Probabilistic Analysis of Packing and Related Partitioning Problems

E. G. Coffman, Jr., D. S. Johnson, and P. W. Shor
AT&T Bell Laboratories
Murray Hill, NJ 07974
G. S. Lueker
Dept. of Information and Computer Science
University of California
Irvine, CA 92717

July 19, 1999

Abstract

In the last 10 years there have been major advances in the average-case analysis of bin-packing, scheduling, and similar partitioning problems in one and two dimensions. These problems are drawn from important applications throughout industry, often under the name of stock cutting. This chapter briefly surveys many of the basic results, as well as the probabilistic methods used to obtain them. The impact of the research discussed here has been two-fold. First, analysis has shown that heuristic solutions often perform extremely well on average, and hence can be recommended in practice, even though worst-case behavior can be quite poor. Second, the techniques in applied probability that have developed for the analysis of bin packing have found application in completely different arenas, e.g., statistics and stochastic models.

1 Introduction

Problems. The problems studied here all involve the partitioning of a set of positive numbers into a collection of subsets satisfying a sum constraint. The following two problems are among the most fundamental. They have wide-ranging applications throughout computer science and operations research [C3],[C4],[D1].

> *Bin Packing* (BP). *Given $c > 0$ and a set $S = \{X_1, \ldots, X_n\}$ with $0 < X_i \leq c$, $1 \leq i \leq n$, partition S into a minimum number of subsets such that the sum of the X_i's in each subset is no more than c.*

The X_i's are usually called items or pieces and are thought of as being packed into bins $B_1, B_2, \ldots$, each with capacity c; the items packed in a bin comprise one of the subsets in a solution to the optimization problem.

> *Multiprocessor Scheduling* (MS). *Given an integer $m \geq 1$ and a set $S = \{X_1, \ldots, X_n\}$, partition S into m subsets such that among all such partitions, the maximum subset sum is minimized.*

Note that the MS problem is complementary to the BP problem in that the objective function and the given parameter are interchanged. The items are now called tasks or jobs, with running times or durations instead of sizes. The bins become processors $P_1, \ldots, P_m$, and the partition becomes a schedule of S on m processors that minimizes the *makespan* c, i.e., the completion time of a latest finishing task. Because of the sequential nature of most heuristics, it is convenient to assume that the set to be partitioned is given as a list $L_n = (X_1, \ldots, X_n)$ from which items are packed or scheduled one by one. If H denotes an MS heuristic, then $H(L_n, m)$ denotes the makespan of the m-processor schedule generated by H for the tasks in L_n. In the BP problem, the bin capacity is only a scale factor, so we take $c = 1$ without loss of generality. Thus, if H denotes a BP heuristic, then $H(L_n)$ denotes the number of unit capacity bins in which H packs the items of L_n.

Merely deciding whether a list of numbers can be partitioned into two subsets with equal sums is NP-complete, so as one would expect, both the BP and MS problems are NP-complete. Thus, one is unlikely to find an algorithm that will solve these problems exactly and efficiently.[1] For this reason, a large literature has built up over the past 20 years on the design and analysis of heuristic or approximation algorithms. Such algorithms are designed to generate optimal or nearly optimal solutions for most problem instances. Quantifying this last statement is the goal of analysis.

Analysis. Early research on BP, MS, and related problems concentrated on combinatorial, worst-case results, as reflected in the survey by Coffman, Garey and Johnson (1984). For example, a scheduling heuristic H would be assessed by determining for each m a least upper bound over all L_n and n on the ratio $H(L_n, m)/OPT(L_n, m)$, where OPT stands for an optimal algorithm, i.e., $OPT(L_n, m)$ denotes the makespan of a solution to the MS problem for the problem instance (L_n, m). Similarly, the ratios $H(L_n)/OPT(L_n)$ were bounded for BP heuristics H. Such results are inherently pessimistic, so probability models were introduced in order to learn more about the probable or average-case behavior of heuristics.

Probabilistic analysis began about 10 years ago, and gained considerable momentum when some striking new results were developed a few years later. In the standard probability model, the X_i's are taken as independent samples of a random variable X with a given distribution $F(x)$. The goal is then an estimate of distributions such as $P\{H(L_n) \leq x\}$, or what is sometimes easier to obtain, expected values such as $E[H(L_n, m)]$, where the expectations are over all n-item samples $L_n = (X_1, \ldots, X_n)$.

Typically, exact analysis of probability models is quite difficult, especially for the more efficient algorithms, so asymptotic techniques have been used. These techniques estimate behavior for large problem instances, i.e. for large n. Also, the estimates often take the form of expressions with terms that are precise only within unspecified multiplicative constants. For example, let $F(x)$ be the uniform distribution on $[0, 1]$. Then as illustrated later, there are BP heuristics H for which it has been proved that $E[H(L_n)] = n/2 + \Theta(\sqrt{n})$. Here, the $\Theta(\cdot)$ notation is simply a relaxation of the concept "is proportional to." Precisely, $f(n) = \Theta(g(n))$ means that there exist constants $\alpha, \beta > 0$ such that for all n large enough,

$$\alpha g(n) \leq f(n) \leq \beta g(n) .$$

If we only know the existence of $\beta > 0$ such that the right-hand inequality is satisfied for all n large enough, then we write the familiar $f(n) = O(g(n))$. A similar restriction to α and the left-hand inequality is denoted $f(n) = \Omega(g(n))$.

We emphasize that usually very little is known about the multiplicative constants hidden in the $\Theta(\cdot)$ terms. One can almost always find some bounds for these constants, but in most cases there is reason to believe that the bounds are very crude.

[1] For a comprehensive treatment of NP-completeness and its implications, see the text by Garey and Johnson [G1].

In the remainder of this section we present a number of fundamental algorithms together with a sampling of results that measure the quality of the packings or schedules produced.

BP algorithms. We begin by describing three algorithms that pack the items in the sequence $X_1, \ldots, X_n$. An item is packed when it is encountered; once packed, it is not moved thereafter. The algorithms are said to be *on-line* because, for each i, $1 \le i \le n$, the rule that decides where X_i is packed is independent of the number and sizes of the remaining items $X_{i+1}, \ldots, X_n$. All three algorithms begin by packing X_1 into B_1.

The simplest of the three algorithms is Next Fit, abbreviated NF. In packing X_i, $i \ge 2$, NF first checks the highest indexed, nonempty bin, say B_j, $j \ge 1$. X_i is packed in B_j if it fits, i.e., if X_i plus the sum of the items already packed in B_j is at most 1. Otherwise, X_i is packed into B_{j+1}, which then becomes the new highest-indexed, nonempty bin.

The two algorithms, First Fit (FF) and Best Fit (BF), improve on NF by checking *all* nonempty bins before starting a new bin, i.e., FF and BF pack an item X_i into an empty bin if and only if X_i does not fit into any nonempty bin. FF packs X_i, $i \ge 2$, into the lowest indexed, nonempty bin, if any, in which X_i fits, while BF packs X_i into a nonempty bin, if any, in which X_i fits best, i.e., with the least unused capacity left over. Ties are resolved by BF in favor of lower indexed bins.

Improved, off-line versions of these algorithms are obtained by first sorting the X_i's into decreasing order; the corresponding NFD, FFD, and BFD algorithms (D stands for decreasing) are simply NF, FF, and BF applied to the list $(X_{(n)}, \ldots, X_{(1)})$, where $X_{(i)}$ denotes the i^{th} smallest item in L_n.

The sidebar summarizes a number of the basic results that have been derived for the above algorithms under the assumption $X \sim U(0,1)$, i.e., $F(x)$ is the uniform distribution on $[0,1]$. The performance metric shown is the *expected wasted space*,

$$(1.1) \quad \begin{aligned} E[W^H(L_n)] &= E[H(L_n) - \sigma(L_n)] \\ &= E[H(L_n)] - \frac{n}{2}\ , \end{aligned}$$

where $\sigma(L_n)$ denotes the sum of the item sizes in L_n. Note that FF and BF and their counterparts FFD

> **Bin Packing (BP)**: $X \sim U(0,1)$
>
> **Next Fit**: $E[W^{NF}(L_n)] \sim \frac{n}{6}$ [C8]
>
> **First Fit**: $E[W^{FF}(L_n)] = \Theta(n^{2/3})$ [S1], [C1]
>
> **Best Fit**: $E[W^{BF}(L_n)] = \Theta(\sqrt{n}\log^{3/4} n)$ [S1]
>
> **Next Fit Decreasing**:
>
> $E[W^{NFD}(L_n)] = (.145\ldots)n$ [H3], [C10]
>
> **First Fit Decreasing (FFD), Best Fit Decreasing (BFD), and Optimal (OPT)**:
>
> $E[W^H(L_n)] = \Theta(\sqrt{n})$
>
> for $H = FFD$ [B2], BFD [B2], or OPT [L5]

and BFD are all asymptotically optimal in the sense that the ratio of expected wasted space $E[W_n^H]$ to the expected occupied space $E[\sigma(L_n)] = n/2$ tends to 0 as $n \to \infty$. This is in sharp contrast to worst-case bounds which show that, for infinitely many $n > 0$, $W^H(L_n)/\sigma(L_n)$ can be as large as $\frac{7}{10}$ for $H = FF$ or BF and as large as $\frac{2}{9}$ for $H = FFD$ or BFD [J1].

The shortcomings of the $\Theta(\cdot)$ results are apparent, since the average-case results do not distinguish between FFD, BFD, and OPT. On the other hand, the average-case results do show that for all n

sufficiently large, $E[FF(L_n)] > E[BF(L_n)]$; a comparable distinction does not appear in the worst-case results.

MS algorithms. We describe three algorithms. The simplest is the on-line List Scheduling (LS) algorithm, which schedules the tasks in the given sequence $X_1, \ldots, X_n$ on the processors $P_1, \ldots, P_m$ with X_1 starting on P_1. LS schedules X_i, $i \geq 2$, on that processor having a smallest workload in the schedule for $X_1, \ldots, X_{i-1}$, with ties broken in favor of lower indexed processors. By the workload of a processor, we mean the total duration of the tasks already scheduled on that processor. As before, LS can be improved by first sorting L_n into decreasing order. LS along with the initial sorting is called the Largest Processing Time (LPT) algorithm.

The third MS heuristic was originally proposed for a somewhat different optimization problem: With the instances (L_n, m) the same as in the MS problem, the objective of the *set-partitioning* (SP) problem is to find a schedule that minimizes the difference in the maximum and minimum processor workloads. Clearly, one expects a good heuristic for SP to be a good heuristic for MS; indeed, the two problems are obviously identical for $m = 2$. The heuristic described below is a *set-differencing method* [K2]. It can be extended to all $m \geq 2$, but we confine ourselves to the case $m = 2$, since it is easier to describe and analyze.

Two tasks X and Y in list L are said to be *differenced in* L when a new list L' is formed from L by replacing X and Y with a task having duration $|X - Y|$. The Largest-First Differencing (LFD) heuristic applied to $L_n = L_n^{(1)}$ for $m = 2$ begins by differencing the largest two tasks in $L_n^{(1)}$ to form $L_n^{(2)}$. Then the largest two tasks are differenced in $L_n^{(2)}$ to form $L_n^{(3)}$. This procedure continues until a list $L_n^{(n)}$ of one task remains. LFD defines a schedule for L_n by requiring that the tasks differenced in $L_n^{(i)}$, $0 \leq i \leq n-1$, be scheduled on different processors and by requiring that the final processor workloads differ by the duration of the task in $L_n^{(n)}$. This schedule is easily developed by working backward through the sequence of differencing operations. First, the task in $L_n^{(n)}$ is put on one or the other of the two processors. Suppose the schedule for $L_n^{(i)}$, $2 \leq i \leq n$, has been formed, and let X and Y be the tasks differenced in $L_n^{(i-1)}$. Then the schedule for $L_n^{(i-1)}$ is formed from the schedule for $L_n^{(i)}$ by removing a task of duration $|X - Y|$, then scheduling X and Y on different processors so as to preserve the processor workload difference, i.e., the duration of the task in $L_n^{(n)}$.

In analogy with (1.1), typical illustrations of probabilistic results can be found in the analysis of the processor idle time averaged over the m processors,

$$(1.2) \qquad A_{n,m}^H = \frac{mH(L_n, m) - \sigma(L_n)}{m} .$$

We assume $m = 2$, so that $A^H(L_n, 2)$ is simply half the difference between the two processor finishing times in the schedule produced by H. A satisfactory analysis of LFD remains an open problem, but Karmarkar and Karp [K2] have studied a randomized, more easily analyzed modification of LFD that we denote LFD*. Results for LS, LPT, and LFD* with $X \sim U(0,1)$ are shown in the sidebar. In the order given, the algorithms are increasingly more complicated and increasingly more difficult to analyze, but they yield schedules that are increasingly more efficient for large n.

Makespan Scheduling (MS)

List Scheduling:

$E[A^{LS}(L_n, 2)] = 1/6$ [F1]

Largest Processing Time:

$E[A^{LPT}(L_n, 2)] \leq \frac{e}{2(n+1)}$ [C2]

Modified Largest Difference First:

There exists a $c > 0$ such that, with a probability that tends to 1 as $n \to \infty$,

$A^{\mathrm{LFD}^*}(L_n, 2) = O(n^{-c \log n})$ [K2]

2 Analytical Techniques

We describe and illustrate below a number of the more important techniques that have been successfully applied to the analysis of BP and MS problems. A more extensive discussion appears in [C4].

2.1 Markov Chains

For the simpler BP and MS heuristics, it is sometimes possible to formulate a tractable Markov chain that represents the element-by-element development of partitions. A state of the Markov chain must represent block sums in a suitable way; given the state space, the transition function is defined by the heuristic. Results for general n are obtained by a transient analysis, while asymptotics for large n are obtained by a steady-state analysis.

To illustrate ideas, consider the average-case analysis of LS on $m = 2$ processors, and assume that $F(x)$ is the uniform distribution on $[0, 1]$. Define V_i as the (positive) difference between the processor finishing times after the first i tasks have been scheduled. The following recurrence is easily verified:

$$V_i = \begin{cases} |V_{i-1} - X_i|, & 1 \leq i \leq n, \\ 0, & i = 0 \ . \end{cases}$$

Since the X_i are i.i.d. random variables, $\{V_i\}_{i \geq 0}$ is a Markov chain. A routine analysis shows that the density for V_i is given by $f_i(x) = 2(1-x)$, for all $i > 2$ [F1]. Then we obtain the result cited in Section 1, viz., $E[A^{LS}(L_n, 2)] = E[V_n]/2 = 1/6$. Since $OPT(L_n, 2) \geq \sigma(L_n)/2$ and $LS(L_n, 2) = [V_n + \sigma(L_n)]/2$, we also have the relative performance bound

$$\frac{E[LS(L_n, 2)]}{E[OPT(L_n, 2)]} \leq 1 + \frac{E[V_n]}{E[\sigma(L_n)]} \ = \ 1 + \frac{2}{3n} \ .$$

As another example, $\{NF(L_n), l_n\}_{n \geq 1}$ is a bivariate Markov chain, where l_n is the level, i.e., sum of item sizes, in the last bin of an NF packing of L_n. An analysis of this chain for $X \sim U(0, 1)$ shows that $E[W^{NF}(L_n)] = \frac{n}{6} + 6$, $n \geq 2$ [H2], thus refining the asymptotic result cited in Section 1. Indeed, an explicit expression for the distribution of $NF(L_n)$ has been derived by Hofri [H2].

Unfortunately, Markov-chain approaches seem to be limited to the relatively simplistic, less efficient heuristics; the state spaces of Markov chains for other heuristics like FF and BF simply become too large and unwieldy.

2.2 Bounds

The immediate advantage of bounds is that they lead to a tractable analysis. The obvious sacrifice is that they provide only partial information. However, this information is often sufficient to choose between alternative heuristics. For example, the results cited in Section 1 for FF, BF, FFD, and BFD were all obtained by bounding techniques, yet they show that for all n sufficiently large, we have $E[FF(L_n)] > E[BF(L_n)] > E[H(L_n)]$, where

H stands for either FFD or BFD. As illustrated below, bounding techniques have appeared in two basic forms.

Bounding the Objective Function. In analyzing the BP heuristic H, it may be possible to find a function $g(L_n)$ such that $g(L_n) \geq H(L_n)$ for all L_n and such that $E[g(L_n)]$ is easily calculated. Then we have the average-case bound $E[H(L_n)] \leq E[g(L_n)]$. A similar approach applies to the analysis of MS heuristics.

As a concrete example, we consider the LPT heuristic and its average idle time, as defined by (1.2). We have [L4],[F2]

$$\begin{aligned} A^{LPT}(L_n, m) &\leq LPT(L_n, m) - \sigma(L_n)/m \\ &\leq \max_{1 \leq i \leq n} \left\{ X_{(i)} - \frac{1}{m} \sum_{k=1}^{i} X_{(k)} \right\} . \end{aligned} \tag{2.1}$$

To see the latter inequality, let i be the largest index such that $X_{(i)}$ runs until the end of the schedule. Then just after $X_{(i)}$ is scheduled the average processor idle time up to the end of the schedule is at most $(m-1)X_{(i)}/m \leq X_{(i)}$. Each task $X_{(k)}$ scheduled after $X_{(i)}$ reduces the average idle time by $X_{(k)}/m$; (2.1) follows easily.

To illustrate the use of the bound (2.1), we show that [F2]

$$A^{LPT}(L_n, m) \to 0 \text{ (a.s.)} \quad \text{as} \quad n \to \infty , \tag{2.2}$$

when $E[X] < \infty$ and $F(x)$ is strictly increasing in $(0, \delta)$ for some $\delta > 0$. Bounding the right-hand side of (2.1) by

$$\begin{aligned} &X_{(\lfloor \epsilon n \rfloor)} \quad + \quad \max \left\{ 0, X_{(n)} - \frac{1}{m} \sum_{k=1}^{\lfloor \epsilon n \rfloor} X_{(k)} \right\} , \\ &0 < F^{-1}(\epsilon) < \delta , \end{aligned} \tag{2.3}$$

we observe that the first term in (2.3) converges (a.s.) to $F^{-1}(\epsilon)$ as $n \to \infty$, and that it can be made arbitrarily small by an appropriate choice of ϵ. Also, since $E[X] < \infty$, $X_{(n)}/n \to 0$ (a.s.). Moreover, $\sum_{k=1}^{\lfloor n\epsilon \rfloor} X_{(k)}/n$ converges (a.s.) to a positive constant as $n \to \infty$ for every $\epsilon > 0$. Thus, the second term within the maximization in (2.3) tends to $-\infty$ (a.s.). We conclude that (2.2) holds.

In some cases, the requirement that a bound hold deterministically for all L_n is too stringent to yield good results. In addition to a bound $H(L_n) \leq g(L_n)$ that always holds, there may exist a sharper bound $g'(L_n)$ such that $H(L_n) \leq g'(L_n)$ except on a set having a small probability q_n. If $q_n \to 0$ sufficiently rapidly that $q_n E[g(L_n)] = o(E[g'(L_n)])$ as $n \to \infty$, then we have

$$\begin{aligned} E[H(L_n)] &\leq (1 - q_n)E[g'(L_n)] + q_n E[g(L_n)] \\ &\sim E[g'(L_n)] . \end{aligned}$$

Dominating Algorithms. A common way to upper-bound $H(L_n)$ is to introduce a simpler, more easily analyzed algorithm H' for which it can be proved that $H'(L_n) \geq H(L_n)$ for all L_n. In this case, H' is said to *dominate* H. A similar approach applies to lower bounds. For example, the MATCH heuristic described below is dominated by both FFD and BFD.

The MATCH packing heuristic iterates the following procedure until all items are packed. Let S denote the set of items that remain to be packed. MATCH first finds a largest item in S, say X. If $|S| = 1$ or if no remaining item fits with X, i.e., $Y + X > 1$ for all $Y \in S - \{X\}$, then MATCH puts X into a bin alone. Otherwise, MATCH puts items X and X' into a bin alone, where X' is a largest remaining item other than X such that $X + X' \leq 1$.

It can be proved without much difficulty that $FFD(L_n) \geq MATCH(L_n)$ and $BFD(L_n) \geq MATCH(L_n)$ for all L_n [L5]. Moreover, MATCH has the following simple analysis when $X \sim U(0,1)$. First, we have that $MATCH(L_n) \leq (n + b)/2$, where b is the number of singleton bins in the MATCH packing. The number of singletons with an item no larger than $1/2$ is at most one, so $MATCH(L_n) \leq (n + b' + 1)/2$ where b' is the number of singletons with an item larger than $1/2$. But an inspection of MATCH shows that b' is equal in distribution to $\max_{1 \le i \le n} \xi_i$ where ξ_i is a symmetric, n-step random walk starting at the origin. Standard results then yield $E[MATCH(L_n)] = n/2 + \Theta(\sqrt{n})$ and hence $E[H(L_n)] = n/2 + \Omega(\sqrt{n})$, where H stands for either FFD or BFD.

2.3 Stochastic Planar Matching

Matching problems in one or more dimensions have arisen in the analysis of several packing heuristics. An example in one dimension was given in Section 2.2. Here, we first define a generalization of this matching problem to two dimensions and then illustrate how it occurs in the analysis of algorithms.

Let n plus points and n minus points be chosen independently and uniformly at random in the unit square. Let M_n denote a maximum *up-right* matching of plus points to minus points such that if a plus at (x, y) is matched to a minus at (x', y'), then $x \le x'$ and $y \le y'$. Let U_n denote the number of points left unmatched by M_n. The problem of determining the distribution of U_n is called the up-right matching problem. Asymptotic bounds on the expected value are given by [L2],[R2],[S1],[C7]

$$E[U_n] = \Theta(\sqrt{n} \log^{3/4} n) . \tag{2.4}$$

To illustrate the applications of (2.4), we consider the upper bound analysis in [S1] of the BF heuristic, assuming that $X \sim U(0,1)$. Define the Modified Best Fit (MBF) heuristic to be the same as BF except that MBF closes a bin to any further items whenever the bin receives an item no larger than $1/2$. Clearly, bins in an MBF packing have at most two items. It is not difficult to prove that MBF dominates BF, so that $E[BF(L_n)] \le E[MBF(L_n)]$.

Next, we describe MBF as a matching procedure. Plot the items of L_n as points in the left half of the unit square so that X_i has a y coordinate $1 - i/n$ and an x coordinate X_i if $X_i \le 1/2$ and $1 - X_i$ if $1/2 < X_i \le 1$. X_i is plotted as a plus point if $X_i \le 1/2$ and as a minus point if $1/2 < X_i \le 1$. Now match a plus point with a minus point if the corresponding items are placed in the same bin by MBF. By definition of MBF, the minus point must be above the plus point, since the item corresponding to the minus point had to be scanned first. Also, the minus point must be to the right of the plus point, since the two items fit into a single bin. An MBF matching is a maximum up-right matching, as is easily verified. However, the model differs from the original one in two respects. First, points are samples in the left half of the unit square, and second, the x coordinate has been discretized so that $x \in \{0, 1/n, \ldots, (n-1)/n\}$. But it is easy to prove that (2.4) still holds; the effects of both differences are limited to changes in the hidden multiplicative constant.

Finally, we observe that $MBF(L_n)$ is the sum of the occupied space $\sigma(L_n)$ and the unoccupied space, the latter quantity being bounded by U_n. Thus, $E[MBF(L_n)] = n/2 + \Theta(\sqrt{n} \log^{3/4} n)$ and hence

$$E[BF(L_n)] = \frac{n}{2} + O(\sqrt{n} \log^{3/4} n) .$$

2.4 Linear Programming

If the item sizes in L_n comprise a discrete set, then BP is easily formulated as an integer program. Let $s_1, \ldots, s_N$ be the different item sizes in L_n and let m_j, $1 \le j \le N$, be the number of items with size s_j. Define the i^{th} possible *configuration* as a sequence of integers $C_{ij} \ge 0$, $1 \le j \le N$, such that $\sum_{j=1}^{N} C_{ij} s_j \le 1$, i.e., a set of items with C_{ij} of size s_j, $1 \le j \le N$, can be packed into a single bin. If M denotes the number of possible configurations, then $OPT(L_n) = \sum_{i=1}^{M} t_i^*$, where $\{t_i^*\}$ solves the integer program: minimize $\sum_{i=1}^{M} t_i$ subject to $t_i \ge$

0, $1 \le i \le M$, and $\sum_{i=1}^{M} t_i C_{ij} \ge m_j$, $1 \le j \le N$.

Relaxations of such integer programs lead to useful bounds for the analysis of optimum solutions. For example, suppose we relax the integer program for L_n so that the t_i can be arbitrary nonnegative reals. Then it is readily shown that

(2.5) $LIN(L_n) \le OPT(L_n) \le LIN(L_n) + N$

where $LIN(L_n)$ denotes a solution to the relaxed problem.

To illustrate the bound, consider the *packing constant* $c = \lim_{n\to\infty} E[OPT(L_n)]/n$. We will show that for general $F(x)$, $E[OPT(L_n)] - nc = O(\sqrt{n})$ [R4]. This will also give us one of the many applications of Kolmogorov-Smirnov statistics to the analysis of BP and MS.

To begin, for some integer $N \ge 1$ to be chosen later, transform the given distribution F to a distribution G consisting of N atoms, each of weight $1/N$, at $s_j = F^{-1}(j/N)$, $1 \le j \le N$. If c_N denotes the packing constant under G, then it is not hard to see that $c_N - 1/N \le c \le c_N$. Note that generating n items X_i according to G can be achieved by taking n uniform samples U_i from $[0,1]$ and setting $X_i = F^{-1}(\lceil NU_i \rceil/N)$, $1 \le i \le n$.

To investigate c_N, consider the one-sided Kolmogorov-Smirnov statistic D_n^-, where

$$\begin{aligned} nD_n^- &= \max_{0\le z\le 1} \{nz - |\{i : U_i \ge z\}|\} \\ &= \max_{0\le z\le 1} \{|\{i : U_i > z\}| - (1-z)n\} \ . \end{aligned}$$

If we remove from L_n the items generated by the largest nD_n^- of the U_i and pack them one per bin, we are left with a list L_n' of items X_i' with sizes in $\{s_j\}_{1\le j\le N}$ such that

$$|\{i : X_i' > s_j\}| \le n(1 - j/N), \ \ 1 \le j \le N \ .$$

We have $OPT(L_n') \le OPT(L_n'')$, where L_n'' contains exactly $\lceil n/N \rceil$ items of each size s_j. Finally, let $LIN(L_n'')$ denote the solution value of the LP relaxation for L_n'' in which we pack exactly n/N (rather than $\lceil n/N \rceil$) of each item size. By (2.5) we have $OPT(L_n'') \le LIN(L_n'') + N$, and by the law of large numbers, we have $LIN(L_n'') = Nc_N$. Hence we obtain the bound

$$OPT(L_n) \le nD_n^- + N + nc_N \le nD_n^- + N + n(c + 1/N) \, .$$

The standard bound, $E[nD_n^-] = O(\sqrt{n})$, along with the choice $N = \sqrt{n}$ then yields $E[OPT(L_n)] - nc = O(\sqrt{n})$, where the hidden constant is independent of the distribution.

Duality theory has played a role in studies of the *perfect packing* problem: For which distributions F do we have the packing constant $c(F) = E[X]$, so that $E[OPT(L_n)]/E[\sigma(L_n)] \to 1$ as $n \to \infty$? The dual of the integer program for BP is: Find a set of nonnegative weights u_j such that $\sum_{j=1}^{N} m_j u_j$ is maximized subject to $\sum_{j=1}^{N} u_j C_{ij} \le 1$, $1 \le i \le M$. Note that the constraint simply requires that for any set of items fitting into a bin, the corresponding sum of weights must be at most 1.

The perfect packing problem was first studied within the class of uniform distributions $U(a,b)$, $0 \le a \le b \le 1$. Motivated by the dual problem above, a weighting-function approach was adopted in [L6]. We say that $u : [0,1] \to [0,1]$ is a *weighting function* if for all finite sequences $x_1, \ldots, x_k$ of positive reals, $\sum_{i=1}^{k} x_i \le 1 \Rightarrow \sum_{i=1}^{k} u(x_i) \le 1$. It is easy to see that if u is a weighting function, then $c(F) \ge E[u(X)]$. Using this relation and a computer program to help in the search, Lueker found weighting functions classifying the distributions $U(a,b)$ such that $c(U(a,b)) = E[X] = \frac{a+b}{2}$. For the general problem, Rhee and Talagrand [R3],[R4] proved strong results drawing on ideas from functional analysis and topology. Courcoubetis and Weber [C9] studied the same problem within the class of on-line algorithms.

3 Related Topics

This section describes some of the more important questions that have grown out of the initial probabilistic studies of BP and MS.

3.1 Variants

The following problem has the same instance L_n as BP.

Bin Covering. Partition L_n into a maximum number of subsets (bins) such that each subset sums to at least 1.

The NF algorithm can be adapted to this problem in an obvious way. Applying standard results in renewal theory to the case $X \sim U(0,1)$, Csirik et al. [C11] analyzed this variant of NF bin-packing to obtain precise estimates of the expected number of bins covered.

The next problem has the same instance (L_n, m) as MS.

Dual Bin Packing. Find a subset $L'_n \subseteq L_n$ of maximum cardinality $C(L_n, m)$ such that L'_n can be partitioned into m subsets with each subset summing to no more than 1.

Asymptotics for $E[C(L_n, m)]$ have been studied in [B3].

3.2 Higher Dimensions

Extensions of BP to two and three dimensions have strong practical motivations, especially in stock-cutting applications. In the two-dimensional *strip packing* problem, the rectangles of a list $L_n = (R_1, \ldots, R_n)$ are to be packed into a unit-width, semi-infinite strip; the heights and widths of all rectangles are at most 1. The packing is to have the properties: (i) the rectangles do not overlap each other or the edges of the strip, (ii) rectangles are placed with their sides parallel to the edges of the strip (90° rotations are disallowed), and (iii) the packing height is minimized, where the packing height is the maximum height reached by the tops of the rectangles in a vertically oriented strip.

In the variant, *two-dimensional bin packing*, horizontal boundaries are also placed at the integer heights of the vertical strip. Each rectangle must now be wholly contained within a unit square, or "bin," between some pair of consecutive integer heights. The objective is now to minimize the number of bins used in the packing.

The probabilistic analysis of two-dimensional packing has recently been surveyed in [C5]. In the most common probability model, all rectangle heights and widths are taken to be independent samples from $U(0,1)$. The heuristics studied have, for the most part, been straightforward extensions of one-dimensional algorithms. For example, any one-dimensional heuristic can be adapted to *level packings* of the strip, in which rectangles are placed along levels, or horizontal baselines. The first level is the bottom of the strip. Each higher level passes through the top of a highest rectangle in the preceding level. Thus, the space between adjacent levels corresponds to a one-dimensional bin. *Shelf packings* [B1] are similar except that levels are preset at heights determined by the distribution of rectangle heights, which is assumed to be given in advance. In general, the probabilistic analysis of level and shelf algorithms extends in natural ways the analysis of one-dimensional bin-packing.

The two-dimensional bin packing algorithm in [K4] is a less obvious generalization of one-dimensional matching; its analysis reduces to that of up-right matching.

3.3 General Bounds

Lower bounds for BP have been useful in estimating the cost of certain restrictions to the design of algorithms. For example, assume $X \sim U(0,1)$.

Shor [S1] proved, as a result in stochastic planar matching, that $E[W^H(L_n)] = \Omega(\sqrt{n \log n})$ for any on-line algorithm. (More recently, Shor [S2] devised an on-line algorithm that achieves this lower bound without knowing n in advance.) It is interesting to note, however, that if we augment the class of on-line algorithms by those that can make use of the number n of items to be packed, then this bound no longer applies; Shor [S1] devised an algorithm of this type which wastes $\Theta(\sqrt{n})$ space on average.

In another example, again under $U(0,1)$, algorithms have been classified by the number of active bins needed during the packing process. Here, an active bin is simply a nonempty bin that has yet to be closed to further items. It has been shown [C6] that if an algorithm H never has more than r bins active at any time, then $E[W^H(L_n)] \geq \frac{n}{16(r+1)}$. Note that $r = 1$ for NF. This result also applies to all other on-line, linear-time algorithms studied in the literature [R1],[L1], i.e., all of these algorithms limit the number of active bins and produce a wasted space whose expected value grows linearly in n.

3.4 Distributions

In studies of BP the emphasis has been on the uniform distribution $U(0,1)$, because it leads to a tractable analysis. Models of MS have concentrated on $U(0,1)$ and exponential distributions, for the same reason. However, there have been many useful results dealing with more general distributions, as illustrated in Sections 2.2 and 2.4. For example, the lower bound on $E[OPT(L_n)]$ under $U(0,1)$ is easily shown to apply to any symmetric distribution on $[0,1]$. These results have also been proved for distributions with decreasing densities, by a technique of decomposing such distributions into a series of symmetric distributions.

In addition to the perfect packing results alluded to in Section 2.4, there have been several cases where the uniform distributions $U(0,b)$ have been successfully handled. Among these are Karmarkar's [K1] analysis of NF and the analysis by Bentley et al. [B2] of FFD. The analysis has often led to anomalous and unexpected results. For example, the analysis of FFD revealed discontinuities at $b = 1/2$ and $b = 1$, as shown by the fact that $E[W^{FFD}(L_n)]$ is $O(1)$ if $0 \leq b \leq 1/2$, $\Theta(n^{1/3})$ if $1/2 < b < 1$, and as noted earlier, $\Theta(\sqrt{n})$ if $b = 1$.

Quite recently, discrete uniform distributions have been studied in depth [C1]. The results have shown that many aspects of average-case behavior are lost in the passage to continuous approximations $U(a,b)$. To illustrate, let $U\{j,k\}$ denote the uniform distribution on the set $\{i/k\}_{1 \leq i \leq j}$, $1 \leq j \leq k$. Results for the continuous case do not suggest the following strong result: For any $U\{j,k\}$ with $j \leq k-2$, there is an *on-line* algorithm A such that $E[A(L_n) - \sigma(L_n)] = O(1)$. The behavior of classical packing rules has also been shown to be much more irregular in the discrete cases. For example, expected wasted space $O(1)$ and $\Theta(n)$ both occur under FFD for specific pairs j,k with $j/k < 1/2$ and with $j/k > 1/2$. As another example, there appear to be j,k such that FF produces $O(1)$ expected wasted space while FFD produces $\Theta(n)$ expected wasted space.

4 Directions for Further Study

There are obvious open problems concerned with more general distributions $F(x)$ and more precise results, e.g., useful bounds on the multiplicative constants hidden in the asymptotic notation. Here, we note a few gaps in the asymptotic theory that have yet to be resolved, even under the usual simplifying assumptions. We begin with two conjectures.

Conjecture 1 *For* $X \sim U(0,b)$, $0 < b < 1$, *we have* $E[W^H(L_n)] = \Theta(n)$ *for* $H = FF$ *or* BF.

A similar conjecture applies to discrete distributions $U\{j,k\}$ for $j \leq k-2$.

The next conjecture refers to the expected processor-idle-time defined in (1.2). We again assume $X \sim U(0,1)$.

Conjecture 2 *There exists an* $\alpha > 0$ *such that* $E[A^{OPT}(L_n,2)] = O(e^{-\alpha n})$.

Strong results on the median of the distribution of $A^{OPT}(L_n,2)$ are proved in [K3]. The analysis applies the second moment method [E1].

The conjectures for one-dimensional packing have their counterparts in higher dimensions. An interesting open problem in two dimensions is the average-case behavior of the FF and BF rules applied to the level algorithms of strip packing.

Industrial applications of three-dimensional packing abound, yet the design and probabilistic analysis of algorithms remains at an early stage [K4],[L3]. For example, the existence of on-line algorithms with sublinear expected wasted space remains an open question.

References

[B1] J. J. Bartholdi, J. H. Vande Vate, and J. Zhang. Expected performance of the shelf heuristic for two-dimensional packing. *Oper. Res. Lett.*, **8** (1989), 11–16.

[B2] J. L. Bentley, D. S. Johnson, F. T. Leighton, C. C. McGeoch, and L. A. McGeoch. Some unexpected expected behavior results for bin packing. In *Proc. Sixteenth Annual ACM Symposium on Theory of Computing*, Washington, DC, 1984, 279–288.

[B3] J. L. Bruno and P. J. Downey. Probabilistic bounds for dual bin packing. *Acta Informatica*, **22** (1985), 333–345.

[C1] E. G. Coffman, Jr., C. A. Courcoubetis, M. R. Garey, D. S. Johnson, L. A. McGeoch, P. W. Shor, R. R. Weber, and M. Yannakakis. "Fundamental discrepancies between average-case analyses under discrete and continuous distributions: A bin-packing case study," *Proc. 23rd ACM Symposium on the Theory of Computing*, 1991, 230–241.

[C2] E. G. Coffman, Jr., G. N. Frederickson, and G. S. Lueker. Expected makespans for largest-first scheduling of independent tasks on two processors. *Math OR*, **9** (1984), 260–266.

[C3] E. G. Coffman, Jr., M. R. Garey, and D. S. Johnson. Approximation algorithms for bin packing – An up-dated survey. In *Algorithm Design for Computer System Design*, G. Ausiello, M. Lucertini, and P. Serafini (eds.), Springer–Verlag, 1984, 49–106.

[C4] E. G. Coffman, Jr. and G. S. Lueker. *Probabilistic Analysis of Packing and Partitioning Algorithms*, Wiley & Sons, New York, 1991.

[C5] E. G. Coffman, Jr. and P. W. Shor. Average-case analysis of cutting and packing in two dimensions. *Eur. J. Oper. Res.*, **44** (1990), 134–144.

[C6] E. G. Coffman, Jr. and P. W. Shor. Packing in two dimensions: Asymptotic average-case analysis of algorithms. *Algorithmica* (to appear).

[C7] E. G. Coffman, Jr. and P. W. Shor. A simple proof of the $O(\sqrt{n}\ \log^{3/4} n)$ up-right matching bound. *SIAM J. Disc. Math.* (to appear).

[C8] E. G. Coffman, Jr., K. So, M. Hofri, and A. C. Yao. A stochastic model of bin-packing. *Information and Control*, **44** (1980), 105–115.

[C9] C. A. Courcoubetis and R. R. Weber. Necessary and sufficient conditions for the stability of a bin packing system. *J. Appl. Prob.*, **23** (1986), 989–999.

[C10] J. Csirik, J. B. G. Frenk, A. Frieze, G. Galambos, and A. H. G. Rinnooy Kan. A probabilistic analysis of the next fit decreasing bin packing heuristic. *Oper. Res. Lett.*, **5** (1986), 233–236.

[C11] J. Csirik, J. B. G. Frenk, G. Galambos, and A. H. G. Rinnooy Kan. Probabilistic analysis of algorithms for dual bin packing problems. Econometrics Institute, Erasmus University, The Netherlands.

[D1] H. Dyckhoff. A typology of cutting and packing problems. *Eur. J. Oper. Res.*, **44** (1990), 145–159.

[E1] P. Erdös and J. Spencer. *Probabilistic Methods in Combinatorics.* Academic Press, New York, 1974.

[F1] William Feller. *An Introduction to Probability Theory and Its Applications*, volume II. John Wiley and Sons, New York, second edition, 1971.

[F2] J. B. G. Frenk and A. H. G. Rinnooy Kan. The asymptotic optimality of the LPT rule. *Math. OR*, **12** (1987), 241–254.

[G1] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, New York, 1983.

[H1] M. Hofri. Two-dimensional packing: expected performance of simple level algorithms. *Inf. Cont.*, **45** (1980), 1–17.

[H2] M. Hofri. A probabilistic analysis of the next fit bin packing algorithm. *J. Algorithms*, **5** (1984), 547–556.

[H3] M. Hofri and S. Kamhi. A stochastic analysis of the NFD bin-packing algorithm. *J. Algorithms*, **7** (1986), 489–509.

[J1] D. S. Johnson, A. Demers, J. D. Ullman, M. R. Garey, and R. L. Graham. Worst-case performance bounds for simple one-dimensional packing algorithms. *SIAM J. Comput.*, **3** (1974), 299–325.

[K1] N. Karmarkar. Probabilistic analysis of some bin-packing algorithms. In *Proc. 23rd Annual Symposium on Foundations of Computer Science*, 1982, 107–111.

[K2] N. Karmarkar and R. M. Karp. The differencing method of set partitioning. Technical Report UCB/CSD 82/113, Computer Science Division (EECS), University of California, Berkeley, December 1982.

[K3] N. Karmarkar, R. M. Karp, G. S. Lueker, and A. M. Odlyzko. Probabilistic analysis of optimum partitioning. *J. Appl. Prob.*, **23** (1986), 626–645.

[K4] R. M. Karp, M. Luby, and A. Marchetti-Spaccamela. A probabilistic analysis of multidimensional bin packing problems. In *Proc. Sixteenth Annual ACM Symposium on Theory of Computing*, 1984, 289–298.

[L1] C. C. Lee and D. T. Lee. Robust on-line bin packing algorithms. Technical report, Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60201, 1987.

[L2] F. T. Leighton and P. W. Shor. Tight bounds for minimax grid matching with applications to the average case analysis of algorithms. In *18th Symposium on Theory of Computing*, 1986, 91–103.

[L3] K. Li and K.-H. Cheng. On three-dimensional packing. *SIAM J. Comput.*, **19** (1990), 847–865.

[L4] R. Loulou. Tight bounds and probabilistic analysis of two heuristics for parallel processor scheduling. *Math. OR*, **9** (1984), 142–150.

[L5] G. S. Lueker. An average-case analysis of bin packing with uniformly distributed item sizes. Technical Report 181, University of California Irvine, Department of Information and Computer Science, 1982.

[L6] G. S. Lueker. Bin packing with items uniformly distributed over intervals $[a, b]$. In *Proc. 24th Annual Symposium on Foundations of Computer Science*, 1983, 289–297.

[R1] P. Ramanan and K. Tsuga. Average-case analysis of the modified harmonic algorithm. *Algorithmica* (to appear).

[R2] W. T. Rhee and M. Talagrand. Exact bounds for the stochastic upward matching problem. *Trans. Amer. Math. Soc.*, **307** (1988), 109–125.

[R3] W. T. Rhee and M. Talagrand. Optimal bin packing with items of random sizes – II. *SIAM J. Comput.*, **18** (1989), 139–151.

[R4] W. T. Rhee and M. Talagrand. Optimal bin packing with items of random sizes – III. *SIAM J. Comput.*, **18** (1989), 473–486.

[S1] P. W. Shor. The average-case analysis of some on-line algorithms for bin packing. *Combinatorica*, **6** (1986), 179–200.

[S2] P. W. Shor. "How to Pack Better Than Best-Fit: Tight Bounds for Average-Case On-Line Bin Packing," *Proc. 32nd Annual Symposium on Foundations of Computer Science*, IEEE Computer Science Press, Los Alamitos, CA, 1991, 752–759.