Approximation Algorithms

Tuesday, April 26, 2022 1:24 PM

decision problems $\leq_p$ ↳ optimization problem (NP-hard)

α-Approximation Alg for Problem Π

A poly-time algorithm that outputs a feasible solution ALG to Π. Let OPT be the optimum solution to Π.

$$\left(\frac{VAL(ALG)}{VAL(OPT)}, \frac{VAL(OPT)}{VAL(ALG)} \right) \leq \alpha.$$

Approx-Alg

Greedy | DP based | LP-based | Random

Greedy Approximation Algs:

Scheduling: → Given: 1) Set of n jobs $\{[n]\}$
(efficiently divide jobs)
2) each job i has a processing time p_i.
3) m machines.

[illegible]

... machines.

Goal: → Find a partition of $[n]$ into $\langle X_1, X_2, \dots, X_m \rangle$ that minimizes max-span

$$\boxed{\max_{i \in [m]} \underbrace{\sum_{j \in X_i} p_j}_{\text{total time for completion by machine } i}} \rightarrow \text{max-span}$$

$$\boxed{\min_{X = \langle X_1, X_2, \dots X_m \rangle} \ \max_{j \in [m]} \ \sum_{i \in X_j} p_i}$$

claim: → SCHEDULING is NP-hard.

PARTITION $\leq_p$ SCHEDULING.

I of partition $S = \{ \quad \}$ → 2 machines

Appx-Alg:

1) for $i = 1$ to n

1) for i = 1 to n
assign job i to machine with minimum load/processing time currently

→ feasible partition ✓
→ Poly-time ✓

Claim: → Algorithm is a 2-approximation algorithm.

Let $X_1, X_2, \ldots, X_n$ be the partition outputed by the alg
& $X_1^*, X_2^*, \ldots, X_n^*$ - optimum-pa

$$\max_{j \in (m)} \sum_{i \in X_j} P_i \leq 2 \max_{j \in (m)} \sum_{i \in X_j^*} P_i$$

$$2\,(\text{max-span})$$

intuitive-lower-bound

1) $\text{max-span} \geq \frac{1}{m} \cdot \sum_{i \in (n)} P_i$
2) $\text{max spax} \geq \max_{i \in (n)} P_i$

$\Rightarrow$ Max-span $\geq$ $\max_{i \in [n]} P_i$

At any point in time, if the alg. assigns job a to machine b

$$X_b^{t+1} = X_b^t \cup \{a\}$$

$$\underline{\sum_{i \in X_b^t} P_i} \leq \frac{1}{m} \cdot \sum_{i \in [n]} P_i \leq \text{max-span}$$

$$+ \quad \underline{P_a} \leq \max_{i \in [n]} P_i \leq \text{max-span}$$

$$\underline{\sum_{i \in X_b^{t+1}} P_i \leq 2 \cdot \text{max-span}}.$$

$\frac{3}{2}$ - Appx- alg

1). Sort the jobs in decreasing order of processing.

2) Run the prev. alg.

claim: $\Rightarrow$ let $\langle x_1, x_2, \ldots x_n \rangle$ = partition claim: let $\langle X_1, X_2, \ldots X_m \rangle$ = partition returned by the alg.

$$\max_{j \in (m)} \sum_{i \in X_j} P_i \leq \frac{3}{2} \text{max-span}.$$

OBS: $n < m$, the $X_1, X_2, \ldots X_n$ is optimum.

CASE: $n > m$

$$\underline{\text{max-span}} \geq 2 \underline{P_{m+1}} \xrightarrow{} m$$

$$\geq 2 \underline{P_j} \quad j \geq m+1$$

Alg assigns job a to machine b.

$$\sum_{i \in X_b^t} P_i \leq \frac{1}{m} \cdot \sum_{i \in (n)} P_j \leq \text{max-span}$$

$\vdash$

$$P_a \leq \frac{1}{2} \text{max-span}$$

$$\sum_{i \in X^{(t+1)}} P_i \leq \frac{3}{2} \text{max-span}$$

$$\sum_{i \in X_b^{(H)}} t_i \leq \frac{1}{2} \cdot \text{max-span}$$

SET-COVER

Given: 1) $S_1, S_2, \ldots, S_m \subseteq U = \{1 \ldots n\}$

2) Each set S_i has a weight or cost w_i.

Goal: Find $C = \{S_1, S_2, \ldots S_r\}$

s.t. $\bigcup_{i \in C(r)} S_i = U$ and

$\sum_{S_i \in C} w_i$ is minimum.

Thm: SET-COVER is NP hard.

VERTEX-COVER $\leq_p$ SET-COVER

Appx-Alg.

while all elements in U are uncovered

choose a set S_i with smallest

choose a set S_i with smallest value of $\frac{w_i}{|S_i \cap R|}$ where

R = set of uncovered elements in U.

Alg runs in poly time.

$$\underbrace{H(d)}_{\text{harmonic number}} = \frac{1}{d} + \frac{1}{d-1} + \cdots 1$$

$$H(d) \in \Theta(\lg d)$$

Let C be the cover returned by the algorithm &

$OPT = \{S_1, S_2, \ldots, S_r\}$ be the optimum-set-cover

$$\boxed{VAL(ALG) = \sum_{S_i \in C} w_i = \sum_{e \in [n]} c_e}$$

If element $e \in U$ was covered by set S_i by the ALG

$$c_e = \frac{w_i}{|S_i \cap \underline{R}|}$$

$\hookrightarrow$ set of uncovered element at that time

$OPT = \{S_1, S_2, \ldots, S_v\}$

$$val(OPT) = \sum_{S_i \in OPT} w_i$$

$w_i \cdot H(e)$

$$VAL(ALG) = \sum_{e \in (n)} c_e \leq \sum_{S_i \in OPT} \sum_{e \in S_i} c_e$$

$\rightarrow H(e) \cdot VAL(OPT)$

---

claim:- $$\sum_{e \in S_i} c_e \leq H(|S_i|) \cdot w_i$$

$S_i = \{s_1, s_2, \ldots s_j, \ldots s_d\}$

order in which elements in S_i are covered by alg.

are covered by alg

$$\frac{w_i}{|S_i \cap R|} \le \frac{w_i}{d-j+1}$$

Alg covered S_j by set $\underline{S_k}$

$$C_{S_j} = \frac{w_k}{|S_k \cap R|} \le \frac{w_i}{|S_i \cap R|} \le \frac{w_i}{d-j+1}$$

$$\sum_{j \in [d]} C_{S_j} \le w_i \sum_{j \in [d]} \frac{1}{d-j+1}$$

$$= w_i \left(\frac{1}{d} + \frac{1}{d-1} + \cdots \frac{1}{1} \right)$$

$$= w_i \, H(d)$$

$$\boxed{\sum_{e \in S_i} C_e = H(|S_i|) \cdot w_i}$$

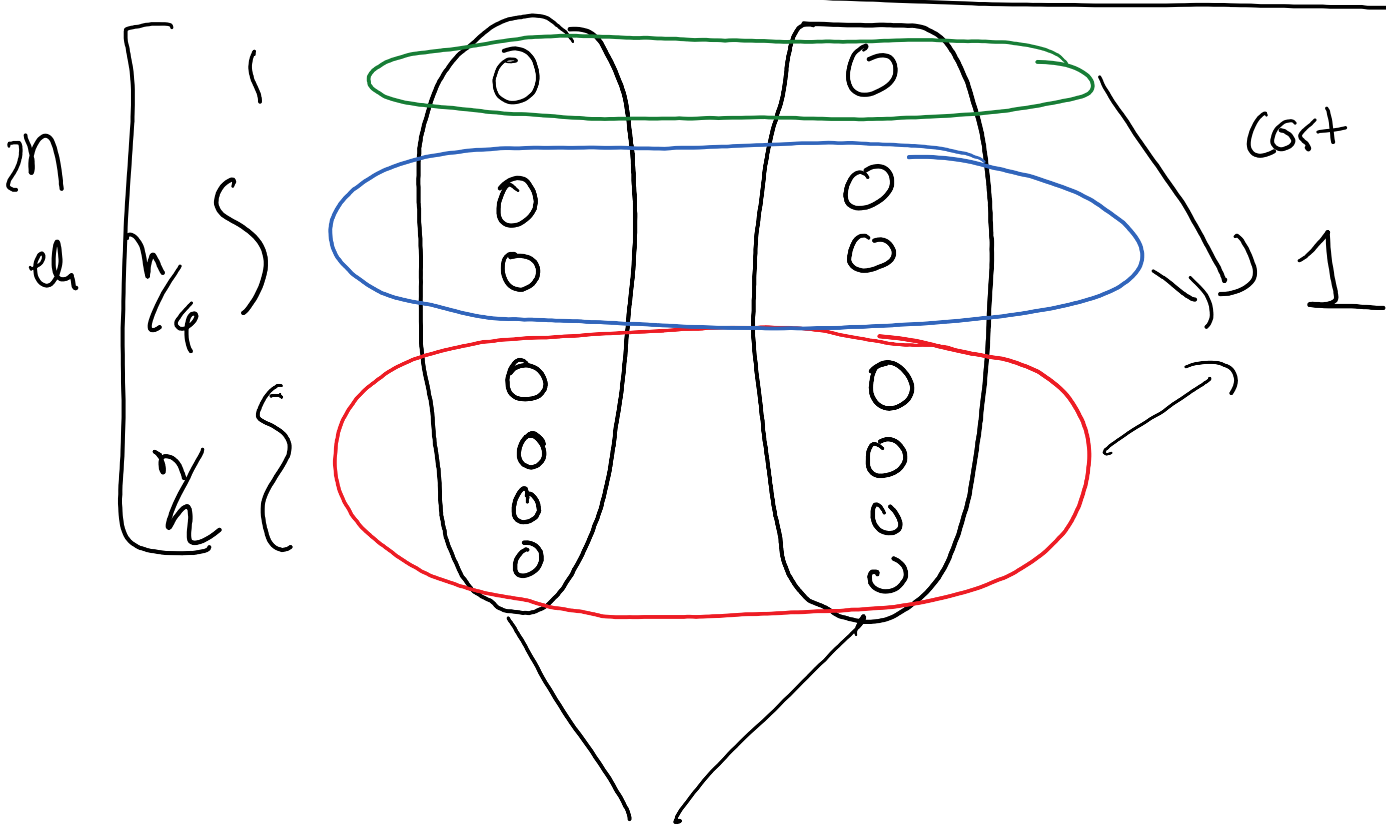
$$\lambda = \max_{i \in [m]} |S_i|$$

$\sum_i C_e$

$$\sum_{e \in S_i} c_e \le H(l_i)\, w_i$$

$$\Rightarrow \quad \sum_{e \in (n)} c_e \le \sum$$

Thm: $\Rightarrow$ Algorithm is a $H(l)$-approximate-algorithm



cost = $1+\varepsilon$

LOWER-BOUND of $\lg(n)$

DYNAMIC PROGRAMMING BASED APPROACH

KNAPSACK

Given:→ 1) Set of n elements

2) each element i has value v_i & weight w_i

3) Knapsack bag of size W.

Goal:→ Find $S \subseteq [n]$ s.t.

$$\sum_{s \in S} w_s \leq W \text{ and}$$

$$\sum_{s \in S} v_s \text{ is maximum}$$

PARTITION $\leq_p$ KNAPSACK

Recall:→ Solve knapsack exactly in time $O(n^2 V)$ → DP-alg

In time $O(n^2 V)$ $\rightarrow$ DP-alg

where $V = \max_{i \in [n]} v_i$

Thm.

For any $\varepsilon > 0$, in time $O(n^3 \varepsilon^{-1})$, we can find $S \subseteq [n]$ that fits in the knapsack &

$$\sum_{s \in S} v_s \geq (1+\varepsilon)^{-1} \sum_{s \in S^*} v_s$$

where S^* = optimum solution

ALG

-- w_i but v. $O(n^3 \varepsilon^{-1})$

, $v_i'' = \lceil v_i / b \rceil \in O(n/\varepsilon)$

4) Solve KNAPSACK on I'' & return solution.

S is the set returned by ALG

S^* is the set - - - - - by OPT.

$$\sum_{i \in S^*} v_i \le \underline{\sum_{i \in S^*}} \left\lceil \frac{v_i}{b} \right\rceil \cdot b$$

$$\le \underline{\sum_{i \in S}} \left\lceil \frac{v_i}{b} \right\rceil b$$

$$\le \sum_{i \in S} v_i + \underline{n b}$$

Recall $b = \frac{\varepsilon}{2n} \cdot V \Rightarrow$ $nb = \frac{\varepsilon V}{2}$

$= \sum v_i + \varepsilon V$

$= \sum_{i \in S} v_i + \quad /2$

$$\sum_{i \in S^*} v_i \leq \sum_{i \in S} v_i + \varepsilon/2 \sum_{i \in S^*} v_i$$

$$\Rightarrow \left(1 - \varepsilon/2\right) \cdot \sum_{i \in S^*} v_i \leq \sum_{i \in S} v_i$$

$$\Rightarrow \sum_{i \in S^*} v_i \leq \left(1 - \varepsilon/2\right)^{-1} \cdot \sum_{i \in S} v_i$$

$$\leq (1+\varepsilon) \sum_{i \in S} v_i$$

for all $\varepsilon \leq 1$

Running time: $\cancel{n \quad O\left(n^2 V\right)}$