Approximation Algorithms Big Tutorial #2

Let Π be an optimization problem.

ALG is a c-approximation for Π if

Π is a min. problem: $\frac{\text{cost}(ALG(x))}{\text{cost}(OPT_\Pi(x))} \leq c \quad \forall$ instances x of Π

Π is a max. problem: $\frac{\text{cost}(OPT_\Pi(x))}{\text{cost}(ALG(x))} \leq c \quad \forall$ instances x of Π

you may also see $c \in (0,1)$, this is just the inverse.

Today: How hard is it to get within some factor of OPT?

Subset Sum

Given: $z_1, \ldots, z_n \in \mathbb{N}$ and $Z \in \mathbb{N}$.

Wanted: Is there some $S \subseteq \{1, \ldots, n\}$

such that $\sum_{i \in S} z_i = Z$?

Remark: We may assume that $Z \geq \frac{1}{2} \cdot \sum_{i=1}^{n} z_i$:

$\{z_1, \ldots, z_n\}, Z$ is a Yes-instance

$\Leftrightarrow$

$\{z_1, \ldots, z_n\}, (\sum_{i=1}^{n} z_i - Z)$ is a Yes-instance.

Theorem T1.1 SUBSET SUM is NP-hard.

Proof by reduction from 3SAT:

For a formula f with n variables and m clauses, we define $2n+m$ integers z_i and set $Z = \underbrace{1\ldots1}_{n \text{ times}}\underbrace{4\ldots4}_{m \text{ times}}$

Idea: The digits that are 4 encode clauses; each can only become 4 if a variable satisfies it. The digits that are 1 encode variables and ensure that each is only assigned one of true or false.

The integers $z_1, \ldots, z_{2n}$ encode variable assignments, the remaining ones are "filler".

Let $f = (x_1 \vee x_2 \vee \overline{x_3}) \wedge (x_1 \vee \overline{x_2} \vee \overline{x_3}) \wedge (\overline{x_1} \vee x_2 \vee x_3)$.

We define $Z = 111444$

	x_1	x_2	x_3	C_1	C_2	C_3	
z_1	1	0	0	1	1	0	} x_1 or $\overline{x_1}$?
z_2	1	0	0	0	0	1	
z_3		1	0	1	0	1	} x_2 or $\overline{x_2}$?
z_4		1	0	0	1	0	
z_5			1	0	0	1	} x_3 or $\overline{x_3}$?
z_6			1	1	1	0	
z_7				2	0	0	
z_8				1	0	0	
z_9					2	0	
z_{10}					1	0	
z_{11}						2	
z_{12}						1	

- Of each pair of variable encodings, we have to pick one.
- Every clause has 3 variables, so picking only numbers from $z_{xj} \dots z_6$ leaves those digits ≤ 3.
- The filler numbers allow every clause digit to become 4, <u>if</u> at least one variable also adds one to that digit. otherwise, 111444 is not obtainable

PARTITION A special case of SUBSET SUM.

Given: $z_1, \ldots, z_n \in \mathbb{N}$.

Wanted: Is there a partition of the numbers into two halves of equal sum?

$\rightarrow$ Equivalent to SUBSET SUM with

$z_1, \ldots z_n \in \mathbb{N}$ and $Z = \sum_{i=1}^{n} z_i \cdot 1/2$.

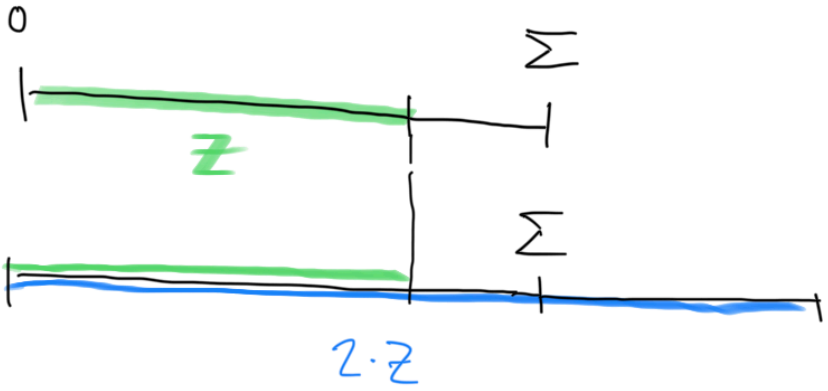
Therefore, PARTITION $\leq_p$ SUBSET SUM.

Is PARTITON $\geq_p$ SUBSET SUM? Yes.

Given: $z_1, \ldots z_n \in \mathbb{N}$ and $Z \in \mathbb{N}$ $\in$ SUBSET SUM

map to

$$z_1, \ldots, z_n, \left(2 \cdot Z - \sum_{i=1}^{n} z_i\right)$$



Bin Packing

Given: $z_1, \ldots, z_n \in \mathbb{Q}$

Wanted: What is the smallest number of bins $b \in \mathbb{N}$ such that $z_1, \ldots, z_n$ can be distributed across b bins $B_1, \ldots, B_b$ with

$$\forall i \in [1,b]: \sum_{j \in B_i} z_j \leq 1 \quad ?$$

Lower bound: Total weight rounded up.
This may require fractional bin assignment and is not always feasible.

First-Fit

- keep a list of used bins, L.
- for $i = 1 \ldots n$ do
  - if z_i still fits into any used bin do
    - put z_i into the first possible bin in L.
  - else
    - add z_i to an empty bin
    - add this bin to L
  - endif
- end for

Theorem T1.2 First-Fit is a 2-approximation for BIN PACKING.

Proof. FF always has at most one open bin in L that has weight $< 1/2$.

By definition, there would otherwise have existed a point in time where some z_i would have fit into an existing bin, but FF opened a new one.

Let k now be the number of bins used by FF, and let $e_i = \sum_{j \in B_i} z_j$ be the weight of B_i.

It follows that

$$\sum_{i=1}^{k} e_i > \frac{k-1}{2}$$

$$\Leftrightarrow \quad k < \left(2 \sum_{i=1}^{k} e_i\right) + 1$$

Since $k \in \mathbb{N}$, it follows that $k \leq \left\lceil 2 \cdot \sum_{i=1}^{k} e_i \right\rceil$. ∎

Thm. T1.3 Unless $P = NP$, there is no polynomial time $(3/2 - \varepsilon)$-approximation for BIN PACKING.

Proof. by reduction from PARTITION.

Let $z_1, \dots z_n \in \mathbb{N}$ be an instance of PARTITION.

We map z_i to $z_i' := \frac{2 \cdot z_i}{\sum_{j=1}^{n} z_j}$ for a BIN PACKING instance. This fits into two bins exactly if the PARTITION instance was a yes-instance. □

Complexity of Approximation.

We consider NP-complete optimisation problems, i.e. the cost function is in P and a solution can be verified in P.

$PTAS = \{\Pi \in NP \mid \forall \varepsilon > 0$: there exists a poly. time $(1+\varepsilon)$-approximation for $\Pi\}$.

"polynomial time approximation scheme"

$f(n)\text{-}APX = \{\Pi \in NP \mid$ there exists a poly time $f(n)$-approximation for $\Pi\}$.

$APX = O(1)\text{-}APX$ (constant factor approximable).

$\log\text{-}APX = \Theta(\log n)\text{-}APX$

$poly\text{-}APX = n^{O(1)}\text{-}APX$.

$$PTAS \overset{?}{\subseteq} APX \subseteq \log\text{-}APX \subseteq poly\text{-}APX.$$

→ Bin Packing $\begin{cases} \in APX \text{ (due to T1.3)} \\ \notin PTAS \text{ (due to T1.2)} \end{cases}$

→ Vertex Cover $\in APX$. Is VC $\in PTAS$?