

On the Sum-of-Squares Algorithm for Bin Packing

JANOS CSIRIK^{*} DAVID S. JOHNSON[†] CLAIRE KENYON[‡] JAMES B. ORLIN[§]
 PETER W. SHOR[¶] RICHARD R. WEBER^{||}

Abstract

In this paper we present a theoretical analysis of the deterministic on-line *Sum of Squares* algorithm (*SS*) for bin packing, introduced and studied experimentally in [8], along with several new variants. *SS* is applicable to any instance of bin packing in which the bin capacity B and item sizes $s(a)$ are integral (or can be scaled to be so), and runs in time $O(nB)$. It performs remarkably well from an average case point of view: For any discrete distribution in which the optimal expected waste is sublinear, *SS* also has sublinear expected waste. For any discrete distribution where the optimal expected waste is bounded, *SS* has expected waste at most $O(\log n)$. In addition, we present a randomized $O(nB \log B)$ -time on-line algorithm *SS**, based on *SS*, whose expected behavior is essentially optimal for all discrete distributions. Algorithm *SS** also depends on a new linear-programming-based pseudopolynomial-time algorithm for solving the NP-hard problem of determining, given a discrete distribution F , just what is the growth rate for the optimal expected waste. An off-line randomized variant *SS*** performs well in a worst-case sense: For any list L of integer-sized items to be packed into bins of a fixed size B , the expected number of bins used by *SS*** is at most $OPT(L) + \sqrt{OPT(L)}$.

^{*}csirik@inf.u-szeged.hu. Department of Computer Sciences, University of Szeged, Szeged, Hungary.

[†]dsj@research.att.com. AT&T Labs, Room C239, 180 Park Avenue, Florham Park, NJ 07932, USA.

[‡]Claire.Kenyon@lri.fr. Laboratoire de Recherche en Informatique, Bâtiment 490, Université Paris-Sud, 91405 Orsay Cedex, France.

[§]zorlin@mit.edu. Sloan School of Management, Massachusetts Institute of Technology, Cambridge, MA, 02139, USA

[¶]shor@research.att.com. AT&T Labs, Room C237, 180 Park Avenue, Florham Park, NJ 07932, USA.

^{||}rrw1@cam.ac.uk. Statistical Laboratory, University of Cambridge, Cambridge CB2 1SB, England.

1. Introduction

In the classical one-dimensional bin packing problem, we are given a list $L = (a_1, \dots, a_n)$ of items, a bin capacity B , and a size $s(a_i) \in (0, B]$ for each item in the list. We wish to pack the items into a minimum number of bins of capacity B , i.e., to partition the items into a minimum number of subsets such that the sum of the sizes of the items in each subset is B or less. Many potential applications, such as packing small information packets into somewhat larger fixed-size ones, involve integer item sizes, fixed and relatively small values of B , and large values of n .

The bin packing problem is NP-hard, so research has concentrated on the design and analysis of polynomial-time approximation algorithms for it, i.e., algorithms that construct packings that use relatively few bins, although not necessarily the smallest possible number. Of special interest have been *on-line* algorithms, i.e., ones that must permanently assign each item in turn to a bin without knowing anything about the sizes or numbers of additional items, a requirement in many applications. In this paper we shall analyze the *Sum of Squares* algorithm, an on-line bin packing algorithm recently introduced in [8] that is applicable to any instance whose item sizes are integral (or can be scaled to be so), and is surprisingly effective.

1.1 Definitions and Notation

First some notation and definitions. Let P be a packing of list L and for $1 \leq h \leq B$, let $N_P(h)$ be the number of bins in P whose contents have total size equal to h . (We shall say that such a bin has *level* h .) We call the vector $\langle N_P(1), N_P(2), \dots, N_P(B-1) \rangle$ the *profile* of packing P .

Definition 1 The sum of squares $ss(P)$ for packing P is $\sum_{h=1}^{B-1} N_P(h)^2$.

The *Sum-of-Squares Algorithm* (*SS*) introduced in [8] is an on-line algorithm that packs each item according to the following simple rule: Let a be the next item to be packed and let P be the current packing. A *legal* bin for a is one that is either empty or has current level no more than $B - s(a)$. Place a into a legal bin so as to yield the minimum possible value of $ss(P')$ for the resulting packing P' , with ties broken in favor of the highest level, and then in favor of the oldest bin with that level. (Our results for *SS* hold for any choice of the tie-breaking rule, but it is useful to have a completely specified version of the algorithm.)

Note that in deciding where to place an item of size s under *SS*, the explicit calculation of $ss(P)$ is not required.

One simply needs to find that i with $N_P(i) \neq 0$ that minimizes $N_P(i+s) - N_P(i)$, $0 \leq i \leq B-s$, where by convention $N_P(0)$ and $N_P(B)$ are defined to be $-1/2$ and $1/2$ respectively. We currently know of no significantly more efficient way to do this in general than to try all possibilities, so the running time for SS is $O(nB)$ overall.

In what follows, we will be interested in the following three measures of L and P .

Definition 2 *The size $s(L)$ of a list L is the sum of the sizes of all the items in L .*

Definition 3 *The number of bins in a packing P , denoted by $|P|$, is $\sum_{h=1}^B N_P(h)$.*

Definition 4 *The waste $W(P)$ of packing P is $\sum_{h=1}^{B-1} N_P(h) \cdot \frac{B-h}{B} = |P| - s(L)/B$.*

Note that the first two quantities are related in that we must have $|P| \geq s(L)/B$. Note also that the count $N_P(B)$ of completely full bins does not contribute to the waste, since the latter is concerned only with unused space in partially-filled bins.

We are in particular interested in the average-case behavior of SS for discrete distributions. A *discrete distribution* F consists of a bin size $B \in \mathbb{Z}^+$, a sequence of positive integral sizes $s_1 < s_2 < \dots < s_J \leq B$, and an associated vector $\bar{p}_F = \langle p_1, p_2, \dots, p_J \rangle$ of rational probabilities such that $\sum_{j=1}^J p_j = 1$. In a list generated according to this distribution, the i th item a_i has size $s(a_i) = s_j$ with probability p_j , independently for each $i \geq 1$. There are three key measures of algorithmic performance. For any discrete distribution F and any algorithm A , let $P_n^A(F)$ be the packing resulting from applying A to a random list $L_n(F)$ of n items generated according to F . We then have

Definition 5 *The expected waste rate for algorithm A and distribution F is*

$$EW_n^A(F) \equiv E \left[W \left(P_n^A(F) \right) \right].$$

Definition 6 *The expected excess rate for algorithm A and distribution F is*

$$EX_n^A(F) \equiv E \left[|P_n^A(F)| - |P_n^{OPT}(F)| \right],$$

where OPT is an algorithm that always produces a packing using the minimum possible number of bins. Note that by definition $EX_n^A(F) \leq EW_n^A(F)$.

Definition 7 *The asymptotic expected performance ratio for A and F is*

$$ER_\infty^A(F) \equiv \limsup_{n \rightarrow \infty} \left(E \left[\frac{|P_n^A(F)|}{|P_n^{OPT}(F)|} \right] \right).$$

1.2 Our results

Let us say that a distribution F is *perfectly packable* if $EW_n^{OPT}(F) = o(n)$ (in which case almost all of the bins in an optimal packing are perfectly packed). By a result of Courcoubetis and Weber [7] that we shall describe in more detail later, the possible growth rates for $EW_n^{OPT}(F)$ when F is perfectly packable are quite restricted: the only possibilities are $\Theta(\sqrt{n})$ and $O(1)$. In the latter case we say F is not only perfectly packable but is also a *bounded waste* distribution. In this paper, we shall present the following results.

1. A theorem that for any fixed perfectly packable distribution F , $EW_n^{SS}(F) = O(\sqrt{n})$.
2. A theorem that if F is a bounded waste distribution, then $EW_n^{SS}(F) = O(\log n)$ or $O(1)$ and that the latter case holds if and only if F satisfies an additional simple combinatorial property. This combinatorial property is satisfied by the discrete uniform distributions $U\{j, k\}$ of $[1, 2, 3, 5, 12]$, which are the main discrete distributions studied to date.
3. A simple $O(nB)$ -time deterministic variant SS' on SS that has bounded expected waste for all bounded waste distributions and $O(\sqrt{n})$ waste for all perfectly packable distributions.
4. A linear-programming based algorithm that, in time polynomial in B and the number of bits required to describe the probability vector \bar{p}_F , determines whether F is perfectly packable, and if so, whether it is a bounded waste distribution. Note that since the running time is polynomial in B rather than in $\log B$, the algorithm technically runs in pseudopolynomial time. Assuming $P \neq NP$, however, we cannot hope for a polynomial time algorithm: the problem solved is NP-hard [3]. Moreover, all previous LP-based approaches took time *exponential* in B .
5. For the case where F is not perfectly packable, lower bound examples and upper bound theorems showing that $1.5 \leq \max_F ER_\infty^{SS}(F) \leq 3$, and that for all lists L , we have $SS(L) \leq 3OPT(L) + 1$, where $A(L)$ is the number of bins used when algorithm A is applied to list L .
6. For any fixed F , a randomized $O(nB)$ -time on-line algorithm SS_F such that $EX_n^{SS_F}(F) = O(\sqrt{n})$ and hence $ER_\infty^{SS_F}(F) = 1$. Algorithm SS_F is based on SS and, given F , can be constructed using the algorithm of (4).
7. A randomized $O(nB \log B)$ -time on-line algorithm SS^* that for any F with bin capacity B has $EW_n^{SS^*}(F) = \Theta(EW_n^{OPT}(F))$ and also $EX_n^{SS^*}(F) = O(n^{1/2})$, the latter implying that $ER_\infty^{SS^*}(F) = 1$. This algorithm works by learning the distribution and using the algorithm of (4) and a variant of the algorithm of (6).
8. Results showing that SS can maintain its good behavior even in the face of a non-oblivious adversary who gets to choose the item size distribution at each step (subject to appropriate restrictions).
9. A randomized $O(nB)$ -time off-line algorithm SS^{**} , based on the algorithms of (4) and (6), that for any fixed bin size B and any list L of items with integer sizes in the interval $[1, B-1]$, will create a packing of L whose expected number of bins is $OPT(L) + O(\sqrt{OPT(L)})$.

Several of these results were conjectured based on experimental evidence in [8], which also introduced the main linear program of (4).

1.3 Previous results

The relevant previous results can be divided into two classes: (1) results for practical algorithms on specific distributions, and (2) more general (and less practical) results about the existence of algorithms. We begin with (1).

The average case behavior under discrete distributions for standard heuristics has been studied in [1, 2, 3, 5, 6, 12]. These papers concentrated on the discrete uniform distributions $U\{j, k\}$ mentioned above, where the bin capacity $B = k$ and the item sizes are $1, 2, \dots, j < k$, all equally likely. If $j = k - 1$, the distribution is symmetric and we have by earlier results that the optimal packing and the off-line First and Best Fit Decreasing algorithms (FFD and BFD) all have $\Theta(\sqrt{n})$ expected waste, as do the on-line First Fit (FF) and Best Fit (BF) algorithms [2, 6].

More interesting is the case when $1 \leq j \leq k - 2$. Now the optimal expected waste is $O(1)$ [2, 3], and the results for traditional algorithms do not always match this. In [2] it was shown that BFD and FFD have $\Theta(n)$ waste for $U\{6, 13\}$, and [4] identifies a wide variety of other $U\{j, k\}$ with $j < k - 1$ for which these algorithms have linear waste. For the on-line algorithms FF and BF, the situation is no better. Although they can be shown to have $O(1)$ waste when $j = O(\sqrt{k})$ [2], when $j = k - 2$ [1, 12], and (in the case of BF) for specified pairs (j, k) with $k \leq 14$ [5], for most values of (j, k) it appears experimentally that the expected waste of BF and FF is linear, and this has been proved for BF and the pairs (8, 11) and (9, 12) [5]. In contrast, $EW_n^{SS}(U\{j, k\}) = O(1)$ whenever $j < k - 1$. On the other hand, BF and BFD can both be implemented to run in time $O(n \log B)$ compared to SS 's current best running time bound of $O(nB)$. (The fastest known implementations of FF and FFD are $\Theta(n \log n)$ and so are slower than SS for fixed B .)

Turning to less distribution-specific results, the first relevant results concerned off-line algorithms. Karmarkar and Karp in [11] presented an off-line deterministic polynomial time algorithm KK that for any list L in our setting guarantees that $KK(L) \leq OPT(L) + O(\log^2 B)$. Although this is a significantly stronger than result (9) above, the Karmarkar-Karp algorithm is far more complicated and expensive than the on-line SS algorithm and its variants. The best running time bound that Karmarkar and Karp could provide for their algorithm was $O(n^8 \log^3 n)$, meaning that it is only of theoretical interest. Also relevant but only of theoretical interest is the off-line approximation scheme of Fernandez de la Vega and Lueker [9]. Like the Karmarkar-Karp algorithm, this would, when specialized to our context, yield $O(1)$ worst-case waste. Moreover, it would run in linear time. Unfortunately, it relies on solving a linear program of size exponential rather than polynomial in B .

For the on-line case, the most general results are those of Rhee and Talagrand [13, 14, 15]. In [14], Rhee and Talagrand proved that for any distribution F (discrete or not) there exists an $O(n \log n)$ on-line randomized algorithm A_F satisfying $EX_n^{A_F}(F) = O(\sqrt{n} \log^{3/4} n)$ and hence $ER_\infty^{A_F}(F) = 1$. (For distributions with irrational sizes and/or probabilities, their results assume a real-number RAM model of computation.) This is a more general result than (6) above, and although the additive error term is worse than the one in (6), the extra factor of $\log^{3/4} n$ appears to reduce to a constant depending only on B when

F is a discrete distribution, making the two bounds comparable. Unfortunately, Rhee and Talagrand only prove that such algorithms *exist*. The details of the algorithms depend on a non-constructive characterization of F and its packing properties given in [13].

In [15], Rhee and Talagrand present a single (constructive) on-line randomized algorithm A that works for all distributions F (discrete or not) and has $EX_n^A(F) = O(\sqrt{n} \log^{3/4} n)$, again with the $\log^{3/4} n$ factor likely to reduce to a function of B for discrete distributions. Even so, for discrete distributions this algorithm is not quite as good as our algorithm SS^* , which itself has $EX_n^{SS^*}(F) = O(\sqrt{n})$ for all discrete distributions and in addition gets bounded waste for bounded waste distributions. Another drawback to the algorithm of [15] lies in its running time: a key step in the algorithm is running the Karmarkar-Karp algorithm on the list of every item seen so far, a step that occurs every time the number of items seen so far equals a power of 2. Thus the overall running time is $O(n^8 \log^3 n)$.

The fastest on-line algorithms previously known that guarantee an $O(\sqrt{n})$ expected waste rate for perfectly packable discrete distributions are due to Courcoubetis and Weber, who used them in the proof of their characterization theorem in [7]. These algorithms are distribution-dependent, but for fixed F run in linear time. At each step, the algorithm must solve a linear program whose number of variables is potentially exponential in B , but for fixed F this takes constant time, albeit potentially a large constant. Moreover, for bounded waste distributions, the Courcoubetis-Weber algorithms have $EW_n^A(F) = O(1)$, whereas the Rhee-Talagrand algorithms cannot provide any guarantee better than $O(\sqrt{n})$. On the other hand, the Rhee-Talagrand algorithms of [14, 15] guarantee $ER_\infty^A(F) = 1$ for all distributions, while Courcoubetis and Weber in [7] only do this for those distributions in which $EW_n^{OPT}(F) = O(\sqrt{n})$.

Thus, although these earlier general approaches rival the packing effectiveness of SS and its variants, and in the case of the offline algorithms actually can do somewhat better, none is usable in practice. The comparative strong points of SS and its variants SS' , SS^* , and SS^{**} are their simplicity, distribution-independence, and the fact that they are sufficiently efficient to be of practical interest.

1.4 Outline of the Paper

The remainder of this paper is organized as follows. In Section 2 we present the details of the Courcoubetis-Weber characterization theorem and prove our result about the behavior of SS under perfect packing distributions. In Section 3 we sketch the proofs of our results for bounded waste distributions. Section 4 covers our linear-programming-based algorithms for characterizing $EW_n^{OPT}(F)$ given F . In Section 5 we discuss our results for linear waste distributions, covering both the suboptimal worst-case behavior of SS and the design of variants to circumvent this. Section 6 discusses the performance of SS in more adversarial situations, and presents our randomized offline variant whose expected performance is asymptotically optimal in the worst case. We conclude in Section 7 by presenting a variety of variants on SS that share its ability to generate sublinear expected waste for any distribution F for which $EW_n^{OPT}(F)$ is itself sublinear.

2. Perfectly Packable Distributions

Theorem 2.1 *Suppose F is a discrete distribution satisfying $EW_n^{OPT}(F) = O(\sqrt{n})$. Then $EW_n^{SS}(F) = O(\sqrt{nB})$.*

The proof of this result relies on the characterization theorem of Courcoubetis and Weber [7], which we now describe. Given a discrete distribution F , a *perfect packing configuration* is an integer-valued, length- J vector $\bar{b} = \langle b_1, b_2, \dots, b_J \rangle$ such that $\sum_{j=1}^J (b_j \cdot s_j) = B$. Such a configuration corresponds to a way of completely filling a bin with items from F . That is, if we take b_i items of size s_i , $1 \leq i \leq J$, we will precisely fill a bin of capacity B . Let Λ_F be the rational cone generated by the set of all perfect packing configurations for F , that is, the closure under rational convex combinations and positive rational scalar multiplication of the set of all such configurations.

Definition 8 *A rational vector $\bar{x} = \langle x_1, \dots, x_J \rangle$ is in the interior of a cone Λ if and only if there exists an $\epsilon > 0$ such that all rational vectors $\bar{y} = \langle y_1, \dots, y_J \rangle$ satisfying $|\bar{x} - \bar{y}| \equiv \sum_{i=1}^J |x_i - y_i| \leq \epsilon$ are in Λ .*

Theorem (Courcoubetis-Weber [7]) *Let \bar{p}_F denote the vector of size probabilities $\langle p_1, p_2, \dots, p_J \rangle$ for a discrete distribution F .*

- (a) *If \bar{p}_F is in the interior of Λ_F , then $EW_n^{OPT}(F) = O(1)$.*
- (b) *If \bar{p}_F is on the boundary of Λ_F , i.e., is in Λ_F but not in its interior, then $EW_n^{OPT}(F) = \Theta(\sqrt{n})$.*
- (c) *If \bar{p}_F is outside Λ_F , then $EW_n^{OPT}(F) = \Theta(n)$.*

Using this result, we prove the following lemma, which is key to the current result and most of those that follow:

Lemma 2.1.1 *Let F be a discrete distribution satisfying $EW_n^{OPT}(F) = O(\sqrt{n})$. There is an algorithm A_F such that given any packing P and an item x randomly generated according to F , A_F will pack x in such a way that for each h with $N_F(h) > 0$, $1 \leq h \leq B - 1$, the probability that $N_F(h)$ increases is no more than the probability that it decreases.*

Note that this implies a bound on the expected increase in $ss(P)$ under A_F . The square of a 0-count can increase by at most 1, and the expected increase in the square of a positive count x is at most

$$\left((x+1)^2 - x^2 \right) + \left((x-1)^2 - x^2 \right) = 2$$

Since a placement changes at most two counts, this means that the expected increase in $ss(P)$ is at most 4. Since SS explicitly chooses the placement of each item so as to minimize the increase in $ss(P)$, we thus must also have that the expected increase in $ss(P)$ under SS is at most 4 at each step. Thus the expected value of $ss(P)$ after packing n items followed according to F is at most $4n$, and Theorem 2.1 follows by an application of the Cauchy-Schwarz inequality.

Thus all that remains is to prove Lemma 2.1.1. The algorithm A_F identified in the lemma depends on the details of the Courcoubetis-Weber Theorem mentioned above. Since

F is perfectly packable, we know by that theorem that there must exist integer vectors b_i and rational α_i satisfying

$$\sum_{j=1}^J (b_{i,j} \cdot s_j) = B, \quad 1 \leq i \leq m \quad (2.1)$$

$$\sum_{i=1}^m (\alpha_i \cdot b_{i,j}) = p_j, \quad 1 \leq j \leq J \quad (2.2)$$

Now since the α_i and p_j are all rational, there exists an integer Q such that $Q \cdot \alpha_i$ and $Q \cdot p_j$ are integral for all i and j . Consider the ideal packing P^* which has $Q\alpha_i$ copies of bins of type \bar{b}_i . Let $L_F = \{x_1, x_2, \dots, x_Q\}$ denote the Q items packed into P^* , and denote the bins of P^* as $Y_1, Y_2, \dots, Y_{|P^*|}$.

Let P be the current packing. We claim that for each bin Y of the packing P^* , there is an ordering $y_1, y_2, \dots, y_{|Y|}$ of the items contained in Y and a special threshold index $last(Y) < |Y|$, which depend on P , such that if we set $S_i \equiv \sum_{j=1}^i s(y_j)$, $0 \leq i \leq |Y|$, then the following holds:

1. P has bins with each level $S_1 \leq S_2 \leq \dots \leq S_{last(Y)}$.
2. If $i > last(Y)$, P has no partially filled bin of level $S_{last(Y)} + s(y_i)$.

That such an ordering and threshold index always exist can be seen from Figure 1, which presents a procedure that, given the current packing P , will compute them. Assume we have chosen such an ordering and threshold index for each bin in P^* . Note that $S_{|Y|} = B$ for all such bins Y , since each is by definition perfectly packed.

Our algorithm A_F begins the processing of an item a by first randomly choosing an appropriate name $r(a) \in L_F$ for it. That is, if a is of size s_j , then $r(a)$ is one of the $Q \cdot p_j$ items in L_F of size s_j , with all such choices being equally likely. Note that this implies that the probability that a is renamed x_i is $1/Q$ for all i , $1 \leq i \leq Q$.

Having chosen $r(a)$, we then determine which bin to place a in as follows. Suppose that in P^* , item $r(a)$ is in bin Y and has index j in the ordering of items in that bin.

- (i) If $j = 1$, place a in an empty bin, creating a new bin with level $s(a) = S_1$.

1. Let the set U of as-yet-unordered items initially be set to Y and let $S = 0$ be the initial total size of ordered items.
2. While $U \neq \emptyset$ and $last(Y)$ is undefined, do the following:
 - 2.1 If there is an item x in U such that P has a partially filled bin of level $S + x$
 - 2.1.1 Put x next in the ordering and remove x from U
 - 2.1.2 Set $S = S + s(x)$.
 - 2.2 Otherwise, set $last(Y)$ to be the number of items ordered so far and exit *While* loop.
3. Complete the ordering by appending the remaining items in U in arbitrary order.

Figure 1: Procedure for ordering items in bin Y

- (ii) If $1 < j \leq \text{last}(Y)$, then put a in a bin of size S_{j-1} , thus increasing its size to S_j .
- (iii) If $j > \text{last}(Y)$, put a into a bin of size $S_{\text{last}(Y)}$ (or into a new bin if $\text{last}(Y) = 0$).

Note that as observed above, if $r(a) \in Y$, then $r(a)$ will take on each of the values y_i , $1 \leq i \leq |Y|$ with probability $p = 1/|Y|$. Thus if $r(a) \in Y$ the probability that the count for level S_i increases equals the probability that it decreases, $1 \leq i < \text{last}(Y)$. The probability that the count for $S_{\text{last}(Y)}$ decreases is at least as large as the probability that it increases (greater if $\text{last}(Y) \leq |Y| - 2$). And for all other levels with positive counts, the probability that a change occurs is 0. Since this is true for all bins Y of the ideal packing P^* , Lemma 2.1.1 and hence Theorem 2.1 follow. ■

3. Bounded Waste Distributions

In order to distinguish the broad class of bounded waste distributions under which SS performs well, we need some new definitions. If F is a discrete distribution, let U_F denote the set of sizes with positive probability under F . We say that a level h , $1 \leq h \leq B - 1$, is a *dead-end* level for F if there is some collection of items with sizes in U_F whose total size is h , but there is no such collection whose total is $B - h$. In other words, it is possible to pack a bin to level h with items from U_F , but once such a bin has been created, it is impossible to fill it completely. Note that the dead-end levels for F depend only on U_F and can be identified in time $O(|U_F|B)$ by dynamic programming. Also, no distribution with $1 \in U_F$ can have a dead-end level, so that in particular the $U\{j, k\}$ do not have dead-end levels. A simple example of a distribution that does have dead-end levels is any F that has $B = 6$ and $U_F = \{2, 3\}$. Here 5 is a dead-end level for F while 1, 2, 3, 4 are not. We say that a level h is *nontrivial* for a distribution F if there is some list L with item sizes from U_F such that the SS packing P of L has more than one bin of level h . In the above $B = 6$ example, it is not difficult to see that there are *no* nontrivial levels.

Theorem 3.1 *If F is a bounded waste distribution with no nontrivial dead-end levels, then $EW_n^{SS}(F) = O(1)$.*

To prove this result we rely on the Courcoubetis-Weber Theorem, Lemma 2.1.1, and the following specialization of a result of Hajek [10].

Lemma 3.1.1 [Hajek's Lemma] *Let S be a state space and let \mathcal{F}_k , $k \geq 1$, be a sequence of functions, where \mathcal{F}_k maps S^{k-1} to probability distributions over S . Let X_1, X_2, \dots be a sequence of random variables over S generated as follows: X_1 is chosen according to $\mathcal{F}_1(\cdot)$ and thereafter X_k is chosen according to $\mathcal{F}_k(X_1, \dots, X_{k-1})$. Suppose $\eta > 0$ and ϕ is a function from S to $[0, \infty)$ such that*

- (a) $E[e^{\eta\phi(X_1)}] < \infty$,
- (b) there is a $\Delta < \infty$ such that for all $N \geq 1$, $P(|\phi(X_{N+1}) - \phi(X_N)| > \Delta) = 0$, and
- (c) there are positive constants D and γ such that for all $N \geq 1$, $E[\phi(X_{N+1}) - \phi(X_N) | \phi(X_N) > D] \leq -\gamma$.

Then there are constants $c > 1$ and $T > 0$ such that for all $N \geq 1$, $E[c^{\phi(X_N)}] < T$.

Note that the conclusion of this lemma implies that there is also a constant T' such that $E[\phi(X_N)] < T'$ for all N . A version of the lemma with the latter conclusion and less general hypotheses was used in the analyses of the Best and First Fit bin packing heuristics in [1, 5, 12]. The added generality is not needed for Theorem 3.1, but will be used in the proofs of subsequent results.

We prove Theorem 3.1 by showing that the potential function $\phi(\bar{x}) = \sqrt{\sum_{i=1}^{B-1} x_i^2}$ satisfies the hypotheses of Lemma 3.1.1, where $\bar{x} = \langle x_1, x_2, \dots, x_{B-1} \rangle$ is a vector of non-negative integers representing the profile of level counts for a packing. It satisfies hypothesis (a) of Hajek's Lemma since the initial state is the empty packing. It is also not difficult to show that it satisfies (b). To prove that it satisfies hypothesis (c), we use the following combinatorial result:

Lemma 3.1.2 *Let F be a bounded waste distribution. Then there are constants M and α such that for any profile \bar{x} with $\phi(\bar{x}) > M$, there is a size $s_i \in U_F$ such that if an item of size s_i is packed by SS into a packing with profile \bar{x} , the resulting profile \bar{x}' satisfies $\phi(\bar{x}')^2 \leq \phi(\bar{x})^2 - \alpha\phi(\bar{x})$.*

This is applied as follows. Let \bar{p} be the vector of probabilities for F . Let $F[i, \epsilon]$ be the distribution which changes p_i to $p'_i = (p_i - \epsilon)/(1 - \epsilon)$ and all other p_j 's to $p'_j = p_j/(1 - \epsilon)$. By the Courcoubetis-Weber theorem and the fact that F is a bounded waste distribution, $F[i, \epsilon]$ is a perfectly packable distribution for all sufficiently small rational $\epsilon > 0$. But by Lemma 2.1.1 this means that the expected increase in $\phi(\bar{x})^2$ is no more than 4 when an item randomly generated according to $F[i, \epsilon]$ is packed by SS . Combining this with the conclusion of Lemma 3.1.2 about the decrease when an item of size s_i is packed, we can conclude that the overall expected decrease is $\Omega(\phi(\bar{x}))$ for all \bar{x} with $\phi(\bar{x}) > M$. This is enough to imply that hypothesis (c) of Lemma 3.1.1 is satisfied, and hence prove Theorem 3.1. ■

Unfortunately, SS does not have bounded expected waste for all bounded waste distributions. Consider the distribution F with $B = 9$, $J = 2$, $s_1 = 2$, $s_2 = 3$, and $p_1 = p_2 = 1/2$. It is easy to see that F is a bounded waste distribution, since 3's by themselves can pack perfectly, and only one 3 is needed for every three 2's in order that the 2's can go into perfectly packed bins. Note, however, that 8 is a nontrivial dead-end level for F , so Theorem 3.1 does not apply. In fact, it can be shown that $EW_n^{SS}(F) = \Omega(\log n)$. Although the formal proof of this requires a detailed combinatorial analysis, the intuition behind it is simple: It is likely that somewhere within a sequence of $n \log n$ items from F there will be $\Omega(\log n)$ consecutive 2's. These are in turn likely to create $\Omega(\log n)$ bins of level 8, and hence, since 8 is a dead-end level, $\Theta(\log n)$ waste.

Fortunately, this is the worst possible result for SS and a bounded waste distribution, although it turns out to be unavoidable for those F that do not satisfy the hypotheses of Theorem 3.1.

Theorem 3.2 *If F is a bounded waste distribution that has nontrivial dead-end levels, then $EW_n^{SS}(F) = \Theta(\log n)$.*

For this result we need to exploit more of the power of Hajek's Lemma (which surprisingly is used in proving the lower bound as well as the upper bound). In addition we need a more sophisticated potential function. Let \mathcal{D}_F denote

the set of dead-end levels for F and let \mathcal{L}_F denote the set of levels that are not dead-end levels for F . We shall refer to the latter as *live* levels in what follows. For a given profile \bar{x} , define $\tau_1(\bar{x}) = \sum_{i \in \mathcal{D}_F} x_i^2$ and $\tau_2(\bar{x}) = \sum_{i \in \mathcal{L}_F} x_i^2$. Note that $\phi(\bar{x}) = \sqrt{\tau_1(\bar{x}) + \tau_2(\bar{x})}$. Let us say that a profile \bar{x}' is *constructible* from a profile \bar{x} under F if there is a way of adding items with sizes in U_F to a packing with profile \bar{x} so that a packing with profile \bar{x}' results. Let $\tau_0(\bar{x}) = \min\{\tau_1(\bar{x}') : \bar{x}' \text{ is constructible from } \bar{x} \text{ under } F\}$. Our new potential function is

$$\sigma(\bar{x}) = \sqrt{1 + \tau_2(\bar{x}) + \tau_1(\bar{x}) - \tau_0(\bar{x})} \quad (3.3)$$

Since the initial state is the empty packing, for which $\sigma = 1$, hypothesis (a) of the lemma is trivially satisfied. The proof that σ satisfies hypothesis (c) is much the same as the argument used to show this for ϕ in the proof of Theorem 3.1. The major difficulty lies in showing that σ satisfies hypothesis (b), that is, that there is a constant Δ such for all profiles \bar{x} , the use of SS to pack an item with size in U_F can neither increase nor decrease $\sigma(\bar{x})$ by more than Δ . We will have to leave the details of this for the full paper. Once we have proved that all hypotheses are satisfied, we can conclude that there exist constants $c > 1$ and $T > 0$ such that for all $N > 0$,

$$E \left[c^{\sigma(X_N)} \right] \leq T. \quad (3.4)$$

The interesting consequences of (3.4) all concern the live levels, and indeed the potential function σ was designed so as to cancel out the effect of the dead-end levels. Note that for each live level h , the component $X_{n,h}$ of the final packing profile X_n satisfies $X_{n,h} \leq \phi(X_n) \leq \sigma(X_n) < c^{\sigma(X_n)} / \log_e c$, and so we have

$$E \left[\sum_{h \in \mathcal{L}_F} X_{n,h} \right] \leq \frac{BT}{\log_e c} \quad (3.5)$$

The second consequence is that for all N , and all $\alpha > 1$,

$$P \left[c^{\sigma(X_N)} > \alpha T \right] < \frac{1}{\alpha}$$

which if we take logarithms base c and set $\alpha = n^2/T$ implies

$$P[\sigma(X_N) > 2 \log_c n] < \frac{T}{n^2}. \quad (3.6)$$

Now since F is a bounded waste distribution, for no item size $s \in U_F$ can s be a dead-end level. Recall also that the maximum count for any live level is by definition bounded by $\sigma(X_N)$. Thus whenever an item is generated according to F , the increase in $ss(P)$ that would be caused by placing it in a new bin is bounded by $2\sigma(X_N) + 1$, and so whatever placement SS decides upon, $ss(P)$ can increase by at most this much. Using (3.6), we can conclude that at any point in the packing process, the probability that the next item's placement increases $ss(P)$ by more than $4 \log_c n + 1$ is at most T/n^2 . Thus, in the process of packing n items, the expected number of such uphill moves is at most T/n .

Now let us consider the dead-end levels. Suppose the count for dead-end level h is $2B(\log_c n + 1)$ or greater and a bin b with level less than h receives an additional item that brings its level up to h . We claim that in the process of attaining this level, bin b must have at one time or another received an item that caused $ss(P)$ to increase by at least

$4 \log_c n + 1$. To see this, let us first revisit the tie-breaking rule used by SS when it must choose between bins with a given level for packing the next item. Clearly the rule chosen has no effect on the amount of waste created, since it has no effect on the packing profiles. For bookkeeping purposes in our analysis, we may thus choose whatever rule we like. The one we shall consider here says that when choosing which bin of a given level h to place an item in, we always pick the bin which most recently attained level h . In other words, the bins for each level will act as a stack, under the "last-in, first-out" rule.

Now consider the bin b mentioned above. In the process of reaching level h , it received less than B items, so it changed levels fewer than B times. Note also that by our tie-breaking rule above, we know that every time the bin left a level, that level had the same count that it had when the bin arrived at the level. Thus at least one of the steps in packing bin b must have involved a jump from a level i to a level j such that $N_F(j) \geq N_F(i) + 2 \log_c n + 2$, meaning that the move caused $ss(P)$ to increase by more than $2(2 \log_c n + 2) + 2 > 4 \log_c n + 1$.

We conclude that

$$\left[E \sum_{h \in \mathcal{D}_F} (X_n, h - 2B(\log_c n + 1)) \right] < \frac{T}{n}$$

and consequently

$$E \left[\sum_{h \in \mathcal{D}_F} X_{n,h} \right] < 2B^2(\log_c n + 1) + \frac{T}{n} = O(\log n) \quad (3.7)$$

for fixed F . Combining (3.5) with (3.7), we conclude that

$$EW_n^{SS}(F) < \left[E \sum_{h \in \mathcal{D}_F} X_{n,h} \right] + \left[E \sum_{h \in \mathcal{L}_F} X_{n,h} \right] = O(\log n).$$

We do not have space here to sketch the full lower bound proof for Theorem 3.2, but here is the key lemma.

Lemma 3.2.1 *Suppose F is a fixed discrete distribution with at least one nontrivial dead-end level, and $D > 0$. Then there is a list L_D consisting solely of items with sizes in U_F , such that the following holds for any packing P with no live-level count exceeding D . Let Q be the packing that results when we use SS to add to P a list $L_{D,H}$ consisting of L_D with each item replaced by H copies of itself. Then Q contains at least H bins with dead-end levels.*

The lower bound proof is based on this lemma, inequality (3.4) above, and the fact that if L_D and $L_{D,H}$ are as in the lemma and we let $m = |L_D|$, there is a constant c depending only on F and D such that a list of length mH^{cmH} generated according to F has probability greater than $1/2$ of containing $L_{D,H}$ as a substring. ■

Although SS misbehaves for bounded waste distributions with nontrivial dead-end levels, a minor modification can eliminate this. The resulting algorithm SS' , which has bounded expected waste whenever the optimal packing does, is defined as follows. Let U be the set of item sizes seen so far and let D_U denote the set of dead-end levels for U . (Initially, U is the empty set.) Whenever an item arrives, we first check if its size is in U . If not, we update U and recompute D_U . After thus screening the item size, we place

the item so as to minimize $ss(P)$ subject to the constraint that no bin with level in D_U may be created unless this is unavoidable.

Theorem 3.3 *If F is a perfectly packable distribution, then $EW_n^{SS'}(F) = O(\sqrt{n})$. If F is a bounded waste distribution, then $EW_n^{SS'}(F) = O(1)$.*

To prove the first conclusion, we first observe that the expected number of items that can arrive before all item sizes have been seen is a constant depending only on F . The first conclusion then follows from the observation that the proof of Lemma 2.1.1 applies to SS' as well as SS . The proof of the second conclusion mimics the proof of Theorem 3.1, except that now we need to take as our initial state X_1 in Hajek's lemma the time at which we first know all the item sizes, and show that there is an η such that condition (a) of the lemma is satisfied. ■

4. Characterizing Distributions

In this section we show how $EW_n^{OPT}(F)$ can be characterized by the solutions to a collection of linear programs in $O(B^2)$ variables and constraints. The first of these LP's was previously presented in [8]. Suppose F has item sizes s_1 through s_J and probabilities p_1 through p_J . Our LP will have JB variables $v(j, h)$, $1 \leq j \leq J$ and $0 \leq h \leq B - 1$, where $v(j, h)$ represents the rate at which items of size s_j go into bins whose current level is h . The constraints are:

$$\begin{aligned} v(j, h) &\geq 0, & 1 \leq j \leq J, & 0 \leq h \leq B - 1 \\ v(j, h) &= 0, & 1 \leq j \leq J, & s_j > B - h \end{aligned} \quad (4.8)$$

$$\sum_{h=0}^{B-1} v(j, h) = p_j, \quad 1 \leq j \leq J \quad (4.9)$$

$$\sum_{j=1}^J v(j, h) \leq \sum_{j=1}^J v(j, h - s_j), \quad 1 \leq h \leq B - 1 \quad (4.10)$$

where the value of $v(j, h - s_j)$ when $h - s_j < 0$ is taken to be 0 by definition for all j . The first set of constraints says that no item can go into a bin that is too full to have room for it. The second set says that all items must be packed. The third says that bins with a given level are created at least as fast as they disappear. The goal is to minimize

$$c(F) \equiv \frac{1}{B} \sum_{h=1}^{B-1} \left((B - h) \cdot \left(\sum_{j=1}^J v(j, h - s_j) - \sum_{j=1}^J v(j, h) \right) \right) \quad (4.11)$$

where note that $\sum_{j=1}^J v(j, h - s_j) - \sum_{j=1}^J v(j, h)$ can be viewed as the rate at which unfilled gaps of size $B - h$ are created.

Our main result about the above LP, to be proved in the full paper, is the following.

Theorem 4.1 *There is a constant α_B , depending on B but not otherwise on F , such that for all sufficiently large n*

$$\left| EW_n^{OPT}(F) - nc(F) \right| \leq \alpha_B \sqrt{n} \quad (4.12)$$

and

$$\lim_{n \rightarrow \infty} E \left[\frac{|P_n^{OPT}(F)|}{s(L_n(F))/B} \right] = 1 + \frac{c(F)}{\sum_{i=1}^J \frac{s_i p_i}{B}} \quad (4.13)$$

From this it follows that F is perfectly packable if and only if $c(F) = 0$. Thus one can determine whether F is perfectly packable in the time it takes to construct and solve the LP given by (4.8) – (4.11). Given its straightforward description, the LP can clearly be constructed in time proportional to its size, so construction time will be dominated by the time to solve the LP. For that, the best algorithm currently available is that of Vaidya [16], which runs in time $O((M + N)^{1.5} NL^2)$, where M is the larger of the number of variables and the number of constraints (the latter including the “ ≥ 0 ” constraints), N is the smaller, and L is a measure of the number of bits needed in the computation if all operations are to be performed in exact arithmetic.

Our LP has JB variables and the number of constraints is $\Theta(JB)$. Thus for our LP the running time is $O((JB)^{2.5} L^2) = O(B^5 L^2)$. To obtain a bound on L , note that all coefficients in the constraints of the LP are 1, 0, or -1 and the coefficients in the objective function are all $O(B)$. This leaves the probabilities p_j to worry about. Let us assume that they have all been expressed in terms of a common denominator $D \geq B$. Note that we can determine $c(F)$ by solving the LP with each p_j replaced by its numerator (the integer Dp_j), and then dividing the answer by D . If we proceed in this way, then all the “probabilities” are integers bounded by D . Following the precise definition of L given in [16] we can then conclude that $L = O(JB \log D)$, giving us an overall running time bound of $O((JB)^{4.5} \log^2 D) = O(B^9 \log^2 D)$. Although the (worst-case) constant B^9 is quite large, in practice we have been able to solve the LP's even for B as large as 1000 using commercial simplex codes [8].

In the remainder of this section, we will show how we can further distinguish between the cases in which $EW_n^{OPT}(F) = \Theta(\sqrt{n})$ and those in which $EW_n^{OPT}(F) = O(1)$, that is, to distinguish cases (a) and (b) in the Courcoubetis-Weber Theorem. Here we need to determine, given that \bar{p}_F is in the cone of perfect packings Λ_F , whether it is also in the interior of Λ_F . Our approach is based on solving J additional, related LP's. The total running time will simply be $J + 1$ times that for solving the original LP, and so we will be able to determine whether $EW_n^{OPT}(F) = O(\sqrt{n})$ and if so, which of the two cases hold, in total time $O(J^{5.5} B^{4.5} \log^2 D) = O(B^{10} \log^2 D)$.

For each i , $1 \leq i \leq J$, let $x_i \geq 0$ be a new variable and let LP_i denote the linear program obtained from (4.8) – (4.11) by (i) replacing (4.10) by

$$\sum_{j=1}^J v(j, h) = \sum_{j=1}^J v(j, h - s_j), \quad 1 \leq h \leq B - 1, \quad (4.14)$$

by (ii) replacing (4.9) by

$$\begin{aligned} \sum_{h=0}^{B-1} v(i, h) &= p_i + x_i \\ \sum_{h=0}^{B-1} v(j, h) &= p_j, \quad 1 \leq j \neq i \leq J, \end{aligned} \quad (4.15)$$

and by (iii) changing the optimization criterion to be “maximize x_i ”. Let $c_i(F)$ denote the optimal objective function value for LP_i . Note that LP_i is feasible for $x_i = 0$ whenever $c(F) = 0$, so that $c_i(F)$ is always well-defined and non-negative in this case.

Theorem 4.2 *If F is a discrete distribution, then \bar{p}_F is in the interior of Λ_F (and hence $EW_n^{OPT}(F) = O(1)$) if and only if $c(F) = 0$ and $c_i(F) > 0$, $1 \leq i \leq J$.*

Proof. Let e_i be the vector with 1 in the i th component and 0's in all other components. Given that Λ is closed under rational convex combinations and scalar multiples, it follows that a vector p is in the interior of a cone Λ if and only if $p \in \Lambda$ and there exists a rational $\epsilon > 0$, such that for every i , the vector $p + \epsilon e_i$ is also in Λ . By the proof of Theorem 4.1 and construction of the linear program LP it is easy to see that \hat{p}_F is in Λ_F if and only if $c(F) = 0$. Similarly, by the construction of the linear programs LP_i , it is easy to see that, given any rational $\epsilon > 0$, $\bar{p}_F + \epsilon e_i$ is in Λ_F , if and only if $\epsilon \leq c_i(F)$. ■

5. Distributions that are Not Perfectly Packable

5.1 Worst-Case Behavior

The implication of Theorem 2.1 that $ER_\infty^{SS}(F) = 1$ when $EW_n^{OPT}(F) = o(1)$ unfortunately does not carry over to the case where $EW_n^{OPT} = \Theta(n)$. In particular, in these cases we can have $ER_\infty^{SS}(F) > 1$, as was first observed in [8], which pointed out that for the distributions FF_m with the single item size 2 and bin capacity $B = 2m + 1$, $\lim_{m \rightarrow \infty} ER_\infty^{SS}(F_m) = 1.5$. We conjecture that this is the worst possible value for $ER_\infty^{SS}(F)$, although at present the best upper bound that we can prove is 3, as implied by the following worst-case result.

Theorem 5.1 *For all lists L ,*

$$SS(L) \leq \frac{3s(L)}{B} + 1 \leq 3OPT(L) + 1.$$

Proof. Suppose the last item of size less than $B/3$ that starts a new bin has size i . (If no such item exists, then all bins are at least $B/3$ full in the final packing and we are done.) Call this item x and let P be the packing just before x was packed. It is sufficient to show that the average level in P is at least $B/3$. If that is so, then the packing of subsequent items cannot reduce the average levels of bins in P to less than $B/3$, and all subsequently started bins, other than the one started with x , must have level $B/3$ or greater.

For $j \leq i$, let Ω_j denote the set of bins in P with levels $j, j+i, \dots, j+\ell_j i$, where ℓ_j is the greatest integer such that $j+\ell_j i < B$. Note that $\Omega_1, \dots, \Omega_i$ is a partition of all bins into i sets, and if we can show that the average level of the bins in each Ω_j is at least $B/3$, we will be done. So fix j and suppose k is the least integer such that either $N_P(j+ki) > 0$ or $j+ki \geq B/3$. If $j+ki \geq B/3$ then the average level of the bins of Ω_j is at least $B/3$. On the other hand, if $j+ki < B/3$, we have $k < \ell_j$. Since, under SS , an item of size i starts a new bin, we must have

$$2N_P(i) + 1 \leq 2(N_P(j+hi+i) - N_P(j+hi)) + 2$$

for $k \leq h \leq \ell_j - 1$, and hence $N_P(j+\ell_j i) \geq \dots \geq N_P(j)$, since all are integers. This means that if we let $t = j+ki$ the average levels of the bins in Ω_j is at least

$$\begin{aligned} \frac{t + (t+i) + \dots + (t+(\ell_j-k)i)}{\ell_j - k + 1} &= \frac{2t + (\ell_j - k)i}{2} \\ &> \frac{j + \ell_j i}{2} \geq \frac{B-i}{2} \geq \frac{B}{3}, \end{aligned}$$

thus proving the theorem. ■

5.2 Modifying SS to Improve Performance

In this section we consider how to modify SS to obtain better average-case performance when $ER_n^{OPT}(F) = \Theta(n)$. For specific fixed distributions, we have the following:

Theorem 5.2 *For any discrete distribution F , there exists a randomized, $O(nB)$ -time variant SS_F of SS such that $EX_n^{SS_F}(F) = O(\sqrt{n})$ and hence $ER_\infty^{SS_F}(F) = 1$.*

Let us first discuss a somewhat simpler $O(nB \log B)$ version, which will be useful in what follows. To construct SS_F all we need do is compute $c(F)$ using the linear program of Section 4. If $c(F) = 0$ then F is perfectly packable and we can use $SS_F = SS$ by Theorem 2.1. If $c(F) > 0$, we proceed as follows. By examining the linear program of Section 4, we can show that if $c(F)$ is added to the probability for items of size 1 (and then all probabilities scaled so that their sum is again 1), the resulting distribution F' has $c(F') = 0$, and hence by Theorem 2.1 $EW_n^{SS}(F') = O(\sqrt{n})$. Given a list L of items generated by F we can simulate the packing of a list generated by F' as follows: At each step, with probability $1/(1+c(F))$ pack the next item from L , and otherwise pack an (imaginary) item of size 1. After n real items have been packed, the packing will be expected to contain $O(\sqrt{n})$ wasted space plus $nc(F)$ space devoted to imaginary items, which is also wasted, for a total of $EW_n^{OPT}(F) + O(\sqrt{n})$. This implies the desired result.

Note that the total number of imaginary items constructed can be as large as $\Theta(nB)$, but since these are all the same size, their packing can be handled by a size-specific priority queue and will take time $O(\log B)$ per item, yielding an overall running time of $O(nB \log B)$. To reduce this to $O(nB)$, one can replace the imaginary items of size 1 with imaginary items of different sizes, generated at appropriate rates so that each completely fills a gap that one expects to be left in an optimal packing. The desired rates can be determined from the values of the variables in an optimal solution to the LP of Section 4. ■

One can use the algorithms of Theorem 5.2 as subroutines in a general learning algorithm SS^* . The packing proceeds in phases, with the k th phase ending after $10B2^k$ real items have been packed, $k = 0, 1, \dots$. During Phase 0, SS^* simply performs SS' , the variant of SS mentioned in Section 3 that has bounded expected waste whenever the optimal packing does. At the end of Phase k , we compute the empirical distribution F_k induced by all the items seen so far, compute $c(F_k)$, and if $c(F_k) \neq c(F_{k-1})$ (with $c(F_{-1}) = 0$ by convention), we “close” the current packing by declaring all partially-packed bins to be off-limits to future items and resetting all the counts to 0. Then in Phase $k+1$ we use SS^{F_k} (SS' if $c(F_k) = 0$). We can show the following:

Theorem 5.3 *For any discrete distribution F with bin capacity B , SS^* runs in time $O(nB \log B)$ and satisfies the following:*

- (a) *If $EW_n^{OPT}(F) = O(1)$, then $EW_n^{SS^*}(F) = O(1)$.*
- (b) *If $EW_n^{OPT}(F) = \Theta(\sqrt{n})$, then $EW_n^{SS^*}(F) = \Theta(\sqrt{n})$.*
- (c) *If $EW_n^{OPT}(F) = \Theta(n)$, then $ER_\infty^{SS^*}(F) = 1$ and $EX_n^{SS^*}(F) = O(\sqrt{n})$.*

The proof is again too lengthy to include here, but a key (and nontrivial) lemma is the following:

Lemma 5.3.1 *Suppose F and F' are discrete distributions with common bin size B and with $U_F = U_{F'}$, and suppose that the probability vectors \bar{p} and \bar{p}' have $|\bar{p} - \bar{p}'| < \epsilon(n)$ where $\epsilon(n) = O(1/\log n)$, and that $EW_n^{OPT}(F) = O(\sqrt{n})$. Then*

$$EW_n^{SS}(F') = O(\max\{n\epsilon(n), \sqrt{Bn}\}).$$

6. Coping with Adversaries

Although Theorem 5.3 is very general, it only covers static instance generation processes, where each item size is generated independently, according to the same distribution. In real world situations, distributions may vary over time, and there may be dependencies between the sizes of items. Interestingly, several of the proof techniques for the above results extend in a limited way to such situations.

Suppose that our item generation process works as follows: Let B be the bin size. For each item x_i , $i = 1, 2, \dots$, the size of item x_i is chosen according to a discrete distribution F_i that has bin size B . An adversary is, however, allowed to choose F_i , given full knowledge of all item sizes chosen so far, the current packing, and the packing algorithm we are using. It would be difficult to do well against such an adversary unless it were somehow restricted, so to introduce a plausible restriction, let us say that such an adversary is *restricted to \mathcal{F}* , where \mathcal{F} is a set of discrete distributions, if all the F_i used must come from \mathcal{F} . As a simple corollary of Lemma 2.1.1 we then have the following.

Theorem 6.1 *Suppose items are generated by an adversary restricted to the set of all perfectly packable distribution. Then the expected waste under SS is $O(\sqrt{n})$.*

Note that without the restriction to perfectly packable distributions, the adversary could force the *optimal* expected waste to be linear.

With even more severe restrictions on \mathcal{F} , one can guarantee *bounded* expected waste against an adversary, since the general hypothesis of Hajek's Lemma allows for adversarial item generation. Thus for instance one can obtain the following.

Theorem 6.2 *Suppose \mathcal{F} is a set of bounded waste distributions none of which has nontrivial dead-end levels, and there is an $\epsilon > 0$ such that every distribution that is within distance ϵ of a member of \mathcal{F} is also a bounded waste distribution. Then if items are generated by an adversary restricted to \mathcal{F} , the expected waste under SS is $O(1)$.*

This latter theorem seems very narrow, but it has an interesting corollary in light of the results of [1, 5, 12].

Corollary 6.2.1 *If items are generated by an adversary restricted to the set of distribution $U\{j, k\}$, $j < k - 1$, then the expected waste under SS is $O(1)$.*

We also have a result for the most powerful adversary, i.e., the one who can generate arbitrary lists of items. Although SS by itself cannot deal very well with such adversaries, the algorithms of Theorem 5.2 can be adapted to create a randomized off-line algorithm SS^{**} that for any fixed bin capacity B does very well for lists L of items of integer size $B - 1$ or less. This algorithm derives the empirical distribution F_L for L , computes $c(F_L)$, randomly generates $|L|$ imaginary items according to F_L and packs them according

to SS^{FL} , and then to the extent possible replaces the imaginary items in the packing by real items from L , putting any leftover items from L into separate bins.

Theorem 6.3 *SS^{**} runs in $O(nB)$ time and satisfies*

$$E[SS^{**}(L)] \leq OPT(L) + O(\sqrt{OPT(L)}).$$

7. Other Sublinear Waste Algorithms

The algorithm SS is not unique in its effectiveness. In this section, we identify other algorithms A that share one or more of the following *sublinearity* properties with SS (where (a) is a weaker form of (b)):

- (a) If $EW_n^{OPT}(F) = O(\sqrt{n})$, then $EW_n^A(F) = o(N)$.
- (b) If $EW_n^{OPT}(F) = O(\sqrt{n})$, then $EW_n^A(F) = O(\sqrt{n})$.
- (c) If $EW_n^{OPT}(F) = O(1)$, then $EW_n^A(F) = O(1)$.

One set of such algorithms are those that replace the objective function $ss(P)$ by a variant that multiplies the squared counts by some function depending only on B and the corresponding level, and then packs items so as to minimize this new objective function. Examples include

$$\sum_{h=1}^{B-1} N_P(h)^2 (B-h), \quad \sum_{h=1}^{B-1} [N_P(h)(B-h)]^2 \quad \text{and} \quad \sum_{h=1}^{B-1} \frac{N_P(h)^2}{h}$$

Such algorithms satisfy both (b) and (c) above, since by Lemma 2.1.1 the expected increase in the objective function at each step is still bounded by a constant (in these cases $4B$, $4B^2$, and 4 respectively). The first of these three variants was actually proposed in 1996 by David Wilson [17], before we had invented the algorithm SS itself. Wilson's unpublished experiments with this algorithm already suggested that it satisfied properties (b) and (c) for the $U\{j, k\}$ distributions. As to which of these variants performs best in practice, our own preliminary experimental studies, using the distributions studied in [8], suggest that there is no clear winner; the best variant depends on the distribution F .

A second class of variants that at least satisfy (a) is obtained by changing the exponent in the objective function.

Theorem 7.1 *Suppose SrP denotes that algorithm that at each step attempts to minimize the function $\sum_{h=1}^{B-1} (N_P(h))^r$. Then for all perfectly packable distributions F ,*

$$EW_n^{SrP}(F) = \begin{cases} O\left(n^{\frac{1}{r}}\right), & 1 < r \leq 2 \\ O\left(n^{\frac{r-1}{r}}\right), & 2 \leq r < \infty \end{cases}$$

The proof again relies on Lemma 2.1.1, although now we need to exploit Taylor series expansions and replace our use of the Cauchy-Schwarz Inequality by the more general Holder's Inequality:

$$\text{If } \frac{1}{p} + \frac{1}{q} = 1, \quad \sum ab \leq \left(\sum a^p\right)^{\frac{1}{p}} \left(\sum b^q\right)^{\frac{1}{q}}$$

Limited experiments with the variants in which $r = 1.5$, 3, and 4 suggest that this class of algorithms might satisfy properties (b) and (c) as well as (a), although we currently do not see how to prove this. Again, our experiments revealed no consistent winner among the variants studied.

Our final class of alternatives to SS are designed to improve the running time, possibly at the cost of packing quality. Recall that J denotes the number of item sizes under F . The $\Theta(nB)$ running time for the naive implementation of SS can be improved to $\Theta(nJ \log B)$ by maintaining for each item size $s \in U_F$ a priority queue Q_s that ranks the levels h by how much $ss(P)$ would increase if an item of size s were placed in a bin of level h in the current packing. The running time can be further improved to $O(nJ)$ by exploiting the fact that the increase in $ss(P)$ due to a particular placement can change by at most 4 each time an item is packed.

Neither of these approaches may be much of an improvement over the naive algorithm for distributions F with large numbers of item sizes, and it remains an open problem as to whether SS can be implemented to run in $o(nB)$ time in general. However, if one is willing to alter the algorithm itself, rather than just its implementation, one can obtain more significant speedups, obtaining overall running times of $O(n \log B)$ or better.

The idea is to perform lazy updating on the priority queues. Note that the organization of each priority queue depends on the counts, and each time an item is packed, one or two counts are changed. Suppose, however, that each time a particular count is changed by the packing of an item, we only update its value in one of the priority queues, cycling through the queues so that each Q_s is never off by more than J in its estimate of any count. We could then mis-estimate the cost of a move by as much as $4J$, but by Lemma 2.1.1 this still leaves the expected increase in the true value of $ss(P)$ bounded by a constant.

There are some technicalities having to do with the fact that if the approximate count in Q_s for level h is less than J , the true count might be 0 and so for safety's sake in this case we can't consider moves that place an item of size s in a bin with level h . However, with some additional work we can construct an algorithm *approxSS* that satisfies (b) and (c) and has running time $O(n \log B)$. With even lazier updating, we can construct an algorithm that satisfies (b) and (c) and has running time $O(n)$. Details will be presented in the full paper.

8. REFERENCES

- [1] S. Albers and M. Mitzenmacher. Average-case analyses of first fit and random fit bin packing. In *Proc. Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 290–299, Philadelphia, 1998. Society for Industrial and Applied Mathematics.
- [2] E. G. Coffman, Jr., C. 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. In *Proceedings 23rd Annual ACM Symposium on Theory of Computing*, pages 230–240, New York, 1991. ACM Press.
- [3] E. G. Coffman, Jr., C. Courcoubetis, M. R. Garey, D. S. Johnson, P. W. Shor, R. R. Weber, and M. Yannakakis. Bin packing with discrete item sizes, Part I: Perfect packing theorems and the average case behavior of optimal packings. *SIAM J. Disc. Math.* To appear.
- [4] E. G. Coffman, Jr., D. S. Johnson, L. A. McGeoch, P. W. Shor, and R. R. Weber. Bin packing with discrete item sizes, Part III: Average-case behavior of FFD and BFD. In preparation.
- [5] E. G. Coffman, Jr., D. S. Johnson, P. W. Shor, and R. R. Weber. Markov chains, computer proofs, and average-case analysis of Best Fit bin packing. In *Proceedings 25th Annual ACM Symposium on Theory of Computing*, pages 412–421, New York, 1993. ACM Press.
- [6] E. G. Coffman, Jr., D. S. Johnson, P. W. Shor, and R. R. Weber. Bin packing with discrete item sizes, Part II: Tight bounds on First Fit. *Random Structures and Algorithms*, 10:69–101, 1997.
- [7] C. Courcoubetis and R. R. Weber. Stability of on-line bin packing with random arrivals and long-run average constraints. *Prob. Eng. Inf. Sci.*, 4:447–460, 1990.
- [8] J. Csirik, D. S. Johnson, C. Kenyon, P. W. Shor, and R. R. Weber. A self organizing bin packing heuristic. In M. Goodrich and C. C. McGeoch, editors, *Proceedings 1999 Workshop on Algorithm Engineering and Experimentation*, pages 246–265, Berlin, 1999. Lecture Notes in Computer Science 1619, Springer-Verlag.
- [9] W. Fernandez de la Vega and G. S. Lueker. Bin packing can be solved within $1+\epsilon$ in linear time. *Combinatorica*, 1:349–355, 1981.
- [10] B. Hajek. Hitting-time and occupation-time bounds implied by drift analysis with applications. *Adv. Appl. Prob.*, 14:502–525, 1982.
- [11] N. Karmarkar and R. M. Karp. An efficient approximation scheme for the one-dimensional bin packing problem. In *Proc. 23rd Ann. Symp. on Foundations of Computer Science*, pages 312–320, 1982. IEEE Computer Soc.
- [12] C. Kenyon, Y. Rabani, and A. Sinclair. Biased random walks, Lyapunov functions, and stochastic analysis of best fit bin packing. *J. Algorithms*, 27:218–235, 1998. Preliminary version under the same title appeared in *Proc. Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 351–358, 1996.
- [13] W. T. Rhee. Optimal bin packing with items of random sizes. *Math. Oper. Res.*, 13:140–151, 1988.
- [14] W. T. Rhee and M. Talagrand. On line bin packing with items of random size. *Math. Oper. Res.*, 18:438–445, 1993.
- [15] W. T. Rhee and M. Talagrand. On line bin packing with items of random sizes – II. *SIAM J. Comput.*, 22:1251–1256, 1993.
- [16] P. M. Vaidya. Speeding-up linear programming using fast matrix multiplication. In *Proceedings, The 30th Annual Symposium on Foundations of Computer Science*, pages 332–337, Los Alamitos, CA, 1989. IEEE Computer Society Press.
- [17] D. B. Wilson. Personal communication, 2000.