Cha 11 近似算法

苏 明

近似算法

- 对那些多项式时间难以达到目标的问题，NP完全问题如何处理？
- **近似算法**

✓ 在多项式时间内运行

✓ 找到接近最优的解；近似比

负载均衡问题

问题描述：

- 给定m台机器，$M_1, \ldots, M_m$ 以及n项作业，每一项作业j有处理时间t_j,要把每一项作业分配给一台机器，使得所有机器的负载尽可能“均衡”.
- 假设A(i)表示分配给机器M_i的作业集，机器M_i工作时间 $T_i = \sum_{j \in A(i)} t_j$

负载均衡问题

- 我们关注$T = \max_i T_i$ 最小
- 关于最小工期分配的问题是*NP*难问题
- 近似算法

设计算法

考虑**贪心算法**

- 与大家排队打饭的情形类似
- 以任意的顺序枚举所有的作业，轮到作业j时，将其分配到**至今负载最小**的机器

设计算法

- List-scheduling algorithm.
  - n个作业按照某种次序.
  - 第j 个作业分配到至今负载最小的机器

```
List-Scheduling(m, n, t1,t2,…,tn) {
   for i = 1 to m {
      Li ← 0            ← load on machine i
      J(i) ← ϕ          ← jobs assigned to machine i
   }

   for j = 1 to n {
      i = argmink Lk           ← machine i has smallest load
      J(i) ← J(i) ∪ {j}        ← assign job j to machine i
      Li ← Li + tj             ← update load of machine i
   }
}
```

采用优先队列实现，复杂度？ O(nlogm)

分析算法

- 这个算法好到什么程度?
- 假设存在最优值T*, 我们的解与T*比较怎样?

- Theorem. [Graham, 1966] Greedy algorithm is a 2-approximation.

分析算法

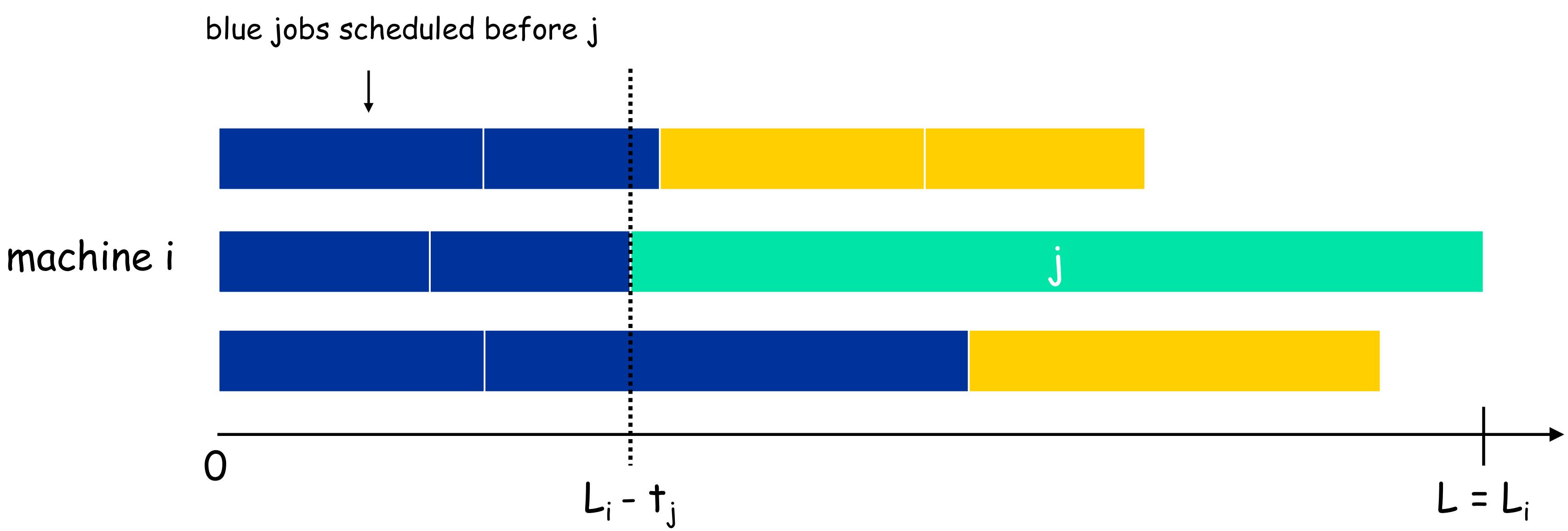
- Lemma 1. The optimal makespan $L^* \geq \max_j t_j$.

- Lemma 2. The optimal makespan

$$L^* \geq \frac{1}{m}\sum_j t_j.$$

分析算法

- Theorem. Greedy algorithm is a 2-approximation.
- Pf. Consider load L_i of bottleneck machine i.
  - Let j be last job scheduled on machine i.
  - When job j assigned to machine i, i had smallest load. Its load before assignment is $L_i - t_j \Rightarrow L_i - t_j \leq L_k$ for all $1 \leq k \leq m$.



分析算法

Sum inequalities over all k and divide by m:

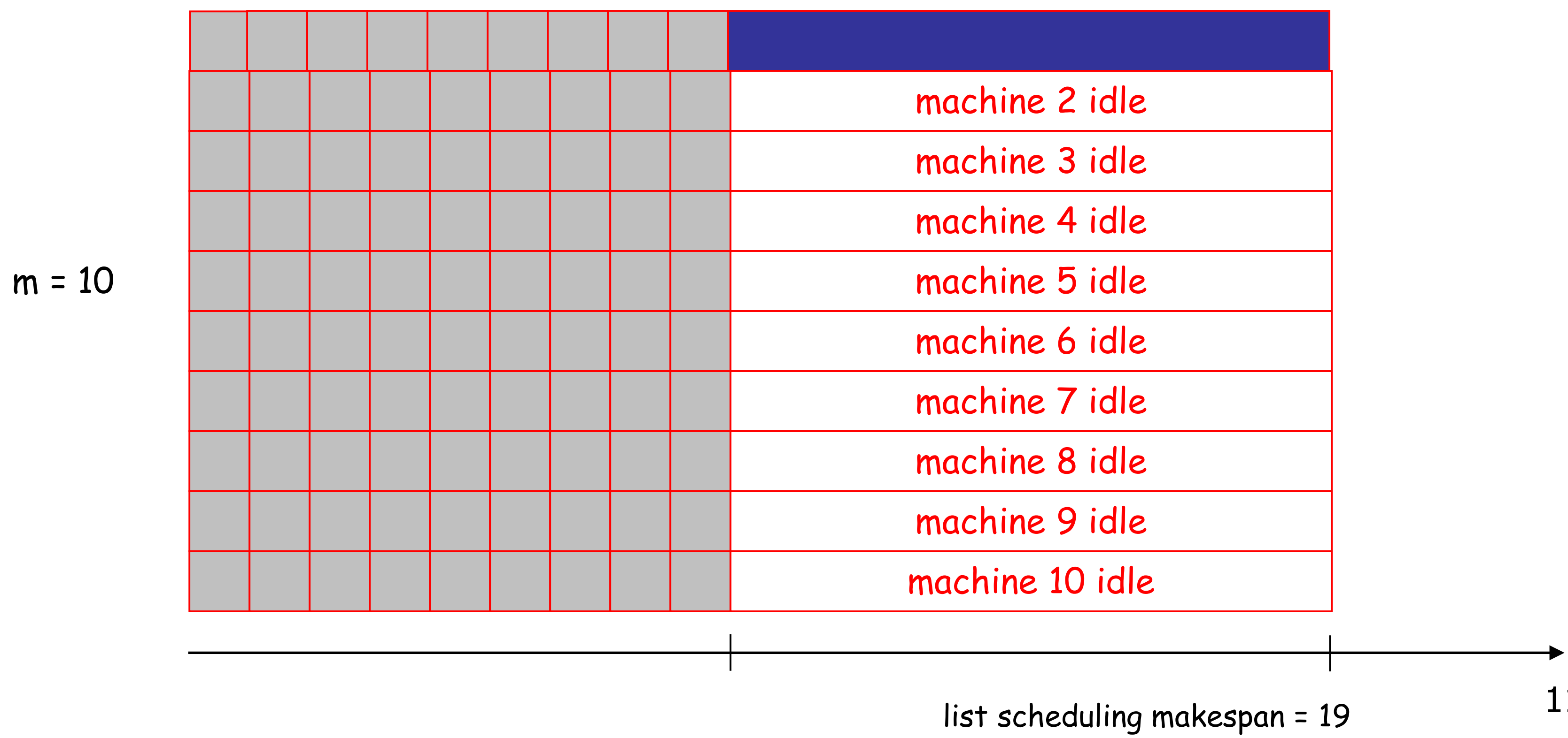
$$\begin{aligned} L_i - t_j &\le \frac{1}{m}\sum_k L_k \\ &= \frac{1}{m}\sum_k L_k \\ \text{Lemma 1} \longrightarrow &\le L^* \end{aligned}$$

$$L_i = \underbrace{(L_i - t_j)}_{\le L^*} + \underbrace{t_j}_{\le L^*} \le 2L^*.$$

Lemma 2

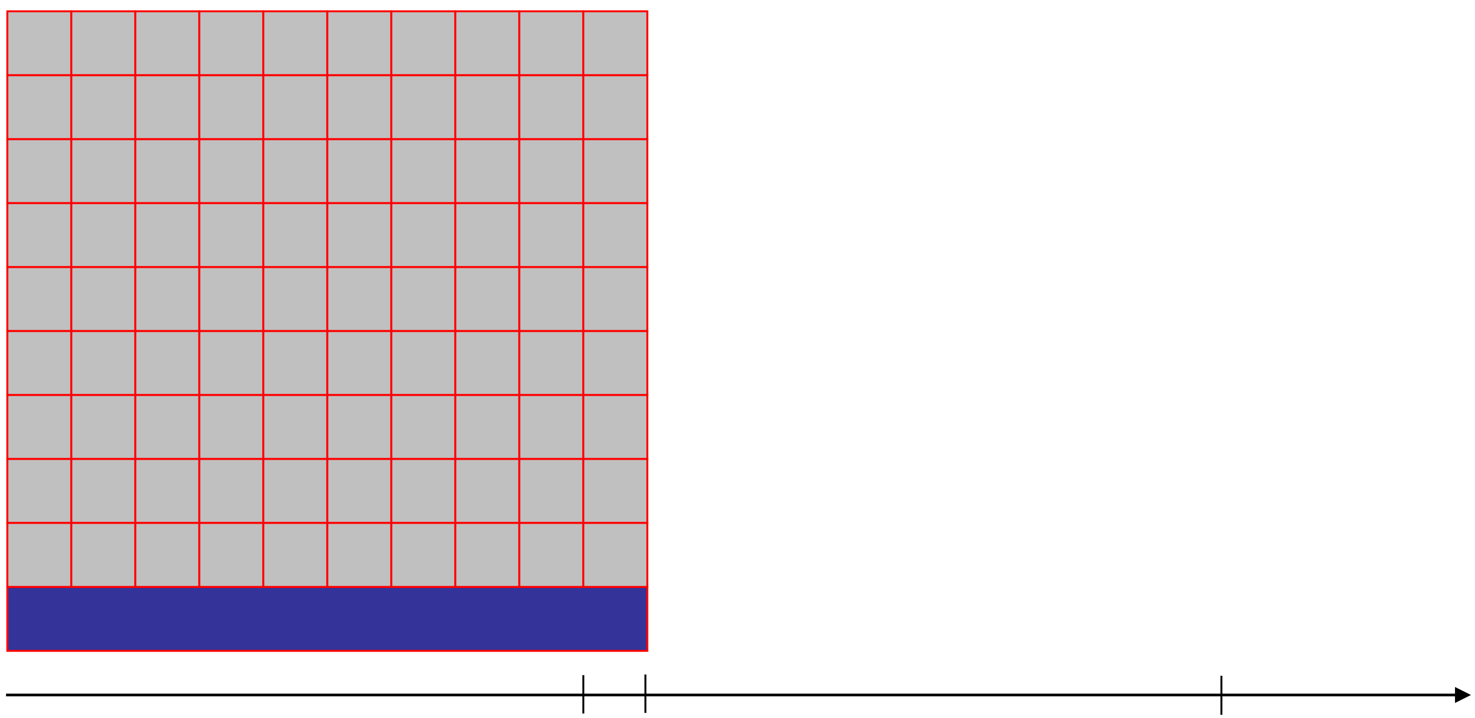
分析算法

- 这个结果紧吗？　　很紧
- Ex: m machines, m(m-1) jobs length 1 jobs, one job of length m



分析算法

最优解：m machines, m(m-1) jobs length 1 jobs, one job of length m



m = 10

optimal makespan = 10

更好近似比的算法

- Any Improvement?
- 原来算法以任意的次序枚举所有的作业
- Sort n jobs in descending order of processing time, and then run list scheduling algorithm.

更好近似比的算法

```
LPT-List-Scheduling(m, n, t₁,t₂,…,tₙ) {
   Sort jobs so that t₁ ≥ t₂ ≥ … ≥ tₙ

   for i = 1 to m {
      Lᵢ ← 0            ← load on machine i
      J(i) ← φ          ← jobs assigned to machine i
   }

   for j = 1 to n {
      i = argmin_k L_k      ← machine i has smallest load
      J(i) ← J(i) ∪ {j}     ← assign job j to machine i
      Lᵢ ← Lᵢ + tⱼ          ← update load of machine i
   }
}
```

算法分析

- 引理11.4 如果有多于m项作业，则$L^* \geq 2\, t_{m+1}$.

- 定理11.5 算法Sorted-Balance产生一个分配方案且满足L<=3/2 L*.

Pf.

$$L_i = \underbrace{(L_i - t_j)}_{\leq L^*} + \underbrace{t_j}_{\leq \frac{1}{2}L^*} \leq \frac{3}{2}L^*.$$

by observation, can assume number of jobs > m

算法分析

- 3/2并不是紧的
- 4/3是紧的
- 例子：m台机器， n=2m+1任务，其中：
  长度为m+1,m+2,...,2m的任务有2个，一个任务的长度为m.

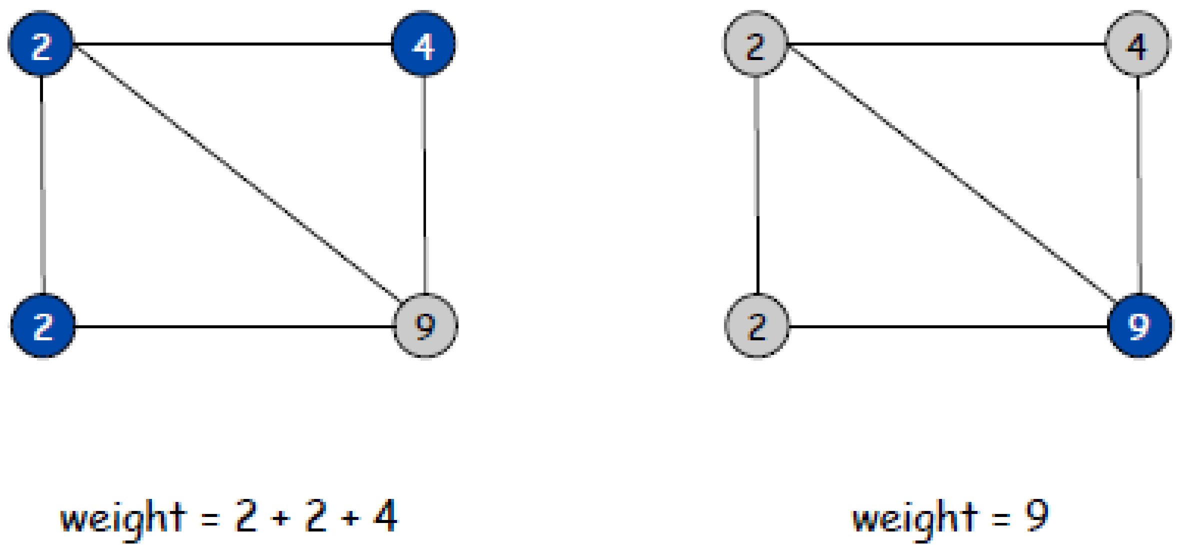
定价法：顶点覆盖

- 图G=(V,E)中的**顶点覆盖**是一个集合$S\subseteq V$,使得每一条边至少有一个端点在S中。假设每一个顶点 i 有一个权$w_i>=0$, 顶点集合S的权记为 $w(S)=\sum_{i\in S} w_i$, 我们希望找到一个**w(s)最小**的**顶点覆盖**S.

当所有权等于1时，确定是否有权重不超过k的顶点覆盖是顶点覆盖的标准判定形式。

顶点覆盖

- Example



顶点覆盖

定价法

- 为每一条边e∈E确定价格p_e>=0.
- 考虑到某种自然的公平规则，如果对每一个顶点i,所有与i关联的边不必支付多于顶点i的费用，即 $\sum_{e=(i,j)} p_e \le w_i$ ，则称价格p_e是公平的。

顶点覆盖

- 命题11.13 对任意的顶点覆盖S*和任意非负的公平价格p_e, 有$\sum_e p_e \le w(S^*)$.

Pf.

$$\sum_{e\in E} p_e \;\le\; \sum_{i\in S^*} \sum_{e=(i,j)} p_e \;\le\; \sum_{i\in S^*} w_i \;=\; w(S^*).$$

each edge e covered by at least one node in S*

sum fairness inequalities for each node in S*

近似算法

- 找到一个顶点覆盖并且同时确定价格
- 如果 $\sum_{e=(i,j)} p_e = w_i$，则称顶点i是紧的(付清的)

近似算法

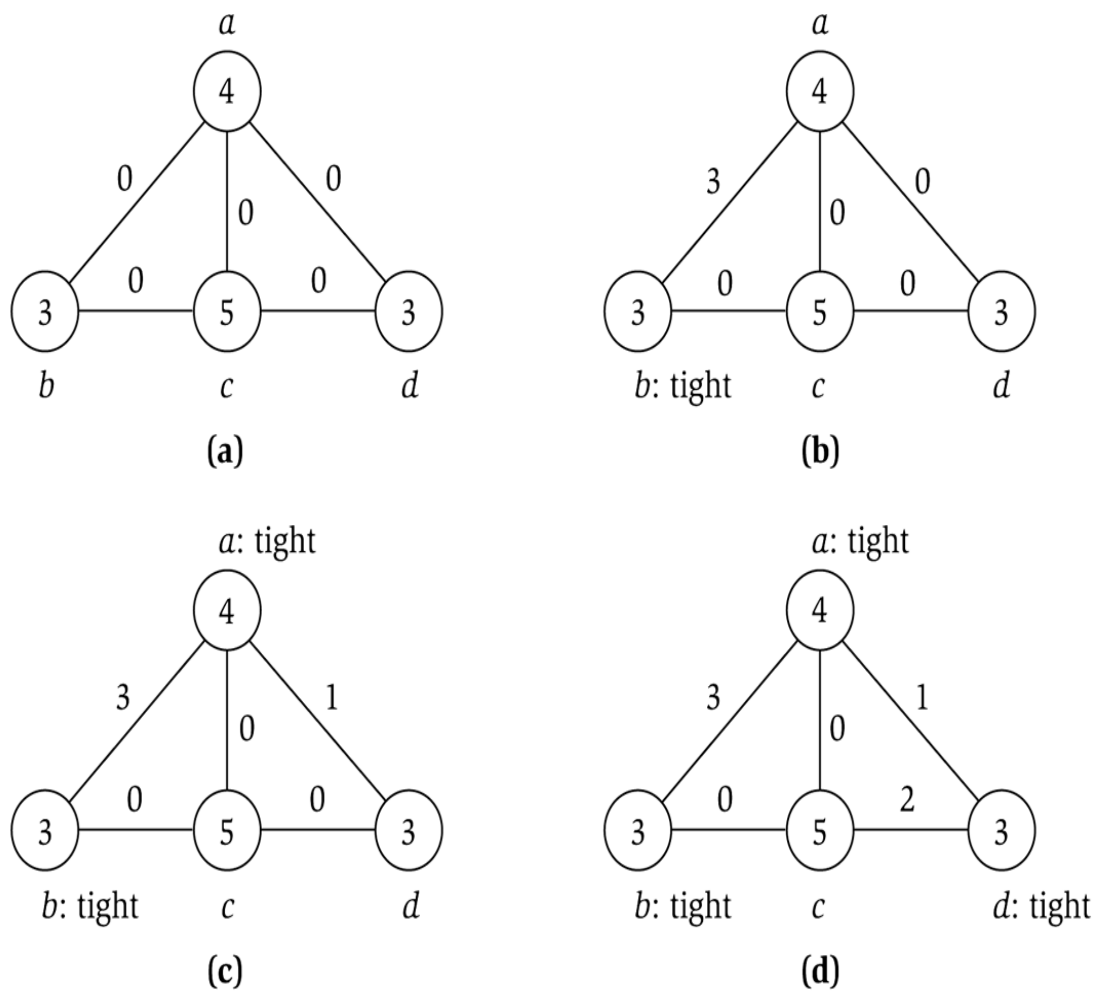
```
Weighted-Vertex-Cover-Approx(G, w) {
   foreach e in E
      pe = 0

   while (∃ edge i-j such that neither i nor j are tight)
      select such an edge e
      increase pe without violating fairness
   }

   S ← set of all tight nodes
   return S
}
```

$$\sum_{e=(i,j)} p_e = w_i$$

近似算法



a
4
0
0
0
3
0
5
0
3
b
c
d
(a)
a
4
3
0
0
3
0
5
0
3
b: tight
c
d
(b)
a: tight
4
3
0
1
3
0
5
0
3
b: tight
c
d
(c)
a: tight
4
3
0
1
3
0
5
2
3
b: tight
c
d: tight
(d)

近似算法

正确性

- 首先，算法返回的S的确是一个顶点覆盖，若不然，假设S不覆盖边e=(i,j),这可以推出i和j都不是紧的，与算法While循环终止条件矛盾。
- 近似比？

近似算法

- Theorem. Pricing method is a 2-approximation.
- Pf.
  - Let S* be optimal vertex cover. We show w(S) ≤ 2w(S*).

$$w(S) = \sum_{i \in S} w_i \underset{\text{all nodes in S are tight}}{=} \sum_{i \in S} \sum_{e=(i,j)} p_e \underset{\substack{S \subseteq V,\\ \text{prices} \geq 0}}{\leq} \sum_{i \in V} \sum_{e=(i,j)} p_e \underset{\text{each edge counted twice}}{=} 2 \sum_{e \in E} p_e \underset{\text{fairness lemma}}{\leq} 2w(S^*). \quad \blacksquare$$

线性规划与舍入：顶点覆盖

线性规划(Linear Programming)

- Max/min linear objective function subject to linear inequalities.
  - Input: integers c_j, b_i, a_{ij} .
  - Output: real numbers x_j.

线性规划

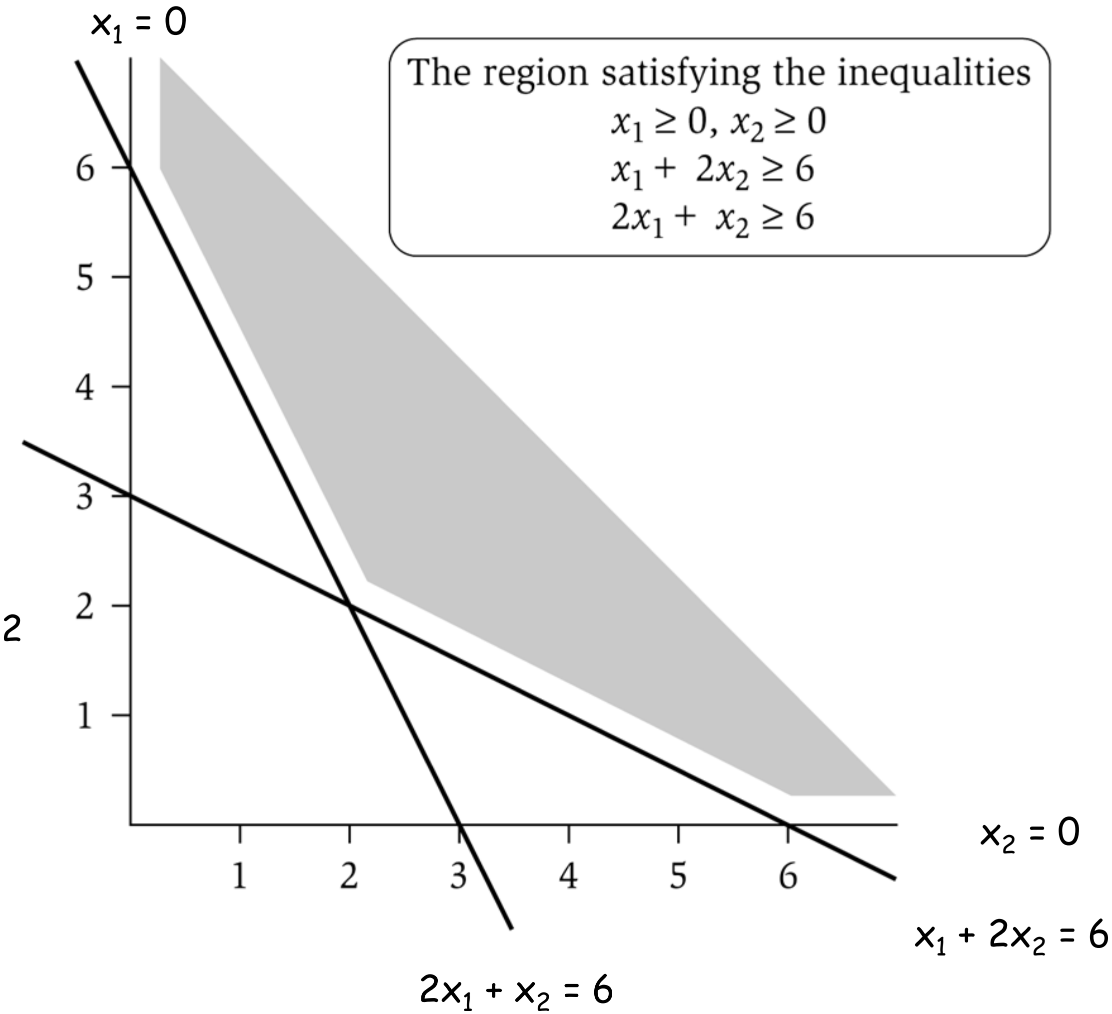
$$
\begin{aligned}
\text{(P)} \quad \max \quad & c^t x \\
\text{s. t.} \quad & Ax \leqslant b \\
& x \geq 0
\end{aligned}
$$

$$
\begin{aligned}
\text{(P)} \quad \text{Min} \quad & \sum_{j=1}^{n} c_j x_j \\
\text{s. t.} \quad & \sum_{j=1}^{n} a_{ij} x_j \geq b_i \quad 1 \leq i \leq m \\
& x_j \geq 0 \quad 1 \leq j \leq n
\end{aligned}
$$

- Simplex algorithm(单纯形法). [Dantzig 1947]
  Can solve LP in practice.
- Ellipsoid algorithm(内点法). [Khachian 1979]
  Can solve LP in poly-time.

线性规划

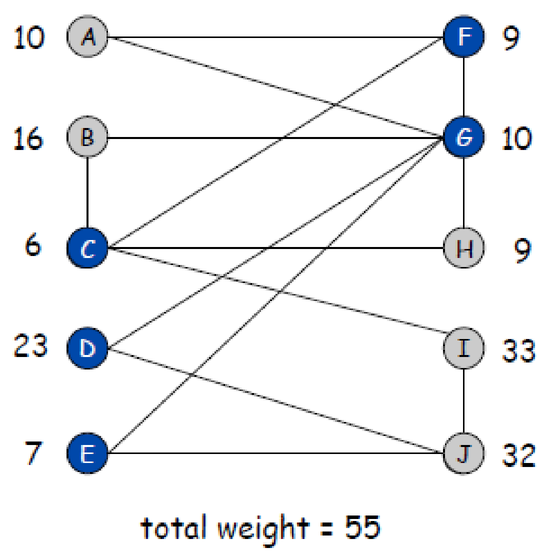
例子



Min: $1.5x_1 + x_2$

顶点覆盖

图G=(V,E)中的**顶点覆盖**是一个集合S V, 使得每一条边至少有一个端点在S中。假设每一个顶点 i 有一个权w_i>=0, 顶点集合S的 权记为 $w(S)=\sum_{i\in S} w_i$, 我们希望找到一个**w(s)最小**的顶点覆盖S.



顶点覆盖

与线性规划的联系？

- Model inclusion of each vertex i using a 0/1 variable x_i.

$$x_i = \begin{cases} 0 & \text{if vertex } i \text{ is not in vertex cover} \\ 1 & \text{if vertex } i \text{ is in vertex cover} \end{cases}$$

- Vertex covers in 1-1 correspondence with 0/1 assignments:
  $S = \{i \in V : x_i = 1\}$
- Objective function: minimize $\Sigma_i w_i x_i$.
- Must take either i or j: $x_i + x_j \geq 1$.

顶点覆盖

- Weighted vertex cover VS 整数规划.

$$
\begin{array}{llllll}
(ILP) & \min & \sum\limits_{i \in V} w_i x_i & & & \\
& \text{s. t.} & x_i + x_j & \geq & 1 & (i,j) \in E \\
& & x_i & \in & \{0,1\} & i \in V
\end{array}
$$

- Observation. If x* is optimal solution to (ILP), then S = {i $\in$ V : x^*_i = 1} is a min weight vertex cover.

顶点覆盖

- 定理11.22 顶点覆盖$\leq_P$整数规划。

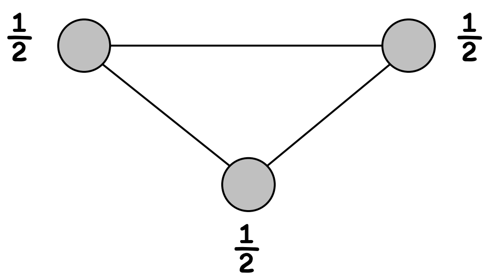
- 问题的转换：把x_i看成 0到1之间的 任意实数：我们可以在 多项式时间内求解。

- 命题11.23 设S*是一个**权最小的顶点覆盖**，那么$w_{LP} <= w(S^*)$.

Pf. 因为x可以取实数值,可在更多选择上最小化

顶点覆盖

同时也意识到：



$$
\begin{array}{llccl}
(LP)\ \min & \sum_{i \in V} w_i x_i & & & \\
\text{s. t.} & x_i + x_j & \geq & 1 & (i,j) \in E \\
 & x_i & \geq & 0 & i \in V
\end{array}
$$

What is the solution?

顶点覆盖

- **取整**：规定$S = \{i \in V : x^*_i \geq \frac{1}{2}\}$

- 命题11.24 这样规定的集合S是一个顶点覆盖，而且$w(S) \le 2w_{LP}$.
- Pf. **[S is a vertex cover]**
  - Consider an edge $(i, j) \in E$.
  - Since $x^*_i + x^*_j \geq 1$, either $x^*_i \geq \frac{1}{2}$ or $x^*_j \geq \frac{1}{2}$ $\Rightarrow$ (i, j) covered.

顶点覆盖

- Pf. **[w(S)<=2w$_{LP}$]**

$$w_{LP} = \sum_i w_i x_i^* \ge \sum_{i \in S} w_i x_i^* \ge \frac{1}{2} \sum_{i \in S} w_i = \frac{1}{2} w(S)$$

- **定理 11.25** 取整的**LP**算法产生一个顶点覆盖**S, w(S)<=2w(S*).**

扩展

- Theorem. [Dinur-Safra 2001] If P $\neq$ NP, then no ρ-approximation for $\rho < 1.3607$, even with unit weights.

- 对不同问题，最好近似算法的近似比不一样