

Approximation Algorithms Lecture 13

Contents

1	Recap	1
2	Content	1
2.1	APTAS for bin packing	1

1 Recap

Fully polynomial time approximation scheme (FPTAS) for knapsack problem

2 Content

2.1 APTAS for bin packing

We will exhibit an APTAS for bin packing with the solution $\leq (1 + \varepsilon)\mathbf{OPT} + 1/\varepsilon^2 + 2$.

We will first write an integer program for bin packing.

Given n_i objects of size s_i , $1 \leq i \leq k$, and $\sum_i n_i = m$.

A configuration c is a k -tuple $(o_1^c, o_2^c, \dots, o_k^c)$ such that $\sum_i o_i s_i \leq 1$, and $o_i \in \mathbb{Z}_{\geq 0}$.

Let x_c be the number of bins which have configuration c . Then $x_c \in \mathbb{Z}_{\geq 0}$.

Then we need to minimize $\sum_c x_c$.

There is another constraint: $\sum_c x_c o_i^c \geq n_i$ for all $1 \leq i \leq k$, which is the integer program.

Relax the integrality constraint on x_c to $x_c \geq 0$, and this will give us a linear program.

Note that in a linear program, the optimal answer is at a vertex. Let the number of different configurations be α (this is also the "dimension" of the LP). Then the optimal solution lies at the intersection of α hyperplanes (inequalities).

We have $\alpha + k$ inequalities in all and at a vertex, we set α inequalities as equalities. At least $\alpha - k$ configurations are set to 0, and so at most k configurations have non-zero x_c .

Our algorithm then becomes:

1. Solve the LP and round up all x_c variables.

Note that solution is \leq LP value + k (since rounding will increase by at most k), which is at most $\mathbf{OPT} + k$.

Note that k is large, and we want to reduce it.

We will group objects to reduce number of distinct sizes as follows: remove all objects of size $\leq \varepsilon$, order objects by decreasing size and form groups of g objects.

Redefine k as the number of distinct sizes now. (Verify this later on). Round up each object in a group to the size of the largest object in that group.

Let I' be this new instance.

If we run the earlier algorithm on I' then we get a solution with number of bins $\leq \mathbf{OPT}' + \lceil \frac{k}{g} \rceil$.

In going from I to I' we only increased the object sizes, hence the solution for I' can also pack the objects in I .

How does \mathbf{OPT}' relate to \mathbf{OPT} ?

Claim 0.1

$$\mathbf{OPT}' \leq \mathbf{OPT} + g$$

Proof. Consider a solution which requires \mathbf{OPT} bins to pack objects of I .

The objects of I' , except those of g_1 can be packed in the spaces occupied by the objects of I . To see this, we can just construct a mapping from g_i to g_{i-1} , and put object $f(o)$ where o is put in I .

Now each group 1 object can be packed in a separate bin, so we are done. \square

So all objects can be packed in $\mathbf{OPT}' + g + \lceil \frac{k}{g} \rceil$ bins.

Here k is the number of distinct sizes obtained after throwing away all objects of size $\leq \varepsilon$. We will choose $g = \lceil \varepsilon^2 k \rceil$, i.e., $g \leq \varepsilon^2 k + 1$.

So the solution we get is $\leq \mathbf{OPT} + \varepsilon^2 k + 1 + \lceil \frac{1}{\varepsilon^2} \rceil \leq \mathbf{OPT} + \varepsilon^2 k + \frac{1}{\varepsilon^2} + 2$.

We have $\varepsilon \leq \frac{\mathbf{OPT}}{k}$ since we have at least k objects of size ε , so the total volume is $\geq \varepsilon k$, whence the total number of bins is at least $\lceil \varepsilon k \rceil$. Then this gets us an upper bound of $(1 + \frac{1}{\varepsilon}) \mathbf{OPT} + \frac{1}{\varepsilon^2} + 2$.

How to handle small objects of size $\leq \varepsilon$?

1. Consider small objects in any order.
2. For each object, check if it can fit into one of the bins. If not, open a new bin.

Note that if I open a new bin, then each bin has objects of size at least $1 - \varepsilon$. Suppose we had r bins opened earlier, then the total size of objects is at least $(1 - \varepsilon)r$, which means that $\mathbf{OPT} \geq (1 - \varepsilon)r$.

This means that $r \leq \frac{\mathbf{OPT}}{1 - \varepsilon}$.

So number of bins in my solution is $r + 1 \leq \frac{\mathbf{OPT}}{1 - \varepsilon} + 1 \leq \mathbf{OPT}(1 + \varepsilon) + 1$. (fix this later on using \mathbf{OPT}').

Note that the running time is dictated by the time required by solving the LP.