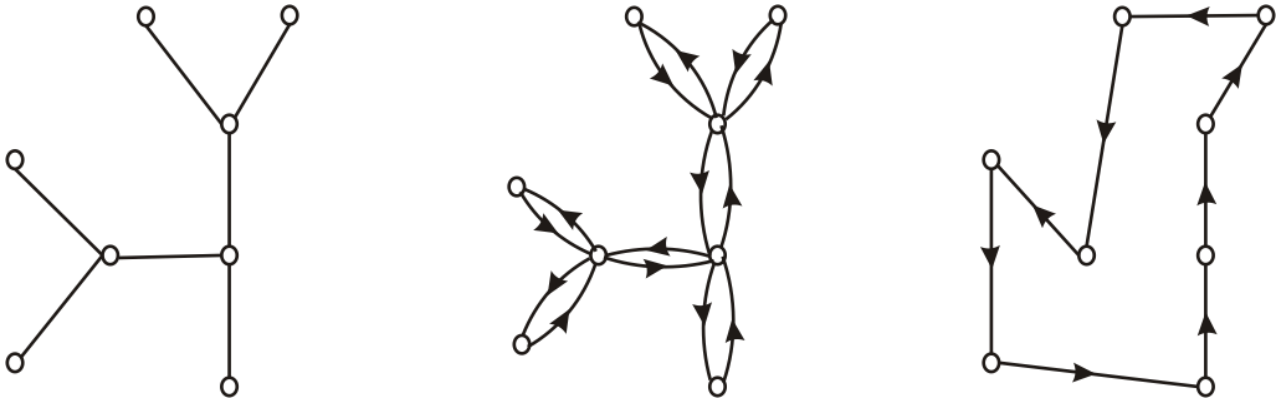
最小生成树法

针对满足三角不等式的货郎问题

最小生成树法MST: 首先, 求图的一棵最小生成树T. 然后, 沿着 T 走两遍得到图的一条欧拉回路. 最后, 顺着这条欧拉回路, 跳过已走过的顶点, 抄近路得到一条哈密顿回路.

例



求最小生成树和欧拉回路都可以在多项式时间内完成, 故算法是多项式时间的.

最小生成树法的性能

定理 对货郎问题的所有满足三角不等式的实例 I,

$$\mathrm{MST}(I) < 2\mathrm{OPT}(I)$$

证 因为从哈密顿回路中删去一条边就得到一棵生成树, 故 T 的权小于$\mathrm{OPT}(I)$. 沿 T 走两遍的长小于$2\mathrm{OPT}(I)$. 因为满足三角不等式, 抄近路不会增加长度, 故

$$\mathrm{MST}(I) < 2\mathrm{OPT}(I).$$

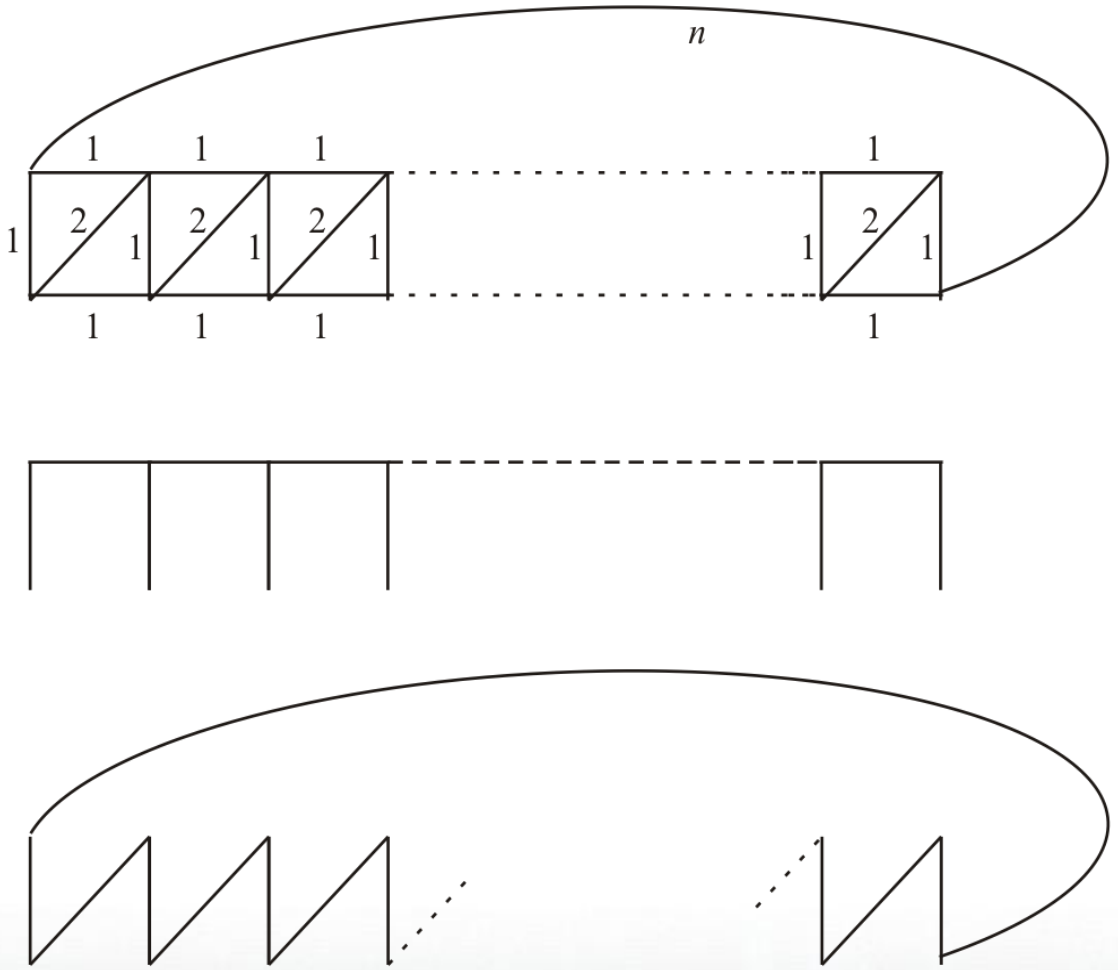
MST是2-近似算法.

紧实例

$\mathbf{OPT}(I) = 2n$

$\mathbf{MST}(I) = 4n-2$

$$= \left(2 - \frac{1}{n}\right) \mathbf{OPT}(I)$$

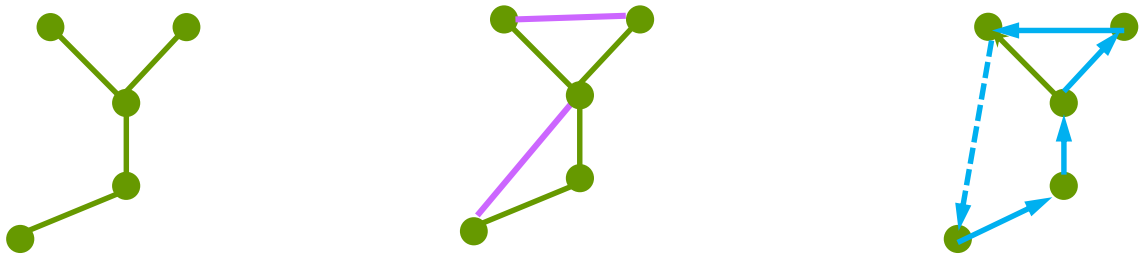


最小权匹配法

最小权匹配法 MM:

首先求图的一棵最小生成树T.

记 T 的所有奇度顶点在原图中的导出子图为 H, H 有偶数个顶点, 求 H 的最小匹配 M. 把 M 加入 T 得到一个欧拉图, 求这个欧拉图的欧拉回路; 最后, 沿着这条欧拉回路, 跳过已走过的顶点, 抄近路得到一条哈密顿回路.



求任意图最小权匹配的算法是多项式时间的, 因此 MM 是多项式时间的.

最小权匹配法的性能

定理 对货郎问题的所有满足三角不等式的实例 I,

$$\mathrm{MM}(I) < \frac{3}{2}\mathrm{OPT}(I)$$

证 由于满足三角不等式, 导出子图 H 中的最短哈密顿回路 C 的长度不超过原图中最短哈密顿回路的长度 $\mathrm{OPT}(I)$. 沿着C 隔一条边取一条边, 得到 H 的一个匹配. 总可以使这个匹配的权不超过 C 长的一半. 因此, H 的最小匹配 M 的权不超过 $\mathrm{OPT}(I)/2$, 求得的欧拉回路的长小于 $(3/2)\mathrm{OPT}(I)$. 抄近路不会增加长度, 得证

$$\mathrm{MM}(I) < (3/2)\mathrm{OPT}(I).$$

MM是3/2 -近似算法

货郎问题的难度

定理 货郎问题(不要求满足三角不等式) 是不可近似的, 除非 **P = NP.**

证 假设不然, 设A是货郎问题的近似算法, 其近似比 $r \le K$, K是常数. 任给图 $G = <V,E>$, 如下构造货郎问题的实例 I_G:
城市集V, $\forall u,v \in V$,

若 $(u,v) \in E$, 则令 $d(u,v)=1$; 否则令 $d(u,v)=Kn$,

其中 $|V| = n$. 若G 有哈密顿回路, 则

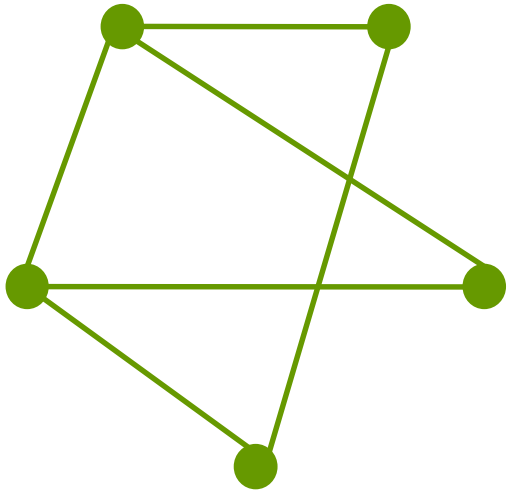
$$\mathrm{OPT}(I_G) = n, \quad \mathrm{A}(I_G) \le r\,\mathrm{OPT}(I_G) \le Kn$$

否则

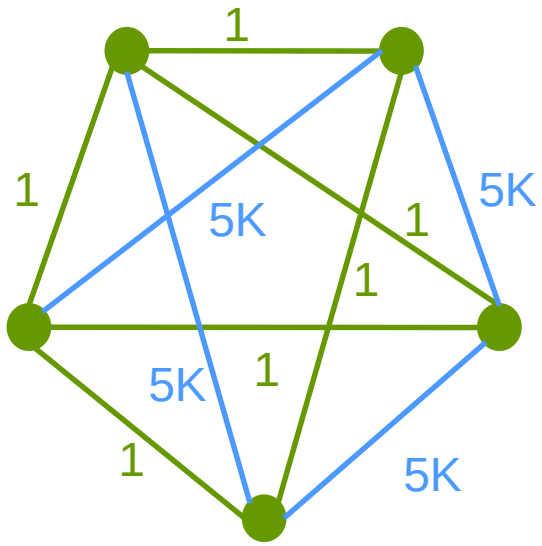
$$\mathrm{OPT}(I_G) > Kn, \quad \mathrm{A}(I_G) \ge \mathrm{OPT}(I_G) > Kn$$

所以, G 有哈密顿回路当且仅当 $\mathrm{A}(I_G) \le Kn$

实例



HC



货郎问题

HC实例有哈密顿回路，则货郎问题实例有长度不超过 *n* 的旅行，即近似算法的解不超过 *Kn*. 若HC实例没有哈密顿回路，则货郎实例的旅行路线长度大于 *Kn*.

证明

下述算法可以判断图 G 是否有哈密顿回路.

算法

1. 由 I 构造货郎问题的实例 I_G
2. 对I_G 运用近似算法 A
3. 若A(I_G) $\le Kn$, 则输出“Yes”；否则输出“No”

由于K是固定的常数, 构造 I_G 可在 $O(n^2)$ 时间内完成且 $|I_G| = O(n^2)$. A是多项式时间的， A 对 I_G 可在 n 的多项式时间内完成计算, 所以上述算法是 HC 的多项式时间算法. 而 HC 是 NP 完全的， 推得 P=NP.

0-1背包问题

0-1背包问题的优化形式:

任给 n 件物品和一个背包，物品 i 的重量为 w_i，价值为 v_i，$1 \le i \le n$，背包的重量限制为 B，其中 w_i，v_i 以及 B 都是正整数.

把哪些物品装入背包才能在不超过重量限制的条件下使得价值最大？即，求子集 $S^* \subseteq \{1,2,\ldots,n\}$ 使得

$$\sum_{i \in S^*} v_i = \max\left\{ \sum_{i \in S} v_i \mid \sum_{i \in S} w_i \le B, S \subseteq \{1,2.\cdots,n\} \right\}.$$

一个简单的贪心算法

贪心算法G-KK

1. 按单位重量的价值从大到小排列物品. 设

$$v_1/w_1 \geq v_2/w_2 \geq \ldots \geq v_n/w_n.$$

2. 顺序检查每一件物品, 只要能装得下就将它装入背包, 设装入背包的总价值为 V.
3. 求 $v_k = \max\{ v_i \mid i = 1, 2, \ldots, n \}$.
   若$v_k > V$, 则将背包内的物品换成物品 k.

实例 (w_i,v_i): (3,7), (4,9), (5,9), (2,2); B=6.

G-KK给出的解是装入(3,7)和(2,2), 总价值为9.

若把第3件物品改为(5,10), 则装入第3件, 总价值为10.

这两个实例的最优解都是装入(4,9)和(2,2), 总价值为11.

G-KK的性能

定理 对0-1背包问题的任何实例 I, 有

$$\mathrm{OPT}(I) < 2\,\mathrm{G\text{-}KK}(I).$$

证 设物品 l 是第一件未装入背包的物品, 由于物品按单位重量的价值从大到小排列, 故有

$$\begin{aligned}\mathrm{OPT}(I) &< \mathrm{G\text{-}KK}(I) + v_l \\ &\le \mathrm{G\text{-}KK}(I) + v_{\max} \\ &\le 2\,\mathrm{G\text{-}KK}(I).\end{aligned}$$

G-KK是2-近似算法.

多项式时间近似方案

算法 PTAS 输入 $\varepsilon > 0$和实例 I.

1. 令 $m = \lceil 1/\varepsilon \rceil$.
2. 按单位重量的价值从大到小排列物品. 设

$$v_1/w_1 \geq v_2/w_2 \geq \ldots \geq v_n/w_n.$$

3. 对每一个 t =1,2,…, m 和 t 件物品, 检查这 t 件物品的重量之和. 若它们的重量之和不超过B, 则接着用 G-KK把剩余的物品装入背包.
4. 比较得到的所有装法, 取其中价值最大的作为近似解.

PTAS是一簇算法. 对每一个固定的 $\varepsilon > 0$, PTAS是一个算法, 记作 PTAS_ε .

例子

取ε=0.1，m=10.

t=1: 尝试 n次，装入背包物品集分别为{1},{2},…,{n}，再使用 G-KK算法

t=2: 尝试 $n(n-1)/2$次，装入物品集分别是是{1, 2},{1, 3}, … ,{n-1, n}，再使用 G-KK算法.

…

t=10: 尝试 C_n^{10} 次，装入物品集为 {1,…9,10}, {1,…9,11}, …, {n-9,n-8,…,n-1,n}，再用G-KK算法.

总计运行G-KK算法次数：

$$C_n^1 + C_n^2 + ... + C_n^{10}$$

PTAS的性能

定理 对每一个$\varepsilon > 0$和 0-1 背包问题的实例 I,

$$\mathrm{OPT}(I) < (1+\varepsilon)\,\mathrm{PTAS}_{\varepsilon}(I),$$

且 $\mathrm{PTAS}_{\varepsilon}$的时间复杂度为$O(n^{1/\varepsilon+2})$.

证 设最优解为S^*. 若 $|S^*| \le m$, 则算法必得到 S^*. 设$|S^*| > m$. 考虑计算中以 S^*中 m 件价值最大的物品为基础, 用G-KK得到的结果 S. 设物品 l 是 S^* 中第一件不在 S 中的物品, 在此之前 G-KK 装入不属于 S^* 的物品 (肯定有这样的物品, 否则应该装入物品 l) 单位重量的价值都不小于 v_l/w_l, 当然也不小于 S^* 中所有没有装入的物品的单位重量的价值, 故有

$$\mathrm{OPT}(I) < \sum_{i\in S} v_i + v_l$$

S 包括 S^* 中 m 件价值最大的物品, 它们的价值都不小于 v_l,

$$v_l \le \sum_{i\in S} v_i/m$$

多项式时间近似方案

$$\begin{aligned}\mathrm{OPT}(I) &< \sum_{i\in S} v_i + v_l \\ &\le \sum_{i\in S} v_i + \sum_{i\in S} v_i/m \\ &\le (1+1/m)\,\mathrm{PTAS}_\varepsilon(I) \\ &\le (1+\varepsilon)\,\mathrm{PTAS}_\varepsilon(I)\end{aligned}$$

时间复杂度. 从 n 件物品中取 t 件(t =1,2,…,m), 所有可能取法的个数为

$$c_n^1 + c_n^2 + \cdots + c_n^m \le m \cdot \frac{n^m}{m!} \le n^m .$$

对每一种取法, G-KK的运行时间 为$O(n)$, 故算法的时间复杂度为 $O(n^{m+1}) = O(n^{1/\varepsilon+2})$.

多项式时间近似方案: 以ε>0和问题的实例作为输入 I, 对每一个固定的 $\varepsilon > 0$, 算法是 $1+\varepsilon$ - 近似的.

伪多项式时间算法与完全多项式时间近似方案

完全多项式时间近似方案: 以 $\varepsilon > 0$和问题的实例 I 作为输入, 时间复杂度为二元多项式 $p(|I|,1/\varepsilon)$, 且对每一个固定的$\varepsilon > 0$, 算法的近似比为 $1+\varepsilon$.

动态规划算法A 记$G_k(d)$: 当只考虑前 k 件物品时, 为了得到不小于 d 的价值, 至少要装入的物品重量

$$G_k(d) = \min\left\{ \sum_{i=1}^{k} w_i x \;\middle|\; \sum_{i=1}^{k} v_i x_i \geq d,\ x_i = 0\text{或}1,\ 1 \leq i \leq k \right\},$$

$0 \leq k \leq n,\ 0 \leq d \leq D,\ D = v_1+v_2+\ldots+v_n,$

约定: $\min\varnothing = +\infty$.

$\mathrm{OPT}(I) = \max\{\, d \mid G_n(d) \leq B \,\}$

动态规划算法

递推公式

$$G_0(d)=\begin{cases}0, & \text{若}d=0,\\ +\infty, & \text{若}d>0,\end{cases}$$

$$G_{k+1}(d)=\begin{cases}\min\{G_k(d),w_{k+1}\}, & \text{若}d\le v_{k+1},\\ \min\{G_k(d),G_k(d-v_{k+1})+w_{k+1}\}, & \text{若}d>v_{k+1},\end{cases}$$

$0\le k\le n\text{-}1,\ 0\le d\le D.$

A的时间复杂度为$O(nD)=O(n^2v_{\max})$, 是伪多项式时间算法.

完全多项式时间近似方案

算法FPTAS (Fully Polynomial-Time Approximation Scheme)

输入 $\varepsilon > 0$和实例 I.

1. 令 $b = \max\left\{\left\lfloor \frac{v_{\max}}{(1+1/\varepsilon)n} \right\rfloor, 1\right\}$.

2. 令 $v_i' = \lceil v_i/b \rceil$, $1 \le i \le n$. 把所有 v_i 换成 v_i', 记新得实例为I'.

3. 对 I'应用算法 A 得到解 S, 把 S 取作实例 I 的解.

定理 对每一个$\varepsilon > 0$和 0-1背包问题的实例 I,

$$\mathrm{OPT}(I) < (1+\varepsilon)\,\mathrm{FPTAS}(I),$$

并且 FPTAS 的时间复杂度为 $O(n^3(1+1/\varepsilon))$.

证明

$$b=\max\left\{\left\lfloor\frac{v_{\max}}{(1+1/\varepsilon)n}\right\rfloor,1\right\}.\quad v_i'=\lceil v_i/b\rceil$$

证　由于

$$(v_i'-1)b < v_i \le v_i'b \qquad (1)$$

对任意的 $T\subseteq\{1, 2, \dots, n\}$

$$0 \le b\sum_{i\in T} v_i' - \sum_{i\in T} v_i < b|T| \le bn \qquad (2)$$

设 I 的最优解为 S^*, 注意到 S 是 I' 的最优解, 故有

$$\mathrm{OPT}(I)-\mathrm{FPTAS}(I) = \sum_{i\in S^*} v_i - \sum_{i\in S} v_i$$

$$= \left(\sum_{i\in S^*} v_i - b\sum_{i\in S^*} v_i'\right) \quad (\le 0,\ \text{由式(1)}\ v_i \le v_i'b)$$

$$+ \left(b\sum_{i\in S^*} v_i' - b\sum_{i\in S} v_i'\right) \quad (\le 0,\ S\text{是关于输入}v_i'\text{的最优解})$$

$$+ \left(b\sum_{i\in S} v_i' - \sum_{i\in S} v_i\right)$$

$$\le \left(b\sum_{i\in S} v_i' - \sum_{i\in S} v_i\right) < bn \qquad (\text{由式(2)})$$

证明

$$b = \max\left\{\left\lfloor \frac{v_{\max}}{(1+1/\varepsilon)n} \right\rfloor, 1\right\} \quad v_i' = \lceil v_i/b \rceil$$

OPT(*I*)–FPTAS(*I*) < *bn*

对每一个$\varepsilon > 0$, 若 $b=1$, 则 I'就是 I, S是 I 的最优解.

设 $b>1$, 则

$$\begin{aligned} &\text{OPT}(I)-\text{FPTAS}(I) \\ &< v_{\max}/(1+1/\varepsilon) \\ &\le \text{OPT}(I)/(1+1/\varepsilon) \quad (\text{因为} v_{\max} \le \text{OPT}(I)) \end{aligned}$$

得 $\text{OPT}(I) < (1+\varepsilon)\text{FPTAS}(I)$.

时间：主要是 A对I'的运算，时间复杂度为

$$O(n^2 v_{\max}/b) = O(n^3(1+1/\varepsilon))$$