Advanced Algorithms (Fall 2024)

Linear Programming Rounding Algorithms

Lecturers: 尹一通，栗师，刘景铖

Nanjing University

Approximation Algorithm based on LP Rounding

- Opti. Problem $X \iff 0/1$ Integer Program (IP) $\xrightarrow{\text{relax}}$ LP

0/1 Integer Program

$$\begin{aligned} \min \quad & c^{\mathrm{T}} x \\ Ax & \geq b \\ x & \in \{0,1\}^n \end{aligned}$$

Linear Program Relaxation

$$\begin{aligned} \min \quad & c^{\mathrm{T}} x \\ Ax & \geq b \\ x & \in [0,1]^n \end{aligned}$$

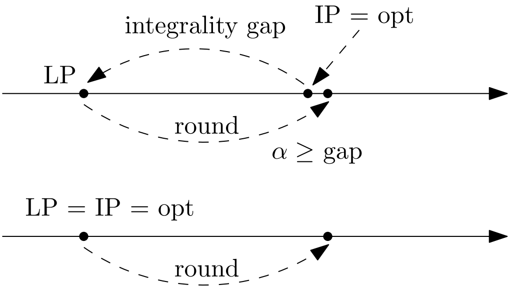
- LP $\leq$ IP
- Integer programming is NP-hard, linear programming is in P
- Solve LP to obtain a fractional $x \in [0,1]^n$.
- Round it to an integral $\tilde{x} \in \{0,1\}^n \iff$ solution for X
- Prove $c^{\mathrm{T}}\tilde{x} \leq \alpha \cdot c^{\mathrm{T}}x$, then $c^{\mathrm{T}} \cdot \tilde{x} \leq \alpha \cdot \mathrm{LP} \leq \alpha \cdot \mathrm{IP} = \alpha \cdot$ opt
- $\implies \alpha$-approximation

IP

$$\begin{aligned} \min \quad & c^{\mathrm{T}} x \\ Ax &\geq b \\ x &\in \{0,1\}^n \end{aligned}$$

LP Relaxation

$$\begin{aligned} \min \quad & c^{\mathrm{T}} x \\ Ax &\geq b \\ x &\in [0,1]^n \end{aligned}$$

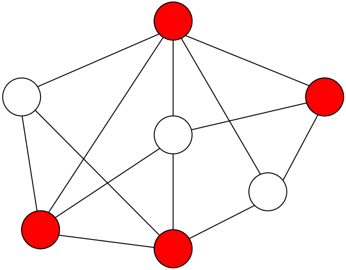


Def. The ratio between IP $=$ opt and LP is called the integrality gap of the LP relaxation.

- The approximation ratio based on this analysis can not be better than the worst integrality gap.

Outline

1. 2-Approximation Algorithm for Weighted Vertex Cover
2. 2-Approximation Algorithm for Unrelated Machine Scheduling
3. Congestion Minimization *



Weighted Vertex Cover Problem

Input: graph $G = (V, E)$, vertex weights $w \in \mathbb{Z}_{>0}^{V}$

Output: vertex cover S of G, to minimize $\sum_{v \in S} w_v$

- $x_v \in \{0,1\}, \forall v \in V$: indicate if we include v in the vertex cover

Integer Program

$$\min \quad \sum_{v \in V} w_v x_v$$

$$x_u + x_v \geq 1 \qquad \forall (u,v) \in E$$

$$x_v \in \{0,1\} \qquad \forall v \in V$$

LP Relaxation

$$\min \quad \sum_{v \in V} w_v x_v$$

$$x_u + x_v \geq 1 \qquad \forall (u,v) \in E$$

$$x_v \in [0,1] \qquad \forall v \in V$$

- IP := value of integer program, LP := value of linear program
- LP $\leq$ IP $=$ opt

Rounding Algorithm

1: Solve LP to obtain solution $\{x_u^*\}_{u \in V}$
$\triangleright$ So, LP $= \sum_{u \in V} w_u x_u^* \leq$ IP

2: **return** $S := \{u \in V : x_u \geq 1/2\}$

Lemma S is a vertex cover of G.

Proof.

- Consider any $(u, v) \in E$: we have $x_u^* + x_v^* \geq 1$
- So, $x_u^* \geq 1/2$ or $x_v^* \geq 1/2 \quad \Longrightarrow \quad u \in S$ or $v \in S$. $\square$

Rounding Algorithm

1: Solve LP to obtain solution $\{x_u^*\}_{u \in V}$ $\triangleright$ So, LP $= \sum_{u \in V} w_u x_u^* \leq$ IP

2: **return** $S := \{u \in V : x_u \geq 1/2\}$

Lemma S is a vertex cover of G.

Lemma $\text{cost}(S) := \sum_{u \in S} w_u \leq 2 \cdot \text{LP}$.

Proof.

$$\begin{aligned}\text{cost}(S) &= \sum_{u \in S} w_u \leq \sum_{u \in S} w_u \cdot 2x_u^* = 2 \sum_{u \in S} w_u \cdot x_u^* \\ &\leq 2 \sum_{u \in V} w_u \cdot x_u^* = 2 \cdot \text{LP}.\end{aligned}$$ $\square$

Theorem The algorithm is a 2-approximation algorithm for weighted vertex cover.

Outline

1. 2-Approximation Algorithm for Weighted Vertex Cover
2. 2-Approximation Algorithm for Unrelated Machine Scheduling
3. Congestion Minimization *

Unrelated Machine Scheduling

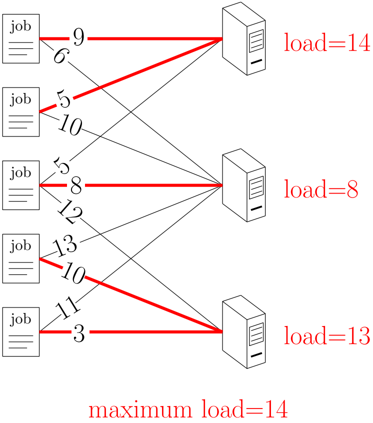
Input: $J, |J| = n$: jobs

$M, |M| = m$: machines

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

Output: assignment $\sigma : J \mapsto M$:, so as to minimize makespan:

$$\max_{i \in M} \sum_{j \in \sigma^{-1}(i)} p_{ij}$$

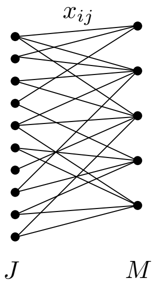


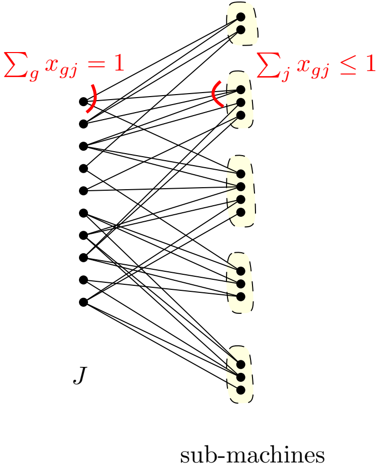
- Assumption: we are given a target makespan T, and $p_{ij} \in [0, T] \cup \{\infty\}$

- x_{ij}: fraction of j assigned to i

$$\begin{aligned} \sum_i x_{ij} &= 1 && \forall j \in J \\ \sum_j p_{ij} x_{ij} &\le T && \forall i \in M \\ x_{ij} &\ge 0 && \forall ij \end{aligned}$$

2-Approximate Rounding Algorithm of Shmoys-Tardos



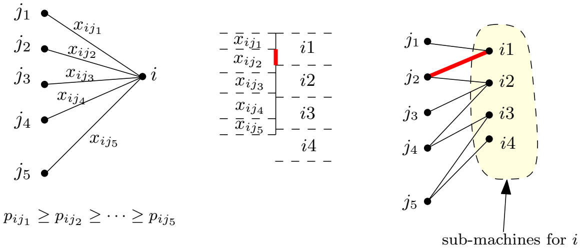


Obs. x between J and sub-machines is a point in the bipartite-matching polytope, where all jobs in J are matched.

- Recall bipartite matching polytope is integral.
- x is a convex combination of matchings.
- Any matching in the combination covers all jobs J.

Lemma Any matching in the combination gives an schedule of makespan $\leq 2T$.

Lemma Any matching in the combination gives an schedule of makespan $\le 2T$.



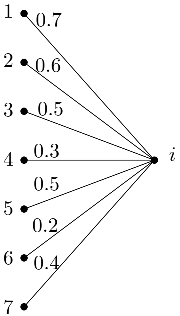
Proof.

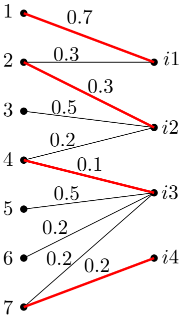
- focus on machine i, let $i_1, i_2, \cdots, i_a$ be the sub-machines for i
- assume job k_t is assigned to sub-machine i_t.

$$\text{(load on } i) = \sum_{t=1}^{a} p_{ik_t} \le p_{ik_1} + \sum_{t=2}^{a} \sum_{j} x_{i_{t-1}j} \cdot p_{ij}$$

$$\le p_{ik_1} + \sum_{j} x_{ij} p_{ij} \le T + T = 2T. \qquad \square$$

- fix i, use p_j for p_{ij}
- $p_1 \geq p_2 \geq \cdots \geq p_7$
- worst case:
  - $1 \to i1, 2 \to i2$
  - $4 \to i3, 7 \to i4$





$$p_1 \leq T$$

$$p_2 \leq 0.7p_1 + 0.3p_2$$

$$p_4 \leq 0.3p_2 + 0.5p_3 + 0.2p_4$$

$$p_7 \leq 0.1p_4 + 0.5p_5 + 0.2p_6 + 0.2p_7$$

$$\begin{aligned} p_1 + p_2 + p_4 + p_7 &\leq T + (0.7p_1 + 0.3p_2) + (0.3p_2 + 0.5p_3 + 0.2p_4) \\ &\quad + (0.1p_4 + 0.5p_5 + 0.2p_6 + 0.2p_7) \\ &\leq T + (0.7p_1 + 0.6p_2 + 0.5p_3 + 0.3p_4 + 0.5p_5 + 0.2p_6 + 0.4p_7) \\ &\leq T + T = 2T \end{aligned}$$

Outline

1. 2-Approximation Algorithm for Weighted Vertex Cover
2. 2-Approximation Algorithm for Unrelated Machine Scheduling
3. Congestion Minimization *

Congestion Minimization

Input: directed graph $G = (V, E)$

k pairs of vertices $(s_1, t_1), (s_2, t_2), \cdots, (s_k, t_k)$

Output: find k paths: P_1 from s_1 to t_1, P_2 from s_2 to t_2, $\cdots$, P_k from S_k to t_k.

cong$(e) := |\{i \in [k] : e \in P_i\}|$.

goal: minimize $\max_{e \in E}$ cong(e)

Q: What if $s_i = s$ for every $i \in [k]$?

A: (Single Source Single Sink) maximum flow problem. Can be solved exactly in polynomial time.

Linear Programming

- $\mathcal{P}_i$: set of paths from s_i to t_i
  assume terminals are distinct
- $\mathcal{P} := \bigcup_{i \in [k]} \mathcal{P}_i$

- $x_{i,e}, i \in [k], e \in E$:
  whether the path P_i uses the edge e or not

Exponential Size LP

$$
\begin{aligned}
&\min \quad C \\
&\sum_{P \in \mathcal{P}_i} x_P = 1 && \forall i \in [k] \\
&C \geq \sum_{P \in \mathcal{P}} x_{\mathcal{P}} && \forall e \in E \\
&x_P \geq 0 && \forall P \in \mathcal{P} \\
&C \geq 1
\end{aligned}
$$

Compact LP

$$
\begin{aligned}
&\min \quad C \\
&C \geq \sum_{i=1}^{k} x_{i,e} \qquad \forall e \in E \\
&C \geq 1
\end{aligned}
$$

(*): $\forall i \in [k]$: capacities $(x_{i,e})_{e \in E}$ support 1 unit flow from s_i to t_i

Equivalent Polynomial-Sized LP

- (*) can be checked using ellipsoid method, or the following LP network flow

Constraints (*) for a fixed i

$$\sum_{e \in \delta^{\text{out}}(v)} f_{i,e} - \sum_{e \in \delta^{\text{in}}(v)} f_{i,e} = \begin{cases} 1 & v = s_i \\ -1 & v = t_i \\ 0 & v \in V \setminus \{s_i, t_i\} \end{cases}$$

$$f_{i,e} \in [0, x_{i,e}], e \in E$$

Lemma The Exponential-Size LP and the Compact LP for congestion minimization are equivalent.

- Easy direction:
  solution for exponential-size LP $\implies$ solution for compact LP

Hard Direction: Solution for Compact LP $\Longrightarrow$ Solution for Exponential-Size LP

- (*) is feasible: in the digraph G with source s_i, sink t_i and edge capacities $x_{i,e}$, the maximum flow has value at least 1.
- We can find $(y_P \geq 0)_{P \in \mathcal{P}_i}$ such that

$$\sum_{P \in \mathcal{P}_i : P \ni i} \leq x_{i,e}, \forall e \in E \qquad \text{and} \qquad \sum_{P \in \mathcal{P}_i} y_P = 1$$

- $(y_P)_{P \in \mathcal{P}}$ is a solution for exponential size LP.
- We assume we are given $(y_P)_{P \in \mathcal{P}}$, using the sparse representation.

Rounding Algorithm

for every $i \leftarrow 1$ to k **do**

  independently and randomly choose P_i so that

$$\Pr[P_i = P] = x_P, \forall P \in \mathcal{P}_i.$$

return $P_1, P_2, \cdots, P_k$

Analysis for a fixed $e \in E$

- $\Pr[e \in P_i] = x_{i,e} := \sum_{P \in \mathcal{P}_i : P \ni e} x_P$
- $\sum_{i \in [k]} x_{i,e} \leq C$
- Let $X_i \in \{0, 1\}$ indicate if $e \in P_i$
- $\mathbb{E}[X_i] = x_{i,e}$
- $\mathsf{cong}(e) = \sum_{i \in [k]} X_i$

Using Chernoff Bound:

$$\Pr\left[\sum_{i\in[k]} X_i \geq (1+\delta)C\right] \leq \left(\frac{e^\delta}{(1+\delta)^{1+\delta}}\right)^C$$
$$\leq \frac{e^\delta}{(1+\delta)^{1+\delta}} \qquad \text{since } C \geq 1$$

- We need to choose a large enough δ so that $\frac{e^\delta}{(1+\delta)^{1+\delta}} \leq \frac{1}{2n^2}$, how big should δ be?
- To get an estimate, we replace e^δ with 1, and $1+\delta$ with δ
- So, we need $\frac{1}{\delta^\delta} = \frac{1}{2n^2}$.
- $\delta = O(\frac{\log n}{\log\log n})$ suffices.

- For some $\delta = O(\frac{\log n}{\log\log n})$, we have $\Pr[\mathsf{cong}(e) \geq (1+\delta)C] \leq \frac{1}{2n^2}$.
- Using Union Bound over all edges $e \in E$

$$\Pr[\exists e \in E, \mathsf{cong}(e) \geq (1+\delta)C] \leq \frac{1}{2n^2} \cdot m \leq \frac{1}{2}$$
$$\Pr[\forall e \in E, \mathsf{cong}(e) < (1+\delta)C] \geq 1 - \frac{1}{2} = \frac{1}{2}$$

- Remarks: the approximation ratio is as bad as $O(\frac{\log n}{\log \log n})$ only when C is a constant.
- As C becomes bigger, the ratio becomes better.
- If $C = \Theta(\log n)$, then the approximation ratio can be $O(1)$.

- The algorithm can be derandomized using the idea of conditional expectation.

Summary

- 2-approximation algorithm for weighted vertex cover
- 2-approximation for unrelated machine scheduling
- $O\left(\frac{\log n}{\log \log n}\right)$-approximation for congestion minimization