More on Approximation Algorithms

Thursday, April 28, 2022 1:44 PM

Recap: →

optimization problems is NP-hard

α- Approximation Alg: →

1) Poly-time
2) it outputs a feasible solution ALG
3) $\left(\frac{VAL(ALG)}{VAL(OPT)}, \frac{VAL(OPT)}{VAL(ALG)}\right) \leq \alpha$

Greedy DP LP randomized

General e.g. ← 1) SCHEDULING → 2 Appx Alg (naive) ; → 3/2 Appx Alg

specific case VERTEX-COVER ← 2) SET-COVER → $H(d) \in O(\lg d)$, d = max size of one of the sets!

3) KNAPSACK → $(1+\varepsilon)$. appx. alg. in $O(n^3 \varepsilon^{-1})$

VERTEX-COVER (weighted)

Given: → 1) Graph $G = (V, E)$

2) w_v associated with every vertex v.

3) $S \subseteq V$ covers E if for all $(u,v) \in E$, either $u \in S$ or $v \in S$.

<u>Goal</u>: $\to$ S s.t S covers E and $\sum_{v \in S} w_v$ is minimum

---

(I) <u>LP-based</u>

$$\underline{x_v} = \begin{cases} 1 & \text{if } v \text{ is in the vertex-cover} \\ 0 & \text{otherwise} \end{cases}$$

\# terms $\leq f$

x_s $\qquad \sum_{\substack{S: \\ e \in S}} x_s \geq 1 \quad e$

Objective $\left\{ \sum_{v \in V} w_v x_v \right.$

Covering constraints $\left\{ \begin{array}{l} x_u + x_v \geq 1 \quad \forall (u,v) \in E \\ \boxed{x_v \in \{0,1\}} \end{array} \right.$

<u>VC: IP</u>

every feasible solution of VC: IP is a valid-vertex-cover

LP-relaxation of VC: IP

$x_v \in \{0, 1\}$

$\underline{0 \le x_v \le 1} \quad \forall v \in V.$ (VC: LP)

<u>Alg.</u>

1) Solve LP ✓ (poly-time ())

2) <u>Round the optimum solution x to LP</u>

$x_v = \underline{0.4}$

$\underline{0} \qquad 1$

$\forall x_v \ge \frac{1}{2}$ $\frac{1}{f}$ set $\underline{y_v} = 1$

$x_v < \frac{1}{2}$ set $y_v = 0$.

Does y satisfy covering constraints?

✓ $\left[\begin{array}{ll} y_u + y_v \ge 1 & \forall (u,v) \in E \\ y_v \in \{0,1\} & \forall v \end{array}\right.$

$\cancel{\Delta}$

$\underline{x_u + x_v} \ge 1 \Rightarrow$ either $\underline{x_u \ge \frac{1}{2}}$ or $x_v \ge \frac{1}{2}$

$\Downarrow$

$y_u + y_v \ge 1$ $\Leftarrow$ either $y_u = 1$ or

$y_v \le f x_v$

$y_v \le 2x_v$

$\forall v \in V$

$\forall v \in V$

$\underline{y_u + y_v \geq 1}$ $\not\Leftarrow$ either $y_u = 1$ or $y_v = 1$

Let $\underline{L^*}$ be the optimum solution to $\underline{VC:IP}$.

1) $L^* \geq \underline{L}$ = optimum solution value to $\underline{VC:LP}$.

<u>Approximation ratio</u>.

$$\underline{\underline{val(ALG)}} = \sum_{v \in V} w_v y_v$$

$$\leq 2 \underline{\sum_{v \in V} w_v x_v} \qquad \text{def}$$

$$= 2L \qquad \text{def } L$$

$$\underline{\underline{\leq 2L^*}} \qquad \text{def } L^* \, (L \leq L^*)$$

2-appx-alg for vertex-cover

$H(\Delta)$-app- - -

Δ = max-deg of a vertex in G

What is the approximation for

↪ what is the approximation for SET-COVER?

f approximation $\quad$ $H(d)$

$$f = \max_{e \in U} f_e$$

f_e = # sets which contains e

1) In poly-time we can get $\min(f, H(d))$ appx for set-cover

SCHEDULING (Gen)

Given:→
1) set $[n]$ of n jobs, each job i has a processing time of t_i.
2) set $[m]$ of m machines
3) each job i can be scheduled on a set of machines J_i

Find:→ A partition $X = (X_1, X_2, \ldots, X_m)$

$$\max_{i \in [m]} \sum_{j \in X_i} x_{ij} \text{ is minimum}$$

(I) LP-based

x_{ij} : processing time of job j on machine i.

max-span objective: $\min L$

$$\sum_j x_{ij} \le L \quad \forall i \in [m]$$

scheduling constraints:

$$\boxed{\sum_{i \in [m]} x_{ij} = t_j} \quad \forall j \in [n]$$

$$\boxed{x_{ij} \in \{0, t_j\}} \quad \forall i, j$$

$$x_{ij} = 0 \quad \forall j, \forall i \notin J_j$$

GS: IP

One to one correspondence between every solution to GS: IP & the scheduling instance..

Consider LP relaxation of GS: IP

$$x_{ij} \in \{0, t_j\} \xrightarrow[\text{Relax}]{} x_{ij} \ge 0 \quad \forall i, j$$

GS: LP

ALG

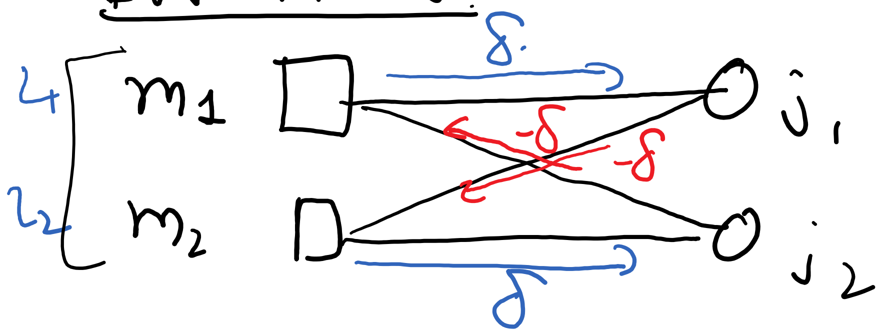
1) Solve GS: LP (poly-time)

2) Round the optimum solution x to GS: LP

Rounding:

Stage 1: "make LP solution as integral as possible"

INTUITION



unnecessarily fractional allocation

1) increase consumption of j_1 for m_1 by δ
2) decrease ... of j_1 for m_2 by δ

1) decrease consumption of j_2 for m_1 by δ
2) increase consumption of j_2 for m_2 by δ

$$\delta = \min(x_{12}, x_{21})$$

one of j_1 or j_2 will be integrally allocated.

Given an allocation x,

$$G(x) = (V(x), E(x))$$

$V(x) = M \cup J$

$(i, j) \in E(x)$ iff $x_{ij} > 0$

If $G(x)$ contains a cycle

$$i_1 \xrightarrow{+\delta} j_1 \xrightarrow{-\delta} i_2 \xrightarrow{+\delta} j_2 \cdots \xrightarrow{-\delta} i_k \xrightarrow{+\delta} j_k \xrightarrow{-\delta} i_1$$

1) increase consumption along (i_ℓ, j_ℓ) by δ
2) decrease consumption along $(i_{\ell+1}, j_\ell)$ by δ

(indices are modulo k)

$$\delta = \min_{\ell \in (k)} x_{i_{\ell+1} j_\ell}$$

$\hookrightarrow$ reduces # edges

$\rightarrow$ $O(mn)$ cycle-eliminations, $G(x)$ is acyclic ($G(x)$ is a tree)

Step-2 (Round fractional x to an

Stage-2 (Round fractional x to an integer solution)

$L^* =$ optimum solution to GS: IP.

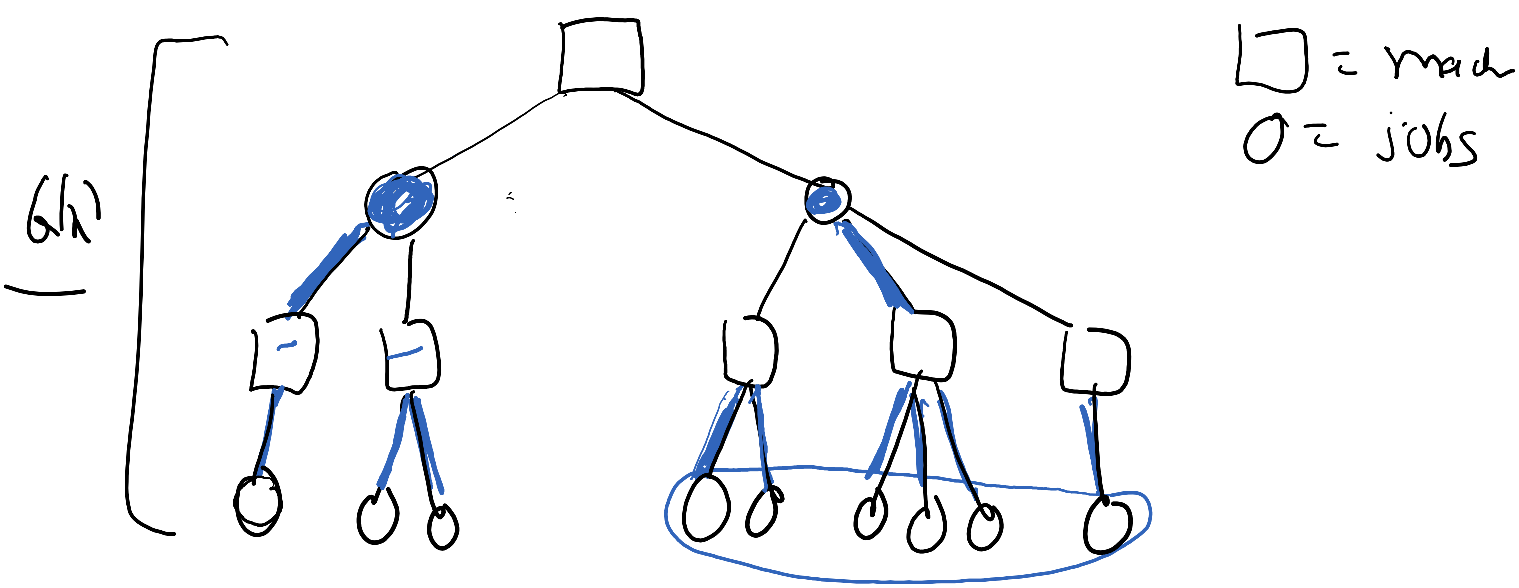
$L =$ optimum solution to GS: LP.

LOWER BOUNDS

#1) $L^* \geq L$ // does not suffice

#2) $L^* \geq \max_j t_j$

$G(x)$ is a tree.



1) assign all leaf jobs j to their

parent machines $P(j)$.

2) assign all internal jobs j to a child machine chosen arbitrarily.

Consider any machine i

x^*_{ij} denote the integer job assignm resulting from the rounding above

$$\sum_{j \in (m)} x^*_{ij} \le \underbrace{\sum_{j : j \in C(i)} x_{ij}}_{\le L} + \underbrace{t_{j'}}_{\le \max_j t_j}$$

$j' = P(i)$ in $G(x)$

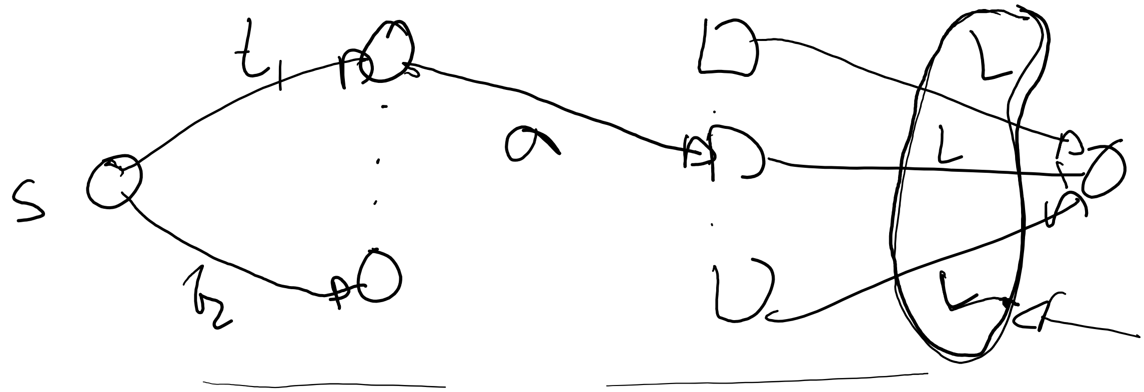
$$\le L + \max_j t_j$$

$$\le L^* + L^*$$

$$= 2L^*$$

2-appx-algorithm for Gen-Job Schedu

Solving LP with flow



$\log(\sum_j t_j)$ $\times$

1) binary search over L

   $0 \le L \le \sum_j t_j$

2) each fixed L, solve a flow-prob to check feasible