CS7800: Advanced Algorithms

Lecture 14: Approximation Algorithms I

- Knapsack
- Maximum Coverage

Jonathan Ullman

10-25-2022

Approximation Algorithms

How to deal with computational intractability?

Exact

$$\text{Given } \underbrace{F, g \geq 0}_{\text{input}} \text{ find } y^* \in \operatorname*{argmax}_{y \in F} g(y)$$

feasible solutions

objective function

Approximate

$$\text{find } \hat{y} \text{ such that } g(\hat{y}) \geq c \cdot g(y^*)$$

approximation ratio $c < 1$

Knapsack Problem

Input: n items with integer values $v_i \geq 0$
integer weights $w_i \geq 0$

integer capacity $W \geq 0$

Objective:
$$\max_{S \subseteq \{1,\dots,n\}} \sum_{i \in S} v_i$$
$$\text{s.t. } \sum_{i \in S} w_i \leq W$$

Recap: ① NP-hard to solve exactly

② Can solve with running times

How?

$O(2^n)$ $\quad$ $O(nW)$ $\quad$ $\boxed{O\left(n \cdot \sum_i v_i\right)}$

Greedy Knapsack

Attempt 1: Most valuable first

bad ex

$v_1 = V \quad w_1 = W$

$v_i = V-1 \quad w_i = 1 \quad (\text{for } i \neq 1)$

greedy gets V

opt get $W(V-1)$

$\frac{V}{W(V-1)}$

a $\frac{1}{W}$-approx

Appempt 2: "Densest" first $\quad d_i = \frac{v_i}{w_i}$ (bang-per-buck)

bad ex

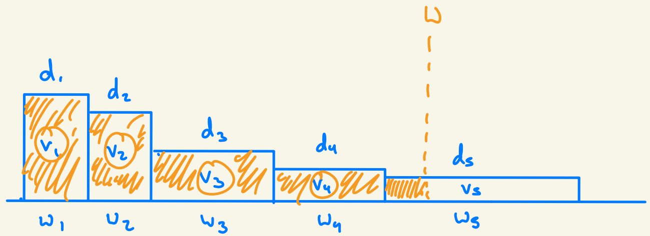
$v_1 = W \quad w_1 = W \quad d_1 = 1$

$v_2 = 1+\varepsilon \quad w_2 = 1 \quad d_2 = 1+\varepsilon$

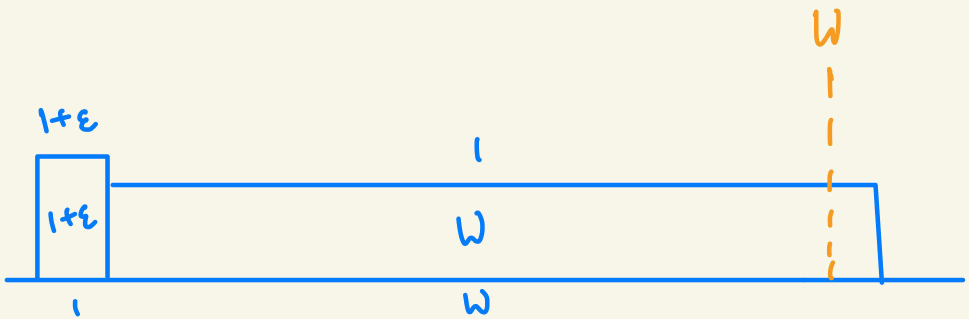
a $\frac{1}{W}$-approx

Fractional Knapsack

<u>Claim:</u> Densest first is optimal for the "fractional" knapsack problem

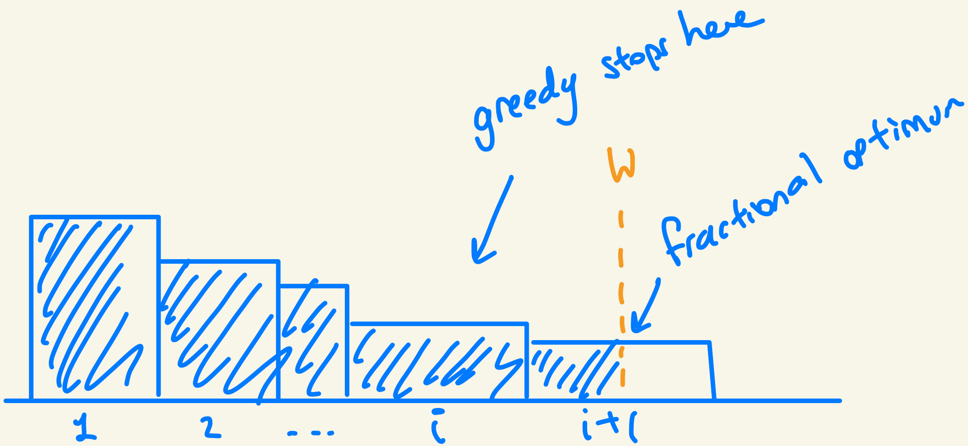


bad example for integral knapsack



Modified Greedy Knapsack

1. Sort by density $\frac{v_1}{w_1} \geq \frac{v_2}{w_2} \geq \dots \geq \frac{v_n}{w_n}$
2. Take items $1, \dots, i$ until running out of space
3. Take the best of $\{1, \dots, i\}$ and $\{i+1\}$



<u>Thm:</u> ModGreedy is a $\frac{1}{2}$-approximation

$$
\begin{aligned}
OPT &\leq OPT_{frac} \\
&\leq \sum_{j=1}^{i} v_j + v_{i+1} \\
&\leq \text{greedy} + v_{i+1}
\end{aligned}
$$

Faster Dynamic Programming for Knapsack

There is an algorithm running in time $O(n \cdot \sum_{i=1}^{n} v_i)$ values are integers

What if we could scale down the values?

① $\alpha = \dfrac{n}{\varepsilon \cdot v_{max}}$ $\left(v_{max} = \max_i v_i \right)$

$\varepsilon \in (0,1)$ is something we choose

② Let $v_i' = \lfloor \alpha v_i \rfloor \in \{0, 1, \ldots, \frac{n}{\varepsilon}\}$

③ Run dynamic programming on $\{(v_i', w_i)\}$

Running time is now $O\left(\frac{n^3}{\varepsilon}\right)$

Thm: ModDP is a $(1-\varepsilon)$-approx

DP Knapsack

1. $\alpha = \dfrac{n}{\varepsilon \cdot v_{max}}$ $\left(v_{max} = \max_i v_i\right)$
2. Let $v_i' = \lfloor \alpha v_i \rfloor \in \{0, 1, \ldots, \frac{n}{\varepsilon}\}$
3. Run dynamic programming on $\{(v_i', w_i)\}$

Thm: ModDP is a $(1-\varepsilon)$-approx

Pf:

Assume $OPT \geq v_{max}$

Key Claim: for any set S $\alpha v(S) \geq v'(S) \geq \alpha v(S) - n$

$v(S) = \sum_{i \in S} v_i$ $v'(S) = \sum_{i \in S} v_i'$

Let A' be the opt for modified inputs

Let A be the opt for original problem

algebra

$$v(A') \underset{\text{clm}}{\geq} \frac{1}{\alpha} v'(A') \underset{\text{optimality}}{\geq} \frac{1}{\alpha} v'(A) \underset{\text{clm}}{\geq} \frac{1}{\alpha}\left(\alpha v(A) - n\right) = v(A) - \varepsilon \cdot v_{max}$$

$$\geq OPT - \varepsilon \cdot OPT$$

$$= (1-\varepsilon) \cdot OPT$$

Alternative DP for Knapsack

Original DP

$OPT(i, u)$ = value of the best solution using items $1, \dots, i$ and knapsack of size u

$$OPT(n, W) = \max\{OPT(n-1, W),\ v_n + OPT(n-1, W-w_n)\}$$

Running Time: $O(n \cdot W)$

$i \in \{0, 1, \dots, n\}$

$t \in \{0, 1, \dots, \sum_j v_j\}$

$OPT(i, t)$ = min wt required to get value t using items $1, \dots, i$

$$OPT(i, t) = \min\{OPT(i-1, t),\ w_i + OPT(i-1, t-v_i)\}$$

Running Time: $O(n \cdot \sum_{i=1}^{n} v_i)$

Maximum Coverage (variant of Set Cover)

Inputs: Sets $S_1, \ldots, S_m \subseteq \{1, \ldots, n\}$

A budget k

Objective: Output sets $\{A_1 \ldots A_k\} \subseteq \{S_1, \ldots, S_m\}$

maximizing $\left| \bigcup_{i=1}^{k} A_i \right|$

Recap: Problem is NP-hard to solve exactly

Greedy Maximum Coverage