JOURNAL OF COMPUTER AND SYSTEM SCIENCES 38, 2–35 (1989)

A Provably Efficient Algorithm for Dynamic Storage Allocation

E. G. COFFMAN, JR.

AT&T Bell Laboratories,
Murray Hill, New Jersey 07974

AND

F. T. LEIGHTON*

Department of Mathematics and Laboratory for Computer Science,
Massachusetts Institute of Technology,
Cambridge, Massachusetts 02139

Received August 25, 1986; revised September 21, 1987

The design and analysis of algorithms for on-line dynamic storage allocation has been a fundamental problem area in computer science for many years. In this paper we study the stochastic behavior of dynamic allocation algorithms under the natural assumption that files enter and leave the system according to a Poisson process. In particular, we prove that for any dynamic allocation algorithm and any distribution of file sizes, the expected wasted space (or fragmentation) in the system at any time is $\Omega(\sqrt{N}\sqrt{\log\log N})$, where N is the expected number of items (or used space) in the system. This result is known to be tight in the special case when all files have the same size. More importantly, we also construct a dynamic allocation algorithm which for *any* distribution of file sizes wastes only $O(\sqrt{N}\log^{3/4}N)$ space with very high probability. This bound is also shown to be tight for a wide variety of file-size distributions, including for example the uniform and normal distributions. The results are significant because they show that the cumulative wasted space in the holes formed by the continual arrival and departure of items is a vanishingly small portion of the used space, at least on the average. This fact is in striking contrast with Knuth's well-known 50% rule which states that the *number* of these holes is linear in the used space. Moreover, the proof techniques establish a surprising connection between stochastic processes, such as dynamic allocation, and static problems such as bin-packing and planar matching. We suspect that the techniques will also prove useful in analyzing other stochastic processes which might otherwise prove intractable. Lastly, we present experimental data in support of the theoretical proofs, and as a basis for postulating several conjectures. © 1989 Academic Press, Inc.

1. INTRODUCTION

The design and analysis of algorithms for dynamic storage allocation has been a fundamental area of research in computer science for many years [Kn]. In a typical

* This author's research was supported by an NSF Presidential Young Investigator Award with matching funds from Xerox and IBM, and in part by a consultant contract at AT&T Bell Laboratories.

0022-0000/89 $3.00
Copyright © 1989 by Academic Press, Inc.
All rights of reproduction in any form reserved.

setting, items (records, files, etc.) of varying sizes enter and leave a storage device in a sequence not known in advance. The storage device is represented by a set of consecutive locations or addresses. At its time of arrival, an item is allocated storage space consisting of a contiguous sequence of unoccupied locations equal in length to the item's size. Some time later, the item may depart, thereby making its space available to other items yet to arrive. An item cannot be moved prior to departure, and thus wasted space builds up over time in the form of interior *holes* alternating with regions of occupied space. This phenomenon is commonly known as *fragmentation.* The basic problem is to design algorithms which minimize, at least on the average, the cumulative wasted space in the interior holes.

Here an algorithm is defined by a rule for deciding where each item is to be placed upon arrival. Often this rule will be expressed by specifying a hole into which the item is to be placed, in which case it will be assumed that the item is placed at the lower (or leftmost) boundary of the hole. For simplicity, we will often assume that the storage device has unbounded capacity, so that the infinite region (a *non*-interior hole) beginning just after the highest occupied location is always available for the storage of a new item. In reality, of course, storage devices have bounded capacity, and there may be times when it is impossible to insert a new item. Among other things, the desire to minimize the likelihood of such a catastrophic event motivates the need to develop algorithms that minimize the cumulative wasted space in the interior holes.

First-fit (FF) is the best known and the most studied algorithm for dynamic allocation. According to FF, an item is placed into the lowest set of consecutive, unoccupied locations (i.e., the first hole) large enough to hold the item. Another well-known algorithm is best-fit (BF). BF stores an item in a smallest hole which is at least as large as the item. FF and BF are examples of *on-line* rules, i.e., rules that decide for each item at its time of arrival which locations it is to occupy throughout its stay in memory. The constraints are that the items must occupy disjoint locations, and arrival times, item sizes, and the times spent in system are not known in advance. Only on-line algorithms will be considered in this paper.

Combinatorial worst-case studies of dynamic storage allocation data back to the mid-sixties. In the texts by Knuth [Kn] and Standish [St] and in a survey by Coffman [C2] references to most of this research can be found. Probabilistic, or average-case results are scant by comparison, but far more important. In fact the relative importance of probabilistic results is even greater in dynamic allocation than in static scheduling problems such as bin packing. The reason is that the worst-case performance of effective heuristics like FF and BF is a very poor indication of probable performance. For example, Robson [Ro] has shown that the ratio of wasted space to occupied space is, in the worst-case, an increasing funcion of the number of items stored under the BF rule. (This is to be contrasted with the constant bounds of bin-packing and scheduling.) On the other and, we shall provide convincing evidence that the ratio of expected wasted space to expected occupied space tends to zero as the number of items stored tends to infinity.

In this paper we adopt the mathematical model of dynamic allocation on the

continuous real line which was defined by Coffman, Kadota, and Shepp [CKS2]. In this model, arrivals are assumed to be Poisson at rate λ and the residence times of items in storage are taken to be independent, exponentially distributed random variables with expected values $1/\mu$. The sizes of items are sampled independently from a distribution $F(x)$. As a convenient normalization, item sizes are considered to be numbers in the unit interval. This model originated with Knuth [Kn] as a basis for his well-known 50% rule; namely, that if the frequency of placing items into holes of exactly the same size can be neglected, then in statistical equilibrium the expected number of holes is one half the expected number of items in storage. The rule applies only to algorithms like FF and BF which place new arrivals into holes, justified to one boundary or the other, but it also extends to interarrival and residence time distributions other than the exponential.

For the continuous Markov model, Coffman, Kadota, and Shepp [CKS1] obtained the distribution of wasted space (cumulative interior hole size) produced by FF under the assumption that all items have the same size. While the analysis remains decidedly nontrivial under this assumption, we note that the general problem degenerates; specifically, an item can be placed into any hole, and there are no "fitting" problems. The form of the result in [CKS1] is rather awkward, and it seems to be very difficult to extract asymptotic information from it. Using other methods they showed that expected wasted space was $O(\sqrt{N \log N})$, where N, the "size" of our problem, is the average number of stored items in statistical equilibrium. Based on Monte Carlo simulations for other item size distributions, notably the uniform, it was conjectured in [CKS2] that the asymptotic expected wasted space under FF remained $o(N)$ for any distribution on [0, 1].

In this paper, we shall prove far stronger and more general asymptotic bounds. We start by proving a lower bound, $\Omega(\sqrt{N \log \log N})$, on the asymptotic expected wasted space produced by any on-line algorithm for any distribution within this model. In tightening the asymptotic results in [CKS1], Aldous [A1] has shown that this bound is achieved by FF in the special case of equal item sizes. Proving upper bounds for arbitrary distributions is much more difficult, however, and is the main thrust of this paper.

In particular, we define and analyze a constrained version of BF, called best-fit-aligned (BFA), which must respect certain prespecified boundaries in storage, i.e., added to the best-fit criterion is the requirement that items be stored wholly within some consecutive pair of the fixed boundaries. We then prove that BFA is optimal (in a stochastic sense) among boundary-respecting allocation algorithms, and more importantly that the expected wasted space for BFA is $O(\sqrt{N \log^{3/4} N})$ no matter what distribution is assumed for item sizes. All that is required is that the distribution be known in advance. For most distributions, such as the uniform distribution, this bound is tight; for all t sufficiently large, the cumulative wasted space is $\Theta(\sqrt{N \log^{3/4} N})$ with a probability that tends very quickly to 1 as $N \to \infty$.

The heart of the proof of the $\Theta(\sqrt{N \log^{3/4} N})$ result involves a mapping from the BFA storage process into a more tractable, matching process whose departure from the original, in terms of expected wasted space, can be bounded. The mapping

technique has a general structure that is potentially applicable to the analysis of other non-Markov processes arising in the stochastic analysis of algorithms. The matching process is then (surprisingly) identified as an application of the *maximum upright matching* problem defined by Karp, Luby, and Marchetti [KLM] in their analysis of two-dimensional bin-packing. Recent results for the upright matching problem by Leighton and Shor [LS, Sh] then provide the desired bounds for the BFA analysis.

At present, there appears to be no on-line algorithm for the dynamic allocation problem with provably better performance. Although it is conceivable that algorithms such as BF and FF perform better, we strongly suspect that this is not the case. Indeed, our experimental evidence suggests that BF and BFA have very nearly the same average case behavior for $F(x)$ uniform on $[0, 1]$, and that the corresponding behavior of FF is dramatically worse. (The extensive experimental data in [CKS2] suggests an estimate of $\Theta(N^{4/5})$ wasted space for FF.) In any case, our results are currently the best *proved* bounds on the performance of optimal dynamic storage allocation.

In addition to a reduction in wasted space, BFA provides a dramatic reduction in the expected number of holes that must be maintained as part of a data structure for implementing the algorithm. In particular, we shall prove that the expected number of potentially usable holes for BFA is $\Theta(\sqrt{N}\log^{3/4} N)$, whereas this expected number is $\Theta(N)$ for BF and FF. (The latter fact is just Knuth's 50% rule.) Hence, for the fastest known implementations of FF, BF, and BFA, this will imply that expected hole-search times can be asymptotically twice as long for FF and BF as for BFA.

Our wasted space bound for BFA has strong consequences for practical storage allocation systems. In particular, we can conclude that there is little need to worry about the wasted space that builds up in interior holes. For even moderately large N, the fraction of wasted space is small, and it quickly tends to 0 as $N \to \infty$. Thus, time consuming compaction (garbage collection) systems, and systems that are designed to create sufficiently large holes when none exist [BCW], will not be worthwhile except in very special circumstances. In fact, we will prove that for $F(x)$ uniform on $[0, 1]$, the probability that the rightmost item in BFA exceeds position $N/2 + c\sqrt{N}\log^{3/4} N$ for sufficiently large c is $O(N^{-\alpha\sqrt{\log N}})$ for some constant $\alpha > 0$. (The value of α depends on c, but for c sufficiently large we can assume that $\alpha = 1$.) Hence, it is extremely unlikely that BFA will require compaction within polynomial time, provided the storage device has $c\sqrt{N}\log^{3/4} N$ more space than the expected used space $(N/2)$. Similar results are true for arbitrary $F(x)$ on $[0, 1]$.

We conclude this section with a brief review of related literature. Our model specialized to equal item sizes seems to have been studied first by Kosten [Ko] almost 50 years ago, although from a different point of view. There, the model represented multiple-server systems with applications to the design of communication systems. A monograph by Newell [Ne] has recently appeared on the analysis of this model.

Kadota, Shepp, and Ziv [KaSZ] designed and analyzed an algorithm of the

first-fit type to handle distributions with a finite number k of possible item sizes. Essentially, storage was partitioned into k disjoint regions, each being reserved solely for items of the same size. The algorithm was undesirable for practical reasons, and it became increasingly awkward as k became large; but it established for a large class of distributions that there were algorithms of simple structure which produced $o(N)$ wasted space.

The properties of optimal dynamic storage allocation have been studied by Benês [Be1, Be2]. Using a dynamic programming approach, the design of optimal algorithms for small memories has led to a number of insights into the general case. In [Be2] Benês illustrates the difficulties in resolving even very simple questions. For example, define optimality as the minimum expected wasted space in the stationary regime and consider the following interesting open problem: Can it be assumed that whenever an optimal algorithm places an item into a hole, it justifies it to one boundary or the other?

Because of the similarities in approximation algorithms and the objectives of the analysis, and because of the common application to storage problems, the recent research in one-dimensional bin-packing is worth mentioning as background for the present paper. (See [CLR] for a recent survey of probabilistic results.) The relation between bin-packing and dynamic allocation is much the same as the relation between static and dynamic data structures. Our dynamics problem is of course fundamentally more difficult. We have already seen an example of this. When item sizes are assumed to be equal, the solution of our problem remains quite difficult [A1, CKS1, Ne], whereas the bin-packing problem becomes completely trivial.

The remainder of the paper is divided into seven sections. In the next section we begin with definitions of the stochastic processes to be analyzed in this paper. We then present a number of well-known limit theorems for several of these processes. Feller [Fe] provides a standard source for most of this material. In Section 3 we prove a general lower bound on expected wasted space for any on-line allocation algorithm. We also show that this bound is achieved by FF when all items have the same size. In Section 4 we turn to the analysis of the BFA algorithm. We begin with definitions and certain important properties of BFA, after which the $\Theta(\sqrt{N}\log^{3/4} N)$ wasted space result is proved. Experimental data is reported in Section 5, and concluding remarks are given in Section 6. Acknowledgments and References follow.

2. Definitions and Preliminary Results

Throughout the paper, we will use the phrase "with very high probability" to mean "with probability exceeding $1-O(N^{-\alpha\sqrt{\log N}})$ for some constant $\alpha>0$." Although we will not worry about computing exact values for α, it is worth noting that α can be set to one for the most important results.

The condition that an event happen with very high probability is quite strong. For example, any collection of $O(N^{\beta})$ very high probability events (possibly depen-

dent) jointly form a very high probability event for constant β. This is because the probability that all of the events happen is at least $1-O(N^{-\alpha\sqrt{\log N}+\beta})=1-O(N^{-\alpha'\sqrt{\log N}})$, where α' is any constant less than α. We will make implicit use of this fact throughout the paper.

To each on-line allocation algorithm we associate a *storage process* $\{Z(t), t\geqslant 0\}$, where $Z(t)$ is the set of occupied intervals at time t. We let $n(t)=|Z(t)|$ denote the number of items in the system. Hereafter, the mean of the exponential residence times is normalized to one *time unit*. The rate of the Poisson arrival process is N items per time unit. By a classical result [Fe, Vol. I, p. 461] this means that N is also the expected number of items in the system in statistical equilibrium. With no significant loss in generality we assume that N is an integer. Since N denotes an instance of our problem, it will hereafter subscript the various processes of interest.

Denoting

$$Z_N(t)=([l_1, l_2), [l_3, l_4), \ldots, [l_{2n-1}, l_{2n})), n=n_N(t),$$

the processes $\{u_N(t)\}$, $\{r_N(t)\}$, and $\{w_N(t)\}$ are defined by

$$w_N(t)=\sum_{1\leqslant i\leqslant n-1}(l_{2i+1}-l_{2i}), \qquad \text{the total hole size (wasted space)}$$

$$r_N(t)=l_{2n}, \qquad \text{the rightmost occupied point,}$$

$$u_N(t)=r_N(t)-w_N(t), \qquad \text{the used space or total mass at time } t.$$

Except where noted otherwise, we shall assume that $Z_N(0)=\varnothing$; accordingly, $n_N(0)=w_N(0)=r_N(0)=u_N(0)=0$. Also, we assume that item sizes have a distribution $F(x)$ in the class of *standard distributions* defined by the properties

$$\text{minimum item size}\geqslant 0$$

$$\text{maximum item size}\leqslant 1$$

$$\text{mean item size}=\tfrac{1}{2}.$$

Other distributions can also be handled but the wasted space measurements must be scaled accordingly. If the mean item size is a constant (independent of N), then this scaling only introduces (hidden) constant factor changes in our bounds for wasted space. We let σ^2 denote the variance of $F(x)$. For random variables with stationary distributions we shall frequently omit the explicit limiting dependence on t, e.g.,

$$\Pr\{r_N\leqslant k\}\equiv\lim_{t\to\infty}\Pr\{r_N(t)\leqslant k\},$$

when this limit exists.

Limit laws. Asymptotic properties of several of the processes related to the storage process $\{Z_N(t)\}$ are well known. Underlying most of these properties is a

large-deviation version of the central limit theorem (e.g., see [Fe, Vol. II, Section XVI.7]). Let X_m be the sum of m i.i.d. random variables each with mean EX and variance Var X. If x_m is a nondecreasing function of m such that $x_m = o(m^{1/6})$, then

$$\Pr\{X_m > EX_m + x_m\sqrt{\text{Var } X_m}\} \sim 1 - \Phi(x_m) \qquad \text{as } m \to \infty,$$
$$EX_m = m\,EX, \qquad \text{Var } X_m = m \text{ Var } X, \tag{2.1}$$

where $\Phi(\cdot)$ is the normal distribution with zero mean and unit variance. Simplifying a classical bound for the tails of $\Phi(\cdot)$ [Fe, Vol. I, p. 175], we have

$$\Pr\{EX_m - x_m\sqrt{\text{Var } X_m} < X_m < EX_m + x_m\sqrt{\text{Var } X_m}\} = 1 - O(e^{-x_m^2/2}). \tag{2.2}$$

Thus, deviations of X_m from the mean which, as functions of m, exceed multiples of the standard deviation become very improbable events as m becomes large. Similar bounds can be established for arbitrary x_m although (2.1) need not hold.

A special case of these results, which is important for our problem, occurs when X_m has a binomial distribution. (Note that in this case X_m can be expressed as a sum of 0–1 random variables.) For example, the number of departures from a fixed set of m items in a time interval $[t, t+T]$ has a binomial distribution with $EX_m = m(1-e^{-T})$ and Var $X_m = me^{-T}(1-e^{-T})$. This follows from the fact that departures of items in any interval $[t, t+T]$ are i.i.d. events each with probability $1-e^{-T}$ of departure, and probability e^{-T} of survival. Hence, (2.1) and (2.2) apply directly to this departure process.

Poisson distributed random variables are another important application of (2.1) and (2.2). Such random variables arise in two places in our model. By definition the number of arrivals $X_N \equiv X_N(T)$ in $[t, t+T]$ has a Poisson distribution with mean and variance NT. A direct asymptotic analysis of the Poisson distribution function shows that (2.1) and (2.2) hold with m replaced by N. Also, by our assumption $n_N(0) = 0$, it is not difficult to prove that $n_N(t)$ is a Poisson distributed random variable (e.g., see [HPS, p. 101]), in this case with mean and variance $N(1-e^{-t})$. Results similar to (2.1) and (2.2) are therefore possible.

For Poisson random variables it will be convenient to have a simple bound for the probability of deviations proportional to the mean. (Here, $x_N = \Omega(\sqrt{N})$ so (2.1) and (2.2) do not apply.) In particular, let N be the mean and variance. Working directly with the Poisson distribution it can be shown by standard techniques [C1] that for any $\alpha > 0$ there is another $\beta > 0$ such that

$$\Pr\{-\alpha N < X_N - N < \alpha N\} = 1 - O(e^{-\beta N}). \tag{2.3}$$

The fact that $n_N(t)$ is a Poisson random variable also makes it easy to estimate the time required for $n_N(t)$ to reach points relatively near the equilibrium mean value $N = \lim_{t\to\infty} N(1-e^{-t})$. For example, consider $t = \frac{1}{2}\log N$, so that $N(1-e^{-t}) = N - \sqrt{N}$, and suppose we want the probability of the event $n_N(t) > N - \sqrt{N}\log^b N$ for some $b > 0$ (such examples appear in Section 4). A direct

calculation involving the Poisson distribution with parameter $N-\sqrt{N}$ shows that there is a constant $\alpha>0$ such that

$$\Pr\{n_N(t)>N-\sqrt{N}\log^b N\}=1-O(e^{-\alpha\log^{2b}N})$$

for any $t\geqslant\frac{1}{2}\log N$. Because of the form of the results that we need from upright matching theory, the choice $b=\frac{3}{4}$ will be convenient in Section 4. In this case we have for all $t\geqslant\frac{1}{2}\log N$ and some $\alpha>0$,

$$\Pr\{n_N(t)>N-\sqrt{N}\log^{3/4}N\}=1-O(N^{-\alpha\sqrt{\log N}}). \tag{2.4}$$

Hence, for $t\geqslant\frac{1}{2}\log N$, $n_N(t)\geqslant N-\sqrt{N}\log^{3/4}N$ with very high probability. As a consequence, note that if we prove that $n_N(t)$ has some other property with very high probability under the assumption $n_N(t)\geqslant N-\sqrt{N}\log^{3/4}N$, then we can prove the same assertion under the assumption $t\geqslant\frac{1}{2}\log N$. We will exploit this fact on several occasions where we prove a hypothesis for "sufficiently large t," while making use of results describing the stationary behavior of $n_N(t)$.

One such result concerns the stationary behavior of displacements in the process n_N. The displacement, $\delta_N(t,T)=n_N(t+T)-n_N(t)$, from $n_N(t)$ in the interval $[t,t+T]$ in statistical equilibrium, is asymptotically normally distributed with the asymptotic mean and variance [Ig1; Fe, Vol. II, p. 335]

$$\begin{aligned}E\,\delta_N(T)&\sim(N-n_N(t))(1-e^{-T})\\ \operatorname{Var}\delta_N(t)&\sim(N+n_N(t)\,e^{-T})(1-e^{-T}).\end{aligned} \tag{2.5}$$

For any constant T we can use this result for estimates of the form (2.2) by taking t sufficiently large. In particular, for $x_N=o(N^{1/6})$ and for all $t\geqslant\frac{1}{2}\log N$ we have

$$\Pr\left\{-x_N<\frac{\delta_N(t,T)-(N-n_N(t))(1-e^{-T})}{\sqrt{(N+n_N(t)e^{-T})(1-e^{-T})}}<x_N\right\}=1-O(e^{-x_N^2/2}). \tag{2.6}$$

As a specific example, to be used again in Section 4, consider $n_N(t)\geqslant N+\sqrt{N}\log^{3/4}N$ and transitions from $n_N(t)$ to $n_N(t+T)\leqslant N$. From (2.5) and (2.6) a routine calculation yields for any fixed T and some constant $\alpha>0$,

$$\Pr\{n_N(t+T)\leqslant N\mid n_N(t)\geqslant N+\sqrt{N}\log^{3/4}N\}=O(N^{-\alpha\sqrt{\log N}}). \tag{2.7}$$

In statistical equilibrium the process of departures from storage is Poisson with parameter N and has the same asymptotic behavior as the arrival process (although the two processes are clearly dependent). It is easily verified from this observation and the earlier results that the number of arrival and departure events in an interval of length T in statistical equilibrium is asymptotically normally distributed with mean $2NT$ and a standard deviation $\Theta(\sqrt{NT})$.

An error of $\Theta(\sqrt{N}\log^{3/4}N)$ in estimating the number of events occurring in some interval will not affect our asymptotic results. Consequently, for large t and any $\beta > 0$ we will be able to treat as equivalent an interval of length βT and an interval in which $2\beta NT$ events (or βNT arrival events) occur, even for results that are true with very high probability.

Finally, consider the total-mass process $\{u_N(t)\}$, which is very simply related to the process $\{n_N(t)\}$. In particular, $u_N(t)$ is equal in distribution to the sum of $n_N(t)$ i.i.d. random variables with the distribution $F(x)$ of item sizes. Since $F(x)$ is a standard distribution we have $Eu_N(t) = En_N(t)/2 = N(1-e^{-t})/2$ and $\operatorname{Var} u_N(t) = \sigma^2 En_N(t) = \sigma^2 N(1-e^{-t})$. Moreover, for the stationary process we have as before a limiting normal distribution [Ig2] and thus

$$\Pr\left\{\frac{u_N - N/2}{\sigma\sqrt{N}} > x\right\} \sim 1 - \Phi(x) \qquad \text{as } N \to \infty.$$

Although the limit process is a Gaussian process, it is not Markov and we do not have the normal displacement probabilities as we did for $\delta_N(T)$. However, we have no need to estimate these probabilities. Indeed, we need only a much weaker version of (2.2): For t sufficiently large (e.g., $t \geqslant \frac{1}{2}\log N$)

$$\Pr\left\{\frac{N}{2} - x_N\sqrt{N\sigma^2} < u_N(t) < \frac{N}{2} + x_N\sqrt{N\sigma^2}\right\} = 1 - o(1), \tag{2.8}$$

where x_N is any increasing function of N.

The only other result we need will be useful in proving lower bounds. It is a straightforward refinement of a well-known result concerning stationary Gaussian processes, which can be found in [CrL, Eq. (12.3.1), p. 272].

LEMMA 2.1. *There exist constants $c_1, c_2 > 0$ such that for all N, k, and $t \geqslant \frac{1}{2}\log N$ the maximum observed mass in the window of kN events occurring after time t satisfies*

$$\frac{N}{2} + c_1\sqrt{N}\sqrt{\log k} < \max u_N(t) < \frac{N}{2} + c_2\sqrt{N}\sqrt{\log k}$$

with probability at least $1-\varepsilon$, where $\varepsilon \to 0$ as $k \to \infty$.

Because of the assumed stationarity, the condition on t is not needed in [CrL]. Our including it here can influence only the constants. For the limiting case $t \to \infty$, the analysis in [CrL] shows that as k, N tend to ∞, both c_1 and c_2 tend to $\sigma\sqrt{2}$, and ε tends to 0.

3. A General Lower Bound

In this section we prove a $\Omega(\sqrt{N \log \log N})$ lower bound for the expected wasted space produced by any on-line algorithm. We then present a simplified proof of Aldous' result [A1] that the bound is achieved by FF when all items have the same size.

As one might expect from the form of the bound, the proof is reminiscent of the well-known "law of the iterated logarithm." However, the similarity is in fact confined to the proof of Lemma 2.1, which our result uses. Aside from this, the two problems are quite different. For the remainder of this section we assume $\sigma = 1$ for simplicity.

THEOREM 3.1. *There is a constant $c > 0$ such that for any constant $\varepsilon > 0$ and any standard distribution of item sizes, the space wasted by any on-line allocation algorithm at any time $t \geq \log N$ is at least $c\sqrt{N}\sqrt{\log \log N}$ with probability at least $1 - \varepsilon$ for all N sufficiently large.*

Proof. Consider the interval of time $[t - \frac{1}{4}\log N, t]$. Because at least $\frac{1}{2}\log N$ time has already elapsed to this point, t is sufficiently large that we can set $k = \frac{1}{4}\log N$ in Lemma 2.1 and postulate a constant b such that the total mass reaches at least $N/2 + b\sqrt{N}\sqrt{\log \log N}$ at some point, say t', during $[t - \frac{1}{4}\log N, t]$ with probability $1 - \varepsilon$ for any constant $\varepsilon > 0$ and all N sufficiently large.

Now consider the rightmost $(b/3)\sqrt{N}\sqrt{\log \log N}$ items in the system at time t'. Since each item has size at most one, the leftmost of these is in position at least $N/2 + (2b/3)\sqrt{N}\sqrt{\log \log N}$. Moreover, at least one of these items still remains in the system at time t, with probability $1 - \varepsilon$ for any constant $\varepsilon > 0$ and N large enough. This is because the probability of any particular item leaving in the course of at most $\frac{1}{4}\log N$ time units is at most $1 - O(e^{-(1/4)\log N})$. And since

$$(1 - e^{-(1/4)\log N})^{(b/3)\sqrt{N}\sqrt{\log \log N}} = o(1),$$

the probability that all $(b/3)\sqrt{N}\sqrt{\log \log N}$ items leave in this interval is $o(1)$.

By the preceding analysis we know that there is an item in position at least $N/2 + (2b/3)\sqrt{N}\sqrt{\log \log N}$ at time t with probability at least $1 - \varepsilon$ for any constant $\varepsilon > 0$ and N large. By (2.8) we know that the total mass at time t is at most $N/2 + (b/3)\sqrt{N}\sqrt{\log \log N}$ with probability at least $1 - o(1)$ for N sufficiently large. Hence, we can conclude that the wasted space at time t is at least $(b/3)\sqrt{N}\sqrt{\log \log N}$ with probability at least $1 - \varepsilon$ for any constant $\varepsilon > 0$ and all N sufficiently large. ∎

For equal item sizes, Aldous [A1] has shown that the bound in the previous theorem is attained by FF. Indeed, he found the limiting distribution and showed that the expected wasted space is $[\sqrt{2} + o(1)]\sqrt{N}\sqrt{\log \log N}$. Unfortunately,

Aldous' argument is very complicated. In what follows we provide a much simpler proof that FF wastes only $O(\sqrt{N}\sqrt{\log\log N})$ space most of the time. However, we do not know how to obtain the exact multiplicative constant with an elementary proof.

THEOREM 3.2. *If all items have the same size, then there is a constant $c>0$ such that for any constant $\varepsilon>0$ and all N sufficiently large, the space wasted by FF at any time t is at most $c\sqrt{N}\sqrt{\log\log N}$ with probability at least $1-\varepsilon$.*

Proof. By our standard-distribution assumptions on $F(x)$ we take $\frac{1}{2}$ as the common item size. Because of the assumed initial condition, $n_N(0)=0$, we shall restrict ourselves to $t\geqslant\frac{5}{2}\log N$, so that the process $n_N(t)$ has had time to build up to points near the equilibrium mean value, N. The arguments below will simplify for small t, since the bound is in fact not tight in that case. Specifically, we consider an interval of $2\log N$ time units starting at $t'\geqslant\frac{1}{2}\log N$ and ending at t. By setting $k=2\log N$ in Lemma 2.1, we know that there is a constant b such that the maximum total mass observed during this interval is at most $N/2+b\sqrt{N}\sqrt{\log\log N}$ with probability $1-\varepsilon$ for any constant $\varepsilon>0$ and all sufficiently large N. Hence, each newly arriving item in this interval occupies a position at most $N/2+b\sqrt{N}\sqrt{\log\log N}$.

We next verify that every item in the system at t' has departed by t with high probability. This is accomplished by observing that there are fewer than $2N$ items in the system at t' with very high probability (see (2.3) applied to $n_N(t)$), so that the probability that one or more of them remains at time t is at most

$$2Ne^{-2\log N}=o(1).$$

Hence, the rightmost item in the system at t occupies a position at most $N/2+b\sqrt{N}\sqrt{\log\log N}$ with probability at least $1-\varepsilon$ for any ε and N large enough. By (2.8) we know that the mass at t is at least $N/2-b\sqrt{N}\sqrt{\log\log N}$ with equivalently high probability. Hence, the wasted space at time t is at most $2b\sqrt{N}\sqrt{\log\log N}$ with probability at least $1-\varepsilon$ for all $\varepsilon>0$ and N sufficiently large. ∎

In fact, FF and BFA are both equal and are both optimal for the special case when all items have the same size. (This is not necessarily true of BF.) The observation that they are equal is trivial. The proof that they are optimal is deferred to Section 4.2, where the necessary tools are developed.

4. THE BEST-FIT ALIGNED ALGORITHM

We begin this section by defining the best-fit aligned (BFA) algorithm and its corresponding Markov storage process. We shall then prove an important property of BFA allocation which linearly orders sets of storage states based on the dis-

tribution of subsequent events. We then introduce the upright matching problem and describe its connection with BFA allocation. Finally, tight bounds on the space wasted by BFA are proved.

4.1. *Definitions and Motivation*

To simplify our descriptions we assume initially that the item-size distribution is uniform over [0, 1]. The adaptation of BFA to a general distribution of item sizes is easy to work out and is safely left until after the analysis.

Under our assumptions BFA partitions storage (i.e., $[0, \infty)$) into a sequence of intervals of consecutive lengths $1/N$, $2/N$, ..., $(N-1)/N$, 1, 1, 1, ... That is, storage is partitioned into the intervals

$$\left[0, \frac{1}{N}\right)\left[\frac{1}{N}, \frac{3}{N}\right)\left[\frac{3}{N}, \frac{6}{N}\right)$$
$$\cdots\left[\frac{(N-1)(N-2)}{2N}, \frac{N-1}{2}\right)\left[\frac{N-1}{2}, \frac{N+1}{2}\right)\left[\frac{N+1}{2}, \frac{N+3}{2}\right)\cdots .$$

In the order given these intervals will be called *cells* 1, 2, 3, ... Cells $N+1$, $N+2$, ... will be defined collectively as the *overflow region.* The BFA algorithm is defined as the modification of BF which respects cell boundaries and which places at most one item in each cell; i.e., a new item is placed completely within the smallest unoccupied cell that is at least as large as the item. Note that we could just as well have called the algorithm first-fit-aligned, since an item is always placed first-fit subject to cell boundaries and occupancy. As a practical matter, the search for an unoccupied cell to contain an item of size x can begin at cell $\lceil Nx \rceil$. Since x is uniform on [0, 1], searches are equally likely to begin at any one of the first N cells.

The definition of BFA was originally motivated by our inability to analyze BF and FF theoretically, and by our empirical observation that FF tends to allocate items in order of increasing size (see also [Pa] and [Sho]). This is precisely the phenomenon that is forced to some degree by the cell boundaries in BFA. Further motivation was obtained when later we were able to analyze BFA theoretically and find its performance to be close to optimal theoretically, much better than FF empirically, and extremely close to BF empirically. Indeed, the data reported in Section 5 are not sufficient to distinguish the performance of BFA and BF. As an added bonus, we found BFA to be a much easier algorithm to implement experimentally, since its cell structure is discrete and since there are provably far fewer holes maintained by BFA than by BF or FF.

As the reader might have observed, the BFA rule can be extended to other partition sizes. For example, a number N^*, smaller or larger than N, could be selected, and a partition into cells of sizes $1/N^*$, $2/N^*$, ..., $(N^*-1)/N^*$, 1, ... could be defined. Indeed, we shall show empirically in Section 5 that with N^* somewhat larger than N modest improvements in performance can be obtained.

We now define the storage process $\{S_N(t);\ t \geqslant 0\}$, where $S_N(t) = s_1 s_2 \cdots s_r$ is a bit

string indicating occupied and interior unoccupied cells. $S_N(t) = 0$ denotes an empty store; otherwise, $s_r = 1$ indicates that the rightmost occupied cell is cell r. Cell i, $1 \leqslant i < r$, is occupied or unoccupied according as $s_i = 1$ or $s_i = 0$, respectively. Transitions out of state $s_1 \cdots s_r \neq 0$ are at rate $n_N(t) = \sum_{1 \leqslant i \leqslant r} s_i$ into states with one of the 1 bits in $S_N(t)$ removed and at rate N into states with a 1 bit added. The cell, j, to become occupied ($s_j = 0$) satisfies $j \leqslant r+1$ and, if $j \leqslant N$, becomes occupied with probability k/N, where $k-1$ is the length of the longest string of 1's ending at position $j-1$. If $j > N$, then for cell j to become occupied we must have $s_N, s_{N+1}, \ldots, s_{j-1} = 1$; in this case cell j becomes occupied with probability k/N, where k is the length of the longest string of 1's ending at position N. Consistent with our earlier assumpton, we assume an initially empty storage device, $S_N(0) = 0$.

The process $\{S_N(t)\}$ is clearly Markov. Moreover, since $\{n_N(t)\}$ is ergodic, $\{S_N(t)\}$ is easily seen to be ergodic. Let S_N be a random variable with the stationary distribution. A direct analysis leading to the distribution of S_N appears to be out of reach. To acquire results asymptotic in N we resort to special techniques.

We adapt our earlier notation as follows. The process $\{r_N(t)\}$ now specifies the highest occupied cell at time t, i.e., $r_N(t) = r$, where $S_N(t) = s_1 \cdots s_r$. Let $h_N(t)$ denote the number of unoccupied cells $i < r_N(t)$. As in the general case, we let $w_N(t)$ denote the cumulative wasted space, viz. the total size of the unoccupied cells counted by $h_N(t)$ plus the sum of the unoccupied subintervals within occupied cells. The process $u_N(t)$ has the same definition (cumulative used space) as before.

In subsequent sections we shall exploit a key simplification in the BFA analysis, viz. that asymptotic expected value results for r_N and h_N provide us with corresponding results for w_N. In effect, the following result supplies us with a discretization of our problem at no cost.

LEMMA 4.1. *For the BFA algorithm*

$$Ew_N = Eh_N + \tfrac{1}{2} + o(1) \qquad \text{as } N \to \infty.$$

Proof. In what follows, we assume that $r_N \geqslant N$. While the reader would find little difficulty in using the bounds of Section 2 to prove that this is true with high probability, it is unnecessary; it is an immediate consequence of the much stronger result of Theorem 4.2 (to be proved later in this section) which implies that $r_N \geqslant N$ with probability exceeding $1 - O(N^{-\alpha})$, $\alpha > 1$. Since $w_N < N$ when $r_N < N$, the case when $r_N < N$ can contribute at most $o(1)$ to the value of Ew_N. Hence, we may assume $r_N \geqslant N$.

When $r_N \geqslant N$ the total space spanned by cells 1 through r_N is

$$\left(\sum_{i=1}^{N} i/N\right) + r_N - N = r_N - \frac{N-1}{2}.$$

The expected occupied space is simply $N/2$, and hence the expected wasted space is simply

$$Ew_N = E(r_N - N) + \tfrac{1}{2} + o(1) = Eh_N + \tfrac{1}{2} + o(1),$$

as claimed. ∎

In fact, w_N and h_N track very closely. In particular, it is not difficult to show that $|w_N - h_N| \leqslant O(\sqrt{N})$ with a probability approaching 1 as $N \to \infty$.

4.2. *A Monotonicity Property*

As might be expected, it becomes very useful to compare different states on the basis of the probable numbers of overflow events that occur in subsequent states. (An *overflow event* occurs whenever an item is placed into a cell $i > N$ of the overflow region.) The result of this subsection shows in fact that certain sets of states can be linearly ordered on this basis. This property will be obtained inductively after first examining the effect of slight perturbations in the starting configuration of the BFA process on its subsequent behavior. Of course, by the ergodicity of $\{S_N(t)\}$ the long-term behavior will be independent of the starting configuration. However, it seems reasonable intuitively that two similar starting configurations should result in similar subsequent behavior even in the near term. For example, if S and S^* are two starting configurations such that $S = \omega_1\, 0\, \omega_2$ and $S^* = \omega_1\, 1\, \omega_2$ for some ω_1 and ω_2 (i.e., S^* is identical to S except that it has one extra item), then it would seem that: (1) the anticipated overflow during a given interval of time for S should be no more than that for S^* and (2) the anticipated overflow for S^* should be at most one more than that for S. In fact, we shall establish this intuition formally, although the proof is not so easy as one might initially think.

Given a BFA process with starting state S, define $P_m(S, k)$ to be the probability that there are k or more items in the overflow region immediately following the mth arrival to the system. The following result formalizes the intuition concerning the effects of slight perturbations in the starting configuration. The proof will be presented after some further discussion.

LEMMA 4.2. *For all k, $m > 0$ and S, S^* such that $S = \omega_1\, 0\, \omega_2$ and $S^* = \omega_1\, 1\, \omega_2$ for some ω_1 and ω_2,*

$$P_m(S, k-1) \geqslant P_m(S^*, k) \geqslant P_m(S, k).$$

The lemma states that, at least in terms of cumulative probability distributions, adding an item to the starting configuration can never decrease the number of anticipated overflow items. The lemma also states that the harm caused by adding an item to the initial state is limited to an increase of at most one in the anticipated number of overflow items.

A standard and attractive method of proving such results is to show that the same hypothesis holds even in the worst case. For example, given a collection of

items with specified arrival times, desired cell locations, and residence times, the task would be to show that inserting an extra item into the starting configuration cannot decrease the number of overflow items and can only increase that number by at most one. Unfortunately, neither result is true. Indeed, it is possible to construct simple examples where inserting an item into the initial configuration *decreases* the number of overflow items from $m/2-1$ to 0! Similarly, there are also examples where adding a single item at the start can increase the number of overflow items from 0 to $m/2-1$. (We leave the construction of these examples as an amusing exercise.) Hence, there are problem instances where slight perturbations in the initial state can have drastic and unexpected consequences.

In light of these examples, it is no longer "intuitively clear" that the lemma is true, and even less clear that it can be proved rigorously. For example, it would seem (at least initially) that any proof of Lemma 4.2 would have to argue that the problem instances with adverse behavior occur very rarely, and that they are more than compensated for by instances with the hoped-for behavior. Of course, this appears hard to do, since it is not at all clear how to characterize bad problem instances, and it is even more difficult to bound the probability of their occurrence.

Fortunately, we do not have to worry about these difficulties. In fact, the proof of Lemma 4.2 is quite simple, employing a long (but not widely) known technique for establishing the stochastic dominance of one random variable over another. The technique is surprisingly powerful; we use it to prove several results in this paper, and we expect it will prove useful in the stochastic analysis of other algorithms as well.

Proof of Lemma 4.2. Consider the sequence of events that can take place starting in $S^* = \omega_1\, 1\, \omega_2$ and leading up to and including the mth arrival. Partition the events into $2m$ blocks, m departure blocks alternating with m arrival blocks, starting with a departure block. Each arrival block consists of a single arrival event. Each departure block consists of some number (between 0 and the number of items in the system, inclusive) of departure events.

The main idea of the proof is to associate in a useful way the possible sequences of arrival and departure blocks starting in $S^* = \omega_1\, 1\, \omega_2$ with those starting in $S = \omega_1\, 0\, \omega_2$. For example, if $S = \omega_1\, 0\, \omega_2$ were changed to $\omega_1'\, 0\, \omega_2'$ as a result of a block of events, then the same events (with corresponding probabilities) would change $S^* = \omega_1\, 1\, \omega_2$ to $\omega_1'\, 1\, \omega_2'$. Moreover, if $\omega_1\, 0\, \omega_2$ were changed to $\omega_1\, 1\, \omega_2$ by an arrival, then the same arrival would change $\omega_1\, 1\, \omega_2$ to $\omega_1\, 1\, \omega_2^*$, where $\omega_2 = \gamma_1\, 0\, \gamma_2$ and $\omega_2^* = \gamma_1\, 1\, \gamma_2$. Lastly, if $\omega_1\, 1\, \omega_2$ changes to $\omega_1\, 0\, \omega_2$, then nothing would happen in the system starting with $\omega_1\, 0\, \omega_2$, and the two systems would become identical. Hence, a sequence of events starting in S^* will be associated with a sequence of events starting in S, so that the corresponding probabilities are the same and so that the result of the process starting in S^* is identical to or contains one more item than the result obtained by starting in S. Once this is done, it will be straightforward to prove first that $P_m(S^*, k) \geqslant P_m(S, k)$, since the set of sample functions leading to k or more overflow items starting in S will be dominated

by the set leading to k or more overflow items starting in S^*; and second that $P_m(S, k-1) \geq P_m(S^*, k)$, since the set of sample functions leading to k or more overflow items starting in S^* will be dominated by the set leading to $k-1$ or more overflow items starting in S.

We now proceed formally to construct the association of probability spaces for the systems starting in S and S^*. We begin by defining the space for the system starting in S^* in terms of a tree Γ^* with depth $2m$. The tree will be finite since the number of items in the system is bounded during the given time period and hence the number of all possible sequences leading up to the mth arrival is also bounded.

Each *node* of Γ^* is associated with a *state* of Γ^*, which is a configuration of items in cells along with a tag that specifies which cell (if any) is *special.* (The special cell will correspond in the obvious way to "the cell with the extra item.") It will be obvious that many nodes of Γ^* can be associated with the same state.

The state of the root of Γ^* is $S^* = \omega_1\, 1\, \omega_2$, where the cell between ω_1 and ω_2 is designated as the special cell. The nodes/states at the first level correspond to all configurations that can be reached by deleting some, none, or all of the items in S^*. These configurations are divided into two types: those for which the item in the special cell was deleted and those for which it was not. If the item in the special cell is deleted, then the corresponding states and all their descendant states will not have a special cell. Otherwise, the designation of the special cell remains unchanged. In either case, each first level edge is assigned a weight equal to the probability that the corresponding events are precisely those that occur before the first arival in the BFA process starting in S^*. Figure 1 illustrates the structure of Γ^* for $N=4$.

Transitions at the second level correspond to arrivals. Since there are N different types of arrivals, each first-level node has N descendant second-level nodes. The state of a second-level node is determined according to the BFA rule based on its predecessor state and the desired cell of the arriving item. The designation of the special cell is changed only if the inserted item would have landed in the special cell were it empty. In this case, the designation of the special cell is changed to become the cell which actually receives the inserted item. As before, each edge is weighted with the corresponding event probability, in this case $1/N$.

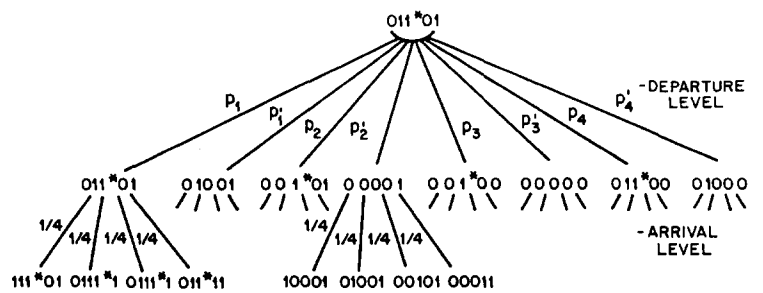

FIG. 1. Illustration of Γ^* for $N=4$, $S^* = 011*01$.

The construction of Γ^* proceeds as above for a total of $2m$ levels. The leaves of Γ^* represent the possible configurations of the BFA process after m arrivals, starting in S^*. The probability of a particular final configuration is simply the sum of the path weights of the leaves of Γ^* corresponding to that configuration (ignoring the designation of special cells). The *path weight* of a leaf is the product of the weights of the edges from the root to the leaf and is equal to the total probability of the sequences corresponding to the path (these sequences for a given path differ only by permutations of the events within departure blocks).

A key observation at this point is that by a simple change to the states of Γ^* we can represent the probability space of events for the system starting in S. In particular, let Γ be the tree that is identical to Γ^* in every way except that nodes associated with states having a special cell are associated instead with the states obtained by removing the item from the special cell. (In Fig. 1 the starred 1's are all replaced by starred 0's.) In what follows we will verify that Γ provides a correct characterization of the probability space of events for BFA starting in S, viz., that the probability of a particular final configuration is simply the sum of the path weights of the leaves of Γ corresponding to that configuration.

Consider a node of Γ associated with configuration v and let v^* denote the configuration associated with the corresponding node in Γ^*. If v^* has no special cell, then by definition $v = v^*$ and the descendants of v are identical to those of v^*, which is precisely as it should be. If $v^* = \beta_1\, 1\, \beta_2$, where the special cell is between β_1 and β_2, then $v = \beta_1\, 0\, \beta_2$. In this case, we must consider arrival and departure events separately. In the case of an arrival it is easily checked that the construction ensures that the descendants of v and v^* (and associated edge weights) match up one-to-one in the proper way. (Of course, the rule for special cell designation was set up for precisely this purpose.)

In the case of a departure block, every set of events from v is represented by *two* descendant edges whose probabilities sum to the probability that the set of events occurs starting in v. The first of these twin edges corresponds to the same set of events starting from v^* in Γ^*, while the second corresponds to this set plus the event that the extra item departs in Γ^*. Since the item departures are independent, we know that this sum provides the correct value for the corresponding event block starting at v.

We note that, in generating a set of sequences differing only in the order of events within departure blocks, we proceed down Γ^* along a unique path. However, in generating such a set according to Γ from the "smaller" initial state S, we may proceed down many paths in general; in each state with a special cell we effectively toss a coin to determine which of two twin edges is to be followed to the next level. The weight of the coin is determined by the probability that in the corresponding state of Γ^* the item in the special cell departs. As noted, Γ accurately describes the BFA process starting in S, because the total probability in leaves associated with the same state has the desired value for each possible final configuration.

It is now a simple matter to complete the proof of the lemma. By definition, $P_m(S^*, k)$ is the sum of the path weights of the leaves of Γ^* associated with con-

figurations having k or more items in the overflow region. Similar interpretations are possible for $P_m(S, k-1)$ and $P_m(S, k)$ with respect to leaves of Γ. Since every leaf of Γ with k or more items in the overflow region corresponds to a leaf of Γ^* with k or more items in the overflow region, we can conlude that $P_m(S^*, k) \geq P_m(S, k)$. By a similar argument we conclude that $P_m(S, k-1) \geq P_m(S^*, k)$, thus completing the proof of the lemma. ∎

In fact, much stronger versions of Lemma 4.2 can be proved. For example, if S and S^* are two configurations such that S^* *dominates* S (i.e., the first i cells of S^* contain at least as many items as the first i cells of S, for all $i \geq 1$), then $P_m(S^*, k) \geq P_m(S, k)$ for all k and m. Proofs of such results are easy generalizations of the technique formulated in the proof of Lemma 4.2. In this particular case, the jth item, counting from left to right, in the system starting in S is associated with the jth item in the system starting in S^*. The remaining rightmost items in the system starting in S^* are treated as items in special cells, much like the extra item in the proof of Lemma 4.2. The remainder of the proof is identical, once it is observed that insertion and deletion preserve the dominance relationship under BFA.

The proof that $P_m(S^*, k) \geq P_m(S, k)$ whenever S^* dominates S can also be used to show that BFA is an optimal on-line allocation policy given any fixed partition of storage into cells of sizes $1/N^*$, $2/N^*$, ..., N^*/N^*, 1, 1, ..., where at most one item can be placed in each cell. The association of items in the two systems proceeds as before. To complete the proof, one needs only observe that insertion of an item in a cell larger than the smallest, sufficiently large available cell preserves the dominance relation. As a special case, this means that BFA (when modified as described in Section 4.5) and FF are optimal when all items have the same size. These results are summarized in the following theorem and corollary.

THEOREM 4.1. *Given a uniform distribution of item sizes, $F(x)$, any N^* and a fixed partition of storage into cells of sizes $1/N^*$, $2/N^*$, ..., N^*/N^*, 1, 1, ..., where at most one item can be placed in a cell, BFA is an optimal allocation policy in the sense that any other policy is at least as likely as BFA to waste w or more space at time t for all w and t.*

COROLLARY 4.1. *If all items are restricted to have the same size, then BFA (as defined for nonuniform distributions in Section 4.5) and FF are optimal in the sense that any other allocation policy is at least as likely to waste w or more space at time t for all w and t.*

4.3. *The Correspondence between BFA and Maximum Upright Matching*

First, we set the stage for the proof of the $\Theta(\sqrt{N} \log^{3/4} N)$ result for the expected wasted space under BFA. For a given initial state, sample functions of the process $\{S_N(t)\}$ can be represented as shown in Fig. 2. In this representation a plus in column i at time $t > 0$ denotes an arrival at time t of an item of size x corresponding to cell i, i.e., $x \in [(i-1)/N, i/N]$. A minus at such a time and column denotes a

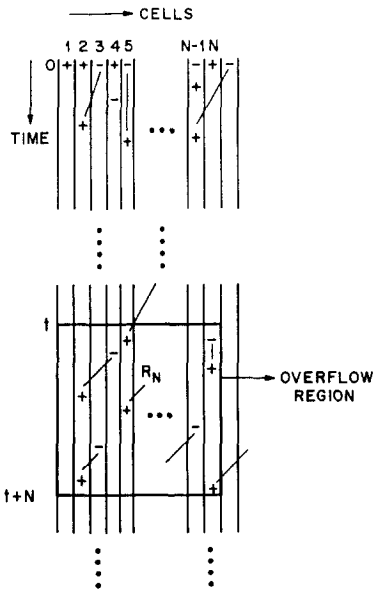

FIG. 2. Sample function for $S_N(t)$.

departure from cell i at time t. At $t=0$, pluses and minuses have a special meaning: each cell has either a plus or a minus at $t=0$ to indicate that it is occupied or unoccupied, respectively, in the initial state.

An edge is drawn between each plus and the lowest minus (above the plus) in the column representing the cell in which the item corresponding to the plus is placed according to BFA. It is readily observed that these edges form a matching, and that every plus is matched to a minus that is above and rightward of the plus. Such a matching is called an *upright matching* [KLM].

The matching of pluses and minuses gives us an explicit representation of the occupied cells in $S_N(t)$. See Fig. 2 for an illustration. Note particularly, that all the pluses unmatched with minuses in the first N columns correspond to arrivals that had to be placed into cells of the overflow region. A quantity of interest then is the number of unmatched pluses in the first N columns of the first T rows, say. In particular, we will be interested in the value $T=N$ since the "half-life" of an item is roughly equal to the time required for N events to transpire.

Let R_N denote the 2-dimensional region restricted to the time interval $[t, t+N]$ for some $t>0$ and cells $\{1, 2, ..., N\}$. It is known [KLM] that if we scan the arrivals (pluses) of R_N in increasing order of arrival time and match each plus with the leftmost unmatched minus to the right and above, then we obtain a maximum matching in R_N. This establishes a correspondence between maximum upright matchings and the matchings of Fig. 2, since BFA matches each plus to the leftmost unmatched minus to the right and above.

Analyzing the expected number of unmatched pluses in a maximum matching of a random upright matching problem has been the subject of much recent work [KLM, LS, Sh]. Leighton and Shor [LS] proved recently that if N points are uniformly selected from an $N \times N$ grid and uniformly labeled as plus or minus (our definition of a random upright matching problem), then the expected number of unmatched pluses in a maximum upright matching is $\Theta(\sqrt{N} \log^{3/4} N)$. In fact, they proved the following stronger result.

LEMMA 4.3 [LS, Sh]. *The number of unmatched points in a maximum matching for a random upright matching problem is* $\Theta(\sqrt{N} \log^{3/4} N)$ *with very high probability.*

Unfortunately, the configuration of pluses and minuses in Fig. 2 differs substantially from that in a random upright matching problem. For example, the (departure) rate at which minuses appear in the time (vertical) dimension is $n_N(t)$, which is different from the fixed (arrival) rate at which pluses appear. Moreover, the distribution of minuses in the horizontal dimension depends greatly on the current state of the system, since no column can contain two consecutive unmatched minuses (no departure can occur from an unoccupied cell).

However, we know from our earlier limit theorems that the relative difference between $n_N(t)$ and N tends to 0 as $N \to \infty$. Moreover, we expect the number of unoccupied cells to be asymptotically negligible when compared to the number of occupied cells. Thus, it is reasonable to hope that the difference between the marginal distributions and their corresponding uniform distributions becomes negligible in an appropriate sense as $N \to \infty$; i.e., that the actual system fails to properly account for unmatched pluses by an expected amount that is at most $\Theta(\sqrt{N} \log^{3/4} N)$. Showing that this is indeed the case is the crux of the proof.

We conclude this section with a trivial upright matching result that we will find useful in the next section. Note the difference in complexity between this result and the corresponding stochastic result proved in Lemma 4.2.

LEMMA 4.4. *In any instance of the upright matching problem suppose the sign or the location of exactly one point is changed. Then the number of unmatched pluses in a maximum upright matching increases by at most* 1.

Proof. Trivial. ∎

4.4. *The Storage Efficiency of BFA*

We are now ready for the next major result, a probabilistic bound on the position of the rightmost item under BFA. We commence with the result for uniform $F(x)$. Nonuniform distributions are considered in Section 4.5 along with extensions of the results to certain variations of BFA.

THEOREM 4.2. *There exists a positive constant* α *such that for all* $t \geq 0$, $r_N(t) \leq N + O(\sqrt{N} \log^{3/4} N)$ *with probability* $1 - O(N^{-\alpha\sqrt{\log N}})$, *and for all* $t \geq \frac{1}{2} \log N$ *the bound is tight, i.e., for all* $t \geq \frac{1}{2} \log N$, $r_N(t) = N + \Theta(\sqrt{N} \log^{3/4} N)$ *with probability* $1 - O(N^{-\alpha\sqrt{\log N}})$. *Moreover,* $Er_N(t) = N + \Theta(\sqrt{N} \log^{3/4} N)$ *for all* $t \geq \frac{1}{2} \log N$.

Proof. For convenience in our association of the BFA process with upright matching, we patition time into consecutive intervals of length $\frac{1}{2}$. In each of these *half-intervals* the expected number of events in statistical equilibrium is N. The theorem will be proved by showing first that during almost all half-intervals, $\Theta(\sqrt{N}\log^{3/4} N)$ items are inserted into overflow cells. Since the expected residence time of any item is 1, it will then be a relatively easy matter to complete the proof by analyzing the arrival/departure process restricted to the overflow region. The details of the analysis are divided into three parts, one for the upper bound, one for the lower bound, and one for the expected value. We commence with the upper bound.

Part I. *Upper Bound.*

In order to analyze the number of items that are likely to be inserted into overflow cells during a half-interval, we must first know which cells are occupied before the first event occurs. In what follows, we claim that without loss of generality, we can assume that precisely the first $N+\sqrt{N}\log^{3/4} N$ cells are initially occupied. This is argued in two steps. First, we argue that precisely the first $\sqrt{N}\log^{3/4} N$ cells in the overflow region can be considered to be occupied without adversely affecting the overflow analysis. That is because the departure and arrival of items in the initial N cells is unrelated to the departure of items in the overflow region. It will be seen that the chief reason for starting with $\sqrt{N}\log^{3/4} N$ overflow items is that throughout the subsequent half-interval we will then have $n_N(t) \geqslant N$ with very high probability.

The second step of the argument follows from Lemma 4.2, i.e., that initially adding items to cells can never decrease the probability that k or more items overflow in any interval of time for any k. Hence, we may as well assume that the first N cells are initially occupied, and then that precisely the first $N+\sqrt{N}\log^{3/4} N$ cells are initially occupied overall.

As pointed out in Section 2 we can apply (2.2) to the number of events in the subsequent half-interval and conclude that there are at most $N+O(\sqrt{N}\log^{3/4} N)$ such events with very high probability. For simplicity we shall look only at the first N events in such very high probability sets. Since the last $O(\sqrt{N}\log^{3/4} N)$ events can only add $O(\sqrt{N}\log^{3/4} N)$ items to the overflow region, the simplification will have no significant impact on the proof.

Now we begin the heart of the argument, relating N events of the Markov BFA process to the random upright matching problem defined in section 4.3. In order to establish the correspondence, we will show how to convert any instance of the upright matching problem into an instance of the BFA problem. The conversion will have two essential features. First, it will transform the probability space associated with the random upright matching problem into the probability space associated with the BFA problem. Second, for all but $O(N^{-\alpha\sqrt{\log N}})$ instances of the upright matching problem, the number of items inserted into overflow cells of the corresponding BFA problem will be at most $O(\sqrt{N}\log^{3/4} N)$ plus a constant times the number of unmatched pluses in the instance of the upright matching problem.

Since we know that all but $O(N^{-\alpha\sqrt{\log N}})$ instances of the random upright matching problem have at most $O(\sqrt{N}\log^{3/4} N)$ unmatched pluses [LS], we will thus be able to conclude that at most $O(\sqrt{N}\log^{3/4} N)$ items overflow in any half-interval with very high probability.

The transformation will operate only on sequences of events; i.e., the interevent times of a Poisson process will be suppressed. In these terms the random upright matching problem can be described by random variables $X_1, ..., X_N$ and $Y_1, ..., Y_N$, where X_i denotes the sign (plus or minus) of the ith event and $1 \leqslant Y_i \leqslant N$ denotes its column. All of the variables are independent and $\Pr\{X_i = +\} = \Pr\{X_i = -\} = \frac{1}{2}$ for all i and $\Pr\{Y_i = j\} = 1/N$ for all $1 \leqslant i, j \leqslant N$.

The first N events of the BFA problem starting with the first $N + \sqrt{N}\log^{3/4} N$ cells occupied can be described by random variables U_i and V_i, denoting the sign and column, respectively, of the ith event. However, as noted in Section 4.3, these variables are not independent. In fact, $\Pr\{U_i = +\} = N/(N + m_i)$, where m_i is the number of items in the system before the ith event. The dependence of U_i on $U_1, ..., U_{i-1}$ arises because m_i is determined by the number of $U_1, ..., U_{i-1}$ which are pluses. Even worse, if U_i is minus, the corresponding V_i is chosen uniformly over the cells which contain an item. Of course, the domain of these cells depends greatly on what values were assigned to $U_1, ..., U_{i-1}$ and $V_1, ..., V_{i-1}$.

Despite these problems, we shall see that it is possible to convert a random instance of the upright matching problem into a random instance of the BFA problem in such a way that the number of overflow items in the latter instance differs by at most a constant factor from the number of unmatched pluses in the former instance. The precise transformation is described by the following procedure.

The events (X_i, Y_i) are scanned in order of increasing index. Let m_i denote the number of items in the system before the ith event, and let $m_i' \leqslant m_i$ denote the number of items in the first N cells before the ith event. For $i = 1, 2, ..., N$ we perform the following steps:

(1) If $m_i < N$, then abort this procedure and assign all subsequent values of (U_j, V_j), $j \geqslant i$ according to the generation procedure of the BFA process. Otherwise, continue to step 2.

(2) If X_i is minus then set U_i to minus.

(3) If X_i is plus then set U_i to plus with probability $1 - \varepsilon_i$ and to minus with probability ε_i, where

$$\varepsilon_i = \frac{m_i - N}{m_i + N}.$$

(4) If U_i was set to plus, then set V_i equal to Y_i.

(5) If U_i was set to minus and setting $V_i = Y_i$ would be an illegal assignment (i.e., if it would mean emptying an already empty cell), then uniformly select one of the $m_i - m_i'$ nonempty overflow cells and set V_i to index that cell.

(6) If U_i is minus and setting $V_i = Y_i$ is a legal assignment over BFA, then set V_i equal to Y_i with probability $1 - \delta_i$ and to any of the $m_i - m_i'$ nonempty overflow cells (selected uniformly) with probability δ_i, where

$$\delta_i = \frac{m_i - N}{m_i}.$$

We must first verify that the transformation produces a random instance of the BFA problem. In particular, we must first check that $U_1, ..., U_N$ have the right distribution of signs. This is confirmed by observing that, by step 3,

$$\Pr\{U_i = +\} = \Pr\{X_i = +\}\left(1 - \frac{m_i - N}{m_i + N}\right) = \frac{1}{2} \cdot \frac{2N}{m_i + N} = \frac{N}{m_i + N},$$

which is the correct value. Note that the condition $m_i \geqslant N$ assures that the probabilities are well defined at this point. Moreover, were the condition ever violated, the procedure would (by definition) properly complete the assignment according to the BFA generation process.

Next, let us check the distribution of $V_1, ..., V_N$. If U_i is plus, then setting $V_i = Y_i$ means that V_i is chosen uniformly from 1, 2, ..., N, which is the correct distribution. If U_i is minus, then V_i should be assigned uniformly over the occupied cells. To verify this, it is sufficient to check that V_i is assigned to the overflow region with probability $(m_i - m_i')/m_i$. This is done by the transformation (provided $m_i \geqslant N$), since by steps 5 and 6,

$$\begin{aligned}
&\Pr\{V_i \text{ is assigned to the overflow region}\} \\
&\quad = \Pr\{V_i = Y_i \text{ would be illegal}\} \\
&\qquad + \Pr\{V_i = Y_i \text{ would be legal}\} \cdot \frac{m_i - N}{m_i} \\
&\quad = \frac{N - m_i'}{N} + \frac{m_i'}{N}\frac{(m_i - N)}{m_i} \\
&\quad = \frac{m_i - m_i'}{m_i},
\end{aligned}$$

as desired. Hence, $V_1, ..., V_N$ have the correct distribution and we have shown that the procedure output is a random BFA instance given a random instance of upright matching as input.

We next show that with very high probability, the number of overflow items for the output BFA problem is not much more than the number of unmatched pluses for the input upright matching problem. We first note that $m_i \geqslant N$ for all $1 \leqslant i \leqslant N$ with very high probability. This is because (2.7) and $m_0 = N + \sqrt{N}\log^{3/4} N$ ensure that $m_i > N$ with very high probability for any particular i. Thus, $m_i > N$ with very

high probability for every i, $1 \leqslant i \leqslant N$, and we can assume that the abort condition in step 1 does not arise.

Provided the abort condition does not arise, the BFA process provides a maximum upright matching for the corresponding random upright matching problem as modified in steps 3, 5, and 6 of the transformation procedure. Hence, the number of items which remain in the overflow region after the Nth event is at most

$$X_{\text{init}} + X_+ - X_-,$$

where $X_{\text{init}} = \sqrt{N} \log^{3/4} N$ is the number of pluses initially in the overflow region, X_+ is the number of unmatched pluses in the modified upright matching problem and X_- is the number of minuses inserted into the overflow region during steps 5 and 6 of the transformation procedure. In what follows, we will use Lemmas 4.3 and 4.4 to show that this quantity is at most $O(\sqrt{N} \log^{3/4} N)$ with very high probability.

By (2.6) we know that $m_i - N = O(\sqrt{N} \log^{3/4} N)$, $1 \leqslant i \leqslant N$, with very high probability. Hence, although the ε_i are not independent, they are uniformly bounded by the relation

$$\varepsilon_i = \frac{m_i - N}{m_i + N} = O\left(\frac{\log^{3/4} N}{\sqrt{N}}\right), \qquad 1 \leqslant i \leqslant N,$$

with very high probability. Applying (2.2) once more, we can thus conclude that X_i and U_i differ for at most $O(\sqrt{N} \log^{3/4} N)$ values of i, $1 \leqslant i \leqslant N$, with very high probability. Hence, with very high probability the modifications to the input upright matching problem consist of changing $O(\sqrt{N} \log^{3/4} N)$ signs in step 3, and moving some minuses into the overflow region in steps 5 and 6. By Lemma 4.4 the changes in step 3 can increase X_+ by at most $O(\sqrt{N} \log^{3/4} N)$. The shift of a minus to the overflow region, on the other hand, cannot increase $X_+ - X_-$ at all, since by Lemma 4.4 such shifts can increase X_+ by at most 1. Therefore, by Lemma 4.3 we can conclude that

$$X_{\text{init}} + X_+ - X_- = O(\sqrt{N} \log^{3/4} N)$$

with very high probability.

As a direct consequence of the preceding analysis we know that the number of newly inserted items (i.e., inserted in the last N events) which remain in the overflow region after the Nth event is at most $O(\sqrt{N} \log^{3/4} N)$ with very high probability. With very high probability this number remaining will differ by at most a constant factor from the total number of items inserted in the overflow region in the last N events. Thus, we conclude that with very high probability, the number of newly inserted items in the overflow region during a half-interval is $O(\sqrt{N} \log^{3/4} N)$.

As in the upper bound proof of Theorem 3.2, we now restrict ourselves to sufficiently large t, where the bound on $r_N(t)$ is tight. Because of the initial state $n_N(0)=0$, the arguments simplify for small t and are easily worked out.

By applying the previous result to $\log N+\log^2 N$ consecutive half-intervals, we can bound the anticipated overflow at any point in time t. As we shall verify below, the first $\log N$ of these intervals are needed to assure that the overflow region has at most $O(\sqrt{N}\log^{3/4} N)$ items, and the last $\log^2 N$ are needed to assure that the rightmost of these is in a cell with an index at most $N+O(\sqrt{N}\log^{3/4} N)$.

In particular, let n_i denote the number of overflow items at time $t-(\log N+\log^2 N-i)/2$, $0\leqslant i\leqslant \log N+\log^2 N$. By (2.3) applied to the process $\{n_N(t)\}$ for t large, the number of items in the system at time $t-(\log N+\log^2 N)/2$ is at most $2N$ with very high probability. By the preceding analysis, the number of new overflow items entered by the end of any of the $\log N+\log^2 N$ half-intervals is at most $O(\sqrt{N}\log^{3/4} N)$ per interval with very high probability. Now the probability that any overflow item departs in a half-interval is $1-e^{-1/2}$. We conclude that in the first $\log N$ half-intervals

$$n_i\leqslant \zeta n_{i-1}+O(\sqrt{N}\log^{3/4} N),\qquad 0\leqslant i\leqslant \log N,$$

with very high probability for any constant $e^{-1/2}<\zeta<1$. Hence, with very high probability the maximum number of overflow items at any time between $t-(\log^2 N)/2$ and t is at most

$$O(\sqrt{N}\log^{3/4} N)(1+\zeta+\zeta^2+\cdots+\zeta^{\log N-1})+2N\zeta^{\log N}=O(\sqrt{N}\log^{3/4} N).$$

Finally, by looking at the last $\log^2 N$ half-intervals we can bound the index of the rightmost occupied cell. Since the maximum number of overflow items in the system at times $t-(\log^2 N/2)$ through t is at most $O(\sqrt{N}\log^{3/4} N)$ with very high probability, it is easy to see that the rightmost cell occupied by any item inserted during this interval is at most $N+O(\sqrt{N}\log^{3/4} N)$. Hence, we need only verify the claim that all of the items in the system at time $t-(\log^2 N)/2$ (some of which could conceivably be in bad positions) are gone by time t with very high probability. This is easy since the probability that all items have departed in time $\log^2 N/2$ is at least

$$[1-e^{-\log^2 N/2}]^{2N}=[1-N^{-\log N/2}]^{2N}=1-O(N^{-\alpha\sqrt{\log N}})$$

for all constant $\alpha>0$. Hence, with very high probability the index, r_N, of the rightmost occupied cell is $N+O(\sqrt{N}\log^{3/4} N)$.

Part II. *Lower Bound.*

We next show that for any time $t\geqslant \frac{1}{2}\log N$, i.e., when the behavior of $\{n_N(t)\}$ is almost stationary, the rightmost item in the BFA process is in cell $N+\Omega(\sqrt{N}\log^{3/4} N)$ with very high probability. The proof is very similar to the corresponding upper bound proof just given, so we will only sketch the proof of the lower bound.

Specifically, we shall prove that with very high probability we have $r_N(t) \geqslant N + c\sqrt{N}\log^{3/4} N$ for any $t \geqslant \frac{1}{2}\log N$, and all sufficiently large N, where $c > 0$ is a sufficiently small constant to be determined. We start by considering the state of the system at the beginning of the interval $[t-\frac{1}{2}, t]$. If for a given $c > 0$ there are $2c\sqrt{N}\log^{3/4} N$ or more items in the overflow region at time $t-\frac{1}{2}$, then there will be at least $c\sqrt{N}\log^{3/4} N < 2e^{-1/2}c\sqrt{N}\log^{3/4} N$ items at time t with very high probability. Thus, we need only consider cases when there are fewer than $2c\sqrt{N}\log^{3/4} N$ items in the overflow region at time $t-\frac{1}{2}$.

By (2.4) we know that with very high probability there are at least $N - c\sqrt{N}\log^{3/4} N$ items in the system overall at time $t-\frac{1}{2}$. Hence, we know that (with very high probability) at least $N - 3c\sqrt{N}\log^{3/4} N$ of the initial N cells are occupied at time $t-\frac{1}{2}$. We can now use Lemma 4.2 to conclude that the probability of having k or more items in the overflow region at time t is at least as large as the probability of having $k + 4c\sqrt{N}\log^{3/4} N$ items in the overflow region at time t starting with precisely the first $N + c\sqrt{N}\log^{3/4} N$ cells occupied. By choosing $k = c\sqrt{N}\log^{3/4} N$, it remains only to show that with very high probability, there will be $5c\sqrt{N}\log^{3/4} N$ items in the overflow region at the end of a half-interval beginning with the first $N + c\sqrt{N}\log^{3/4} N$ cells occupied.

To prove the latter result, we use the same transformation procedure as in the proof of the upper bound. The analysis is also similar, except that we have to be careful with the constant factors. With very high probability, we can restrict our attention to the first N events, introducing a potential change of $2c\sqrt{N}\log^{3/4} N$ in the number of overflow items. As before, $m_i > N$ for all i, $1 \leqslant i \leqslant N$, with very high probability. Using (2.6) and (2.2) as in the proof of the upper bound, we can conclude that with very high probability at most $2c\sqrt{N}\log^{3/4} N$ signs are changed during the transformation procedure. This introduces another potential change of $2c\sqrt{N}\log^{3/4} N$ in the number of overflow items. Moving minuses into the overflow region can only increase the number of items that overflow during the interval so we can neglect their effect.

By Lemma 4.3 we know there exists a constant $c > 0$ such that for all N sufficiently large, $18c\sqrt{N}\log^{3/4} N$ items overflow during this interval with very high probability. Since $\frac{1}{2}$ of these items are still present at time t with very high probability, we obtain the desired result.

Part III. *Expected Value.*

To show that $Er_N(t) = N + O(\sqrt{N}\log^{3/4} N)$, it suffices to analyze the tail probabilities when $r_N(t)$ is very large; e.g., $r_N(t) \geqslant 3N$, for the upper bound proved in Part I easily implies that

$$\begin{aligned} Er_N(t) &= \sum_{k \geqslant 0} \Pr\{r_N(t) \geqslant k\} \\ &\leqslant N + O(\sqrt{N}\log^{3/4} N) + \sum_{k \geqslant 3N} \Pr\{r_N(t) \geqslant k\}. \end{aligned} \tag{4.1}$$

For simplicity we again partition time into intervals, in this case of size $1/N$. Note that during such an interval the number in system changes by at most $O(1)$ with probability approaching 1. With no essential loss in generality, we assume that $t = j/N$ is a multiple of $1/N$.

Now $r_N(t) = l$ implies that there must have been some interval $[(j-i-1)/N, (j-i)/N]$, $0 \leqslant i < j$, such that an item arrived, was placed in position l and remained in the system for at least i/N units. For large l, the placement into position l is very unlikely, since such a placement implies that there are at least $l-N$ items in the system (the lower bound is achieved for some $l \geqslant N$ only if the first $N-1$ cells are empty and the arriving item has a size in $[(N-1)/N, 1]$). By the normal limit law of Section 2, the probability that there were $l-N$ or more items in the system at any time in $[(j-i-1)/N, (j-i)/N]$ is at most $O(e^{-(l-2N)^2/(2N)})$. Therefore, since the probability that an item arrives in a given interval and stays for at least i/N time units is $[1-O(1/N)]\, e^{-i/N}$, we have an overall bound on the probability of this event occurring of $O(e^{-i/N} \cdot e^{-(l-2N)^2/(2N)})$. Summing these probabilities over i and $l \geqslant k$ gives an upper bound on the probability that the rightmost occupied position is at least k at time t. Thus, since

$$\sum_{i=0}^{j-1} e^{-i/N} \sum_{l \geqslant k} e^{-(l-2N)^2/(2N)} = O(N) \cdot O(Ne^{-(k-2N)^2/(2N)}),$$

we have

$$\Pr\{r_N(t) \geqslant k\} = O(N^2 e^{-(k-2N)^2/(2N)}).$$

Hence, the sum in (4.1) is clearly $O(1)$ and the upper bound on $Er_N(t)$ follows.

Finally, $Er_N(t) = N + \Omega(\sqrt{N} \log^{3/4} N)$ follows immediately from the lower bound argument in Part II, so we have proved $Er_N(t) = N + \Theta(\sqrt{N} \log^{3/4} N)$. ∎

It is now clear that, in conjunction with the argument of Lemma 4.1, we have

COROLLARY 4.2. *With probability* $1 - O(N^{-\alpha\sqrt{\log N}})$ *for some* $\alpha > 0$ *we have*

$$w_N = \Theta(\sqrt{N} \log^{3/4} N)$$

and

$$h_N = \Theta(\sqrt{N} \log^{3/4} N).$$

By adjusting the constant factor in the $O(\sqrt{N} \log^{3/4} N)$ term of the upper bound, it is possible to make α arbitrarily large (e.g., α could be set to 1). We do not know whether a similar result is true for the lower bound. In any case, the general form of the bound is correct, since with probability $O(N^{-\alpha\sqrt{\log N}})$, the claimed bounds are provably incorrect for α that grow with N.

4.5. *Extensions of BFA*

BFA and its analysis can be easily extended to handle nonuniform distributions. Let $F(x)$ be an arbitrary distribution on $[0, 1]$ for which we can partition $[0, 1]$ into intervals $[m_0, m_1]$, $[m_1, m_2]$, ..., $[m_{N-1}, m_N]$, where $0 = m_0 \leqslant m_1 \leqslant \cdots \leqslant m_N = 1$ and

$$\int_{m_{i-1}}^{m_i} dF(x) = \frac{1}{N}, \qquad 1 \leqslant i \leqslant N.$$

In the corresponding BFA rule, the first N cells have sizes m_i $(1 \leqslant i \leqslant N)$, while all remaining cells have size 1.

The upper bounds in Theorems 4.1 and 4.2 and Corollaries 4.1 and 4.2 carry over to this general version of BFA. The lower bounds carry over as well, provided $F(x)$ is a standard distribution and $0 = m_1 < m_2 < \cdots < m_N = 1$.

As noted in Section 3, the size of the partition of storage can be taken as a design parameter. Returning to the uniform distribution, for example, storage can be subdivided into cells with lengths $1/N^*$, $2/N^*$, ..., $(N^* - 1)/N^*$, 1, 1, ... Otherwise, the new rule, BFA*, operates just as BFA. Note that if $N^* > N$ we can expect the average number, $E(h_{N^*})$, of interior unoccupied cells to increase but their sizes to become stochastically smaller. The object, of course, is to select an N^* that makes the second effect more pronounced, so that expected wasted space actually decreases. In the next section we will see that an optimization over $N^* > N$ does in fact reduce the expected wasted space, at least experimentally.

For $N^* - N = O(\sqrt{N} \log^{3/4} N)$ (which our optimization in the next section satisfies) it is easy to see that our earlier bounds still apply. However, Lemma 4.1 relating $E(w_n)$ and $E(h_n)$ no longer applies when $N^* > N$. On the other hand, for small $N^* - N$ the inequality $r_{N^*}(t) \geqslant N^*$ still holds with high probability, so that by the argument in Lemma 4.1 we have

$$E(w_{N^*}) \sim \sum_{i=1}^{N^*} i/N^* + E(r_{N^*} - N^*) - \frac{N}{2}$$

and

$$E(w_{N^*}) \sim E(h_{N^*}) - \frac{N^* - N}{2} + \frac{1}{2}.$$

For a substantial improvement, therefore, we need $E(h_{N^*})$ to increase by substantially less than $(N^* - N)/2$.

Another way to implement a finer subdivision is simply to scatter extra $O(\sqrt{N} \log^{3/4} N)$ cells uniformly over the storage area, while retaining the same subdivision of item sizes; i.e., if an item has a size in $[(i-1)/N, i/N]$, $i \geqslant 1$, then it is placed into a smallest unoccupied cell with size at least i/N. We can obtain the same trade-offs as before and the relation between $E(w_n)$ and $E(h_N)$ can be extended to this case. However, our bounds do not cover this variation of BFA. Nor does our

analysis cover the extension of BFA whereby two or more items are allowed in the same cell, so long as their cumulative size does not exceed the cell's size. We leave the design and analysis questions concerning these latter two variations as problems for future research. The major experimental evaluation effort that would be entailed falls outside the scope of this paper.

5. Experimental Results

For a number of algorithms, simulations were designed for the purpose of getting some idea of asymptotic behavior within our mathematical model. (Other such studies of storage allocation, much less oriented to asymptotic behavior, can be found in the work of Page [Pa] and Shore [Sho].) Our simulations were quite informal and, because of costs, only a small number of experiments were run. Thus, the results can only be taken as suggestive of general behavior. On the other hand, we shall mention instances of corroboration with independent simulations and known results which will enhance the credibility of conjectures based on these data.

We shall be investigating five algorithms: FF, BFA, BFA*, BF, and FFRS (first-fit, randomized search). The first four have already been defined. The fifth is the same as FF, except that the linear scan for the first adequate hole begins differently. When storing a new item under the FFRS rule, a hole in storage is first picked at random. The scan for a sufficiently large hole is started at this point and continued toward the highest addressed interior hole in storage. At that point the search, if it must continue, wraps around and resumes with the lowest addressed hole in storage.

Intuitively, under FF smaller holes tend to accumulate near the base of storage, where the hole searches begin. Thus, the average number of holes tested in a linear-scan hole search will be much larger than $N/4$, where $N/2$ is the expected number of holes. (Indeed, as we shall see, experiments suggest that the average is asymptotically $N/2$.) In randomizing the starting point of the hole search, FFRS produces a hole size distribution which is nearly independent of the starting address; thus, the expected number of tests in the hole search should be reduced. While this improvement has been borne out by simulation studies, we shall see that it is accompanied by a major sacrifice in storage utilization (see also [Pa, Sho]).

To ensure fast algorithms for simulation purposes, hole searches were implemented with the help of binary search trees. For FF and FFRS the vertices (holes) were ordered by the starting address, and for BF they were ordered by size. With standard tree balancing techniques, $\Theta(\log N)$ expected search times are easily implemented in both cases, although for our purposes tree balancing was not thought to be worth the additional mechanism required.

Figure 3 summarizes the results on expected wasted space for the BF, FF, and FFRS algorithms and uniform item size distributions. The curves for BFA and BFA*, were they shown, would be extremely close to the BF curve. Therefore, we have provided a separate, more detailed comparison of these three algorithms,

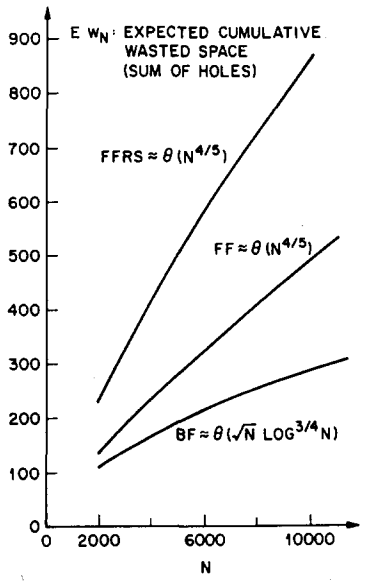

FIG. 3. Comparison of algorithms.

which we will get to shortly. In the meantime, our comparisons of FF and FFRS to BF also apply to a comparison of FF and FFRS to BFA and BFA*.

To facilitate visual comparisons, the curves in Fig. 3 are all straight plots. However, log–log plots were also used to help estimate the asymptotic behavior labeling the curves.

For each algorithm there were 5 iterations of the simulation. Data for $N = 2000i$, $1 < i < 5$, were collected for each algorithm and averaged over the 5 runs. As might be expected, FF and BF exhibited the slowest convergence; $1500N$ and $2000N$ events (arivals and departures) were simulated in each run for FF and BF, respectively. Only $1000N$ events appeared necessary for FFRS.

The estimate $\approx\Theta(N^{4/5})$ shown for FF matches that given in [CKS2] for simulations based on an implementation, compiler, and machine which differed in each case from our own.

Intuitively, FF can be expected to produce more small, hard-to-use holes than FFRS, but it should compensate by producing a higher percentage of large, easy-to-use holes. A similar comparison can be expected between FF and BF, with BF having the larger variation. The bottom line of these trade-offs between FFRS and FF and between FF and BF is remarkably in favor of FF in the first case and BF in the second. While the curves of expected wasted space under FFRS and FF appear to have the same growth rate (as judged from log–log plots), the latter shows a storage efficiency that is about 50% better. Among the three algorithms, BF is decidedly the best with data to be presented below suggesting that expected wasted space is $\Theta(\sqrt{N}\log^{3/4} N)$.

TABLE I

Comparison of BF and BFA*

Range average N	$0.62\sqrt{N}\log^{3/4} N$	BFA	N^*	BFA*	BF
2000	127	130	2120	116–118 117	109–117 112
4000	192	188	4175	171–174 172	159–179 169
6000	243	243	6210	211–215 214	202–226 216
8000	288	288	8260	248–252 250	246–263 254
10,000	328	329	10,290	273–279 277	270–326 286

Note. $2N \times 10^3$ event simulations for BF; $N \times 10^3$ event simulations for BFA*.

Table I reports the data from BFA* experiments for two choices of N^*. The first was simply $N^* = N$, corresponding to the basic BFA algorithm. The second was obtained by means of a somewhat coarse optimization over N^*. The performance of BFA* was tested for several values of N^* close to the one chosen in Table I. The results indicated that as N^* was increased from N the expected wasted space decreased monotonically to a point near the chosen value of N^*, and increased monotonically thereafter. As an illustration of this property, the entire function is estimated in Table II for $N = 8000$.

The simulation of BFA* was simplified according to our earlier observations in Lemma 4.1 and Section 4.5; from a simulation of the process $\{S_N(t)\}$ the data shown in Table I are estimates of

$$E(w_{N^*}) \sim E(h_{N^*}) - \frac{N^* - N}{2} + \frac{1}{2}.$$

TABLE II

Optimization of N^*

$N^* - N$:	0	25	50	100	150	200	250	300	350	400
Ew_{N^*}:	288	281	272	263	254	254	253	258	263	280

Note. $N = 8000$.

The BFA (i.e., $N^* = N$) data were in fact obtained prior to the proposed and subsequently proved Theorem 4.2. From the averages we concluded originally that expected wasted space was asymptotically $\Omega(\sqrt{N}\sqrt{\log N})$ and $O(\sqrt{N}\log N)$. Table III shows how closely the $N = N^*$ data fit $0.62\sqrt{N}\log^{3/4} N$; as can be seen the fit is fairly close for $N < 6000$ and almost exact for $N \geqslant 6000$.

Table I also gives the BF data (the averages in Table T are plotted in Fig. 3). As can be seen, although BF improves measurably over BFA ($N^* = N$), it is quite close in performance to BFA* for the values of $N^* > N$ shown. Indeed, for large N the comparison favors BFA* slightly, although the nature of the data makes this a risky conclusion; the size of the simulation needs to be increased to cope better with the variation in the BF data. On the other hand, it seems safe to say that expected storage utilization is very unlikely to be an important issue in the choice between BF and BFA*.

There are two independent criteria where BFA (or BFA*) is plainly more desirable than BF. The first, which is demonstrable analytically, is the speed of the algorithm. While an implementation of BF must maintain and search $N/2$ holes on the average, BFA deals with only $\Theta(\sqrt{N}\log^{3/4} N)$ holes (unocuppied cells) on the average. Theorem 4.2 and the simulations of BFA also provided data on expected search lengths, i.e., the expected number of occupied cells that were encountered in linear scans for the unoccupied cells where new arrivals were stored. The numbers in Table III are averages over 5 simulations and fit rather closely the function $0.3\sqrt{N}\log^{3/4} N$. These data show that a simple, linear-scan implementation of BFA may be quite feasible in circumstances where a linear scan with BF would be far too time-consuming. But even if both algorithms used a balanced-tree search, the search times under BFA would be approximately $\log E(h_N) \sim \frac{1}{2}\log N$, which is one half the expected search time under BF. Also, the search tree under BFA would be smaller than that under BF by a factor of about $\sqrt{N}$.

The second major advantage of BFA is suggested by the observed smaller excursions of $r_{N^*}(t)$. This is indicated in part by the ranges of the data shown in Table I. When a fixed capacity for storage must be selected, a significantly smaller value may be chosen for BFA* than for BF, assuming the same confidence level for exceeding the capacity within a given time interval. For a concrete example of this effect we examined the total storage occupancy after $k \times 10^6$ events, for each $k = 11, 12, ..., 20$, in the output of the BF and BFA* simulations for $N = 4, 6, 8, 10$ ($\times 10^3$). We imagined the storage capacity to be set at the maximum point ever

TABLE III

BFA Search Times

N	2000	4000	6000	8000	10,000
$0.3\sqrt{N}\log^{3/4} N$	61	93	118	159	176
Data	61	92	116	158	175

reached by BFA* and determined for each N how many of the 10 BF samples exceeded this value; i.e., how many times did BF overflow a smallest storage that was sufficient for BFA*. The result was 3 out of the 10 times for $N = 4, 6$ ($\times 10^3$) and 5 out of the 10 times for $N = 8, 10, 10$ ($\times 10^3$).

6. Final Remarks

Proving that there is a constant c such that the rightmost item in BFA occupies a location below $N/2 + c\sqrt{N}\log^{3/4} N$ with very high probability at any point in time has powerful implications for practical dynamic allocation. Among other things, the result means that a finite disk with capacity $N/2 + c\sqrt{N}\log^{3/4} N$ is (with very high probability) sufficient to store items from the uniform $[0, 1]$ distribution (or any other distribution, provided that the cell boundaries are set appropriately) for any polynomial amount of time without ever wasting more than $O(\log^{3/4} N/\sqrt{N}) = o(1)$ of the disk, and without ever being unable to legally insert an item! This is a very powerful statement, particularly considering that allocation schemes commonly used in practice tend to waste $\Theta(1)$ of the disk (i.e., linear space) and to spend a constant fraction of their time reallocating files for compaction purposes.

Of course, the key question left open by this paper concerns the average wasted space of BF for distributions such as the uniform on $[0, 1]$. For this distribution, the experimental data strongly suggest that BF wastes $\Theta(\sqrt{N}\log^{3/4} N)$ space on average. It would be especially nice if these bounds could be proved for arbitrary distributions, since then there would be a distribution-independent algorithm that is provably good. (Recall that the cell boundaries in BFA are set in accordance with the distribution of item sizes.)

Acknowledgments

We have benefited from the help of many colleagues. We would particularly like to thank J. A. Reeds, M. I. Reiman, L. A. Shepp, P. Shor, and A. Weiss for many useful discussions.

References

[Al] D. Aldous, Some interesting processes arising as heavy traffic limits in an $M/M/\infty$ storage process, *Stoch. Proc. Appl.* **22** (1986), 291–313.

[BCW] B. S. Baker, E. G. Coffman, Jr., and D. E. Willard, Algorithms for resolving conflicts in dynamic storage allocation, *J. Assoc. Comput. Mach.* **32** (1985), 327–343.

[Be1] V. E. Benês, Models and problems of dynamic memory allocation, *in* "Applied Probability and Computer Science—The Interface, Vol. I" (R. L. Disney and T. J. Ott, Eds.), Birkhäuser, Cambridge, MA, 1982.

[Be2] V. E. Benês, Optimal memory allocation: Problems, principles, and an example, unpublished manuscript, AT & T Bell Laboratories, Murray Hill, NJ, 1974.

[C1] H. CHERNOFF, A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations, *Ann. Math. Statist.* **23** (1952), 493–507.

[C2] E. G. COFFMAN, JR., An introduction to combinatorial models of dynamic storage allocation, *SIAM Rev.* **25** (1983), 311–325.

[CKS1] E. G. COFFMAN, JR., T. T. KADOTA, AND L. A. SHEPP, A stochastic model of fragmentation in dynamic storage allocation, *SIAM J. Comput.* **14** (1985), 416–425.

[CKS2] E. G. COFFMAN, JR., T. T. KADOTA, AND L. A. SHEPP, On the asymptotic optimality of first-fit storage allocation, *IEEE Trans. Software Engrg.* **SE-11** (1985), 235–239.

[CLR] E. G. COFFMAN, JR., G. S. LUEKER, AND A. H. G. RINNOOY KAN, An introduction to the probabilistic analysis of sequencing and packing algorithms, *Management Sci.*, in press.

[CrL] H. CRAMÈR AND M. R. LEADBETTER, "Stationary and Related Stochastic Processes," p. 272, Wiley, New York, 1967.

[Fe] W. FELLER, "An Introduction to Probability Theory and Its Applications," Vol. I, 3rd ed., Vol. II, 2nd ed., 1971, Wiley, New York, 1968, 1971.

[HPS] P. G. HOEL, S. C. PORT, AND C. J. STONE, "Introduction to Stochastic Processes," Houghton–Mifflin, Boston, 1972.

[Ig1] D. L. IGLEHART, Limit diffusion approximations for the many server queue and the repairman problem, *J. Appl. Probab.* **2** (1965), 429–441.

[Ig2] D. L. IGLEHART, Weak convergence of compound stochastic processes, *Stochastic Process. Appl.* **1** (1973), 11–31.

[KaSZ] T. T. KADOTA, L. A. SHEPP, AND J. ZIV, Asymptotic efficiency of dynamic storage systems, Bell Laboratories, Murray Hill, NJ, submitted for publication, 1983.

[KLM] R. M. KARP, M. LUBY, AND A. MARCHETTI-SPACCAMELA, Probabilistic analysis of multidimensional bin-packing problems, *in* "Proceedings, 16th ACM Symp. on Theory of Computing, 1984," pp. 289–298.

[Kn] D. E. KUTH, "Fundamental Algorithms," Vol. 1, 2nd ed., Section 2.5, p. 435ff, Addison–Wesley, Reading, MA, 1973.

[Ko] L. KOSTEN, Uber Sperrungswahrscheinlichkeiten bei Staffelschaltungen, *Elec. Nachr. Tech.* **14** (1937), 5–12.

[LS] F. T. LEIGHTON AND P. SHOR, Tight bounds for minimax grid matching, with applications to the average-case analysis of algorithms, *in* "Proceedings, 18th Symp. on Theory of Computing, May 1986."

[Ne] G. F. NEWEL, "The $M/M/\infty$ *Service System with Ranked Servers in Heavy Traffic*," Springer-Verlag, New York, 1984.

[Pa] I. P. PAGE, Optimal fit of arbitrary sized segments, *Comput. J.* **25** (1982), 32–33.

[Ro] J. M. ROBSON, Worst-case fragmentation for first-fit and best-fit storage allocation strategies, *Comput. J.* **20** (1977), 242–244.

[Sh] P. SHOR, "Random Planar Matching and Bin-Packing," Ph. D. thesis, Department of Mathematics, Massachusetts Institute of Technology, Cambridge, MA, 1985; The average-case analysis of some on-line algorithms for bin-packing, *in* "Proceedings, 15th Conf. on Foundations of Computer Science, 1984," pp. 193–200.

[Sho] J. E. SHORE, On the external storage fragmentation produced by first fit and best fit allocation policies, *Comm. ACM* **18** (1975), 433–440.

[St] T. A. STANDISH, "Data Structure Techniques," Section 6.2, p. 249ff, Addison-Wesley, Reading, MA, 1980.