APPROXIMATION ALGORITHMS FOR BIN PACKING: A SURVEY[*]

E. G. Coffman, Jr. M. R. Garey D. S. Johnson

The classical one-dimensional bin packing problem has long served as a proving ground for new approaches to the analysis of approximation algorithms. In the early 1970's it was one of the first combinatorial optimization problems for which the idea of worst-case performance guarantees was investigated. It was also in this domain that the idea of proving lower bounds on the performance of online algorithms was first developed, and it is here that the probabilistic analysis of approximation algorithms has truly flowered. The chapter surveys the literature on worst case and average-case behavior of approximation algorithms for one-dimensional bin packing, using each type of analysis to put the other in perspective.

INTRODUCTION

In the classical one-dimensional bin packing problem, we are given a sequence L = $(a_1;\ a_2;\ldots;a_n)$ of items, each with a size $s(a_i) \in (0;\ 1]$ and are asked to pack them into a minimum number of unit-capacity bins (i.e., partition them into a minimum number m of subsets $B_1, B_2, \ldots, B_m$ such that $\Sigma a_i \in B_j\ s(a_i) \leq 1$, $1 \leq j \leq m$).

This NP-hard problem has many potential real-world applications, from loading trucks subject to weight limitations to packing television commercials into station breaks to stock-cutting problems, where the bins correspond to standard lengths of some material, say cable, lumber, or paper, from which items must be cut.

NEXT FIT

Perhaps the simplest algorithm for the classical one-dimensional bin packing problem is Next Fit (NF). This is a bounded-space online algorithm in which the only partially-filled bin that is open is the most recent one to be started, i.e., the nonempty bin B_j in the current packing with the largest index j. (In this and subsequent discussions, we assume that bins are indexed B_1, B_2, … in the order in which they are created, i.e., receive their first items.) Let level(B) be the sum of the sizes of the items in bin B. In packing item a_i, Next Fit tests whether $s(a_i) \leq 1 - \text{level}(B_j)$. If so, it places a_i in bin B_j, leaving that bin open. Otherwise, it closes bin B_j and places a_i in a new bin B_{j+1}, which now becomes the open bin.

[*] In *Approximation Algorithms for NP-Hard Problems*, D. Hochbaum (ed.), PWS Publishing, Boston (1996), 46-93

This algorithm can be implemented to run in linear time, and it is not difficult to show that for all lists L, NF(L) $\leq$ 2 OPT(L) - 1. Furthermore, there exist lists L with arbitrarily large values of OPT(L) such that NF(L) = 2 OPT(L) - 1.

FIRST FIT

Next Fit can be made to suffer because of the bounded-space constraint inherent in its definition. For now we shall go directly to the historically more important First Fit algorithm, in which the restriction is removed entirely and we consider all partially-filled bins as possible destinations for the item to be packed. The particular rule followed is implicit in the algorithm's name: we place an item in the first (lowest indexed) bin into which it will fit, i.e., if there is any partially-filled bin B_j with level(B_j)+s(a_i) $\leq$ 1, we place a_i in the lowest-indexed bin having this property. Otherwise, we start a new bin with a_i as its first item. Note that in removing the bounded-space restriction, we forfeit the benefits of a linear running time enjoyed by Next Fit. We don't have to settle for the naive quadratic-time implementation, however, as it is possible to construct the First Fit packing in time O(n log$n$) using an appropriate data structure.

BEST FIT, WORST FIT, AND ALMOST ANY FIT ALGORITHMS

How crucial is the packing rule used by First Fit to the improved worst-case behavior ? It turns out that other packing rules can do essentially as well. The most famous of these rules is the one used by the Best Fit (BF) algorithm. In this packing rule, which like the First Fit packing rule can be implemented to run in time O(n log$n$), item a_i is packed in the partially-filled bin B_j with the highest level level(B_j) $\leq$ 1 - s(a_i), ties broken in favor of lower index.

Consider the algorithm Worst Fit (WF), in which each item a_i is packed in the partially-filled bin with the lowest level (ties broken by index), assuming it fits, and otherwise starts a new bin.

Surprisingly, it takes only a slight modification to this algorithm to improve it dramatically. Consider the close variant Almost Worst Fit (AWF), in which a_i is placed in the partially-filled bin with the second lowest level (ties broken by index) unless there is only one bin into which it fits, in which case it goes in that bin. If a_i fits in no partially-filled bin, it of course starts a new bin. The worst-case behavior of Almost Worst Fit is just as good as that for First and Best Fit. Again, however, there can be significant differences on individual lists.

More generally, let us say that an online bin packing algorithm is an Any Fit (AF) algorithm if it never starts a new bin unless the item to be packed does not fit in any partially-filled bin in the current packing, and that it is in addition an Almost Any Fit (AAF) algorithm if it never packs an item into a partially-filled bin with the lowest level unless there is more than one such bin or that bin is the only one that has enough room.