180. \end{aligned}$$

$X = D, Y = E, K = I$. All four possibilities may occur, but the Table assumes that (1) holds, in which case we have a bound of

$$\begin{aligned} \text{VAL}[f_1(D) \in D\text{-PIECES}](D, I) + \text{VAL}_1(E, I) &= (1/4)(1/5) \\ &= 1/4 + 1/5 - 1/20 = 2/5, \text{ as given.} \end{aligned}$$

This is the worst possible since if (2) holds we have a bound of

$$\begin{aligned} \text{VAL}[f_1(D) \notin D\text{-PIECES}](D, I) + \text{VAL}_1(E, I) \\ &= 1/5 + 1/5 = 2/5, \end{aligned}$$

if (3) holds we have a bound of

$$\begin{aligned} \text{VAL}[g_2(D) \geq 2/3](D, I) + \text{VAL}_1(E, I) \\ &= 11/108 + 1/5 = 163/540 < 216/540 = 2/5, \end{aligned}$$

and if (4) holds the bound is

$$\begin{aligned} \text{VAL1}(D, 1) + \text{VAL}[g_2(E) \geq 2/3](E, 1) \\ = 1/4 + 11/108 = 38/108 = 190/540 < 216/540 = 2/5. \end{aligned}$$

$X = D$, $Y = E$, $K = 11$. Again the Table assumes (1) holds, in which case the bound is

$$\begin{aligned} \text{VAL}[f_1(D) \notin D\text{-PIECES}](D, 11) + \text{VAL1}(E, 11) - (1/4)(1/5) \\ = 5/18 + 1/5 - 1/20 = 77/180, \text{ as given.} \end{aligned}$$

This is the worst possible since in (2), (3), and (4) the bounds are

$$\begin{aligned} \text{VAL}[f_1(D) \notin D\text{-PIECES}](D, 11) + \text{VAL1}(E, 11) \\ = 41/180 + 1/5 = 77/180, \\ \text{VAL}[g_2(D) \geq 2/3](D, 11) + \text{VAL1}(E, 11) \\ = 7/54 + 1/5 = 178/540 < 231/540 = 77/180, \text{ and} \\ \text{VAL1}(D, 11) + \text{VAL}[g_2(E) \geq 2/3](E, 11) \\ = 5/18 + 11/108 = 205/540 < 231/540 = 77/180. \end{aligned}$$

Thus all the Table entries for VAL2 obey (5.9F) and Claim 5.9.26 is proved. \square

We apply Table 5.2 to the proof of property (5.9C') with an analogue of Claim 5.9.22 (the proof is also an analogue and hence is omitted):

CLAIM 5.9.27. Suppose $\text{cont}_{p^*}(j) \in [x_1, \dots, x_n]$ contains disjoint pairs (x_{i_m}, x_{j_m}) , $1 \leq m \leq M$, each satisfying a Discounting Relation, and that

$$\sum_{m=1}^M \text{VAL2}(x_{i_m}, x_{j_m}, K) + \sum_{i \neq i_m, j_m} \text{VAL1}(x_i, K) \leq 11/9.$$

Then property (5.9C') holds for BIN_j in CASE K.

Claims 5.9.22 and 5.9.27 will take care of all but 5 of the legal configurations we must consider, two in CASE I and three in CASE II. For these we will have to consider subcases, depending on the settings of the parameters, and derive upper bounds by arguments about conditional worst case settings, specifically tailored to each subcase.

We now introduce the terminology and abbreviations we shall use in the case analysis. LHS will stand for the left hand side of inequality (5.9C'). Our goal is to show that LHS $\leq 11/9$ for each legal configuration. VAL1, when written without an argument, will stand for the upper bound on LHS obtained by using Claim 5.9.22 and the w1-worst case settings from Table 5.1. VAL2, when written without an argument, will stand for the upper bound on LHS obtained by using Claim 5.9.27 and the w2-worst case settings from Table 5.2. Since we will be doing a separate enumeration for each CASE, we shall drop the argument

referring to CASE when we write VAL1 and VAL2 with arguments. As in the proof of Theorem 5.7, we shall write the configuration $[x_1, \dots, x_n]$ as $|x_1, \dots, x_n|$, and if $\{x_1, \dots, x_n\}$ is in that configuration, we shall refer to piece x_i as $x_{i|i}$, and identify the piece with its size.

CLAIM 5.9.28. (5.9C') holds for all non-empty, non-A-bins in P^* , in both CASE I and CASE II.

Proof of Claim. We proceed by an enumeration of all legal configurations, one complete enumeration for each CASE. Recall that the requirement for a given configuration $[x_1, \dots, x_n]$ to be legal is that

$$u = \sum_{i=1}^n u(x_i) < 1,$$

where u is defined as in Theorem 5.7. We begin with CASE I:

CASE I: $C \cap \mathcal{B} \neq \emptyset$ One-Piece Bins:

The class of such bins is w1-dominated by $[B_1]$, and so Claim 5.9.24 applies, yielding $VAL1 \leq VAL1(B) = 1/2 < 11/9$.

Two-Piece Bins:

This class is w1-dominated by $[B_1, B_2]$, yielding $VAL1 \leq 2(VAL1(B)) = 1 < 11/9$.

Three-Piece Bins:

1) Two or more B's:

$[B_1, B_2, B_3]$. This is impossible since $u = 3/3 = 1$.

$[B_1, B_2, C_3]$. $VAL1 = 2/2 + 13/36 = 49/36 > 11/9$.

Moreover, even though we have that (B_2, C_3) obeys Relation 2, $VAL2 = VAL1(B) + VAL2(B, C) = 1/2 + 3/4 = 5/4 > 11/9$. So we must resort to subcases:

SUBCASE 1: $f_1(B_1) \notin B\text{-PIECES}$.

In this case we can replace $VAL1(B)$ by $7/18 =$

$\text{VAL}[\text{f1}(B) \notin \text{B-PIECES}](B)$ as given in Claim 5.9.26, yielding
 $\text{LHS} \leq 7/18 + 3/4 = 41/36 < 11/9.$

SUBCASE 2: $B' = \text{f1}(B_1) \in \text{B-PIECES}.$

By Claim 5.9.17A, $B' \leq B_1 \leq 1-B_2-C_3 < 1-(1/3)-(1/4) = 5/12.$

We now are in a situation analogous to that of Claim 5.9.20.

Since B' would not fit in any A-bin, an A-bin which contains a C-piece in PF must have a gap above its A-piece of less than $5/12$, and so, since $u(C) + u(D) = 1/4 + 1/5 = 9/20 > 5/12$, cannot contain a C and a D in P^* . Thus if the settings of $g_1(C_3)$ and $g_2(C_3)$ are as advocated in Table 5.2, $f(C_3)$ can be at most $\{C, E\}$ or $\{D, D\}$ instead of $\{C, D\}$, reducing the contribution of B_2 and C_3 by at least $1/20$ to a value of at most $\text{VAL2}(B, C) - 1/20 = 7/10$. Therefore, in this case the worst case settings for C_3 will actually have

$g_1(C_3) = 0$, and $f(C_3) = \{C\}$, (the settings for B_2 remain as in the Table, and the discount is still cancelled). The contribution of B_2 and C_3 is then $7/18 + 1/3 = 13/18$. But now we have $\text{LHS} \leq \text{VAL1}(B) + 13/18 = 1/2 + 13/18 = 22/18 = 11/9.$

$[B_1, B_2, X_3]$, for $X = D$ or E .

No matter what X is, (B_2, X_3) obeys Relation 2, so $\text{LHS} \leq \text{VAL1}(B) + \text{VAL2}(B, X) \leq 1/2 + 23/36 = 41/36 < 11/9.$

2. Bins with one or fewer B's: This, the class of all remaining three-piece configurations, is clearly w1-dominated by $[B_1, C_2, C_3]$, for which $VAL1 = 1/2 + 2(13/36) = 22/18 = 11/9$.

Four-Piece Bins:

1) Bins with two or more B's: Impossible, since the remaining pieces must be at least E's so $u \geq 2/3 + 4/11 = 34/33 > 1$.

2) Bins with one B, two or more C's: Impossible, since $u \geq 1/3 + 2/4 + 2/11 = 67/66 > 1$.

3) Legal configurations with one B:

$[B_1, C_2, D_3, D_4]$. $VAL1 = 1/2 + 13/36 + 2/4 = 49/36 > 11/9$.

However, $B_1+D_4 \leq 1-(1/4)-(1/5) = 11/20$, so $2B_1+D_4 \leq 11/10 - 1/5 < 1$, and (B_1, D_4) obeys Relation 2. Also, $C_2+D_3 \leq 1-(1/3)-(1/5) = 7/15$, so that $3C_2+D_3 \leq 7/5 - 2/5 = 1$, and (C_2, D_3) obeys Relation 3. Thus LHS $\leq VAL2(B, D) + VAL2(C, D) \leq 23/36 + 19/36 = 42/36 < 11/9$.

$[B_1, C_2, X_3, E_4]$, for $X = D$ or E .

No matter what X is, $B_1+X_3 \leq 1-(1/4)-(2/11) = 25/44$, and so $2B_1+X_3 \leq 25/22 - 2/11 < 1$, and (B_1, X_3) obeys Relation 2,

so that $LHS \leq 13/36 + 1/5 + 23/36 = 6/5 < 11/9$.

$[B_1, D_2, D_3, D_4]$. $VAL1 = 1/2 + 3/4 = 5/4 > 11/9$. However,
 $B_1+D_2 \leq 1-2(1/5) = 3/5$, so $2B_1+D_2 \leq 6/5 - 1/5 = 1$, and
 (B_1, D_2) obeys Relation 2. Thus $LHS \leq 2/4 + 23/36$
 $= 41/36 < 11/9$.

$[B_1, D_2, D_3, E_4]$, $[B_1, D_2, E_3, E_4]$, and $[B_1, E_2, E_3, E_4]$.

This set of configurations is w_1 -dominated by the first, for which $VAL1 = 1/2 + 2/4 + 1/5 = 6/5 < 11/9$.

4) Bins with no B's, three or more C's:

$[C_1, C_2, C_3, C_4]$. Impossible, since $u = 4/4 = 1$.

$[C_1, C_2, C_3, D_4]$. $VAL1 = 39/36 + 1/4 = 4/3 > 11/9$.
 (C_3, D_4) obeys Relation 3. However, $2VAL1(C) + VAL2(C, D) =$
 $13/18 + 19/36 = 45/36 > 11/9$, so we must examine subcases,
observing first that by Claim 5.9.14 no $g_1(C_i) > 1$:

SUBCASE 1: $g_1(C_i) = 0$ for at least one of the C-pieces.

It is easy to verify that $VAL[g_1(C) = 0](C) = 1/3$, a decline
from the total contribution of the C_i 's to the above mentioned
 $45/36$ of $13/36 - 1/3 = 1/36$ for each C_i with $g_1(C_i) = 0$. Thus,
even if only one C_i is so impaired, $LHS \leq 44/36 = 11/9$.

SUBCASE 2: All $g_1(C_i)$'s = 1, some $g_2(X_i) \geq 2/3$.

This immediately adds $4/27$ to the COST we had in the worst case settings that yielded $VAL_2 \leq 45/36$, and the most it can add to the WT is $(1/3)(1/4) = 1/12$, by cancelling the discount that is allowed by the w2-worst case settings for C3 and D4. Thus the bound on LHS is reduced by at least $4/27 - 1/12 = 7/108$, to a value of at most $45/36 - 7/108 = 128/108 < 11/9$.

SUBCASE 3: All $g(C_i)$'s = 1, $g_2(X_i)$'s = 0, and at least one of the following holds:

- A) $f_1(C_i)$ does not contain a C-piece, for some C_i ,
- B) $f_2(C_i)$ does not contain a D-piece, for some C_i ,
- C) $f_1(D_4)$ does not contain a D-piece.

Since the g_1 and g_2 settings are all identical to the original worst case settings that yielded $VAL_2 = 45/36$, the total COST remains the same. But since one of the three possibilities must hold, the total WT must be reduced by at least the $(2/3)(1/4 - 1/5) = 1/30$ needed to demote a $1/3$ discounted D to a $1/3$ discounted E, so $LHS \leq 45/36 - 1/30 < 44/36 = 11/9$.

SUBCASE 4: All $g_1(C_i)$'s = 1, all $g_2(X_i)$'s = 0, and none of A), B), or C) holds.

Since $g_1(C_3) = 1$, C3 went in a Deficit Bin in PF. Since any C-piece will fit as the first non-A piece in a Deficit Bin, we can conclude that no C-piece in \mathcal{F} exceeds C3. Thus $3f_1(C_1) \leq 3C_3$, and since $f_1(D_4) \leq D_4$ by Claim 5.9.16,

$(f_1(C_1), f_1(D_4))$ obeys Relation 3. ($f_i(C_1)$ is a C-piece by hypothesis). However, by the same Claim we have that $f_2(C_3)$ does not exceed the top piece in P^* in the Deficit Bin into which C_3 goes, and $f_1(C_3)$ does not exceed the bottom piece by the SIZE LEMMA. Since $f_1(D_4) + C_3$ was too big for the gap in that A-bin (else $f_1(D_4)$ would have gone in the bin rather than a non-A-bin in PF), we thus have $f_1(C_3) + f_2(C_3) \leq C_3 + f_1(D_4)$, and so $3f_1(C_3) + f_2(C_3) \leq 3C_3 + f_1(D_4) \leq 3C_3 + D_4 \leq 1$. So $(f_1(C_3), f_2(C_3))$ also obeys Relation 3. Thus the total WT is made up of 3 C-pieces, 2 normal D's, and 2 D's discounted by $1/3$, and LHS $\leq 3/3 + 2/4 + (2/3)(2/4) - 3(2/9) = 11/6 - 2/3 = 7/6 < 11/9$.

$[C_1, C_2, C_3, E_4]$. $VAL1 = 3(13/36) + 1/5 = 77/60 > 11/9$. However, (C_3, E_4) obeys Relation 3, so LHS $\leq VAL2 = 2(13/36) + 89/180 \leq 219/180 < 22/18 = 11/9$.

5) Bins with no B's, two or fewer C's: This, the class of all remaining four-piece bins, is clearly w1-dominated by $[C_1, C_2, D_3, D_4]$, for which $VAL1 = 2(13/36) + 2/4 = 44/36 = 11/9$.

Five-Piece Bins:

- 1) Bins with one or more B's: Impossible, since we would have $u \geq 1/3 + 8/11 = 35/33 > 1$.
- 2) Bins with no B's, two or more C's: Impossible, since we would have $u \geq 2/4 + 6/11 = 46/44 > 1$.
- 3) Bins with no B, one C, and two or more D's: Impossible, since then $u \geq 1/4 + 2/5 + 4/11 = 223/220 > 1$.
- 4) Legal configurations with one C: By the above, this class is w1-dominated by $[C1, D2, E3, E4, E5]$, for which $VAL1 = 13/36 + 1/4 + 3/5 = 109/90 < 11/9$.
- 5) Bins with no B's, no C's, five D's: Impossible, since then $u = 5/5 = 1$.
- 6) Bins with no B's, no C's, no more than four D's: This, the class of all remaining five-piece bins, is clearly w1-dominated by $[D1, D2, D3, D4, E5]$, for which $VAL1 = 4/4 + 1/5 = 6/5 < 11/9$.

CASE II: $C \cap J = \emptyset$ One-Piece Bins:

This class is w1-dominated by $[B_1]$, for which
 $VAL1 = 1/2 < 11/9$.

Two-Piece Bins:

This class is w1-dominated by $[B_1, B_2]$, for which
 $VAL1 = 2/2 = 1 < 11/9$.

Three-Piece Bins:

1) Two or more B's:

$[B_1, B_2, B_3]$. Impossible - See CASE I.

$[B_1, B_2, X_3]$, for $X = C, D, \text{ or } E$.

No matter what X is, (B_2, X_3) obeys Relation 2, LHS \leq
 $VAL1(B) + VAL2(B, X) \leq 1/2 + 47/72 = 83/72 < 11/9$.

2) Bins with no more than one B: This, the class of all remaining three-piece configurations, is clearly w1-dominated by $[B_1, C_2, C_3]$, for which $VAL1 = 1/2 + 2(5/18)$
 $= 19/18 < 11/9$.

Four-Piece Bins:

- 1) Bins with two or more B's: Impossible - See CASE I.
- 2) Bins with one B, two or more C's: Impossible - See CASE I.
- 3) Legal configurations with one B:

$[B_1, C_2, D_3, X_4]$, for $X = D$ or E .

No matter what X is, $B_1+X_4 \leq 1-(1/4)-(1/5) \leq 11/20$, so
 $2B_1+X_4 \leq 11/10 - 2/11 < 1$ and (B_1, X_4) obeys Relation 2, and
 $VAL2 \leq 5/18 + 5/18 + 47/72 = 87/72 < 11/9$.

$[B_1, X_2, E_3, E_4]$, for $X = C, D$, or E .

No matter what X is, we still have $VAL1 = 1/2 + 5/18 + 2/5 =$
 $106/90 < 11/9$.

$[B_1, D_2, D_3, D_4]$. $VAL1 = 1/2 + 3(5/18) = 4/3 > 11/9$.

However, $B_1+D_2 \leq 1-(2/5) = 3/5$, so $2B_1+D_2 \leq 6/5 - 1/5 = 1$ and (B_1, D_2) obeys Relation 2. Thus LHS $\leq VAL2 = 47/72 + 2(5/18) = 87/72 < 11/9$.

$[B_1, D_2, D_3, E_4]$. $VAL1 = 1/2 + 10/18 + 1/5 = 113/90 > 11/9$.

Since no pair of the X_i 's necessarily obeys any of the Discounting Relations, we must again resort to subcases:

SUBCASE 1: $f_1(B_1) \notin B\text{-PIECES}$.

In this case we can replace $VAL1(B)$ by $1/3 = VAL[f_1(B) \notin B\text{-PIECES}](B)$, reducing the contribution of B_1 to $VAL1$ by $1/2 - 1/3 = 1/6$ and yielding $LHS \leq 113/90 - 15/90 = 98/90 < 11/9$.

SUBCASE 2: $f_1(B_1)$ is a B -piece $B_o \leq 2/5$.

Then by Claim 5.9.20, $f_2(D_2)$ and $f_2(D_3)$ must contain E 's instead of D 's, if, indeed, they are non-empty. With this the case, it is easy to see that in the worst case they will be empty, and $g_1(D_2) = g_1(D_3) = 0$. Thus $VAL[B_o \leq 2/5](D) = 1/4$, a decline of $5/18 - 1/4 = 1/36$ from the Table values, and $LHS \leq 113/90 - 2/36 = 108/90 < 11/9$.

SUBCASE 3: $f_1(B_1)$ contains a B -piece $B_o > 2/5$, and some $g_2(X_i) \geq 2/3$.

The most that having some $g_2(X_i)$ be non-zero can add to WT is the $1/20$ obtained by promoting $f_1(E_4)$ from a D to an E , since

there are no discounts to cancel. But the increase in COST is at least $4/27$, for a net loss of $4/27 - 1/20 = 53/540$, so $LHS \leq 113/90 - 53/540 = 625/540 = 105/90 < 11/9$.

SUBCASE 4: $f_1(B_1)$ contains a B-piece $B_0 > 2/5$, for all $X_i g_2(X_i) = 0$, and $f_1(E_4) = \emptyset$.

This case has $1/5$ taken from the WT as given in the Table, at no corresponding decrease in the cost, so $LHS \leq 113/90 - 1/5 = 95/90 < 11/9$.

SUBCASE 5: $f_1(B_1)$ contains a B-piece $B > 2/5$, for all $X_i g_2(X_i) = 0$, and $f_1(E_4) \neq \emptyset$.

Then $f_1(E_4)$ must contain an E-piece $\leq E_4$, and so $D_i + f_1(E_4) \leq D_i + E_4 \leq 1 - (2/5) - (1/5) = 2/5$, for $i = 2$ or 3 . So if D_i went in a Deficit Bin in PF, since $f_1(E_4)$ did not fit on top of it but went to a non-A-bin, the original gap must have been $\leq 2/5$, and since $g_2(D_i) = 0$, $f_2(D_i)$ can only contain an E-piece if it is non-empty. Proceed as in Subcase 1.

4) Bins with no B-pieces: This, the class of all remaining four-piece configurations, is clearly w1-dominated by $[C_1, C_2, C_3, C_4]$, which, even if it were legal, still has $VAL1 = 4(5/18) = 10/9 < 11/9$.

Five-Piece Bins:

- 1) Bins with one or more B's: Impossible - See CASE I.
- 2) Bins with two or more C's: Impossible - See CASE I.
- 3) Bins with one C, two or more D's: Impossible - See CASE I.
- 4) Legal configurations with at least three E's: By the above, this class contains all legal configurations containing a C-piece, and is clearly w1-dominated by $[C1, D2, E3, E4, E5]$, for which $VAL1 = 2(5/18) + 3/5 = 52/45 < 11/9$.
- 5) Bins with no B's, no C's, two or fewer E's:

$[D1, D2, D3, D4, D5]$. Impossible - See CASE I.

$[D1, D2, D3, D4, E5]$. $VAL1 = 4(5/18) + 1/5 = 59/45 > 11/9$. Even though $(D4, E5)$ obeys Relation 4, we still have $VAL2 = 3(5/18) + 77/180 = 227/180 > 11/9$, so we must again resort to subcases:

SUBCASE 1: Some $g2(x_i) \geq 2/3$.

This adds at least $4/27$ to COST, while adding at most $1/20$ to the VAL1 WT, by possibly promoting $f1(E5)$ to a D-piece.

Thus LHS $\leq 59/45 + 1/20 - 4/27 = (708+27-80)/540 = 655/540 < 660/540 = 11/9.$

SUBCASE 2: All $g_2(X_i)$'s = 0, $f_1(E_5) = \emptyset$.

This reduces the VAL1 WT by $1/5$ without reducing COST, and so LHS $\leq 59/45 - 1/5 = 50/45 < 11/9.$

SUBCASE 3: All $g_2(X_i)$'s = 0, $f_1(E_5) \neq \emptyset$.

Then $f_1(E_5)$ is an E-piece $\leq E_5$, and for each $i \leq 4$, $f_1(E_5) + D_i \leq E_5 + D_i \leq 1 - (3/5) = 2/5$. Thus, if D_i goes in a Deficit Bin in PF, since $f_1(E_5)$ did not fit on top of it in that bin, the original gap must have been $\leq 2/5$. Proceeding as we did in subcase 2 for $[B_1, D_2, D_3, E_4]$, we can thus conclude $VAL[Subcase 3](D) = 1/4$, a decline from $VAL1(D)$ of $5/18 - 1/4 = 1/36$, leaving LHS $\leq 118/90 - 4(1/36) = 108/90 < 11/9.$

$[D_1, D_2, D_3, E_4, E_5]$. $VAL1 = 3(5/18) + 2/5 = 37/30 > 11/9.$

Since no Discounting Relations necessarily hold between any pairs from this configuration, we must for the last time look at subcases:

SUBCASE 1: Some $g_2(X_i) \geq 2/3$.

This again can add at most $1/20$ to WT for each $4/27$ added to COST, and so LHS $\leq 37/30 + 1/20 - 4/27 = (666+27-80)/540 = 613/540 < 660/540 = 11/9.$

SUBCASE 2: All $g_2(X_i)$'s = 0, some $f_1(E_i) = \emptyset$.

This again cause a dead loss to LHS of at least $1/5$ and so

$LHS \leq 37/30 - 1/5 = 31/30 < 11/9.$

SUBCASE 3: All $g_2(x_i)$'s = 0, no $f_1(E_i) = \emptyset$.

If both $D_1+E_4 > 2/5$ and $D_2+E_5 > 2/5$, we would have $D_3 < 1/5$, which is impossible for a D-piece, So say $D_1+E_4 \leq 2/5$.

Then we have $D_1+f_1(E_4) \leq 2/5$, so if D_1 went in a Deficit Bin in PF the gap was $\leq 2/5$ and $f_2(D_1)$ was at most an E-piece, making us get a larger LHS by setting $g_1(D_1)$ to 0 and $f_2(D_i)$ to \emptyset . But this reduces D_1 's contribution to LHS by $5/18 - 1/4 = 1/36$ from the Table value, and we have $LHS \leq 37/30 - 1/36 = 217/180 < 11/9$.

Since this exhausts the possible five-piece configurations, we are now done with CASE II as well as CASE I, Claim 5.9.28 is proven. \square

We now can conclude from Claims 5.9.28 and 5.9.21 that property (5.9C) holds for all non-empty BIN_j in P^* . Since f and g obey (5.9A) and (5.9B) by Claims 5.9.12 and 5.9.13, Lemma 5.10 tells us that $FF(L) \leq (11/9)L^* + 4$, and so Theorem 5.9 is proved. \square

Now although the weighting function W_d "works" for arbitrary AAF algorithms in the sense that Lemma 5.5 applies, the properties of f_1 , f_2 , g_1 , and g_2 , which were so vital to the

above proof, depended very heavily on the nature of FF, so there is no simple way of generalizing Theorem 5.9 to the whole class of algorithms. However, we do have

COROLLARY 5.9.1. For all decreasing lists L,

$$BF(L) \leq (11/9)L^* + 4.$$

Proof. As in the above proof, we can assume $\text{Range}(\text{size}_L) \subseteq (2/11, 1]$. But then $\text{Range}(\text{size}_L) \subseteq [1/6, 1]$, and so by Theorems 3.15 and 5.9,

$$BF(L) \leq FF(L) \leq (11/9)L^* + 4. \quad \square$$

Finally, combining the upper bounds provided by Theorem 5.9 and its corollary via Lemma 3.1, with the lower bounds presented in Theorem 4.1, we get

COROLLARY 5.9.2. If $1/2 < t \leq 1$, then

$$R[BFD, t] = R[FFD, t] = 11/9.$$

CHAPTER 6. OTHER OFF-LINE ALGORITHMS

SECTION 6.1. The GROUPING Rule

Chapters 3, 4, and 5 were all devoted to a single preprocessing rule, the DECREASING rule. In this chapter we consider alternatives. The first we shall describe is an application of the same principles we used in defining GROUP-X FIT in Chapter 2, to get a linear-time approximation to the DECREASING rule just as that algorithm was a linear-time approximation to BEST FIT.

Given a schedule of intervals X , as defined in Section 2.3, our rule is:

GROUPING-X RULE: L is ordered so that if $\text{size}(a) \in X_i$, $\text{size}(b) \in X_j$, and $i < j$, then $\text{rank}_L(a) > \text{rank}_L(b)$.

If X has k groups, a list of length n can be so grouped in about $n \log_2(k)$ comparisons. Moreover, if we are going to use the GROUP-X FIT packing rule, we would probably have to make many of those comparisons anyway, in the course of deciding which bin the pieces should go in. So, letting GROUP-X FIT GROUPED (GXFG) be the algorithm which preprocesses according to

the GROUPING-X rule and then packs according to GROUP-X FIT, we have that GXFG takes time comparable to that of as GXF (which has no preprocessing), and can have better worst case behavior:

THEOREM 6.1. For $m = \lfloor 1/t \rfloor \geq 1$ and $\{1/(m+2), 1/(m+1), 1/m\} \subseteq X$,

$$R[GXFG, t] = 1 + 1/(m+1).$$

Proof. A general lower bound example for all possible X is given in Figure 6.1. No matter what X is, the ϵ in the examples can be made small enough so that the GXFG packing comes out as pictured. For the upper bound when X is as stated, let L be a list with $\text{Range}(\text{size}_L) \subseteq (0, 1/m]$ which is ordered in accordance with the GROUPING-X rule. Let PX be a GXF-packing of L using $\text{GXF}(L)$ bins. We shall prove that

$$(6.1A) \quad L^* \geq [(m+1)/(m+2)] \cdot [\text{GXF}(L) - 1],$$

and the upper bound will follow via Lemma 3.1. Divide the list into segments $L = L_1 \bullet L_2 \bullet L_3$, where

$$\text{Range}(\text{size}_{L_1}) \subseteq [1/(m+1), 1/m],$$

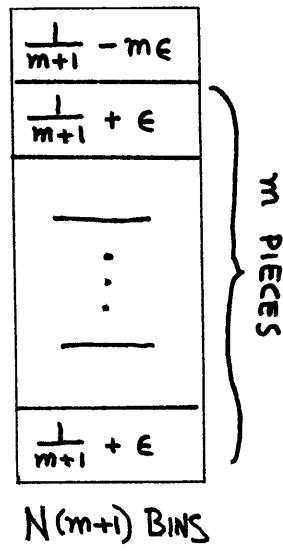
$$\text{Range}(\text{size}_{L_2}) \subseteq [1/(m+2), 1/(m+1)], \text{ and}$$

$$\text{Range}(\text{size}_{L_3}) \subseteq (0, 1/(m+2)).$$

This must be possible because $\{1/(m+2), 1/(m+1), 1/m\} \subseteq X$.

OPTIMAL PACKING

$$L^* = N(m+1)$$



GXFG - PACKING

$$GXFG(L) = N(m+2)$$

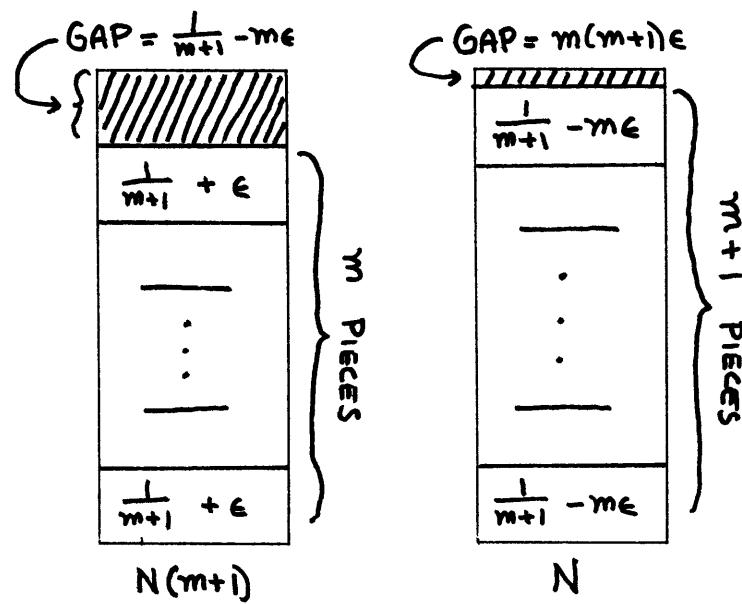


FIGURE 6.1. Lists L with $\text{Range}(\text{size}_L) \subseteq (0, -\frac{1}{m}]$, for which

$$\frac{GXFG(L)}{L^*} = 1 + \frac{1}{m+1}, \left\{ \frac{1}{m+2}, \frac{1}{m+1}, \frac{1}{m} \right\} \subseteq X.$$

Now if $b = \text{piece}_{\mathbf{P}X}(\#PX, 1) \in \text{PIECES}(L3)$, then $\text{size}(b) < 1/(m+2)$, and when b was assigned, $\text{gap}_{\mathbf{P}}(\#PX) = 1$. Thus by Lemma 2.11 and the fact that $1/(m+2) \in X$, we must have $\text{gap}_{\mathbf{P}}(j) < 1/(m+2)$, $1 \leq j \leq \#PX$, so that $L^* \geq W(L) > [(m+1)/(m+2)] \cdot [GXF(L) - 1]$, and (6.1A) holds, as desired. (Note that this is basically the same argument we used in proving Lemma 4.2 for decreasing lists and AF algorithms.)

Thus the only problem is if b is a piece from $L1$ or $L2$, that is if $GXF(L) = GXF(L12)$, where $L12 = L1 \cdot L2$. Thus, since $L^* \geq L12^*$, it will be sufficient to show that

$$(6.1B) \quad L12^* \geq [(m+1)/(m+2)] \cdot [GXF(L12) - 1].$$

So let PX' be a GXF-packing of $L12$, using $GXF(L12)$ bins. Divide it into two segments as shown in Figure 6.2. $PX1$ is the first L^* bins and $PX2$ is all the non-empty bins to the right of $PX1$. We may assume that $PX2$ is not vacuous, else (6.1B) would be immediate.

Now by size constraints we know that $L12^* \geq \lceil |PIECES(L1)|/m \rceil$. Let $j_1 = \text{MAX}\{j : \text{piece}_{\mathbf{P}X'}(j, 1) \in \text{PIECES}(L1)\}$. By Lemma 2.13 all BIN_j 's with $1 \leq j \leq j_1$ must contain m pieces from $L1$, and so, $j_1 \leq \lceil |PIECES(L1)|/m \rceil \leq L12^*$. Thus, since BIN_{j_1} is the rightmost bin containing pieces from $L1$, all the

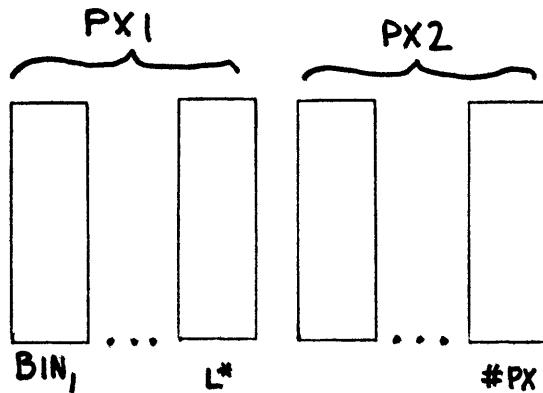


FIGURE 6.2. GXF-Packing PX' of $L_{12} = L_1 * L_2$.

pieces in PX_2 must be from L_2 and hence have size $< 1/(m+1)$.

Let us return our attention to PX_1 . By Lemma 2.13, since $\#PX_1 < \#PX'$, PX_1 must contain at least $(m)(\#PX_1) = (m)(L_{12}^*)$ pieces. But by size constraints $|PIECES(L_{12})| \leq (m+1)(L_{12}^*)$. Thus the number of pieces in PX_2 can be at most L_{12}^* . Since PX_2 can be considered a packing of a sublist of L_{12} , all of whose pieces are $< 1/(m+1)$ in size, Lemma 2.12 tells us that all its bins except the last must contain at least $m+1$ pieces, and so $\#PX_2 \leq \lceil L_{12}^*/(m+1) \rceil$. Thus

$$\begin{aligned} \text{GXF}(L_{12}) &= \#P_{X1} + \#P_{X2} \leq L^* + \lceil L_{12}^*/(m+1) \rceil \\ &\leq [(m+2)/(m+1)] L_{12}^* + 1, \end{aligned}$$

and (6.1B) follows. \square

Note that this is one theorem which does not make a special case out of $t > 1/2$. For such t it yields $R[GXF, t] = 1.5$. If we use an AF packing rule S instead of GF along with the Grouping-X rule (algorithm SXG), we will get the same results for $t \leq 1/2$, but for $t > 1/2$ and $\{1/4, 1/3, 1/2\} \subseteq X$ we have a stronger result:

THEOREM 6.2. Suppose $S \in \text{AF}$, $1/3 < t \leq 1$, and $\{1/2, 1/3, 1/4\} \subseteq X$. Then

$$R[SXG, t] = 4/3.$$

Proof. The lower bound examples are given in Figure 6.3. To prove the upper bound, let L be a list ordered in accordance with the GROUPING-X rule. We shall show that $S(L) \leq (4/3)L^* + 2$, and the bound will follow via Lemma 3.1.

Since $1/4 = (4/3 - 1)/(4/3) \in X$, we may assume that $\text{Range}(\text{size}_L) \subseteq (1/4, 1]$, using the same type of reasoning as was used to prove Lemma 4.2. So let us divide L into segments $L = LA \bullet LB \bullet LC$, where

$$L = \left\langle \frac{1}{3} - \epsilon, \frac{1}{3} - (N+1)\epsilon, \frac{1}{3} - 2\epsilon, \frac{1}{3} - N\epsilon, \dots, \frac{1}{3} - Ne, \frac{1}{3} - 2\epsilon, \frac{1}{3} - \epsilon, \dots, \frac{1}{3} - \frac{N+1}{2}\epsilon, \frac{1}{3} - \frac{N+1}{2}\epsilon, \frac{1}{3} - (N+1)\epsilon, \dots, \frac{1}{3} - (N+1)\epsilon \right\rangle$$

OPTIMAL PACKING

$$L^* = N + 1$$

$\frac{1}{3} - (N+1)\epsilon$
$\frac{1}{3} + (N+1-i)\epsilon$
$\frac{1}{3} + i\epsilon$

$i = 1, N$

$\frac{1}{3} - \frac{N+1}{2}\epsilon$
$\frac{1}{3} - \frac{N+1}{2}\epsilon$
$\frac{1}{3} + (N+1)\epsilon$

1 BIN

SXG - PACKING

$$SXG(L) = \frac{4}{3}N + 1$$

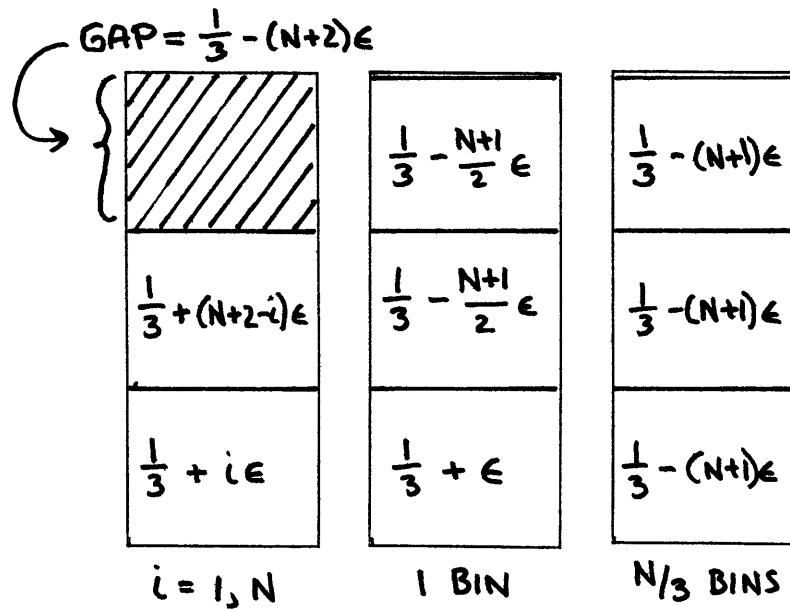


FIGURE 6.3. Lists L for which $\lim_{L^*} \frac{SXG(L)}{L^*} = \frac{4}{3}$, for all $S \in AF$, $\times 2 \begin{Bmatrix} 1 & 1 & 1 \\ 2 & 3 & 4 \end{Bmatrix}$.

$\text{Range}(\text{size}_{L_A}) \subseteq (1/2, 1]$,
 $\text{Range}(\text{size}_{L_B}) \subseteq (1/3, 1/2]$,
 $\text{Range}(\text{size}_{L_C}) \subseteq (1/4, 1/3]$,

and X-pieces, X-bins, $X \in \{A, B, C\}$, are defined in the usual way. The list can be so segmented since $\{1/2, 1/3\} \subseteq X$. Let PS be an S-packing of L, and P* an optimal packing ordered so that if BIN is the a-bin in PS for $a \in A\text{-PIECES}$, then it is also the a-bin in P*. See Figure 6.4. We divide PS and P* into segments as shown, with

$\text{PS}_A = A\text{-bins of } PS$,
 $\text{PS}_B = \{\text{BIN}_j : |\text{A-PIECES}| < j \leq L^*\}$,
 $\text{PS}_X = \{\text{BIN}_j : L^* < j \text{ and } \text{level}_{PS}(j) > 0\}$,
 $\text{P}_A^* = A\text{-bins of } P^*$,
 $\text{P}_B^* = \text{non-empty non-}A\text{-bins of } P^*$.

By construction we thus have $\#\text{PS}_A = \#\text{P}_A^*$ and $\#\text{PS}_B = \#\text{P}_B^*$, so that $\#PS = L^* + \#\text{PS}_X$. Let $x(B)$ be the number of B-pieces in $\text{cont}(\text{PS}_X)$ and $x(C)$ the number of C-pieces in C-bins of PSX (we do not count any C-pieces that might be in B-bins of PSX).

Now since $\{1/3, 1/2\} \subseteq X$, the generation of the S-packing will proceed much as it did for a decreasing list, with each

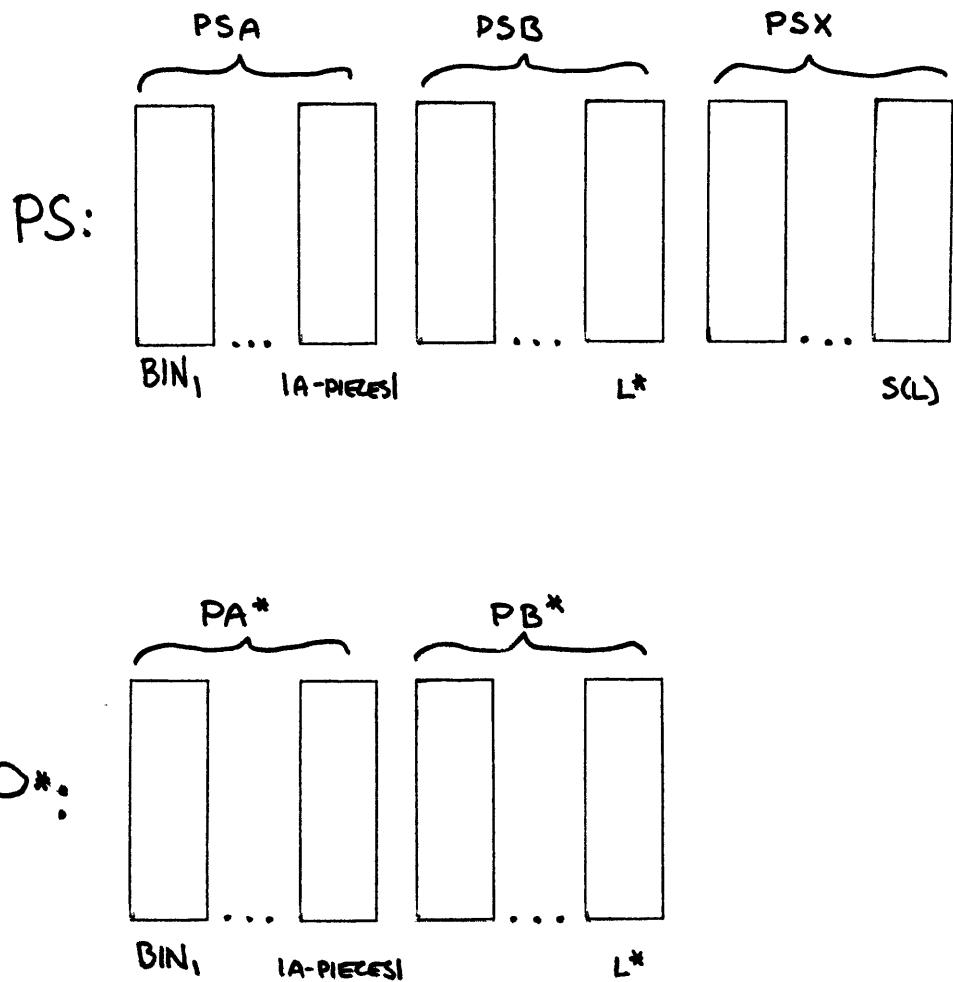


FIGURE 6.4. PS and P*.

B-bin receiving two pieces before the next bin is started, and each C-bin (except possibly the last) receiving 3 C-pieces.

Thus clearly

$$\#PSX \leq \lceil x(B)/2 \rceil + \lceil x(C)/3 \rceil.$$

We shall show that

$$(6.2A) \quad \lceil x(B)/2 \rceil + \lceil x(C)/3 \rceil \leq L*/3 + 2,$$

and the Theorem will follow.

Now by our above reasoning about the generation of the S-packing, we can conclude that

$$x(B) + x(C) \leq |\text{PIECES}(L)| - \#PA* - 2(\#PB*).$$

On the other hand, no A-bin can contain more than one additional piece larger than $1/4$, and no bin can contain more than 3 such pieces, so

$$|\text{PIECES}(L)| \leq 2(\#PA*) + 3(\#PB*).$$

We thus conclude that

$$x(B) + x(C) \leq \#PA* + \#PB* = L*,$$

and so if $x(B) = 0$, (6.2A) is immediate. We thus assume $x(B) > 0$.

Since from this point on the proof for arbitrary $S \in AF$ is rather involved, we shall first go through it for $A \in AAF$, and then extend to the whole of AF . So let

$n(A)$ = total number of A-pieces in L ,

$n(B)$ = total number of B-pieces in L ,

$n(C)$ = total number of C-pieces in L ,

$PS(B)$ = number of B-pieces in A-bins in PS ,

$P*(B)$ = number of B-pieces in A-bins in $P*$.

Now the bins of PS all must contain two B's each in PS , since there is at least one B-piece in PS . Hence $BIN_{n(A)+1}$ thru BIN_L are packed in PS with as many B-pieces as possible. It follows that there must be at least $x(B)$ B-pieces which are in A-bins in $P*$ but not in A-bins in PS . Now when such a piece b was assigned by S , it went either into a new bin, in which case the AF Constraint applied, or into a 1B-bin, which, since

all the other bins were either A-bins or 2B-bins, must have been the bin with unique minimum non-zero level, in which case the AAF Constraint applied. Thus, since b originally fit in an A-bin, the only way it could have failed to go in one now without violating the AF (AAF) Constraint, is if some B-piece were already in that bin. From this we can conclude that $PS(B) \geq x(B)$, and that

$$n(B) = PS(B) + 2[L^* - n(A)] + x(B) \geq 2[L^* + x(B) - n(A)].$$

If $x(C) = 0$, then since there must be at least $x(B)$ A-bins which contained B-pieces in P^* but not in PS , we have $L^* \geq x(B) + PS(B) \geq 2x(B)$, and so

$$\lceil x(B)/2 \rceil + \lceil x(C)/3 \rceil \leq x(B)/2 + 1 \leq L^*/4 + 1,$$

an even better result than that required for the Theorem. On the other hand, if $x(C) > 0$ we still have $x(C) \leq$

$$\begin{aligned} n(C) &= |\text{PIECES}(L)| - n(B) - n(A) \\ &\leq 2n(A) + 3[L^* - n(A)] - n(B) - n(A) \\ &\leq 3L^* - 2n(A) - 2[L^* + x(B) - n(A)] \\ &\leq L^* - 2x(B), \end{aligned}$$

and so

$$\begin{aligned} \left\lceil \frac{x(B)}{2} \right\rceil + \left\lceil \frac{x(C)}{3} \right\rceil &\leq \frac{x(B)}{2} + \frac{L^* - 2x(B)}{3} + 2 \leq \frac{L^*}{3} - \frac{x(B)}{6} + 2 \\ &\leq \frac{L^*}{3} + 2. \end{aligned}$$

And so (6.2A) holds if $S \in \text{AAF}$.

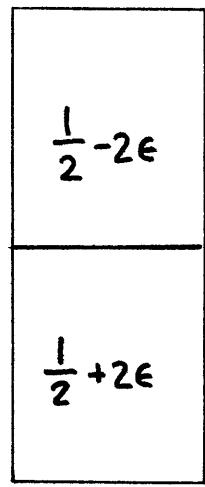
The above observation about the case when $x(C) = 0$ is no longer valid if we allow S to be an arbitrary AF algorithm. For such an S , we can no longer be assured that $PS(B) \geq x(B)$. If a B-piece b went in an A-bin in P^* , it can still go in a non-A-bin in PS even though that original A-bin does not already contain a B-piece. All we know is that by the AF Constraint b cannot start a new bin unless the original A-bin already contains a B-piece. Hence if, for instance, P^* is made up entirely of A-bins, $PS(B)$ need only be $\geq x(B)/2$. Thus we can have $x(B)$ as large as $2L^*/3$, and the number of excess bins can be as large as $L^*/3$. See Figure 6.5 for an example which realizes this bound for $S = \text{WORST FIT}$, and Corollary 6.2.1 for a consequence.

Taking these possibilities into account we now extend our upper bound proof to arbitrary $S \in \text{AF}$. First we define some additional quantities:

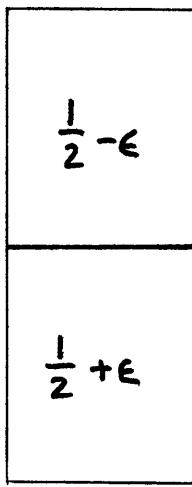
$$L = \left\langle \underbrace{\frac{1}{2} + 2\epsilon, \dots, \frac{1}{2} + 2\epsilon}_{2N \text{ Pieces}}, \underbrace{\frac{1}{2} - \epsilon, \dots, \frac{1}{2} - \epsilon}_{N \text{ Pieces}}, \underbrace{\frac{1}{2} + \epsilon, \dots, \frac{1}{2} + \epsilon}_{N \text{ Pieces}}, \underbrace{\frac{1}{2} - 2\epsilon, \dots, \frac{1}{2} - 2\epsilon}_{2N \text{ Pieces}}, \frac{1}{2} - \epsilon, \frac{1}{2} - 2\epsilon, \dots, \frac{1}{2} - \epsilon, \frac{1}{2} - 2\epsilon \right\rangle$$

OPTIMAL PACKING

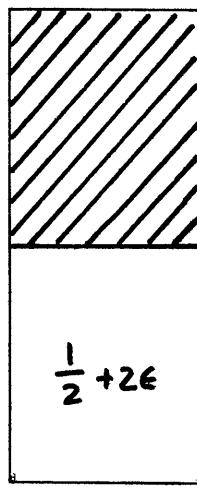
$$L^* = 3N$$



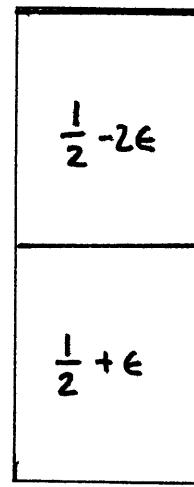
2N BINS



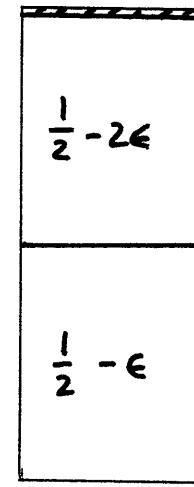
N



2N



N



N

WFXG - PACKING

$$WFXG(L) = 4N$$

$b_1(B)$ = number of non-A-bins in P^* with two C-pieces,

$b_0(B)$ = number of non-A-bins in P^* with three C-pieces.

Recall that we may assume $x(B) > 0$, and so every bin in PSB contains two B-pieces. Thus, since no bin in P^* can contain more than 3 pieces, we have

$$PS(B) = P^*(B) - x(B) - b_1(B) - 2(b_0(B)), \text{ and}$$

$$P^*(B) \geq x(B).$$

We consider two cases, depending on the relation between $x(B)$ and $PS(B)$:

First, suppose $x(B) \leq 2[PS(B)]$. We then have

$$x(B) \leq 2[PS(B)] = 2P^*(B) - 2x(B) - 2(b_1(B)) - 4(b_0(B)).$$

Thus we can conclude that

$$x(B) \leq (2/3)[P^*(B) - b_1(B) - 2(b_0(B))].$$

If $x(C) = 0$, we then have

$$\lceil x(B)/2 \rceil + \lceil x(C)/3 \rceil \leq \lceil P^*(B)/3 \rceil \leq L^*/3 + 1,$$

and so (6.2A) holds. If $x(C) > 0$, we still have that

$$x(C) \leq n(C) \leq L^* - P^*(B) + b_1(B) + 2(b_0(B)),$$

and so $\lceil x(B)/2 \rceil + \lceil x(C)/3 \rceil \leq$

$$\leq \frac{P^*(B) - b_1(B) - 2b_0(B)}{3} + \frac{L^* - P^*(B) + b_1(B) + 2b_0(B)}{3} \rightarrow +2$$

$\leq L^*/3 + 2$, and again (6.2A) holds.

Suppose on the other hand that $x(B) > 2[PS(B)]$. Then by the AF Constraint, less than half of the $x(B)$ B-pieces which were in A-bins in P^* but not in PS can be bottom pieces in B-bins in PS . Thus at least $d(B) = x(B) - 2[PS(B)]$ B-pieces must be in non-A-bins in P^* . Substituting for $PS(B)$ we get

$$x(B) = 2[P^*(B) - x(B) - b_1(B) - 2b_0(B)] + d(B)$$

SECTION 6.1 - Page 360

and we can conclude that

$$x(B) = (2/3)[P^*(B) - b_1(B) - 2b_0(B)] + d(B)/3.$$

Since there must be at least $d(B)/2$ non-A-bins in P^* , we thus have $x(B) \leq (2/3)L^*$, and (6.2A) follows if $x(C) = 0$. If $x(C) > 0$, we still have

$$x(C) \leq n(C) - [P^*(B) - P_S(B)],$$

since no C-piece can start the first excess C-bin until all A-bins which had room for B-pieces in P^* have received either a B- or a C-piece in P_S . Substituting for $n(C)$ and $P_S(B)$ we get

$$\begin{aligned} x(C) &\leq L^* - P^*(B) + b_1(B) + 2b_0(B) - P^*(B) + [x(B) - d(B)]/2. \\ &\leq L^* - P^*(B) + b_1(B) + 2b_0(B) - d(B)/2, \end{aligned}$$

since $P^*(B) \geq x(B) > x(B)/2$. We thus have

$$\lceil x(B)/2 \rceil + \lceil x(C)/3 \rceil \leq$$

$$\frac{P^*(B) - b_1(B) - 2b_0(B) + [d(B)/2]}{3} + \frac{L^* - P^*(B) + b_1(B) + 2b_0(B) - [d(B)/2]}{3}$$

$$\leq L*/3 + 2,$$

and so (6.2A) holds in all cases if $S \in \text{AF}$ and Theorem 6.2 is proven. \square

COROLLARY 6.2.1: There exists an interval I , namely $I = (1/3, 1]$, such that for all $x \notin \{1/4, 1/3, 1/2\}$

$$R[WFXG, I] > R[FFXG, I].$$

This follows from our remarks in the middle of the above proof about the case when $x(C) = 0$ and $S \in \text{AAF}$, and Figure 6.5 for WF. The actual values are $R[WFXG, (1/3, 1)] = 4/3$ and $R[FFXG, (1/3, 1)] = 5/4$. The significance of this Corollary derives from the fact that we were not able to find any intervals which would similarly distinguish between WFD and FFD (indeed, if $\text{Range}(\text{size}_L) \subseteq (1/4, 1]$, by Theorem 3.9 we have $|FFD(L) - WFD(L)| \leq 1$), and thus the DECREASING rule seems to be a better leveller of ANY FIT algorithms than the GROUPING rule, even though we chose the latter as an approximation to it.

SECTION 6.2. The INCREASING Rule

The final preprocessing rule we shall consider might be called the "ultimate" leveller. It in fact guarantees that all AF algorithms will generate identical packings when applied to a list preprocessed in accord with it. Unfortunately this packing will usually be among the worst possible for any ordering, and we would not recommend it for practical use. It is of theoretical interest because it is a natural sort of idea, simply the opposite of the DECREASING rule:

INCREASING RULE: L is in increasing order, that is
 $\text{size}(a) > \text{size}(b) \implies \text{rank}_L(a) > \text{rank}_L(b)$.

A 2-part algorithm consisting of a packing rule S and the INCREASING rule will be called S INCREASING, or simply SI . If S \in AF, it should be clear that the SI -packing will be the same as the NFI-packing, since if a piece starts a new bin it cannot have fit in the previous one, and so neither can any of its successors, all of which are at least as large. That this packing will generally be bad can be seen from the fact that all bins containing a piece larger than $1/2$ will contain only that piece, and hence there will be much unused bin-capacity.

The following Theorem gives a lower bound on the worst case

behavior of such algorithms which is only slightly better than the upper bound for AAF algorithms with no preprocessing at all. And although we suspect that this is also the upper bound and so using the INCREASING rule does yield a tiny improvement in worst case behavior, in the average case we would have to expect the ANY FIT INCREASING algorithms to be considerably and consistently worse than the ANY FIT algorithms with no preprocessing at all.

THEOREM 6.3. If $S \in AF \cup \{NkF: k \geq 1\}$, then

$$R[S1] \geq 1 + \sum_{k=2}^{\infty} \frac{1}{c_k} = 1 + \frac{1}{2} + \frac{1}{6} + \frac{1}{42} + \dots = 1.69103\dots$$

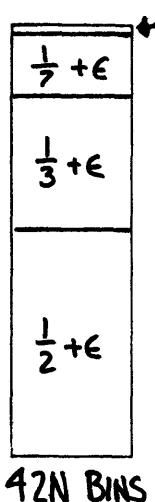
where $c_k = 1/[(c_{k-1})(c_{k-1}+1)]$, and $c_1 = 1$.

PROOF: We can construct an arbitrarily long list L for each partial sum so that $S1(L)/L*$ realizes the given sum. This is a fairly straightforward process once the idea is grasped. We merely present, in Figure 6.6, an example realizing the sum of the first four terms.

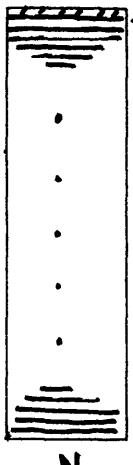
The above lower bound was originally thought to be the upper bound on the worst case behavior of FIRST FIT, until Ullman discovered the example we presented in Theorem 2.6 [Gr3].

OPTIMAL PACKING

$$L^* = 42N$$



$42N$ BINS



42 PIECES of SIZE $\frac{1}{42-3\epsilon}$

SI-PACKING

$$SI(L) = 42N \left(1 + \frac{1}{2} + \frac{1}{6} + \frac{1}{42}\right)$$

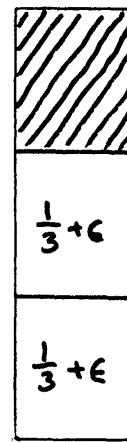
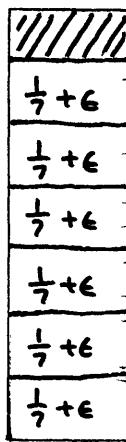


FIGURE 6.6. Lists L for which $SI(L)/L^* = 1 + \frac{1}{2} + \frac{1}{6} + \frac{1}{42}$,
for all $s \in AFU \{NkF\}$.

Similar lower bounds can be generated for the cases when the pieces must all be $\leq 1/n$, but we leave that as an exercise for the reader.

This completes our study of 2-part algorithms. Table 6.1 summarizes the bounds on worst case behavior that we have proved for them. The entries for the algorithms involving GROUPING are made under the assumption that the schedule of intervals X contains at least those numbers specified by the relevant theorem. The ANY FIT and ALMOST ANY FIT entries give the worst values possible for any member of the corresponding class of algorithms. Timings are omitted for these entries, although by Theorem 2.10, we know that all such algorithms require at least $O(n \log n)$. Where the precise value of $R[S, t]$ is not known, the best upper and lower bounds are given.

ALGORITHM S	TIME	$R[S, t], t \in (1/2, 1]$	$R[S, t], m = \lfloor 1/t \rfloor \geq 2$
1. NEXT FIT 2. WORST FIT 3. ANY FIT	$O(n \log n)$ $O(n \log n)$	2.0	$1 + \frac{1}{(1/t)-1}$
4. NEXT-2 FIT 5. GROUP-X FIT	$O(n)$ $O(n)$	$[1.7, 2.0]$	
6. FIRST FIT 7. BEST FIT 8. ALMOST ANY FIT	$O(n \log n)$ $O(n \log n)$	1.7	$1 + \frac{1}{m}$
9. ANY FIT INCREASING		$[1.691\dots, 1.7]$	$[1 + 1/m - ?, 1 + 1/m]$
10. GROUP-X FIT GROUPED	$O(n)$	1.5	
11. ANY FIT X-GROUPED		$4/3 = 1.333\dots$	$1 + \frac{1}{m+1}$
12. FIRST FIT DECREASING 13. BEST FIT DECREASING	$O(n \log n)$ $O(n \log n)$	$11/9 = 1.222\dots$	$\left[1 + \frac{1}{m+2} - \frac{2}{m(m+1)(m+2)}, \right.$
14. ANY FIT DECREASING		$[1.222\dots, 1.25]$	$\left. 1 + \frac{1}{m+2}\right]$

Table 6.1. Summary of asymptotic worst case bounds proven for 2-Part bin-packing algorithms.

SECTION 6.3. More Complicated Algorithms

Of all the algorithms considered so far, FIRST FIT DECREASING (and presumably the rest of the AFD algorithms) has the value of $R[S]$ closest to 1. However, these algorithms still can require as many as $11/9$ times the optimal number of bins. The question arises, are there any polynomial time algorithms S with $R[S] < 11/9$? In particular, are there any fairly simple (though not necessarily 2-part) algorithms of this type which might have practical applications?

One candidate is mentioned in [Gr3]:

BEST BIN FIT: Set $I = 1$, LIST = PIECES(L).

1. If LIST is empty, halt.
2. Pack bin I with that subset of LIST which fits leaving the smallest possible remaining gap in the bin. Delete that subset from LIST.
3. Set $I = I + 1$
4. Go to 1.

This algorithm, unfortunately, has a number of failings.

The first is that it is not necessarily a polynomial time algorithm. The second step involves the solution of the polynomial complete problem SUBSET SUM [Ka1], so if the

algorithm can be implemented in polynomial time, we can find an optimal packing in polynomial time as well, and this algorithm would probably be unnecessary, especially since it is definitely not optimal itself.

Graham [Gr3] asserts the following result, and it is not difficult to reconstruct the proof:

$$\text{THEOREM 6.4. } R[\text{BBF}] \geq \sum_{n=1}^{\infty} 1/(2^n - 1) = 1.606695\dots$$

PROOF: For each partial sum we can construct a sequence of lists $\{L_i\}$ such that $L_i* \rightarrow \infty$ and $\lim BBF(L_i)/L_i*$ lies between that partial sum and the overall sum. Figure 6.7 gives a series for the sum of the first two terms and should indicate how to proceed for the rest of the partial sums. Graham conjectures that this lower bound is also the upper bound and we tend to agree.

Although the above algorithm is more or less a failure, it does suggest some ideas that might enable us to improve on FFD. The basic problem with it (other than timing) is that it allows the packing of the larger pieces to be postponed until there are no longer any small pieces to fill in the gaps, the same problem that our use of the DECREASING rule in FFD was designed to eliminate. By trying to get the best of both worlds, we arrive

Let $d(1) < 1/64$, $d(i) = d(i-1)/4$ for $2 \leq i \leq 4N+1$, $N \equiv 0 \pmod{3}$.

OPTIMAL PACKING

$$L^* = 4N + 1$$

$\frac{1}{8} - 4d(i)$
$\frac{1}{8} + d(i)$
$\frac{1}{4} + d(i)$
$\frac{1}{2} + d(i)$

$$i=1, 4N+1$$

BBF-PACKING

$$\text{BBF}(L) = 19N/3$$

$\frac{1}{8} - 4d(4k+5)$
$\frac{1}{8} + d(4k+4)$
$\frac{1}{8} - 4d(4k+3)$
$\frac{1}{8} + d(4k+2)$
$\frac{1}{8} - 4d(4k+1)$

$$k=0, N-1$$

$\frac{1}{8} - 4d(1)$
$\frac{1}{2} + d(1)$
$\frac{1}{4} + d(3k+4)$
$\frac{1}{4} + d(3k+3)$
$\frac{1}{4} + d(3k+2)$

$$1 \text{ BIN}$$

$\frac{1}{8} - 4d(1)$
$\frac{1}{2} + d(1)$
$\frac{1}{4} + d(3k+4)$
$\frac{1}{4} + d(3k+3)$
$\frac{1}{4} + d(3k+2)$

$$K=0, \frac{4N}{3}-1$$

$\frac{1}{8} - 4d(1)$
$\frac{1}{2} + d(1)$
$\frac{1}{4} + d(3k+4)$
$\frac{1}{4} + d(3k+3)$
$\frac{1}{4} + d(3k+2)$

$$i=2, 4N+1$$

$$\text{FIGURE 6.7. } \text{Lists } L \text{ for which } 1 + \frac{1}{3} < \lim_{L^*} \frac{\text{BBF}(L)}{L^*} = 1 + \frac{1}{3} + \frac{1}{4} < \sum_{n=1}^{\infty} \frac{1}{2^n - 1}$$

at the following series of algorithms (indexed by k):

MOST- k FIT: Set $I = 1$, LIST = PIECES(L).

1. Remove the largest piece from LIST and put it in Bin I .
2. If LIST is empty, halt.
3. If the smallest piece in LIST will not fit in Bin I ,
set $I = I + 1$ and go to 1.
4. Add to Bin I that subset of LIST with k or fewer
pieces which fits with the least gap remaining,
and delete that set from LIST.
5. Go to 2.

The reader can observe that MOST-1 FIT will yield the FIRST FIT DECREASING packing of L , and so in a sense all of these algorithms are simply generalizations of FFD. For $k \geq 2$, they can be implemented in $O(n^k)$ if we are willing to allocate $O(n^k)$ storage to store an ordered list of all the sums of subsets of L with k or fewer elements. If space is constrained, they can be implemented in time $O(n^{k+1})$, with the various subsets being tested each time we make an assignment.

We have as yet been unsuccessful in proving upper bounds on the worst case behavior of these algorithms, except to observe that they all obey the $17/10$ upper bound since for any list L there is a permutation L' of L such that the FF-packing of L' is

the same as the MkF -packing of L . (This is the same sort of idea as was used to prove Lemma 2.0.)

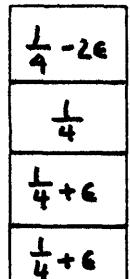
We suspect that the algorithms are much better than this. For instance, we have found no list L for which $M2F(L)/L^*$ exceeds $7/6$. The lists given in Figure 4.1 which yielded an $11/9$ ratio for FFD are packed optimally by M2F. The list in Figure 4.3 that yielded a $71/60$ ratio for FFD yields only a $68/60$ ratio for M2F. The lists which do attain the $7/6$ ratio are those given in Figure 4.2 with $n = 3$. For clarity, we reproduce this example in Figure 6.8, with n fixed at 3.

Similarly, the worst example we have been able to find for MOST-3 FIT is given by Figure 4.2 with $n = 4$ and yields a ratio of $23/20$. In fact, Figure 4.2 yields the best lower bound examples known for all $k \leq 5$, the values for $k = 4$ and $k = 5$ being $17/15$ and $5/42$ respectively. Figure 6.9 gives a set of lists for which $MkF(L)/L^* = 10/9$, for all $k \geq 2$, and this is the best lower bound known for $k \geq 6$.

Nevertheless, 11% excess bins is considerably better than 22%, and so a distinct improvement on FFD may well be possible, although apparently at a considerable increase in cost. And noting that the $10/9$ example in Figure 6.9 would in fact be optimally packed by FFD, we can suggest the following avenue for possible further improvement:

OPTIMAL PACKING

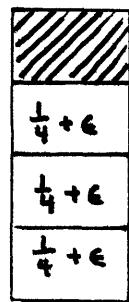
$$L^* = 12N$$



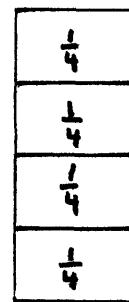
12N BINS

M2F-PACKING

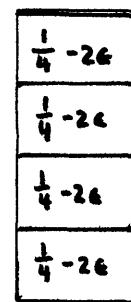
$$M2F(L) = 14N$$



8N



3N

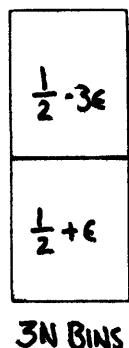


3N

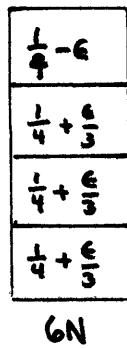
FIGURE 6.8. Lists L for which $M2F(L)/L^* = 7/6$.

OPTIMAL PACKING

$$L^* = 9N$$



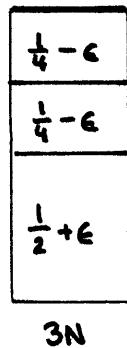
3N BINS



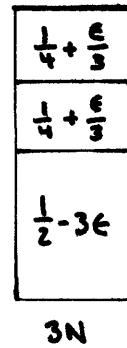
6N

MkF-PACKING

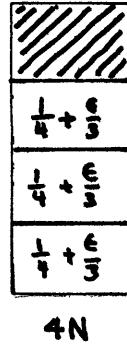
$$MkF(L) = 10N$$



3N



3N



4N

FIGURE 6.9. Lists L for which $\frac{MkF(L)}{L^*} = \frac{10}{9}$ for all $k \geq 2$.

Pack L according to MkF for $1 \leq k \leq j$, and
take as your final packing the best of the j trials.

We leave the investigation of this and the other algorithms
we have suggested for further research.

CHAPTER 7. EMPIRICAL TESTS OF AVERAGE CASE BEHAVIOR

SECTION 7.1. The experimental environment

In this Chapter, we hope to give a general idea of how the average behavior of the various bin-packing algorithms we have introduced might compare with the worst case behavior we have so extensively analyzed in Chapters 1 thru 6. Do our worst case bounds actually have practical significance? Do the rankings of the algorithms according to $R[S, t]$ carry over into the average case?

In an attempt to answer these questions, we have chosen to empirically test the algorithms by running them on computer generated sample lists and averaging the results over a number of runs.

In this section we present a general description of our research procedures and the implementation of the various algorithms. Section 7.2 will then present the experimental results in summary and graphical form and compare them with our earlier worst case results.

Our experiments were performed on the M.I.T. Artificial Intelligence Laboratory's PDP-10 Time-Sharing System. The programs for the implementations of the various algorithms were

written in LISP and run as compiled LAP code. We were primarily interested in the behavior of the algorithms with respect to the number of bins they used and so did not consider the issue of comparative computer running times important enough for us to devote a lot of effort toward making our implementations as fast as possible. Thus, for instance, although most of our implementations of AF algorithms required only $O(n \log n)$ comparisons, each had an overhead component which, though its coefficient was small, was $O(n^2)$. GROUP-X(k) FIT was implemented in $O(nk)$ rather than $O(n \log k)$. Thus none of the results presented in this chapter will concern timing questions, and we will restrict this introductory discussion to those factors in our experimental design which have a bearing on the packings our algorithms generated.

First let us describe the way we produced the lists which the algorithms were required to pack. The basic machinery behind all our list generators was the random number generator supplied by MACLISP and named RANDOM. This provides a computer word full of "random" octal digits which we converted into a number with 6 places decimal accuracy in $(0,1]$. Figure 7.1 portrays the decile distribution of the numbers generated, based on 10,000 consecutive calls. We call this distribution "Uniform," although it is probably only an approximation to the real uniform distribution. To generate numbers according to an

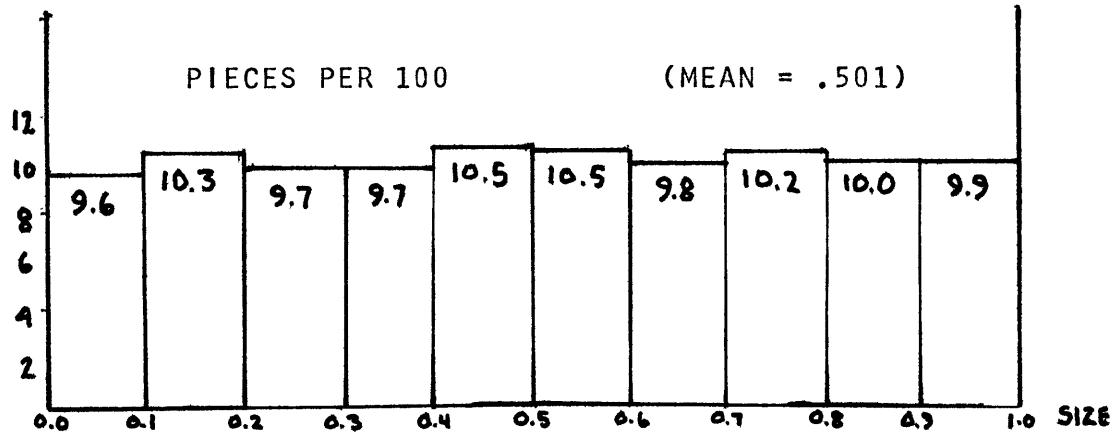


FIGURE 7.1. Decile Breakdown of "Uniform" Distribution

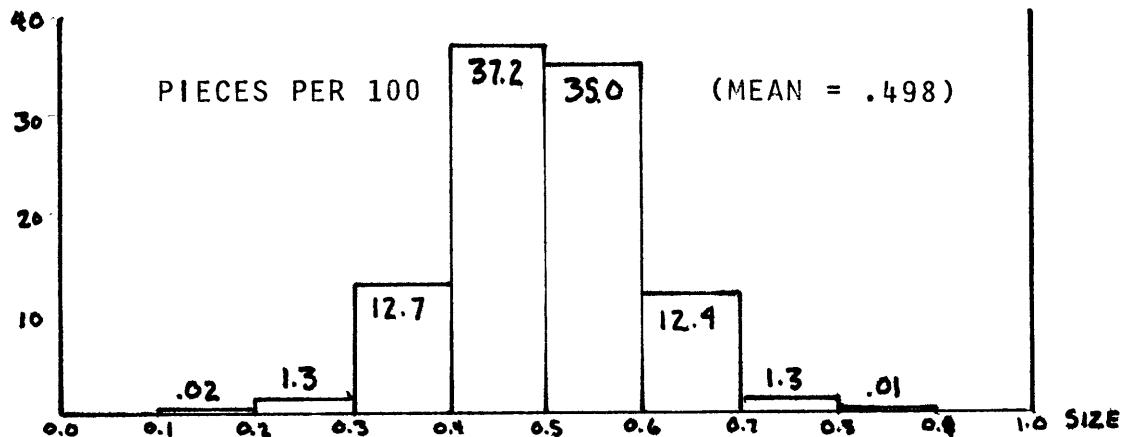


FIGURE 7.2. Decile Breakdown of "Normal" Distribution.

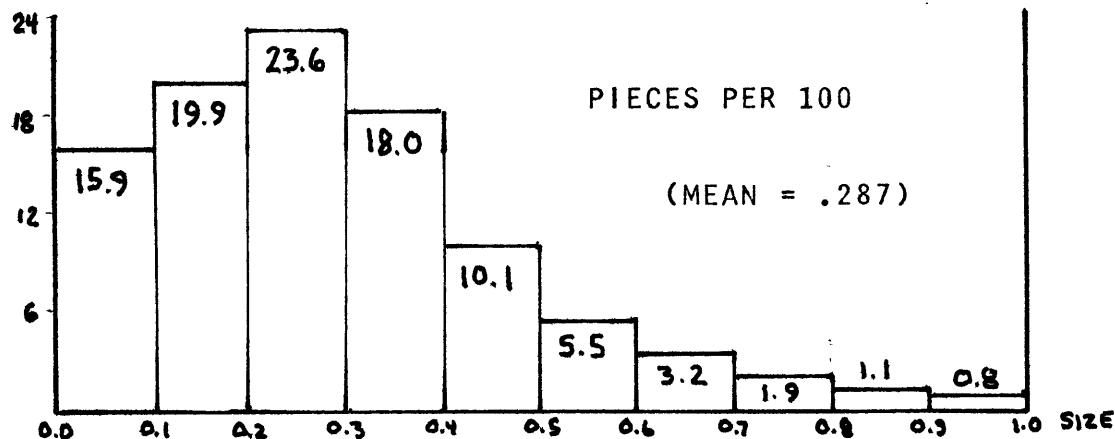


FIGURE 7.3. Decile Breakdown of "Chopped (2,5)" Distribution.

approximation to a "Normal" distribution, we averaged together 10 of our "Uniform" numbers. Figure 7.2 gives a decile view of the distribution actually obtained.

In both these distributions, the numbers are all generated independently, insofar as the basic RANDOM gives independent values. There is a problem with this, however. For, assuming as we do that the polynomial complete problems cannot be done in polynomial time, there is no reasonable way to compute the actual value of L^* , given a list of say 100 or so pieces. Yet, in order to get average case measures which are comparable to our worst case bounds, we want to find values for $S(L)/L^*$.

For the purposes of this investigation, we have chosen to take for L^* the best lower bound we can find. This will be $|w(L)|$ or the number of A-pieces (numbers $> 1/2$), whichever is greater. In general this will probably be a slight under-estimate of the actual value, so our average ratios will be a little higher (and in some special cases, perhaps more than a little higher) than the real ratios actually are. This effect is fortunately bounded by the fact that the best packing produced by any of the algorithms yields an upper bound on L^* , and this often is reasonably close to our lower bound estimate.

In an effort to get around this difficulty, we designed an additional distribution of pieces, which we call the "Chopped" distribution. It is obtained by assuming that the optimal

packing has all its bins completely filled, and breaking each up independently into a random number of pieces, which are then assigned random values, suitably adjusted so that they add up to 1. To get a list of pieces fitting this distribution, we must specify the number of bins we wish the optimal packing to have, and the range in the number of pieces allowed per bin, for instance, 50 bins with from 2 to 5 pieces per bin. This allows us to know L^* in advance, and by shuffling (using "Algorithm P" [Kn2, p.125]) the list, we can for all practical purposes render its original structure irrecoverable. This in a sense stacks the deck against the possibility of our algorithms finding an optimal packing, since there are now probably far fewer different optimal packings than there would be for a completely random list, but presumably it will not prevent our algorithms from getting close to optimal, which is what we are interested in anyway. In addition, the distribution, pictured in Figure 7.3, does not look altogether that unreasonable, although there is no reason to believe that it, any more than our other two, would actually be encountered in practical situations.

SECTION 7.2. Average Case Results

Our principal investigation dealt with the uniform distribution. The results summarized in Table 7.1 will therefore stand as a basic reference point for our other, less extensive, empirical studies. The entries in the table give the average value of

$$(\#PS/L^* - 1.0) \times 100\%,$$

where PS is the packing resulting when algorithm S was applied to list L, for S = NF, N2F, GX(8)F, AWF, FF, BF, GX(8)FG, AWFD, FFD, and BFD, and for L with piece sizes uniformly distributed in the intervals (0,1), (0,1/2), and (0,1/4). For each interval, we used our "uniform" list generator to generate 25 different lists of length 200, and ran all 10 algorithms on each of the lists, averaging the results to get the table entries, and using the lower bound estimate for L^* described in the previous section. Worst case bounds proved in the first two chapters are included for comparison purposes (if none was proved in a particular case, then the best lower bound on worst case behavior known is listed).

	UNIFORM (0,1.0)	UNIFORM (0,.50)	UNIFORM (0,.25)
NF	31.1 [100.]	18.8 [100.]	7.4 [50.0]
N2F	21.9 [70.0]	8.5 [50.0]	2.2 [25.0]
AWF	10.4 [70.0]	4.8 [50.0]	1.4 [25.0]
FF	7.0 [70.0]	2.2 [50.0]	0.6 [25.0]
GX(8)F	5.8 [70.0]	2.2 [50.0]	0.8 [25.0]
BF	5.6 [70.0]	2.0 [50.0]	0.5 [25.0]
GX(8)F	2.1 [50.0]	0.4 [33.3]	0.3 [20.0]
AWFD	2.0 [22.2]	0.2 [18.3]	0.2 [15.0]
FFD	1.9 [22.2]	0.1 [18.3]	0.2 [15.0]
BFD	1.9 [22.2]	0.1 [18.3]	0.2 [15.0]

Table 7.1. Percentage of excess bins required on the average in bin-packings of 25 lists with piece sizes "uniformly" distributed within the stated ranges.
[Worst case bounds are inserted in brackets.]

COMMENTARY ON TABLE 7.1

The difference between the entries for the DECREASING algorithms in the second and third columns is not significant. In both cases the packings generated were optimal for all but one of the 25 lists, but in the former the lists had optimal lengths only half that for the latter. In addition, the one list for which the algorithms did require an "excess" bin in column three had piece size total $W(L) = 25.98$, so our estimate of L^* was 26, although in truth L^* may well have been 27, the number of bins actually used by our algorithms.

The principal conclusion to be drawn from these results is clearly that these bin-packing algorithms apparently are much better than their worst case behavior bounds would lead us to expect. BEST FIT, which we could only guarantee would do no worse than 70% excess bins, actually yields less than 6%. FIRST FIT DECREASING, for which we went to so much trouble to prove a worst case bound of 22.2%, actually uses no more than 2% excess on the average. And if no piece exceeds $1/2$, although in the worst case its behavior could be as bad as 18.3%, its average behavior is to all intents and purposes optimal.

Results for "Normal" and "Chopped" Distributions (Table 7.2)

The fact that average case behavior is better than worst case behavior is perhaps not unexpected, due to the rather complicated nature of many of the worst case examples. However, the magnitude of the difference is somewhat surprising. The question arises, is this all dependent on the fact that we used a uniform distribution? Table 7.2 gives the companion results for our "Normal" and "Chopped" distributions, this time based on just 10 lists for each range.

As we can see, the general outline of the average case behavior remains the same in the chopped distribution, and in the normal case for the interval $(0,1)$, although in this case the ratios are all a little higher. This may indicate either that the algorithms are worse for this type of distribution, or that our estimate of L^* is worse, although probably a combination of the two is involved. However, a definite anomaly turns up in the normal case when all piece sizes are in $(0,1/2)$. Here the ANY FIT DECREASING algorithms are significantly worse than they were when bigger pieces were allowed, and in fact BEST FIT is practically just as good as BFD. What has happened?

	CHOPPED (2,5)	NORMAL (0,1.0)	NORMAL (0,.50)	NORMAL (0,.25)
NF	24.8	37.0	14.8	6.7
N2F	13.0	27.2	9.4	4.4
AWF	6.8	13.2	6.8	4.0
FF	4.0	9.8	6.0	4.0
GX(8)F	4.2	8.2	5.6	4.0
BF	3.6	7.8	5.8	4.0
GX(8)F	2.0	3.9	4.4	2.8
AWFD	2.0	3.2	4.4	2.0
FFD	2.0	3.2	4.4	2.0
BFD	2.0	3.2	4.4	2.0

Table 7.2. Percentage of excess bins required on the average for 10 lists with "Chopped" and "Normal" distributions of piece sizes.

Using Worst Case Information (Table 7.3)

The explanation is actually quite simple, and brings out the point that knowing the significant features of the piece distribution can be of help in choosing which algorithm to use. Here the important feature is that all the pieces are fairly close to their mean, which is of the form $1/n$, where here $n = 4$. Thus the typical list will to a certain extent simulate our worst case example for all pieces in the interval $(0, 1/n + \epsilon]$. (See Figure 4.2.) In the optimal packing the bins will tend to have n pieces, half of them less than the mean, and half greater. But a DECREASING algorithm will put $n-1$ of the bigger pieces per bin until they run out, and since there are not very many especially small pieces, perhaps none of the remaining pieces will fit, so these bins will end up with just $n-1$ pieces.

To bolster the above argument, we ran an experiment with the pieces uniformly distributed throughout the interval $(.2, .3)$, with results as shown in Table 7.3, averaged over 5 trials of lists of length 200.

	UNIFORM (.15,.35)	UNIFORM (.20,.30)
NF	14.9	14.6
BF	6.7	6.7
AWF	7.4	7.1
BFD	4.3	10.7

Table 7.3: Percentage of excess bins required on the average for 5 lists with pieces "uniformly" distributed in narrow ranges centered on 1/4.

Note that now BFD is distinctly worse than BF. In fact, all the algorithms seem to be worse than they were when the pieces were in the larger interval $(0,1/2)$. This is apparently because the effectiveness of the algorithms depends in large part on the availability of small pieces to fill in the gaps in earlier bins, although it also may again reflect the fact that the optimal packing is significantly longer than our lower bound estimate.

Table 7.3 also shows the values obtained when the range of the distribution is opened up slightly to $(.15,.35)$. We note that all the algorithms yield about the same ratios as they did in the narrower range, except for BFD, which shows an immense improvement. Thus this phenomenon of the DECREASING rule being

counter-productive is a very local one.

Average Case Behavior as a Function of Maximum Piece Size
(Figures 7.4 and 7.5)

Although lowering the upper bound on piece size can only improve the worst case behavior of a bin packing algorithm, it still need not be true that the algorithms' average case behavior improve monotonically as the upper bound on the piece size decreases. Figures 7.4 and 7.5 show this graphically for the "Uniform" and "Normal" distributions. We plot the average percentage of excess bins when the piece range is $(0, t)$ as t goes from .125 to 1.0. The appearance of at least some smoothness in the curves may well be deceptive, as values were calculated at only eight points (those not in Tables 7.1 and 7.2 are based on only 10 runs on lists of length 200). For both distributions there are local maxima for t 's other than 1.0, but at present we are not sure whether to assign the blame for these to our algorithms, or to the fact that our estimate of L^* is too low for those particular t .

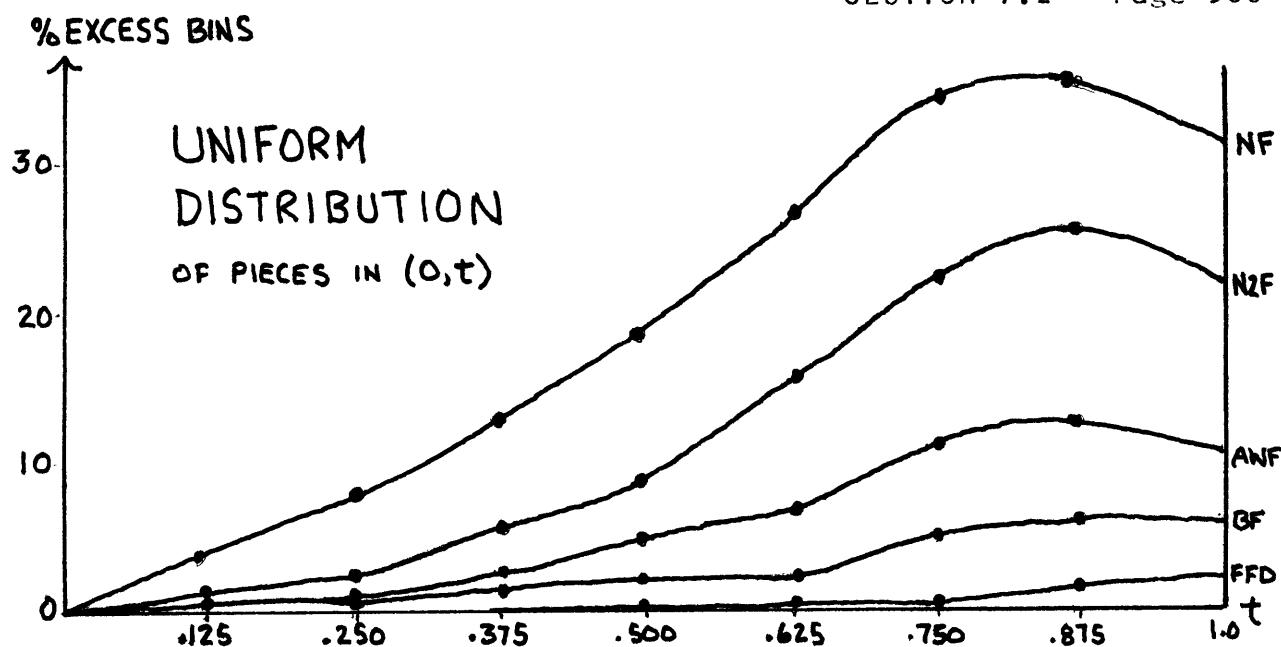


FIGURE 7.4. Average case behavior as a function of upper on the "Uniform" distribution.

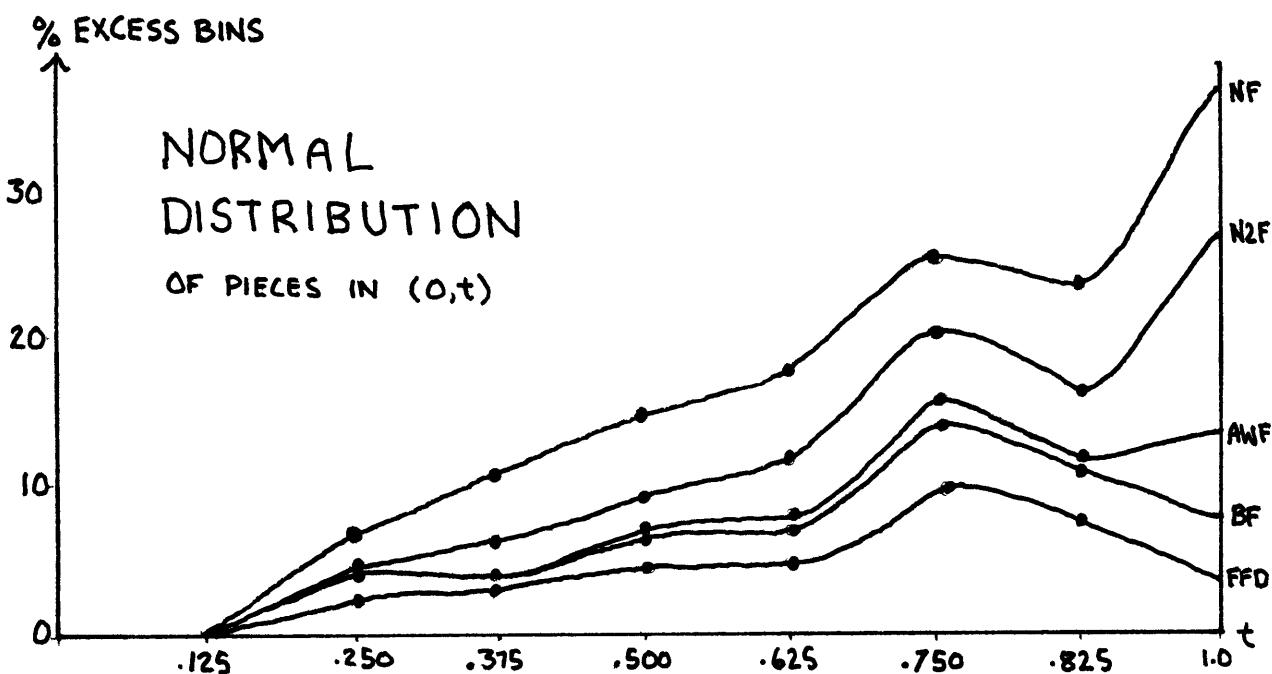


FIGURE 7.5. Average case behavior as a function of upper on the "Normal" distribution.

WORST FIT, ALMOST WORST FIT, and ANY FIT (Table 7.4)

In Section 2.1 we showed that by a slight modification of the algorithm WORST FIT, we could obtain a second algorithm, ALMOST WORST FIT, with significantly improved worst case behavior. We also introduced the incompletely determined algorithm ANY FIT, which was allowed to make any assignment which did not violate the AF Constraint, and showed that it had the same worst case behavior as WORST FIT. Again a simple modification yielded another algorithm, ALMOST ANY FIT, which has the same worst case behavior as AWF.

Now in Tables 7.1 and 7.2 the only one of these algorithms represented is AWF, and so we conducted a brief study to see how these algorithms might compare in practice. To simulate the possible behavior of AF, we implemented it so that it worked like FF, except that after each assignment the sequence of non-empty bins was randomly shuffled. AAF was implemented similarly, with additional machinery to insure that the AAF Constraint was obeyed. The results of our test, which consisted of lists of length 200 with piece sizes "uniformly" distributed in $(0,1)$, are presented in Table 7.4. BF has been added as a reference point, since these particular lists yielded better results than those in Table 7.1.

Uniform
(0.0,1.0)

WF	14.5
AWF	9.9
AF	7.7
AAF	6.7
BF	4.8

Table 7.4: Percentage of excess bins required on the average for 10 lists with piece size "uniformly" distributed.

Note that adding the AAF Constraint has a significant effect on average case behavior as well as worst case behavior. On the other hand, AF does better than AWF, even though its worst case behavior is worse. This is probably because, although AF can occasionally choose the bin which is the worst fit, most of the time it will choose bins which are neither the worst nor second worst fits.

GROUPING Algorithms (Figure 7.6)

In Sections 2.3 and 6.1 we introduced linear time algorithms GXF and GXFG which we said were "approximations" to BF and BFD respectively. We showed that if lists are restricted to those with no pieces larger than 1/2 in size, and the interval schedule is $X_1 = \{0, 1/3, 1/2, 1\}$, then $GX_1 F$ has the same worst case behavior as BF. In addition, if $X_2 = \{0, 1/4, 1/3, 1/4, 1\}$, then $GX_2 FG$ has better worst case behavior than BF, though not quite as good as BFD.

We were skeptical that such results would actually be mirrored by the algorithms' average behavior, and a simple experiment verified this. For 20 lists of length 200 with a "Uniform" distribution of piece sizes in (0,0.5), the average percentages of excess bins were 0.6% for BFD, 2.7% for BF, 17.2% for $GX_2 FG$, and 25.6% for $GX_1 F$. The linear time algorithms, despite their speed, are really no competition.

However, as suggested in Section 2.3, it is possible to get good results out of such algorithms, if we are more clever about our choice of interval schedule. It was suggested that by reducing the mesh of the schedule, we would get better approximations to BF (and BFD). We defined a series of schedules

$$x(k) = \left\{ 0, \frac{1}{2^k}, \frac{2}{2^k}, \dots, \frac{2^{k-1}}{2^k}, 1 \right\},$$

and predicted that the average case behavior of $GX(k)F$ would approach that of BF as $k \rightarrow \infty$, and similarly for $GX(k)FG$ and BFD. Tables 7.1 and 7.2 contain entries for $GX(8)F$ and $GX(8)FG$, and these can be seen to be quite close to those for BF and BFD respectively. Figure 7.6 graphs the average case behavior of $GX(k)F$ and $GX(k)FG$ as $k \rightarrow \infty$, and we can see our prediction validated.

FIGURES 7.7 and 7.8: NEXT-k FIT as a function of k

In Section 1.2 we showed that although NEXT-2 FIT was a significant improvement over N1F in worst case behavior, the algorithms NkF for $k \geq 2$ all had the same worst case behavior. However, we predicted that as k increased, the average case behavior of NkF would approach that of FF. Figure 7.7 presents the results of a test of this hypothesis using 10 lists of length 200 with piece sizes "Uniformly" distributed in $(0,1)$, giving the percentage of excess bins required for FF and NkF , $k = 1, 2, 3, 5, 10, 15$, and 25. Again our prediction is validated.

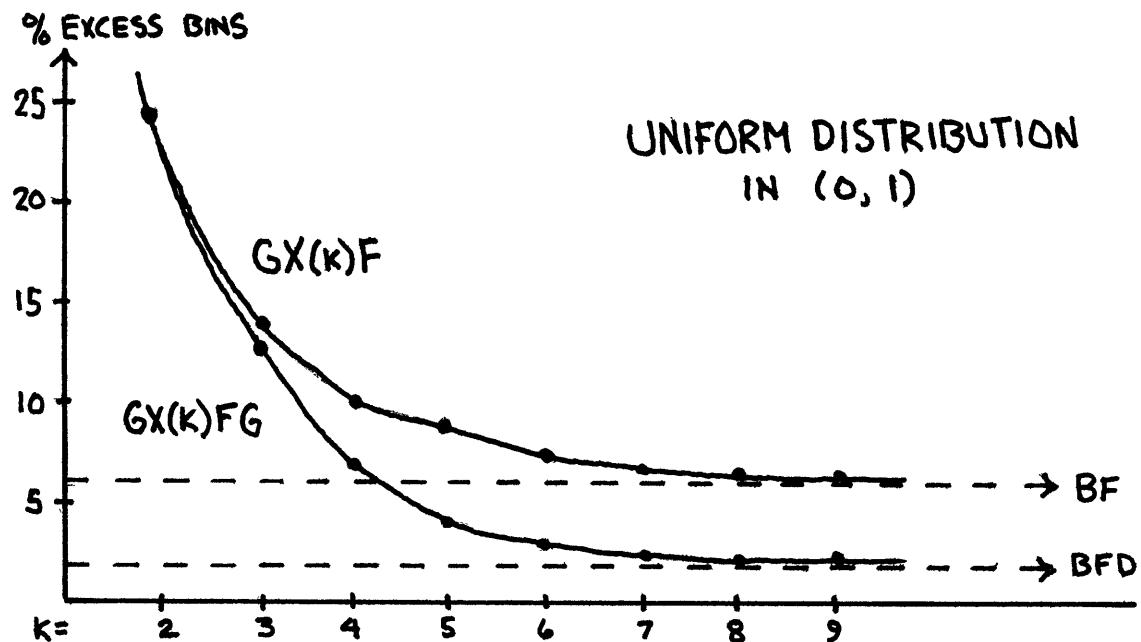


FIGURE 7.6. Average case behavior of $GX(k)F$ and $GX(k)FG$ as a function of k .

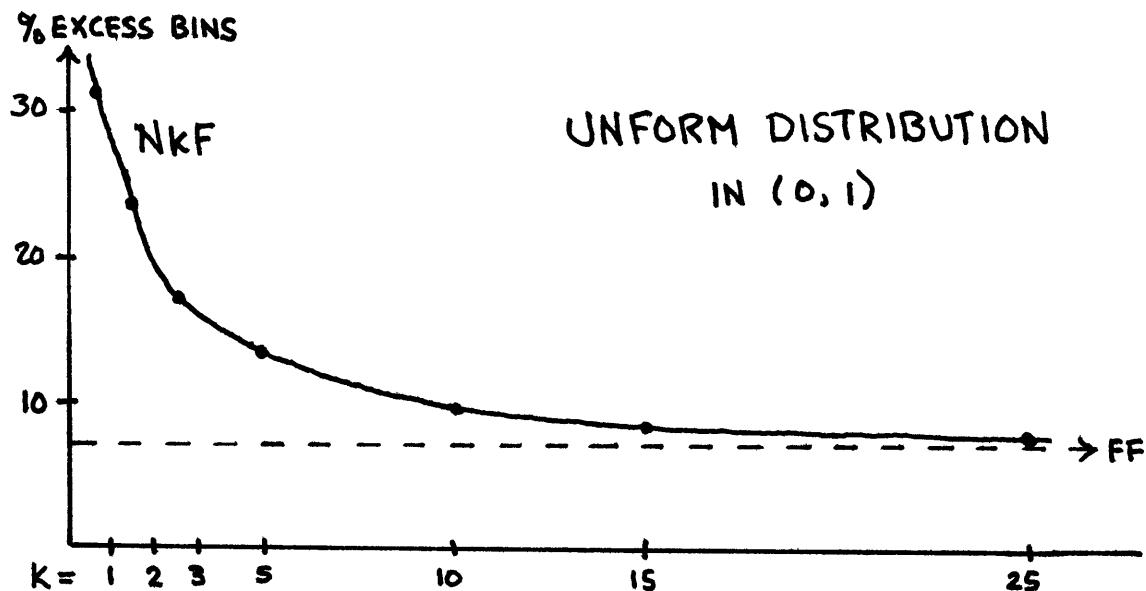


FIGURE 7.7. Average case behavior of NEXT-k FIT as a function of k .

However, one might question our use of lists of length 200, since the length of the list could have a significant effect on the results. If the list had 100 pieces and hence a packing of around 50 bins, N25F would examine around half the bins FF can consider in choosing its assignment. However, if the list length were 1000, N25F would only be looking at about 1/20 of the bins that FF can consider. Therefore, we tested out the possibility that list length might effect average case results by running N1F, N2F, N25F, and some of the other algorithms on 10 samples each of lists of length 100, 200, 500, and 1000, with piece sizes "Uniformly" distributed in (0,1). The results are graphed in Figure 7.8.

Note that there is a slight increase in the percentage of excess bins for the NkF algorithms as list length increases, but the other algorithms seem to be settling down to a constant value. In fact, BF experiences a considerable improvement at lengths 500 and 1000 over its behavior for 100 and 200. FF was not included in this study since our implementation of it was too slow.

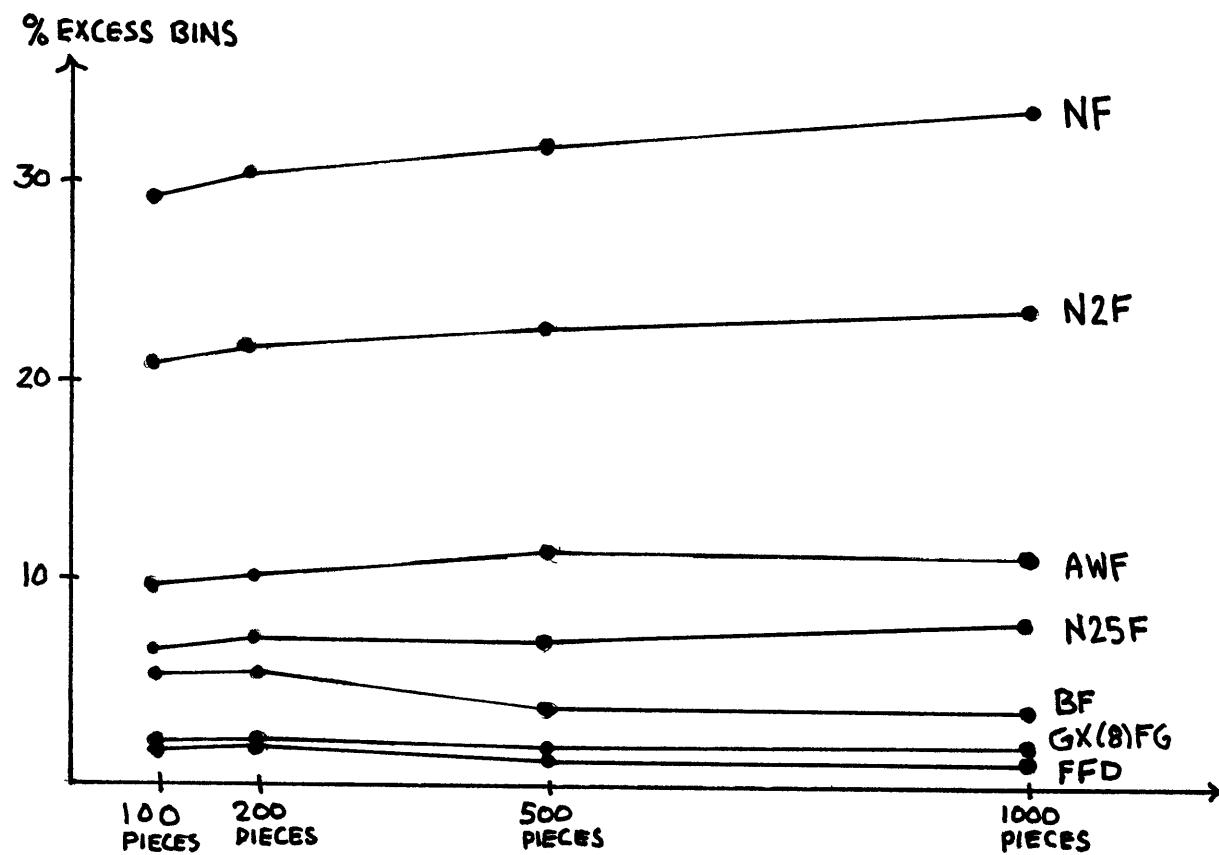


FIGURE 7.8. Average case behavior as a function
of list length.

This concludes our study of average case behavior. A number of questions have been raised, many of which have been referred to in the text. For instance, how does the distribution of piece sizes effect the ratio $L^*/W(L)$, on the average? If L^* is significantly greater than $W(L)$, how does this effect the ratio $S(L)/L^*$ for the various algorithms? Why is BEST FIT better than FIRST FIT?

Some of these questions might well be answered by a probabilistic analysis of the algorithms' expected behavior, and we would be very interested in the results of such a study. However, our worst case analysis can be of considerable help also. Many aspects of our average case results become intelligible in the light of such analysis, for instance, the apparent anomaly in the behavior of DECREASING algorithms when all piece sizes are close to $1/n$, and the differences in behavior between the nearly identical algorithms, WF and AWF. We were able to explain both of these in terms of the mechanisms that cause worst case behavior.

An additional value of our worst case analysis is that it provides us with an intuition as to which algorithms should behave nicely, and indeed has led us to propose several new algorithms which seem to do quite well on random lists. If the average case results presented here are reflected by the behavior of the various algorithms on the more idiosyncratic

lists encountered in realistic situations, we suspect that algorithms like BFD and the linear-time approximations to it that we have proposed will be of real practical use.

BIBLIOGRAPHY

- [Ad1] G. M. Adel'son-Vel'skii and Ye. M. Landis, "An algorithm for the organization of information," Soviet Math. Dokl. 3 (1962), 1259-1262.
- [Br1] A. R. Brown, Optimum Packing and Depletion, Macdonald and American Elsevier Inc., New York, N. Y., 1971.
- [Co1] S. A. Cook, The complexity of theorem-proving procedures, Proceedings of the 3rd Annual ACM Symposium on the Theory of Computing, 1971.
- [De1] A. Demers, private communication.
- [Ga1] M. R. Garey, R. L. Graham, and J. D. Ullman, "Worst-case analysis of memory allocation algorithms," Proceedings of the 4th Annual ACM Symposium on the Theory of Computing, 1972.
- [Ga2] M. R. Garey, private communication.
- [Gr1] R. L. Graham, "Bounds for certain multiprocessing anomalies," Bell. Sys. Tech. Jour. 45, no. 9 (1966) 1563-1581.
- [Gr2] R. L. Graham, "Bounds on multiprocessing timing anomalies," SIAM Jour. of App. Math. 17, no. 2 (1969) 416-429.

- [Gr3] R. L. Graham, "Bounds on multiprocessing anomalies and related packing algorithms," *Proceedings of the Spring Joint Computer Conference*, 1972.
- [Jol] D. S. Johnson, "Approximation algorithms for combinatorial problems," *Proceedings of the 5th Annual ACM Symposium on the Theory of Computing*, 1973.
- [Kal] R. M. Karp, "Reducibility among combinatorial problems," in R. E. Miller and J. W. Thatcher (ed), Complexity of Computer Computations, Plenum Press, New York, N. Y., 1972.
- [Kn1] D. E. Knuth, The Art of Computer Programming, Vol. 1, Addison-Wesley, Reading, Mass. 1969.
- [Kn2] D. E. Knuth, The Art of Computer Programming, Vol. 2, Addison-Wesley, Reading, Mass., 1969.
- [Ni1] J. Nievergelt and E. M. Reingold, "Binary Search Trees of bounded balance," *Proceedings of the 4th Annual ACM Symposium on the Theory of Computing*, 1972.
- [U11] J. D. Ullman, "The performance of a memory allocation algorithm," Technical Report No. 100, Princeton University, Princeton, N. J., 1971.

BIOGRAPHY

The author was born on December 9, 1945, and at age two demonstrated his capacity for abstract thought with his first words: "That's the ceiling up there." After logically disproving the existence of Santa Claus at age five, he entered a more or less normal childhood until his graduation from 8th Grade, at which time he was voted the most likely to be replaced by a computer.

His next four years were spent at Riverside High School in Milwaukee, Wisconsin, where he took up swimming and distance running, placing 4th in the City mile and the State MAA mathematics contest. He graduated in 1963 and traveled East to attend Amherst College. There he honored in Mathematics, captained the cross-country team, and secretary-treasurered the Literary Magazine. In his sophomore year he joined Psi Upsilon fraternity, in his junior year he joined Phi Beta Kappa, and in his senior year he joined Sigma Xi and graduated first in his class.

In 1967, after a summer at Los Alamos, he entered MIT as an NDEA Fellow in Mathematics, with an interest in Artificial Intelligence. In the summer of 1968 he wrote a Master's Thesis under Seymour Papert entitled "Look-ahead strategies in one-person games with randomly generated game trees." His next

BIOGRAPHY - Page 401

course of study turned out to be Basic Training.

After a distinguished academic career in the military (he was valedictorian of his Advanced Individual Training class, and somehow managed to survive through OCS), he married Susan B. Walker of Smith College, and settled into a comfortable sequence of desk jobs, first at the Registrar's Office of the Infantry School, and then in a bunker in Seoul, Korea.

Returning to MIT in September of 1971, he became interested in the Analysis of Algorithms, and in February 1972 began the research that led to this thesis. During this time he has been associated with both the Artificial Intelligence Laboratory and Project MAC. To keep his sanity in the middle of proofs with 23 induction hypotheses, he has resumed his long-distance running, and this year finished the Boston Marathon about the same time he finished Chapter 4.

He has accepted an appointment to the staff of the Bell Laboratories Mathematics and Statistics Research Center in Murray Hill, N. J., and hopes to avoid being replaced by that computer for a few more years yet.