PTAS: Knapsack & Bin packing

Knapsack problem:

input: n items, item i has size $s_i \geq 0$, profit/value $v_i \geq 0$

knapsack cap B

Goal: Find a subset S of items s.t. $\sum_{i \in S} s_i \leq B$ and $\max \sum_{i \in S} v_i$

WLOG we assume $s_i \leq B$.

In the frac. setting you can take frac of an item.

It can be shown that in frac. setting the greedy alg sorts the item based on $\frac{v_i}{s_i}$: $\frac{v_1}{s_1} \geq \frac{v_2}{s_2} \geq \cdots \geq \frac{v_n}{s_n}$ and packs items in this order gives opt.

The int. knapsack is NP-hard

How good is greedy for knapsack?

Can be arbitrarily bad: $s_1 = 1$ $v_1 = 2$ Greedy: 2

$s_2 = B$ $v_2 = B$ opt: B

Modified Greedy for knapsack

Return the better of the following two:

- opt of Greedy
- most profitable item

It can be shown that the modified greedy is a 2-approx.

lemma: Suppose greedy picks items $1 \cdots i^*-1$ but cannot pic item i^*: $\frac{v_1}{s_1} \geq \frac{v_2}{s_2} \cdots \geq \frac{v_{i^*-1}}{s_{i^*-1}} \geq \frac{v_{i^*}}{s_{i^*}}$. Then $v_1 + v_2 + \cdots + v_{i^*}$

$$V_1 + V_2 + \cdots + V_{i^*-1} + \frac{B - (s_1 + \cdots + s_{i^*-1})}{s_{i^*}} V_{i^*} \geq \text{opt}$$

Proof: Exercise.

Corollary: at least one of $\{V_1 + V_2 + \cdots + V_{i^*-1},\ V_{i^*}\} \geq \frac{\text{opt}}{2}$

Corollary: if all v_i's $\leq \varepsilon \text{opt}$ then Greedy is $(1-\varepsilon)$-approx for knapsack.

Pseudo-polytime for knapsack:

$A[i, v]$: the size of smallest knapsack using items $1 \cdots i$ with total value v.

$0 \leq i \leq n$ ↓ $\leq \sum_i v_i$

Goal: find largest V s.t. $A[n, V] \leq B$. Say $V = \max_i v_i$ then $V \leq nV$

$$A[i, v] = \min \{ \underbrace{A[i-1, v]}_{\text{don't pick}},\ \underbrace{A[i-1, v-v_i] + s_i}_{\text{Pick item } i} \}$$

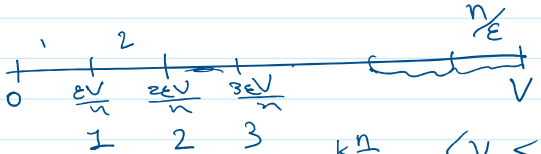
Runtime: if V is the largest value, time $O(n^2 V)$ ($A[i, nV]$)

This is not polytime. We use scaling and rounding to turn this into a PTAS.

instead of all values $0 \cdots V$ we focus on a poly-size many values by scaling:

PTAS for knapsack

- let $k = \frac{V}{n/\varepsilon}$ $\quad \frac{V}{k} = \frac{n}{\varepsilon}$



$k\frac{n}{\varepsilon} < V < k\frac{n}{\varepsilon} + \left(\frac{n}{\varepsilon}\right)$

- for each item i let $v_i' = \left\lfloor \frac{v_i}{k} \right\rfloor$

Run DP ... (s_i, v_i')

- for each item i let $v_i' = \lfloor \frac{v_i}{k} \rfloor$
- Run DP with (s_i, v_i')
- let S' be the set items returned by DP
- return S'

Theorem: This is an FPTAS for knapsack

proof: let S^* be an opt sol with value opt^*.

by def v_i': $\quad k v_i' \le v_i \le k(v_i'+1)$ (by $v_i' = \lfloor \frac{v_i}{k} \rfloor$)

$\rightarrow \quad k v_i' \le v_i \quad \& \quad v_i \le k v_i' + k$

$$\rightarrow opt^* = \sum_{i \in S^*} v_i \le \sum_{i \in S^*} k v_i' + nk \qquad (1)$$

Note that S' is opt under value v_i'. So

$$\sum_{i \in S'} v_i' \ge \sum_{i \in S^*} v_i' \qquad (2)$$

$$\sum_{i \in S'} v_i \ge \sum_{i \in S'} k v_i' \underset{\text{(by def } v_i')}{\ge} \underset{\text{by (2)}}{\ge} \sum_{i \in S^*} k v_i' \underset{\text{by (1)}}{\ge} opt^* - nk = opt^* - \varepsilon V$$

$$\ge (1-\varepsilon) opt^*$$

Since $V \le opt$

So the sol found is $\ge (1-\varepsilon) opt^*$

run time: $O(n^2 \lfloor \frac{V}{k} \rfloor) = O(n^3/\varepsilon)$

Bin Packing

Input: set of items $0 < s_n \le s_{n-1} \le \dots \le s_1 < 1$

Goal: find a min # of unit-size bins into which all items can be packed.

(i.e. partition items into k groups $G_1 \dots G_k$ s.t. $\forall j: \sum_{i \in G_j} s_i \le 1$ and k is minimized

Theorem: There is no α-approx for bin packing for any

Theorem: There is no α-approx for bin packing for any $\alpha < 3/2$ unless $P=NP$.

Proof: reduction from Partition:

Set of integers/values $s_1 \ldots s_n$. Can we partition them into two parts with equal sum?

Suppose $\sum s_i = 2\cdot$

Perfect packing into 2 bins = partition

if there is an α-approx for bin packing with $\alpha < \frac{3}{2}$ then you can decide between

- need ≥ 3 bins
- need ≤ 2 bins

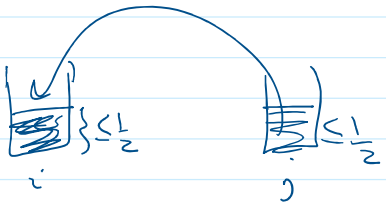
Greedy for Bin packing: First-Fit

Start putting items into bins by trying the first bin into which you can fit it and use a new bin if needed.

Theorem: Cost of F.F. is $\leq 2\text{opt}(+1)$

let $S_I = \sum s_i$ for instance I.

Fact: $\text{opt} \geq S_I$

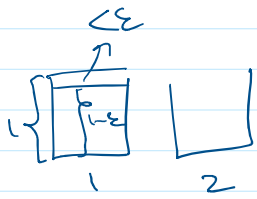


Claim: in the sol of F.F. at most one bin is $\leq \frac{1}{2}$ full

Theorem follows easily from this claim. & the fact above.

Theorem (Johnson '73): if items are sorted in non-inc order then F.F. uses at most $\leq \frac{11}{9} S_I + 4$ bins.

Can be improved if all items are small, i.e. $\leq \varepsilon$:



then in F.F. all the bins except possibly the last are $\geq (1-\varepsilon)$ full. $\longrightarrow$ $\text{F.F.}(I) \leq \frac{1}{1-\varepsilon} \cdot S_I + 1$ $\quad \frac{1}{1-\varepsilon} \leq 1+2\varepsilon$

$\leq (1+2\varepsilon)\,\text{opt}(+1)$ $\quad 1 \leq$

So we can get good approx if all items are small.

We first show how to handle the case when all items are small # of distinct item sizes.

We first show how to handle the case when all items are big (i.e. $\geq \varepsilon$) and there are a small # of distinct item sizes.

This case can be solved optimally in fact!

Lemma: Let $\varepsilon > 0$ be fixed and suppose all items have size $\geq \varepsilon$ and there are at most g (g is a fixed const) distinct item sizes. Then this can be solved opt. in polytime.

Proof: Note that each bin can have $m \leq \lceil \frac{1}{\varepsilon} \rceil$ items. (m is const). The # of diff. bin types (how many from each item type) is at most $g^m = R$ which is $O(1)$.

Since opt $\leq n$, # of diff. feasible sol's is at most # of non-neg sol's to $x_1 + x_2 + \cdots + x_R = n$

$\leq n^{O(R)}$. So this case can be solved using exhaustive search. ∎

Main Lemma: If all items have size $\geq \varepsilon$ (i.e big) for some fixed ε, then there is a PTAS for Bin Packing

We prove this lemma soon. First see how this helps.

Theorem: For any fixed $\varepsilon > 0$, there is an appx for BinPack that returns a sol of cost $\leq (1+2\varepsilon)\text{opt} + 1$ (APTAS).

Proof: Suppose I is the given instance.

Let I' be the big items. Clearly $\text{opt}(I') \leq \text{opt}(I)$

By main lemma, we can find a packing O' of size $(1+\varepsilon)\text{opt}(I')$ for I'. Then use F.F on O' to pack small items.

If F.F does not open any new bins $\longrightarrow$ final answer $\leq (1+\varepsilon)\text{opt}(I') \leq (1+\varepsilon)\text{opt}(I)$ ✓

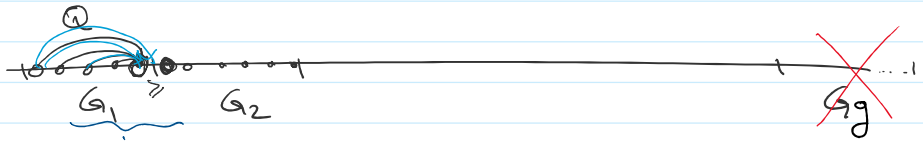
O.w. every bin (except the last) is at least $(1-\varepsilon)$-full. $\longrightarrow$ total # of bins $\frac{1}{1-\varepsilon} \cdot \text{opt} + 1 \leq (1+2\varepsilon)\text{opt} + 1$

Proof of main lemma

Let I_L be the given instance (big ones).

Order the items in non-inc. size & partition them

let I_L be the given instance (big ones).
Order the items in non-inc. size & partition them into $g = \lceil 1/\varepsilon^2 \rceil$ groups each having $Q = \lfloor n\varepsilon^2 \rfloor$ items.



Round up items in G_i to the largest value in G_i; $2 \le i \le k$
Discard G_1 too. Call this instance I'.

Claim: $opt(I') \le opt(I_L)$

we build another instance I'' that "dominates" I' and obviously $opt(I'') \le opt(I_L)$

Start building I'' from I_L: discard items in G_k
round down each item in each group to the smallest in that group. Clearly $opt(I'') \le opt(I_L)$
Also both I' & I'' have $g-1$ groups, furthermore all items in the i'th group of I'' are $\ge$ the items in the i'th group of I'. So $opt(I') \le opt(I'') \le opt(I_L)$

So we can solve I in polytime since they are all big & only g distinct sizes. We can pack all the items of G_1 into new bins to obtain a packing for I_L.
the packing of I_L uses at most $opt(I') + Q \le opt(I_L) +$ for G_1 $\lfloor \varepsilon^2 n \rfloor$

We know $opt(I_L) \ge \varepsilon n$ (because all items are big $\rightarrow$ total sum $\ge \varepsilon n$)

$\rightarrow$ total bins used for $I_L \le opt(I_L) + \varepsilon\, opt(I_L) \le (1+\varepsilon)\, opt(I_L)$

Open Question: for bin packing can we have $opt+1$?
$opt + O(1)$?

can we have opt+1 ?

opt + O(1) ?

best known : opt + O(log n)

from 80's: using LP-rand

opt + O(log² n)