* HW2 - due Thurs Sep 11
* Exam 1 on Sep 18

Plan for today: Knapsack $(1-\epsilon)$ approx, Markov Decision Process

Knapsack Problem

Given a budget B ← integer

n jobs: Size s_i and value v_i (assume $s_i \le B$)

Goal: Find $A \subseteq \underbrace{[n]}_{\{1,2,\ldots,n\}}$ s.t.

$$\sum_{i \in A} s_i \le B \quad \text{and we maximize} \quad \sum_{i \in A} v_i$$

Thm 1: We can solve Knapsack in time $O(n^2 \cdot \underbrace{\max_i v_i}_{})$

good for small values, but not polytime in general

Approx Algorithm

What if we don't want to assume values & want a polytime algo?

Give up on optimality.

Given a fixed error parameter ϵ (think $\epsilon = 0.01$)

↑ constant

Design an Alg with value $\ge (1-\epsilon)$ Optimum and run time $\text{poly}\left(n, \log B, \sum_i \log v_i, \sum_i \log s_i\right)$.

E.g. $n^{1/\epsilon} (\log B \cdot \log v_{max})^{10}$

Such an Alg is called Polytime Approx Scheme (PTAS), i.e., runtime is poly(# input bits) assuming ϵ is a constant

Such an alg is called Polytime Approximation (PTAS), if

is poly(# input bits) assuming ϵ is a constant

Thm: For knapsack, there exists a PTAS (infact, depency on ϵ is just poly($1/\epsilon$))

Plan: Replace v_i by $\tilde{v}_i$ such we can apply pseudo-polytime algo

Then argue the obtained soln is $(1-\epsilon)$ approx.

Proof: Define $K = \dfrac{\epsilon \max_i v_i}{n}$

let $\tilde{v}_i := K \left\lfloor \dfrac{v_i}{K} \right\rfloor \implies v_i \geq \tilde{v}_i \geq v_i - K$

Moreover, for any set S of jobs

$$\sum_{i \in S} \tilde{v}_i \geq \sum_{i \in S} (v_i - K) \geq \sum_{i \in S} v_i - \epsilon v_{max}$$

$$\geq \sum_{i \in S} v_i - \epsilon \, opt \quad \text{since } opt \geq v_{max}$$

Alg: (1) Run Pseudo-polytime algo with jobs having value $\dfrac{\tilde{v}_i}{K}$ & size s_i and budget B. (Simplified instance)

(2) Return the obtained set S of jobs

Runtime: $O(n^2 v_{max}) = O\left(n^2 \cdot \dfrac{\max \tilde{v}_i}{K}\right) = O\left(n^2 \cdot \dfrac{\max v_i}{K}\right) = O(n^3/\epsilon)$

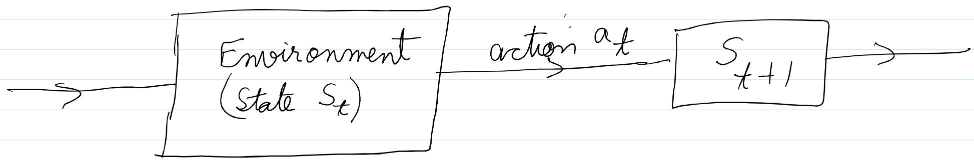
Ignores bit-complexity

Alg's value: Want to show $\sum_{i \in S} v_i \geq (1-\epsilon)\, opt$

Exercise on HW-2

Markov Decision Process (MDP)

Problems involving uncertainty : If Alg takes an action, the output is uncertain



Defn :
- Finite state space $\mathcal{S}$
- Finite action space $\mathcal{A}$
- Start state $s_1 \in \mathcal{S}$
- Transition prob $p_a(s, s')$ for all $a \in \mathcal{A}$, $s, s' \in \mathcal{S}$

  where $\sum_{s' \in \mathcal{S}} p_a(s, s') = 1$

  Input size $= O(|\mathcal{S}|^2 \cdot |\mathcal{A}|)$
- Rewards $r_a(s) \in \mathbb{R}$

Alg (Policy) π : It tells us what action $a_t^\pi \in \mathcal{A}$ to take at t-th step given prior states.

Goal : Given a time horizon T, design an alg (policy) that plays actions to maximize $\mathbb{E}[\text{Total reward}]$

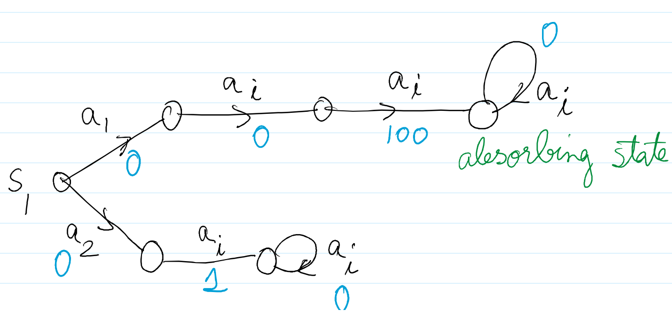
If s_t^π denotes t-th state of policy π then goal is

$$\max_\pi \mathbb{E}\left[\sum_{t=1}^{T} r_{a_t^\pi}(s_t^\pi)\right]$$

Eg 1:

$\mathcal{S} = 6$ states

$\mathcal{A} = \{a_1, a_2\}$

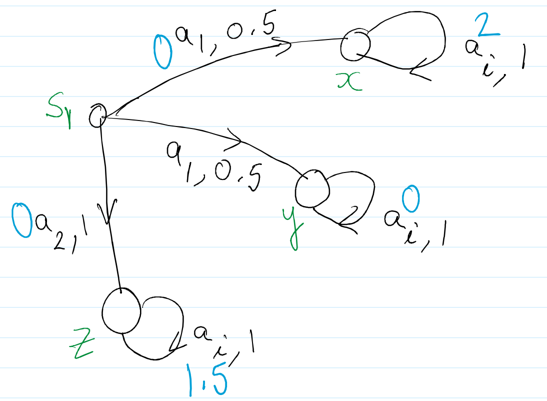


If $T=1$, what's the opt policy? a_i $\therefore$ reward $=0$

If $T=2$ " a_2, a_i reward $=1$

If $T=3$ " a_1, a_i, a_i reward $=100$

Eg 2:



$\mathcal{S} = \{s_1, x, y, z\}$

$\mathcal{A} = \{a_1, a_2\}$

If $T=1$: a_i has reward $=0$

If $T=2$: $a_2, a_i \Rightarrow$ rew $=1.5$

a_1, a_i : $\frac{1}{2}(0+2) + \frac{1}{2}(0+0) = 1$

Q: what's the optimal policy given an MDP and time horizon T?

Thm: DP can find opt policy in time $O(|\mathcal{A}| \cdot |\mathcal{S}|^2 \cdot T)$

Obs 1: We can always assume that the optimal policy is deterministic.

Obs 2: Optimal action only depends on current state and remaining number of time steps.

(Markovian property)

(Does not depend on history beyond knowing current state).

Define: $V^*(s,t) \leftarrow \mathbb{E}\left[\begin{array}{l}\text{Reward of optimal policy} \\ \text{starting at state } s \in \mathcal{S} \text{ and} \\ \text{time horizon } t\end{array}\right]$

Base case $\leftarrow$ Easy exercise

Assume we have computed $V^*(s,t')$ for $t' < t$.

$$V^{\pi}(s,t) = r_{a_1^{\pi}}(s) + \sum_{s' \in \mathcal{S}} p_{a_1^{\pi}}(s,s') * V^{\pi}(s', t-1)$$

$$V^*(s,t) = \max_{a \in \mathcal{A}} \left\{ r_a(s) + \sum_{s' \in \mathcal{S}} p_a(s,s') \cdot V^*(s', t-1) \right\}$$

Solved via DP ∎