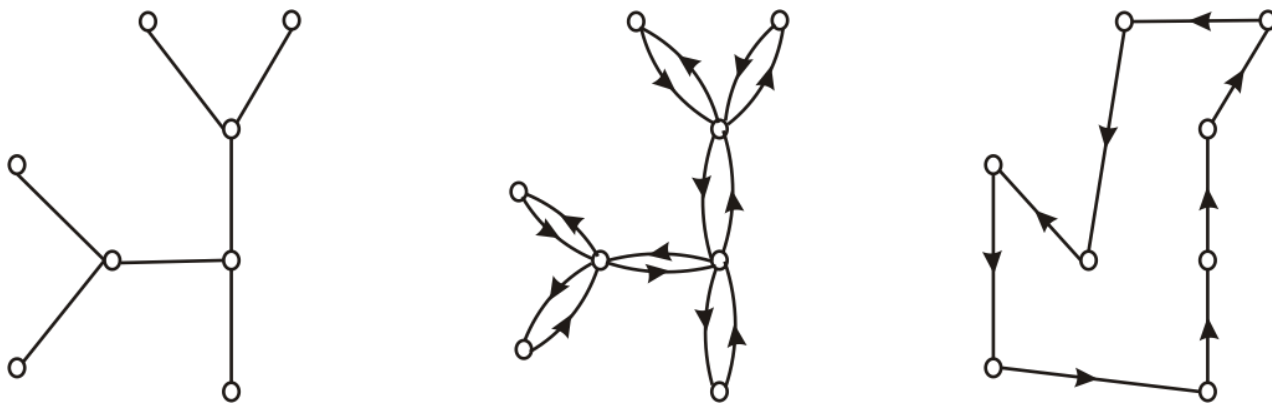


最小生成树法

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.



北京大学



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 ,

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

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 $\text{OPT}(I)$. 沿 T 走两遍的长小于 $2\text{OPT}(I)$. 因为满足三角不等式, 抄近路不会增加长度, 故

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

MST是2-近似算法.



北京大學

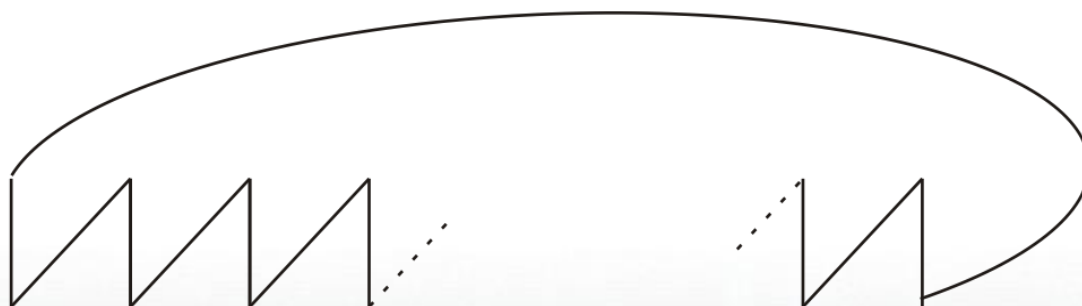
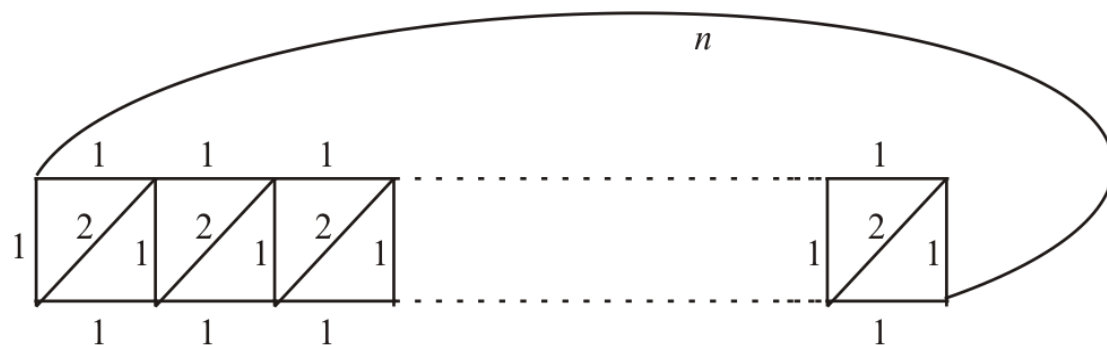


紧实例

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

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

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



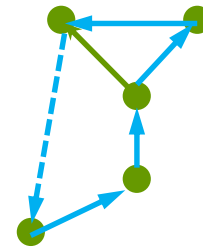
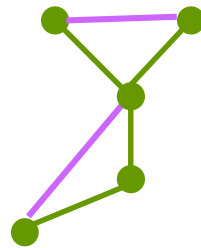
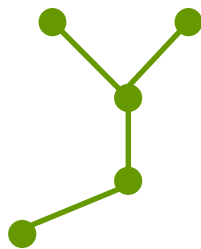
北京大学

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.



北京大學

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 ,

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

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 $\text{OPT}(I)$. 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 $\text{OPT}(I)/2$, 求得的欧拉回路的长小于 $(3/2)\text{OPT}(I)$. 抄近路不会增加长度, 得证

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

MM是3/2 -近似算法



北京大學

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 $r \leq K$, K 是常数. 任给图 $G = \langle V, E \rangle$, 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

$$\text{OPT}(I_G) = n, \quad A(I_G) \leq r \text{OPT}(I_G) \leq Kn$$

否则

$$\text{OPT}(I_G) > Kn, \quad A(I_G) \geq \text{OPT}(I_G) > Kn$$

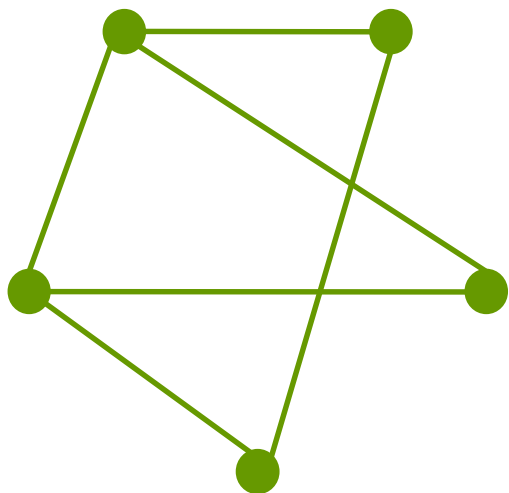
所以, G 有哈密顿回路当且仅当 $A(I_G) \leq Kn$



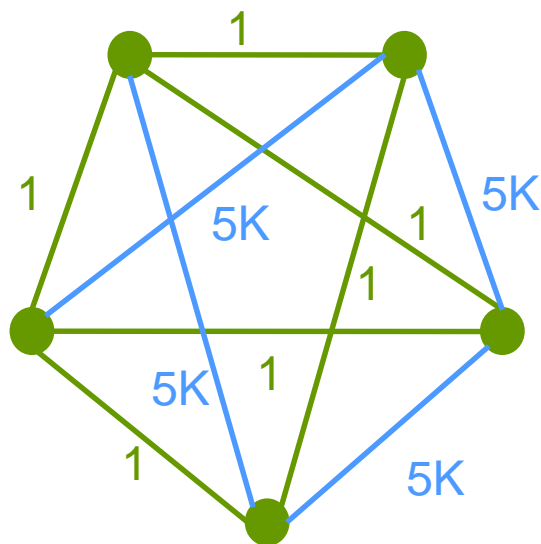
北京大學



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 $A(I_G) \leq Kn$, 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 $P=NP$.



北京大學



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 , $1 \leq i \leq n$, 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 $S^* \subseteq \{1, 2, \dots, n\}$ 使得

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

下面不妨设所有物品的重量 $w_i \leq B$
否则将这个物品排除在外



北京大學

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

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

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 $v_k = \max\{v_i \mid i = 1, 2, \dots, n\}$.

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.



北京大學



G-KK的性能

定理 对0-1背包问题的任何实例 I , 有

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

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

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

G-KK是2-近似算法.



北京大學

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

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

3. 对每一个 $t = 1, 2, \dots, m$ 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 .



北京大學



例子

取 $\varepsilon=0.1$, $m=10$.

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

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

...

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

总计运行 G-KK 算法次数:

$$C_n^1 + C_n^2 + \dots + C_n^{10}$$



北京大学

PTAS的性能

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

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

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

证 设最优解为 S^* . 若 $|S^*| \leq m$, 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

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

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 ,

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



北京大學

多项式时间近似方案

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

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

$$c_n^1 + c_n^2 + \dots + c_n^m \leq m \cdot \frac{n^m}{m!} \leq n^m.$$

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 $\varepsilon > 0$ 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.



北京大學

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

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

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

约定: $\min \emptyset = +\infty$.

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



北京大學

动态规划算法

递推公式

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

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

$$0 \leq k \leq n-1, \quad 0 \leq d \leq D.$$

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



北京大學

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 $v_i' = \lceil v_i/b \rceil$, $1 \leq i \leq n$. 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 ,

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

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



北京大學

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

$$(v_i' - 1)b < v_i \leq v_i' b \quad (1)$$

对任意的 $T \subseteq \{1, 2, \dots, n\}$

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

设 I 的最优解为 S^* , 注意到 S 是 I' 的最优解, 故有

$$\begin{aligned} \text{OPT}(I) - \text{FPTAS}(I) &= \sum_{i \in S^*} v_i - \sum_{i \in S} v_i \\ &= \left(\sum_{i \in S^*} v_i - b \sum_{i \in S^*} v_i' \right) \quad (\leq 0, \text{ 由式(1) } v_i \leq v_i' b) \\ &\quad + \left(b \sum_{i \in S^*} v_i' - b \sum_{i \in S} v_i' \right) \quad (\leq 0, S \text{ 是关于输入 } v_i' \text{ 的最优解}) \\ &\quad + \left(b \sum_{i \in S} v_i' - \sum_{i \in S} v_i \right) \\ &\leq \left(b \sum_{i \in S} v_i' - \sum_{i \in S} v_i \right) < bn \quad (\text{由式(2)}) \end{aligned}$$



证明

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

$$\text{OPT}(I) - \text{FPTAS}(I) < bn$$

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

$$\text{OPT}(I) - \text{FPTAS}(I)$$

$$< v_{\max}/(1+1/\varepsilon)$$

$\leq \text{OPT}(I)/(1+1/\varepsilon)$ (因为所有物品的重量都
小于等于背包限重, $v_{\max} \leq \text{OPT}(I)$)

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

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



北京大學