Approximation Algorithms
Lecture Notes
Lan Guo

Bin Packing

Minimum Bin Packing
Instance: Finite set I of rational numbers $\{a_1, a_2, \ldots, a_n\}$ with $a_i \in (0, 1]$ for i = 1, …, n.
Solution: A partition $\{B_1, B_2, \ldots, B_k\}$ of I such that $\sum_{a_i \in B_j} a_i \le 1$ for j = 1,…, k, and capacity of B_i, $c(B_i) \le 1$.
Measure: The cardinality of the partition, i.e., k.

We define that a bin is open if we can put item into it; otherwise, it is defined as closed.

Next Fit
The first item a_1 is placed into bin B_1. Let B_j be the last used bin, when the algorithm considers item a_i: it assigns a_i to B_j if B_j has enough room, otherwise, closes B_j and assigns a_i to a new bin B_{j+1}.

For example, suppose we have {0.3, 0.9, 0.2}. The first item 0.3 will go to the first bin B_1. When 0.9 is considered, it is assigned to B_2, since B_1 doesn't have enough room and therefore it is closed. The last item 0.2 is assigned to B_3, since B_2 doesn't have enough room and therefore it is closed.

Analysis of *Next Fit*: Let's say that *Next Fit* uses h bins. The sum of items sizes in each consecutive bins is greater than 1 (otherwise, we can put them together). In addition, we know that optimal value of *Next Fit*, OPT $\ge \lceil \sum a_i \rceil$. In some case, OPT $> \lceil \sum a_i \rceil$. For example, 5 items with size 0.8, we need 5 bins but the sum of item sizes is 4. Let's consider two case for bin number h.

Case 1:

h is even: $c(B_1) + c(B_2) \ge 1$

$+c(B_3) + c(B_4) \ge 1$

…

$$+c(B_{h-1}) + c(B_h) \geq 1$$

$$\sum a_i \geq h/2$$

➔ $h \leq 2\sum a_i \leq 2\lceil \sum a_i \rceil \leq 2OPT$

Case 2:

h is odd: $c(B_1) + c(B_2) \geq 1$

$+c(B_3) + c(B_4) \geq 1$

…

$+c(B_{h-2}) + c(B_{h-1}) \geq 1$

$$\sum a_i \geq (h-1)/2 + c(B_h)$$

➔ $\lceil 2\sum a_i \rceil \geq \lceil h - 1 + 2c(B_h) \rceil$

➔ $\lceil 2\sum a_i \rceil \geq h$

➔ $h \leq \lceil 2\sum a_i \rceil \leq 2\lceil \sum a_i \rceil \leq 2OPT$

Therefore, the approximation ratio for *Next Fit* is 2.

A worst-case scenario is a set of 4n items {½, 1/2n, ½,1/2n, …, ½, 1/2n }. The optimal solution, OPT = n +1, while *Next Fit* algorithm returns 2n. The optimal solution will put 2 items with weight ½ together into 1 bin and 2n items with weight 1/2n together into 1 bin, therefore, it uses n + 1 bins. *Next Fit* will put every adjacent 2 items together, which needs 2n bins.

First Fit
Throw each a_i into the first available bin, assuming that all bins are open.

This algorithm returns a solution h such that h ≤ 1.7OPT + 2. [1]

First Fit Decreasing
1. Sorts the items in non-increasing order with regards to their sizes 2. Processes as *First Fit*.

First Fit Decreasing (FFD) finds a solution h such that h ≤ 1.5 OPT + 1.

Analysis of Algorithm: Let's partition the ordered list of items $\{a_1, a_2, \ldots, a_n\}$ according to their value, into the following sets:

$A = \{a_i \mid a_i > 2/3\}$,

$B = \{a_i \mid ½ < a_i \leq 2/3\}$,

$C = \{a_i \mid 1/3 < a_i \leq ½\}$,

$D = \{a_i \mid a_i \leq 1/3\}$.

Case 1: There is one bin with all items from D. We know that:

1. It has to be the last one.
2. All bins except the last one have used more 2/3 of their capacities. Otherwise, we can put items from D into them.

$2(h-1)/3 + c(B_h) \leq \sum a_i \leq \lceil \sum a_i \rceil \leq OPT$

$\Rightarrow h \leq 3\ OPT/2 + 1 - 3c(B_h)/2$

$\Rightarrow h \leq 3\ OPT/2 + 1$

If there are m bins with all items from D, where $m > 1$, Condition 2 still holds. Therefore, we can reach the same conclusion.

Case 2: There is no bin with all items from D. In this case, we can throw out all items from D without changing the total number of bins. We can conclude that:

1. No bin has more than 2 items.
2. Any bin with 1 item from A can't accommodate any other item.
3. Any bin with 1 item from B can accommodate only another item from C.
4. Any bin with one item from C can accommodate either one item from B or one item from C, but not both.

From the conclusion above, we know that FFD will put at most 2 items in a bin. Observe that FFD processes items by non-increasing order with respect to their weight. Therefore, it puts each item from C with the largest possible item from B that might fit with it and that does not already share a bin with another item. This implies that the solution by FFD and the optimal solution are the same.

A better bound for FFD gives a solution h such that $h \leq 11/9\ OPT + 4$. [2]

References:

[1] Johnson D. S. (1972), "Near-optimal bin packing algorithms", Technical Report MAC TR-109, Project MAC, MIT, Cambridge.

[2] Baker, B. S. (1983), "A new proof for the first fit decreasing algorithm", Technical Report, Bell Laboratories, Murray Hill.