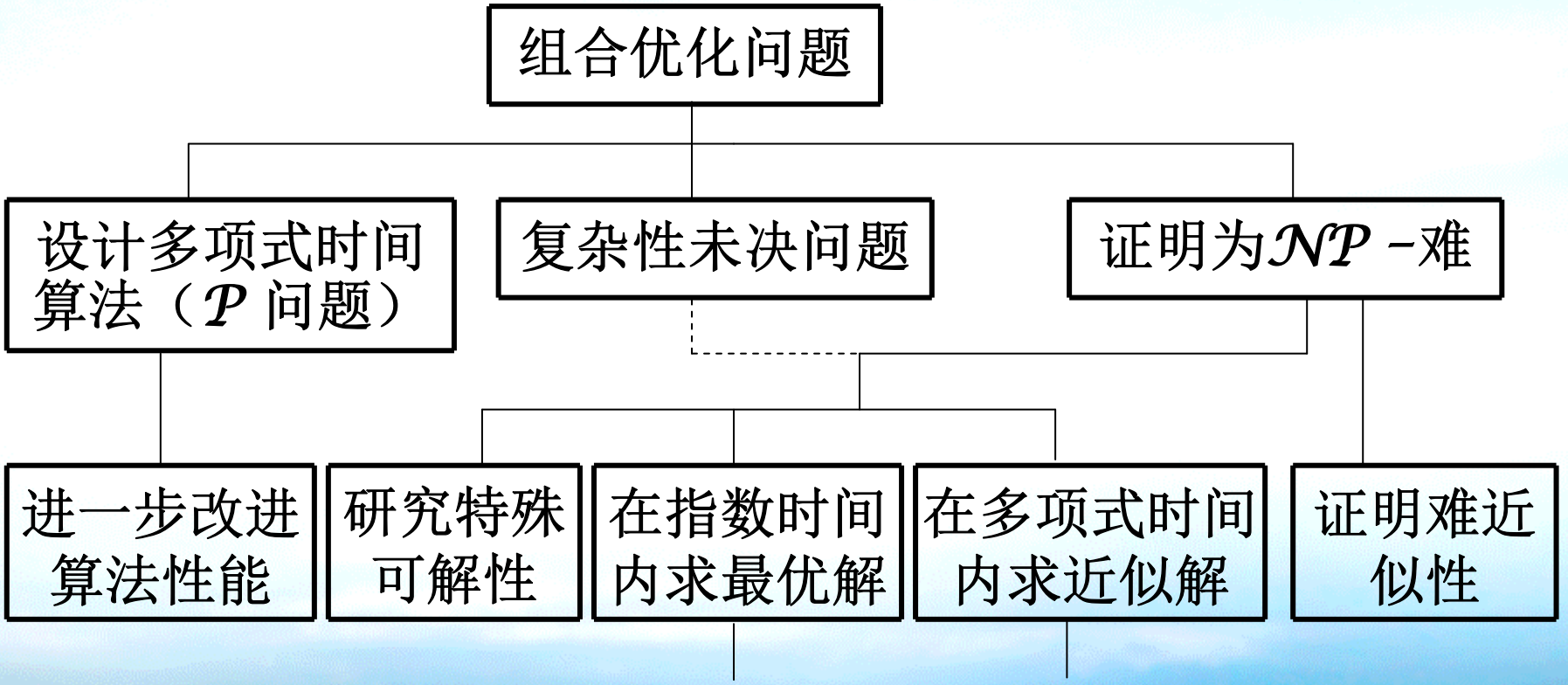
组合优化

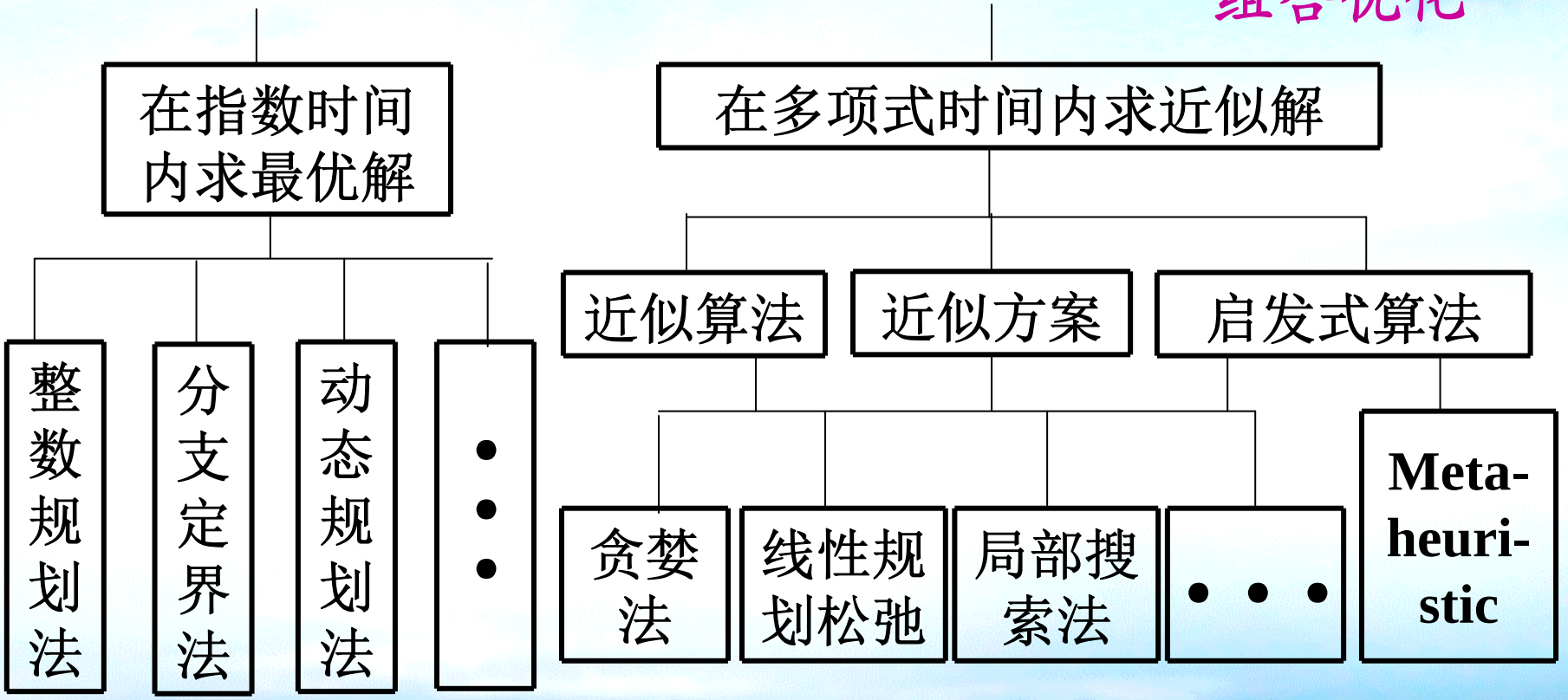
浙江大学数学系　谈之奕

组合优化问题的求解方法

组合优化问题的求解方法



组合优化问题的求解方法



整数规划

- 组合优化问题多可用整数规划描述。建立整数规划的主要步骤有确定决策变量、给出目标函数、列出约束条件
- 整数规划求解也是$\mathcal{NP}$—难的，但可借助整数规划的算法或软件求解一些具体的实例，也可利用整数规划的理论和方法分析解决问题
- 一个组合优化问题可能存在多个整数规划描述，需根据实际情况选择、完善和优化。复杂组合优化问题实例的整数规划求解往往需结合问题性质

LINGO 15.0：
Optimization Modeling Software for Linear, Nonlinear, and Integer Programming

CPLEX Optimizer：
High-performance mathematical programming solver for linear programming, mixed integer programming, and quadratic programming

IBM

背包问题

- 现有 n 件物品，物品 j 的价值为 p_j，大小为 w_j，背包容量为 C。要求选择若干物品放入背包，在放入背包物品大小之和不超过背包容量前提下使放入背包物品价值之和尽可能大
  - 决策变量 $x_j = \begin{cases} 1 & \text{放入第 } j \text{ 种物品} \\ 0 & \text{其他} \end{cases}$ $j = 1, 2, \cdots, n$
  - 整数规划

$$\max \sum_{j=1}^{n} p_j x_j$$ 放入背包物品价值之和

$$s.t. \quad \sum_{j=1}^{n} w_j x_j \le C$$ 放入背包物品大小之和不超过 C

$$x_j = 0,1, \quad j = 1, \cdots, n$$

指派问题

- 设有 n 项任务需分配给 n 位员工，每人完成其中一项，员工 i 完成任务 j 所需时间为 c_{ij}，如何分配可使完成所有任务所用总时间最少
  - 决策变量 $x_{ij}=\begin{cases}1, & \text{员工 } i \text{ 被分配完成工作 } j\\ 0, & \text{其他}\end{cases}$ $i,j=1,2,\cdots,n$
  - 整数规划

$$\min \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}$$ 完成所有任务所用总时间

$$s.t. \quad \sum_{j=1}^{n} x_{ij}=1,\ i=1,\cdots,n$$ 每位员工完成一项工作

$$\sum_{i=1}^{n} x_{ij}=1,\ j=1,\cdots,n$$ 每项工作由一位员工完成

$$x_{ij}=0 \text{ 或 } 1,\ i,j=1,\cdots,n$$

整数规划 $\in \mathcal{NP}$-C

指派问题 $\in \mathcal{P}$?

指派问题

- 记决策变量向量为 $\mathbf{x}=(x_{11},x_{12},\cdots,x_{1n},x_{21},\cdots,x_{2n},\cdots,x_{n1},\cdots,x_{nn})^T$，整数规划的系数矩阵为 $\min\{\mathbf{cx}\mid \mathbf{Ax}=\mathbf{b}, x_{ij}\in\{0,1\}\}$

$$\mathbf{A}=\begin{pmatrix} 1\ 1\ \cdots\ 1 & 0\ 0\ \cdots\ 0 & & 0\ 0\ \cdots\ 0 \\ 0\ 0\ \cdots\ 0 & 1\ 1\ \cdots\ 1 & \cdots & 0\ 0\ \cdots\ 0 \\ \vdots\ \vdots\ \ \vdots\ \ \vdots & \vdots\ \vdots\ \ \vdots\ \ \vdots & & \vdots\ \vdots\ \ \vdots\ \ \vdots \\ 0\ 0\ \cdots\ 0 & 0\ 0\ \cdots\ 0 & & 1\ 1\ \cdots\ 1 \\ \mathbf{I} & \mathbf{I} & \cdots & \mathbf{I} \end{pmatrix}$$

$$\min \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}$$

$$s.t.\ \sum_{j=1}^{n} x_{ij}=1,\ i=1,\cdots,n$$

$$\sum_{i=1}^{n} x_{ij}=1,\ j=1,\cdots,n$$

$$0\le x_{ij}\le 1\quad i,j=1,\cdots,n$$

- 若矩阵的所有子式均为**0**或 ±1，则称该矩阵为全幺模（**totally unimodular**）矩阵
- 系数矩阵为全幺模矩阵的整数规划的松弛线性规划的最优解必为整数解

TSP问题

浙江大学 ZheJiang University

组合优化

- 决策变量 $x_{ij}=\begin{cases}1, & \text{离开城市 } i \text{ 后到达城市 } j\\ 0, & \text{其他}\end{cases}$ $i,j=1,2,\cdots,n$

决策变量能表示环游，同时能计算环游长度

- 整数规划

$$\min \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}$$

$$s.t. \quad \sum_{j=1}^{n} x_{ij}=1, \quad i=1,\cdots,n$$ 离开城市 i 后到达另一个城市

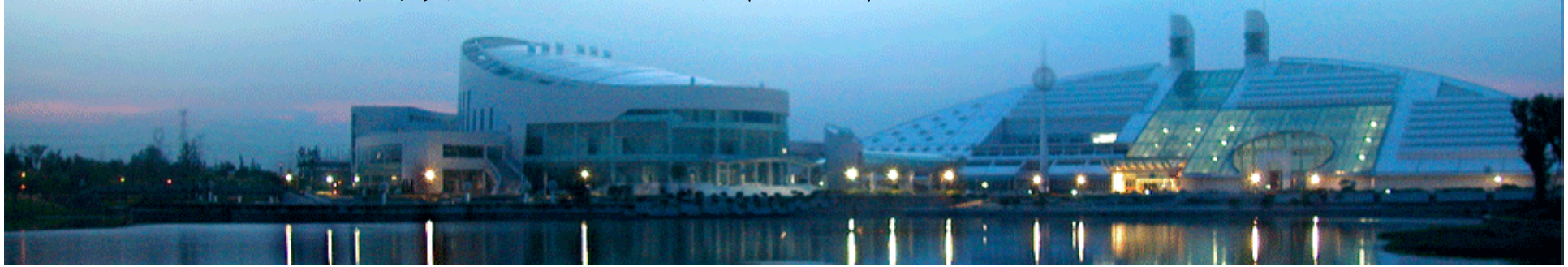
$$\sum_{i=1}^{n} x_{ij}=1, \quad j=1,\cdots,n$$ 从一个城市来到城市 j

$$x_{ij}=0,1,\ i,j=1,\cdots,n$$

问题可行解与其整数规划可行解之间未一一对应

指派问题 $\in \mathcal{P}$

TSP $\in \mathcal{NP}$-C ?



TSP问题

- **TSP**环游不允许出现经过城市数小于的子环游
- 若存在一相继经过城市 $i_1, i_2, \cdots$ 的子环游，则

$$x_{i_1 i_2} = x_{i_2 i_3} = \cdots = x_{i_{k-1} i_k} = x_{i_k i_1} = 1$$

$$\sum_{i,j \in \{i_1, i_2, \cdots, i_k\}} x_{ij} \geq k$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 1, \cdots, n$$

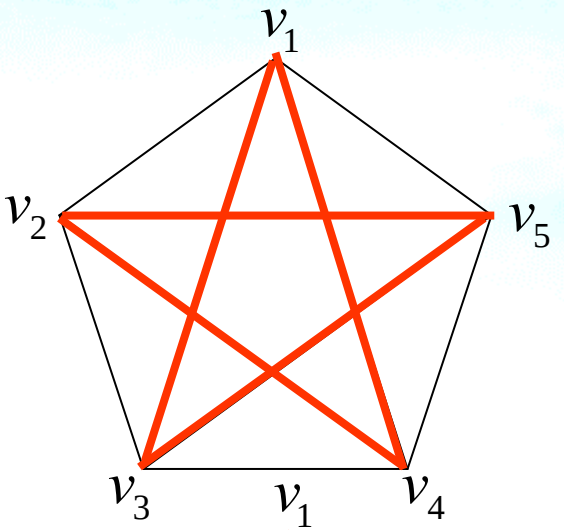
$$\sum_{i=1}^{n} x_{ij} = 1, \quad j = 1, \cdots, n$$

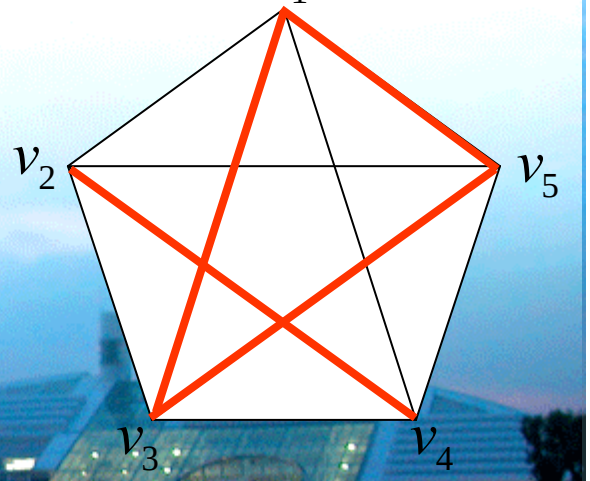
$$x_{13} = x_{35} = x_{52} = x_{24} = x_{41} = 1$$

$$x_{ij} = 0, (i, j) \notin \{(1,3), (3,5), (5,2), (2,4), (4,1)\}$$

$$x_{13} = x_{35} = x_{51} = x_{24} = x_{42} = 1$$

$$x_{ij} = 0, (i, j) \notin \{(1,3), (3,5), (5,1), (2,4), (4,2)\}$$





TSP问题的整数规划

- 为使整数规划的解不存在任意子环游，需增加约束以消除子环游

$$\sum_{i,j\in S} x_{ij} \le |S| - 1, \quad \forall \varnothing \ne S \subset \{2,3,\cdots,n\}$$

  - 增加约束后的整数规划约束个数为 $O(2^n)$
- 增加 $n-1$ 个实决策变量 $u_2, u_3, \cdots, u_n$ 和 $(n-1)^2$ 个约束

$$(n-1)x_{ij} + u_i - u_j \le n-2, \quad i, j = 2,3,\cdots,n$$

也可消除子环游

Miller CE, Tucker AW, Zemlin RA. Integer programming formulation of traveling salesman problems. *Journal of the ACM*, 7, 326-329, 1960

MTZ整数规划

- 若解含有子环游，则约束不被满足
  - 若含有子环游，则必有一子环游 $(i_1\ i_2\cdots i_k)$不经过城市 1
  - 若该子环游仅含一个城市 i，则 $x_{ii}=1$

$$(n-1)x_{ii}+u_i-u_i=n-1>n-2$$

矛盾

  - 若该子环游至少含有两个城市，则 $x_{i_j i_{j+1}}=1\left(i_{k+1}=i_1\right)$,

$$(n-1)x_{i_j i_{j+1}}+u_{i_j}-u_{i_{j+1}}=n-1+u_{i_j}-u_{i_{j+1}},\ j=1,2,\cdots,k$$

$$\sum_{j=1}^{k}\left((n-1)x_{i_j i_{j+1}}+u_{i_j}-u_{i_{j+1}}\right)=\sum_{j=1}^{k}\left(n-1+u_{i_j}-u_{i_{j+1}}\right)=k(n-1)>k(n-2)$$

$$(n-1)x_{ij}+u_i-u_j\le n-2,\ \ i,j=2,3,\cdots,n$$

矛盾

MTZ整数规划

- 对任意可行环游，存在一组可行解满足约束
  - 对任意可行环游，选定城市 1为起点，若城市 i是环游中的第 k 个经过的城市，取 $u_i = k$，$2 \le u_i \le n$，x_{ij}的取值同前定义
  - 若对某个 i, j 组合，$x_{ij} = 0$，则

$$(n-1)x_{ij} + u_i - u_j = u_i - u_j \le n-2$$

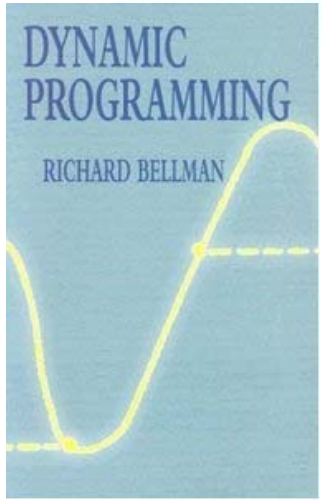
  - 若对某个 i, j 组合，$x_{ij} = 1$，则 $u_i - u_j = -1$，

$$(n-1)x_{ij} + u_i - u_j = (n-1) - 1 = n-2$$

$$(n-1)x_{ij} + u_i - u_j \le n-2,\ i, j = 2, 3, \cdots, n$$

动态规划

- 动态规划（**dynamic programming，DP**）是求解多阶段决策优化问题的一种数学方法和算法思想
- 动态规划的基本思路是将需求解的实例转化为规模较小的实例，并利用递推关系导出两者最优解之间的关系，从而可由初始条件出发逐步求得最优解

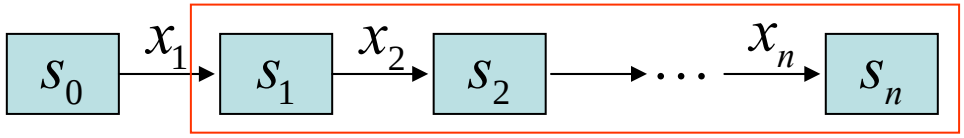


Bellman RE, *Dynamic Programming*, Princeton University Press, 1957

Richard Ernest Bellman (1920-1984)
美国运筹学家

最优化原理

- 最优化原理（**Principle of Optimality**）
  - 一个过程的最优决策 $(x_1^*, x_2^*, \cdots, x_n^*)$具有如下的性质：无论初始状态 s_0 及初始决策 x_1如何确定，之后的决策$(x_2^*, \cdots, x_n^*)$对于以 x_1^*所造成的状态 为初始状态的后部过程而言，必为最优策略



$$f_n(s_0; x_1, x_2, \cdots, x_n)$$

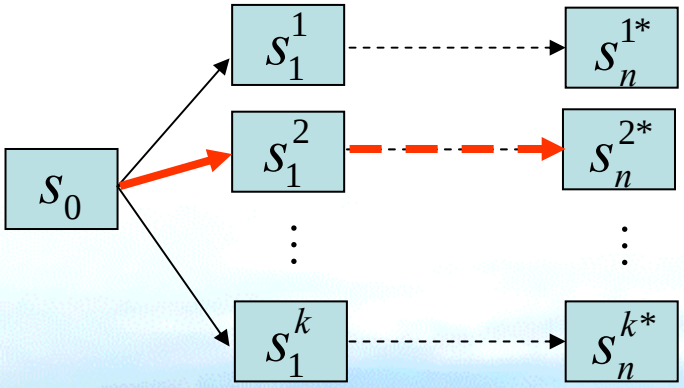
$$f_n(s_0; x_1^*, x_2^*, \cdots, x_n^*) = f_1(s_0; x_1^*) + f(s_1^*; x_2^*, \cdots, x_n^*)$$

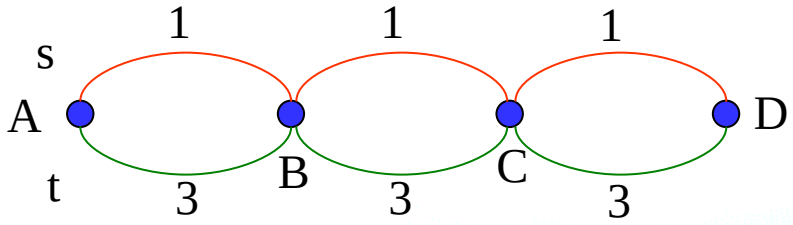
$$f_n(s_0; x_1^*, \hat{x}_2, \cdots, x_n) = f_1(s_0; x_1^*) + f(s_1^*; \hat{x}_2, \cdots, x_n)$$

PRINCIPLE OF OPTIMALITY. *An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision.*

最优化原理

- 最优化原理（**Principle of Optimality**）：一个 n 阶段过程的最优策略可以这样构成：首先求出以初始决策 x_1 造成的状态 s_1 为初始状态的 $n-1$ 阶段子过程的最优策略，然后在附加第一阶段效益（或费用）的情形下，从所有可能初始决策得到的解中选择最优者





最短路的子路也是最短路

路长定义为边长的模4加法时，t为A到D的最短路，但C到D的最短路仍为s，最优化原理不再成立

背包问题的动态规划

组合优化

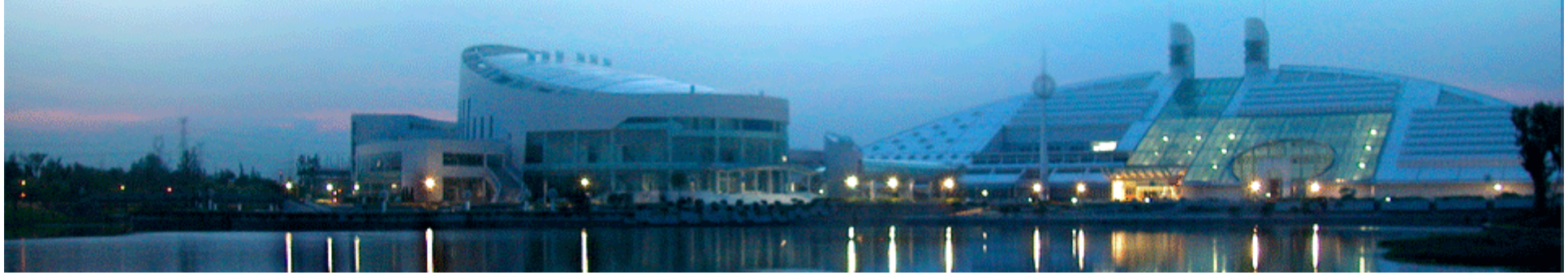
- 依次考虑所有物品，每个物品对应动态规划的一个阶段，可能决策为该物品放入与不放入两类
- 构造一系列背包问题实例 $I(k,w), 0 \le k \le n, 0 \le w \le C$，其中背包容量为 w，物品数为 k，包含原实例中的前 k 个物品
- 记 $C^*(k,w)$ 为实例 $I(k,w)$ 的最优解，原实例的最优值即为 $C^*(n,C)$

递推关系

- $C^*(k,w)$ 满足递推关系

$$C^*(k,w)=\begin{cases} C^*(k-1,w) & \text{若} w_k > w \\ \max\{C^*(k-1,w), C^*(k-1,w-w_k)+p_k\} & \text{若} w_k \le w \end{cases}$$

  - 若 $w_k > w$，则物品 k 不能放入容量为 w 的背包中，因此 $I(k,w)$ 的最优值与 $I(k-1,w)$ 的最优值必相同
  - 若 $w_k \le w$，则物品 k 可以放入容量为 w 的背包中，$I(k,w)$ 的最优解有两种可能
    - 若物品 k 未放入背包，$I(k,w)$ 的最优值必与 $I(k-1,w)$ 的最优值相同
    - 若物品 k 放入背包，背包剩余容量为 $w-w_k$，可放入的物品必为 $I(k-1,w-w_k)$ 的最优解中放入的物品



动态规划DPS

- $C^*(k,w)$ 满足初始条件

$$C^*(0,w)=0,\ w=0,\cdots,C,\ C^*(k,0)=0,\ k=0,\cdots,n$$

- 由初始条件和递推关系可逐步求得 $I(n,C)$，外层循环为 k 自 0到 n，内层循环为 w 自 0到 C
- 动态规划**DPS**的时间复杂性为 $O(nC)$，背包问题的实例规模为 $n+\lceil \log_2 L \rceil$，其中 $L=\max\left\{\max_{j=1,\cdots,n} p_j, \max_{j=1,\cdots,n} w_j, C\right\}$，因此**DPS**是伪多项式时间算法，背包问题是普通意义下的 $\mathcal{NP}$ −完全问题

动态规划DPS

- 背包问题实例

  $C=5, n=4$

  $p_1=3, p_2=4, p_3=5, p_4=6$

  $w_1=2, w_2=3, w_3=4, w_4=5$

- 最优值为 7，最优解为物品1，2 放入背包

w \ k	0	1 +3	2 +4	3 +5	4 +6
0	0	0	0	0	0
1	0	0	0	0	0
2 w_1	0	3	3	3	3
3 w_2	0	3	4	4	4
4 w_3	0	3	4	5	5
5 w_4	0	3	7	7	7

$$C^*(k,w)=\begin{cases} C^*(k-1,w) & \text{若} w_k > w \\ \max\left\{C^*(k-1,w), C^*(k-1,w-w_k)+p_k\right\} & \text{若} w_k \le w \end{cases}$$

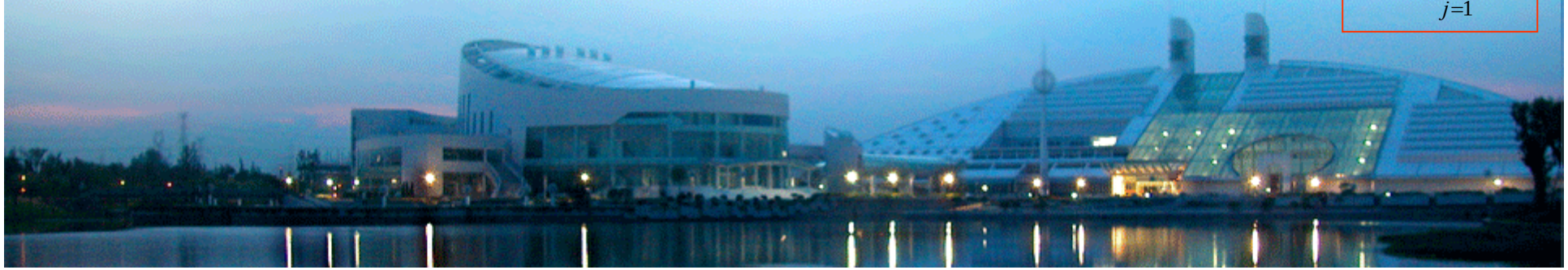
动态规划DPV

- 记 $y(k,q)$ 为在前 k 个物品中选出若干物品，使其价值之和恰为 q 的前提下，选出物品大小之和所能取到的最小值
  - 若价值之和恰为的物品子集不存在，则令 $y(k,q)=C+1$
  - 原实例的最优值即为 $\max\{q \mid y(n,q)\le C\}$
- $y(k,q)$ 满足递推关系

$$y(k,q)=\begin{cases} y(k-1,q), & \text{若} q<p_k, \\ \min\{y(k-1,q),y(k-1,q-p_k)+w_k\}, & \text{若} q\ge p_k. \end{cases}$$

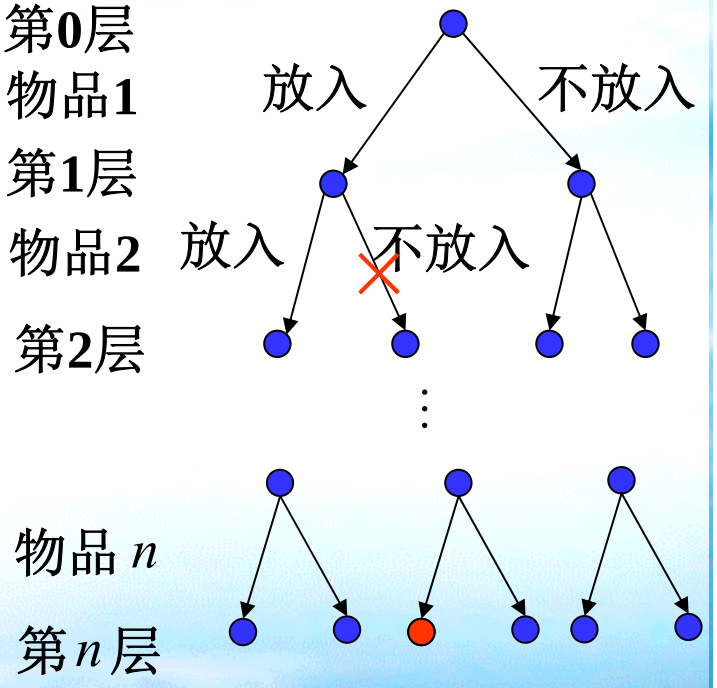
- $y(k,q)$ 满足初始条件 $y(k,0)=0,k=0,\cdots,n,y(0,q)=C+1,q=1,\cdots,P$
- **DPV的时间复杂性为 $O(nP)$，与DPS同为伪多项式时间算法，两者的时间复杂性不存在明确的优劣关系**

$$P=\sum_{j=1}^{n}p_j$$



分枝定界法

- 背包问题的分枝定界法
  - 实例求解过程可用一颗高度为 n 的二叉树表示。各层依次对应一个物品，最后一层的每个节点对应一个解
  - 分枝：每个节点有两条出边连结两个子节点，分别代表该节点下一层对应物品放入背包与不放入背包
  - 剪枝
    - 该枝不存在可行解
      - 按该枝含义确定放入背包的物品大小之和超过背包容量
    - 该枝不存在最优解　　如何提前判断



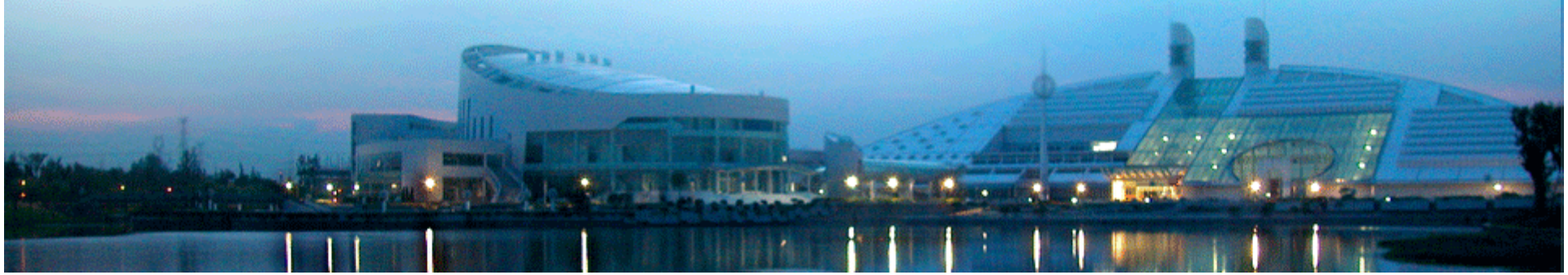
分枝定界法

- 定界　　下界对所有节点均有效，上界仅对该枝有效
  - 任一个已获得的可行解的目标值均是最优值的下界
  - 对每一枝，求该枝所有可行解目标值的上界，若该上界不大于下界，则该枝不存在更好的可行解
    - 按该枝含义，所有确定已放入物品和所有未确定物品价值之和

      上界越小越好，下界越大越好

      上界和下界的计算尽量简单
    - 该枝对应的整数规划的松弛线性规划的最优值

      只考虑未确定物品



松弛线性规划

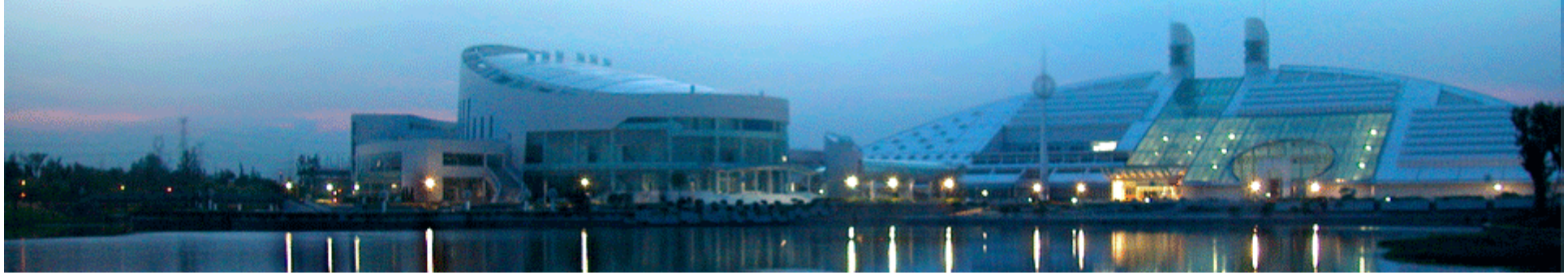
- 松弛线性规划最优解 $\mathbf{x}=(x_1^*,x_2^*,\cdots,x_n^*)^T$ 的性质
  - $\sum_{j=1}^{n} w_j x_j^* = C$
    - 若不然，必存在 $i, x_i^* < 1$。令 $x_i' = x_i^* + \varepsilon$ 可得一更好的解
  - 若 $\frac{p_i}{w_i} > \frac{p_k}{w_k}$，则若 $x_i^* < 1$，则 $x_k^* = 0$
    - 若不然，令 $x_i' = x_i^* + \frac{w_k}{w_i}\varepsilon, x_k' = x_k^* - \varepsilon$，

      $$\sum_{j=1}^{n} p_j x_j' - \sum_{j=1}^{n} p_j x_j^* = p_i \frac{w_k}{w_i}\varepsilon - p_k\varepsilon > 0$$ 矛盾
  - $\frac{p_j}{w_j}$ 称为物品 j 的价值密度，最优解应优先放入价值密度大的物品

$$\max \sum_{j=1}^{n} p_j x_j$$
$$s.t. \quad \sum_{j=1}^{n} w_j x_j \le C$$
$$x_j = 0,1, \quad j = 1,\cdots,n$$

$$\max \sum_{j=1}^{n} p_j x_j$$
$$s.t. \quad \sum_{j=1}^{n} w_j x_j \le C$$
$$0 \le x_j \le 1, \quad j = 1,\cdots,n$$

$$p_j > 0, j = 1,\cdots,n, \sum_{j=1}^{n} w_j > C$$

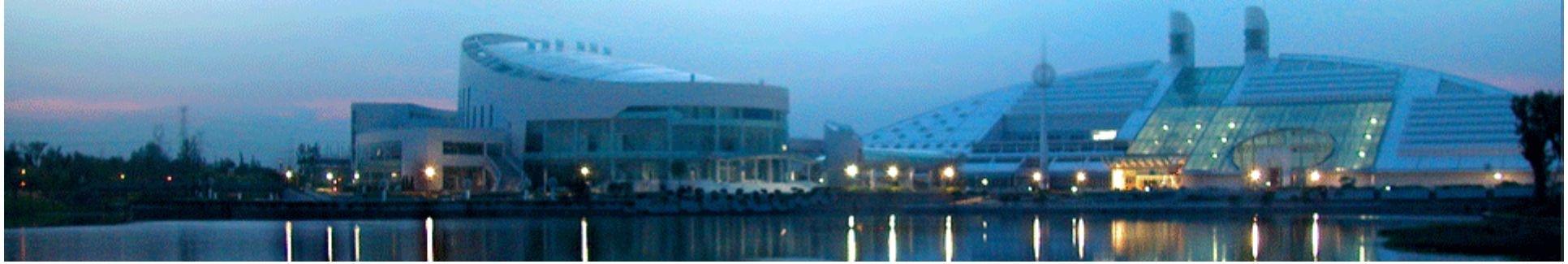


松弛线性规划

- 假设物品按价值密度非增顺序排列，即 $\frac{p_1}{w_1} \geq \frac{p_2}{w_2} \geq \cdots \geq \frac{p_n}{w_n}$
  - 若 $\frac{p_i}{w_i} = \frac{p_k}{w_k}$，可将物品 i 和物品 k 合并为一个物品
- 记 $j = \min\left\{k \middle| \sum_{i=1}^{k} w_i > C\right\}$ 为按价值密度非增顺序第一个不能全部放入背包的物品
- 松弛线性规划的最优解为

$$x_i^* = 1, i = 1, 2, \cdots, j-1, \quad x_j^* = \frac{1}{w_j}\left(C - \sum_{i=1}^{j-1} w_i\right), \quad x_i^* = 0, i = j+1, \cdots, n$$

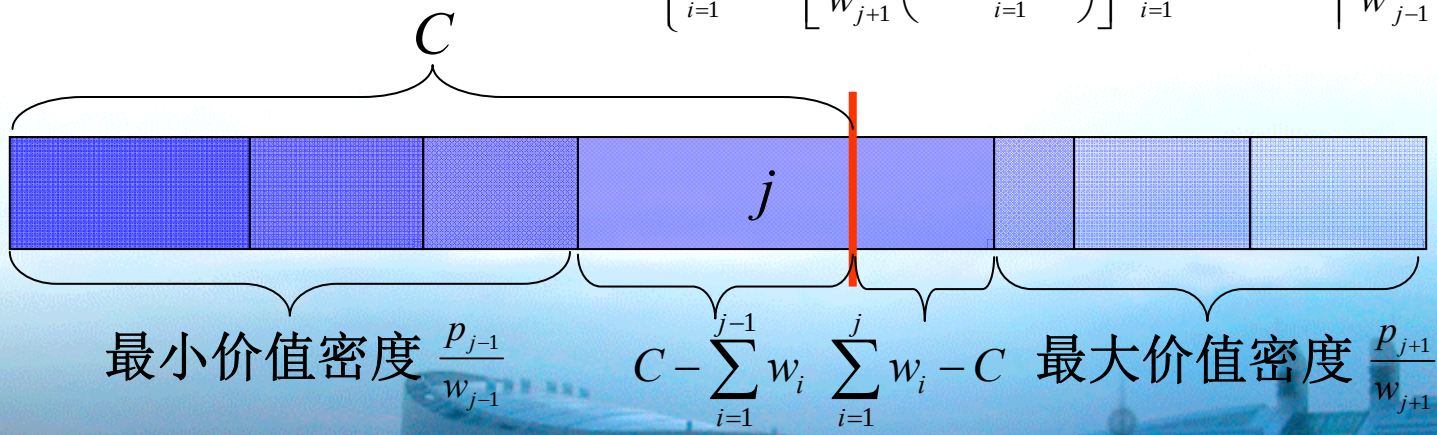
- 松弛线性规划的最优值为 $\sum_{i=1}^{j-1} p_j + \frac{p_j}{w_j}\left(C - \sum_{i=1}^{j-1} w_i\right)$。由于物品价值均为整数，背包（整数规划）最优值的上界也可取为 $UB_1 = \sum_{i=1}^{j-1} p_i + \left\lfloor \frac{p_j}{w_j}\left(C - \sum_{i=1}^{j-1} w_i\right) \right\rfloor$



上界

- 背包最优解可能不放入物品 j 而放入之后大小适中的物品，也可能因放入物品 j 而不得不取出之前的部分物品
  - 若不放入物品 j，最优值上界为 $\sum_{i=1}^{j-1} p_i + \frac{p_{j+1}}{w_{j+1}}\left(C - \sum_{i=1}^{j-1} w_i\right)$
  - 若放入物品 j，最优值上界为 $\sum_{i=1}^{j-1} p_i + p_j - \frac{p_{j-1}}{w_{j-1}}\left(\sum_{i=1}^{j} w_i - C\right)$
- 背包最优值上界可取为 $\max\left\{\sum_{i=1}^{j-1} p_i + \left\lfloor \frac{p_{j+1}}{w_{j+1}}\left(C - \sum_{i=1}^{j-1} w_i\right)\right\rfloor, \sum_{i=1}^{j-1} p_i + p_j - \left\lceil \frac{p_{j-1}}{w_{j-1}}\left(\sum_{i=1}^{j} w_i - C\right)\right\rceil\right\}$

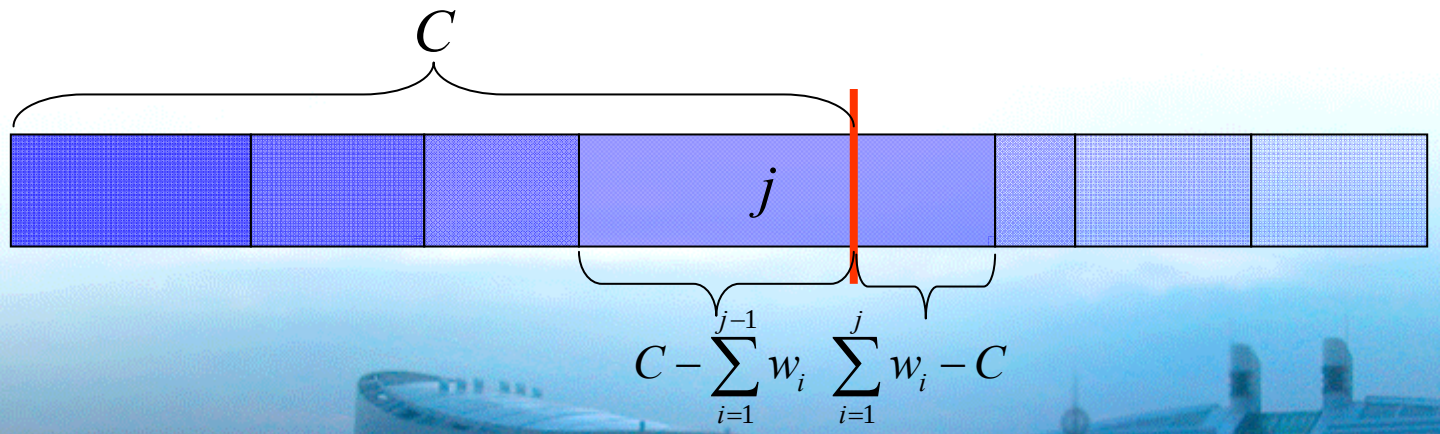
优于 UB_1



下界

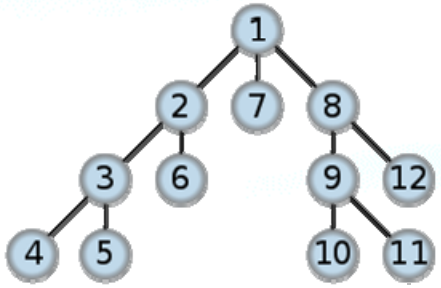
- 放入前 $j-1$ 个物品为可行解，目标值为 $LB_1 = \sum_{i=1}^{j-1} p_i$
- 再放入物品 j 之后某个大小适中的物品，或放入物品 j 而取出之前的某个物品可能为更好的可行解

$$LB_2 = \max\left\{LB_1, \max_{k=j+1,\cdots,n}\left\{LB_1 + p_k \,\middle|\, \sum_{i=1}^{j-1} w_i + w_k \le C\right\}, \max_{k=1,\cdots,j-1}\left\{LB_1 + p_j - p_k \,\middle|\, \sum_{i=1}^{j-1} w_i + w_j - w_k \le C\right\}\right\}$$

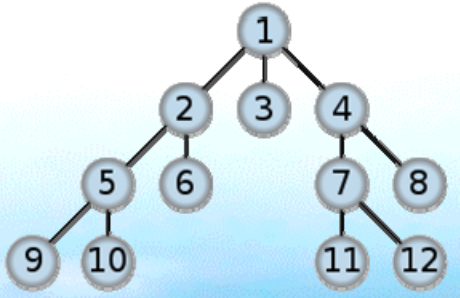


分枝定界法

- 从根节点开始，按深探法（**Depth-first search, DFS**）或广探法（**Breadth-first search, BFS**）给定的顺序检查每个节点
  - 对每个节点，按节点含义计算上界和下界
  - 若下界大于当前下界，则修正当前下界
  - 若当前节点不满足剪枝条件，则进行分枝
- 若所有节点都检查完毕且不可再分枝，当前下界即为实例的最优值
- 分枝定界法是指数时间算法，但实际效果未必劣于伪多项式时间动态规划



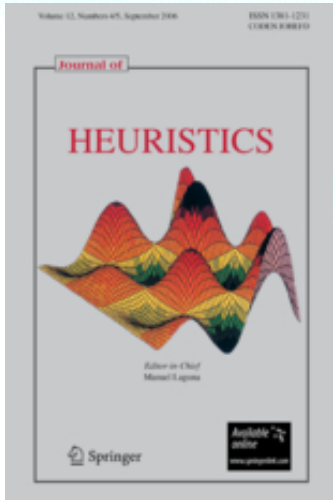
深探法



广探法

近似算法

- 可给出$\mathcal{NP}$－难问题任一实例最优解的算法总需要指数时间。较为现实的思路是“用精度换时间”，在多项式时间或可接受的实际运行时间内得到一个目标值与最优值较为接近的可行解
  - 近似算法（**approximation algorithm**）：算法的时间复杂性可通过分析确定（一般要求多项式时间），算法给出的近似解与最优解目标值之间的差距可通过证明严格估计
  - 启发式算法（**heuristic**）：无法说明算法的时间复杂性，或无法估计算法给出的近似解与最优解目标值之间的差距



Journal of Heuristics

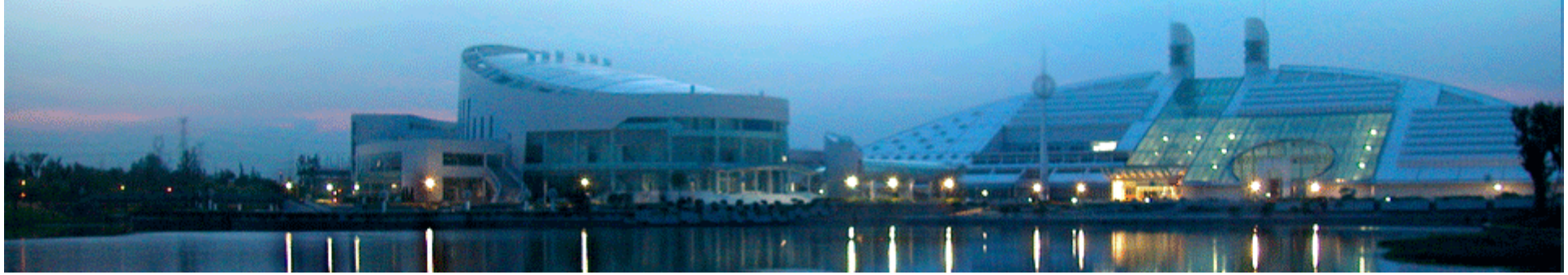
(ευρισκειν)
to find

最坏情况界

- 设Π是一个极小化优化问题，A是它的一个算法。对 Π的任意实例 I，算法 A 给出一个可行解，其目标值为 $C^A(I)$，实例 I 的最优目标值为 $C^*(I)$
- 称 $r_A=\inf\{r\geq 1 \mid C^A(I)\leq rC^*(I),\forall I\}$或 $r_A=\sup_I\left\{\frac{C^A(I)}{C^*(I)}\right\}$ 为算法的最坏情况界（**worst-case ratio**）
  - 若算法 A的最坏情况界为 r_A，则对该问题的任意实例 I，均有 $C^A(I)\leq r_AC^*(I)$
  - 最坏情况界越接近于1，说明算法给出的可行解目标值越接近于最优值，算法近似性能越好

最坏情况界

- 最坏情况界的证明方法
  - （上界）A 的最坏情况界不超过 r：对任意实例 I，$C^A(I) \le rC^*(I)$
    - 对很多优化问题，不可能得到 $C^*(I)$的简单表达式。为此，给出$C^*(I)$的一些易计算的下界$C^*_{LB}(I)$，即 $C^*(I) \ge C^*_{LB}(I)$，并证明 $C^A(I) \le rC^*_{LB}(I)$
  - （下界）A 的最坏情况界至少为 r：构造实例 I 或一族实例 I_n，使得 $\frac{C^A(I)}{C^*(I)} = r$ 或 $\frac{C^A(I_n)}{C^*(I_n)} \to r(n \to \infty)$
  - 若上界与下界相等，称界为紧（**tight**）的。若两者不相等或只得到其中一个，算法近似性能无法完全确定



背包问题的近似算法

- 对极大化目标的优化问题，定义 A的最坏情况界为 $r_A = \inf\{r \geq 1 | C^*(I) \leq rC^A(I), \forall I\}$
- 基于贪婪（**Greedy**）思想的算法
  - 将物品按价值密度非增的顺序排列，即有 $\frac{p_1}{w_1} \geq \frac{p_2}{w_2} \geq \cdots \geq \frac{p_n}{w_n}$
  - 按上述顺序将物品依次放入背包，直至第一个不能放入的物品为止
- 记 $j = \min\left\{k \mid \sum_{i=1}^{k} w_i > C\right\}$，算法将前 $j-1$个物品放入背包 $C^G = \sum_{i=1}^{j-1} p_j$

物品	1	2
价值	2	$2M$
大小	1	$2M$
背包容量 $2M$		

$C^G(I_1) = 2$，$C^*(I_1) = 2M$

$\frac{C^*(I_1)}{C^G(I_1)} = M \to \infty (M \to \infty)$

证明最坏情况界之前可通过构造实例对最坏情况界作出估计

背包问题的近似算法

浙江大学 ZheJiang University

组合优化

- 算法改进
  - 将物品 j 之后可以放入背包的物品放入背包　　改进无效
- 基于复合思想的改进算法
  - 运行基于贪婪思想的算法 G
  - 将 G 所得目标值与最大物品价值 $p_{\max}$ 进行比较，取优者作为输出

$$C^C(I)=\max\left\{\sum_{i=1}^{j-1}p_j, p_{\max}\right\}$$

物品	1	2
价值	2	$2M$
大小	1	$2M$
背包容量 $2M$		

$$C^G(I_1)=2,\quad C^*(I_1)=2M$$

$$\frac{C^*(I_1)}{C^G(I_1)}=M\to\infty(M\to\infty)$$

复合算法

- 复合算法的最坏情况界为 **2**
  - 用松弛线性规划的最优值作为最优值的上界

$$C^*(I) \le \sum_{i=1}^{j-1} p_i + \frac{p_j}{w_j}\left(C - \sum_{i=1}^{j-1} w_i\right) \le \sum_{i=1}^{j-1} p_i + \frac{p_j}{w_j} w_j \le \sum_{i=1}^{j-1} p_i + p_{\max}$$

$$\frac{C^*(I)}{C^C(I)} \le \frac{\sum_{i=1}^{j-1} p_i + p_{\max}}{\max\left\{\sum_{i=1}^{j-1} p_i, p_{\max}\right\}} \le 2$$

物品	1	2	3
价值	1	M	M
大小	1	M	M
背包容量 $2M$			

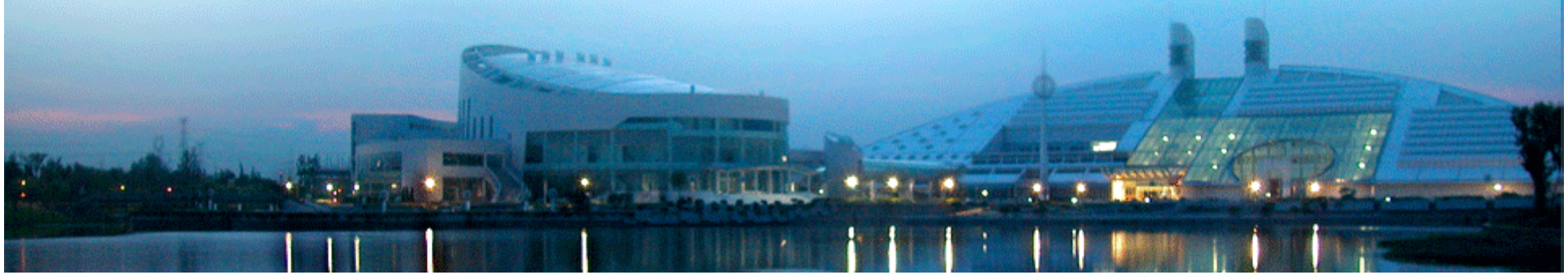
$$C^C(I) = M+1, C^*(I) = 2M$$

$$\frac{C^*(I)}{C^C(I)} = \frac{2M}{M+1} \to 2 (M \to \infty)$$

$$C^C(I) = \max\left\{\sum_{i=1}^{j-1} p_j, p_{\max}\right\} \quad j = \min\left\{k \mid \sum_{i=1}^{k} w_i > C\right\}$$

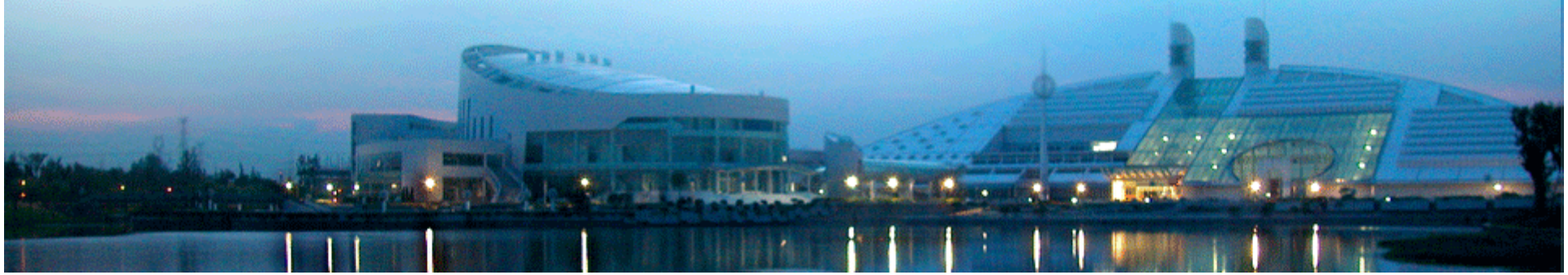
最坏情况分析

- 最坏情况界的特点
  - 对问题的不同实例，算法给出的近似解目标值与最优值的比值不全相同，最坏情况界仅是这些比值的最大值，无法充分反映算法性能的全貌
  - 若算法的某种改变仅能改进部分实例的可行解，而不能改进最坏情况实例的可行解，算法的最坏情况界不会减小
- 算法设计与最坏情况界证明思路
  - 算法给出的可行解目标值的恰当描述与最优值的合适估计
  - 最坏情况界上界和下界证明互为借鉴
  - 善于构造典型实例，重视其在算法设计与改进中的作用
  - 精心思考、勇于尝试、反复比较、不断修正



平均情况界

- 假设实例中的数据服从一定概率分布，$\frac{C^A(I)}{C^*(I)}$为一随机变量，其期望$\mathbb{E}\left(\frac{C^A(I)}{C^*(I)}\right)$即为算法 A 的平均情况界（**average-case ratio**）
  - 平均情况界依赖于所假设的实例中数据所服从的概率分布
  - 在多数情况下，缺乏足够的依据来判断一个实例中的数据来自何种分布
  - 即便已得到某一算法在某个分布下的平均情况界，对来自该分布的一个实例 I，也无法对$\frac{C^A(I)}{C^*(I)}$的值作出估计
  - 平均情况界的证明比最坏情况界更为复杂



TSP的难近似性

- 对**TSP**问题，不存在最坏情况界为有限常数的多项式时间近似算法，除非$\mathcal{P}=\mathcal{NP}$
  - 设**TSP**问题存在最坏情况界小于M的多项式时间算法A
  - 任给$\mathcal{NP}$–完全问题**HC**的实例I_{HC}：图$G=(V,E)$。构造**TSP**实例I_{TSP}如下：城市数$n=|V|$，城市之间距离

$$c_{ij}=\begin{cases}1, & \text{若} v_i v_j \in E,\\ nM-n+1, & \text{若} v_i v_j \notin E。\end{cases}$$

    - 若G中存在**Hamilton**圈，则$C^*(I_{TSP})=n$
    - 若G中不存在**Hamilton**圈，则任一**TSP**环游中至少有一对距离为$nM-n+1$的相邻城市，$C^*(I_{TSP})\geq(n-1)+(nM-n+1)=nM$

TSP的难近似性

浙江大学 ZheJiang University

组合优化

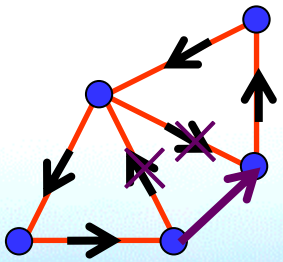
- 用算法 A 求解实例 I_{TSP}，则 $\dfrac{C^A(I_{TSP})}{C^*(I_{TSP})} < M$
  - 若 $C^A(I_{TSP}) \geq nM$，则 $C^*(I_{TSP}) > \dfrac{C^A(I_{TSP})}{M} \geq n$，$I_{HC}$ 的答案为“否”
  - 若 $C^A(I_{TSP}) < nM$，则 $C^*(I_{TSP}) \leq C^A(I_{TSP}) < nM$，$I_{HC}$ 的答案为“是”
  - 由 A 可给出**HC**问题的多项式时间算法　　矛盾

$C^*(I_{TSP}) = n$ ⇐ I_{HC} 的答案为“是” ⇐ $C^*(I_{TSP}) < nM$ ⇐ $C^A(I_{TSP}) < nM$

$C^*(I_{TSP}) \geq nM$ ⇐ I_{HC} 的答案为“否” ⇐ $C^*(I_{TSP}) > n$ ⇐ $C^A(I_{TSP}) \geq nM$

度量TSP

- 度量TSP（metric TSP）：城市之间的距离满足三角不等式，即 $c_{ij} \le c_{ik} + c_{kj}$
- 基于相关的多项式可解问题算法设计近似算法



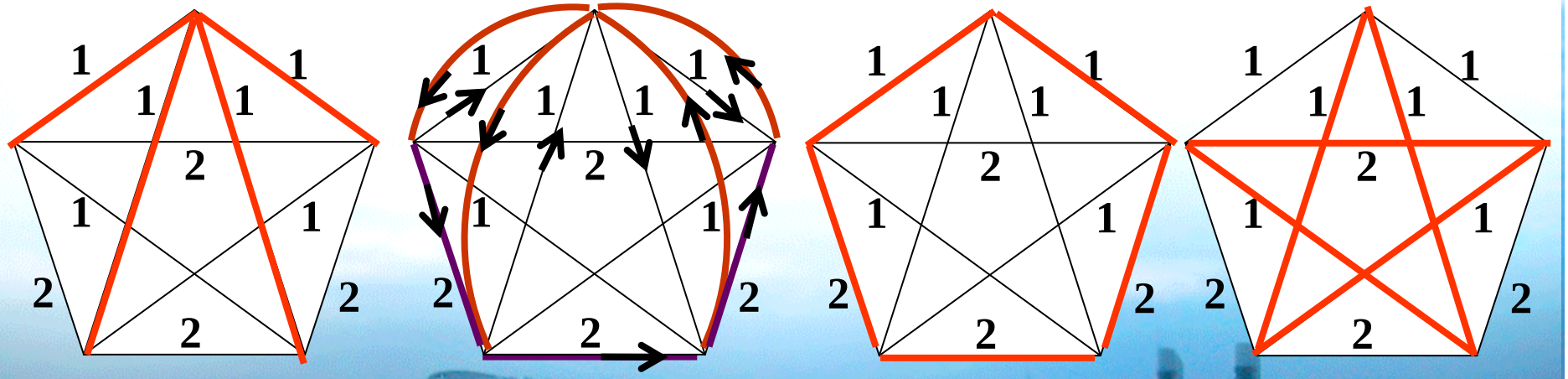
最小生成树加倍算法

- 最小生成树加倍算法
  - 构造赋权完全图 K_n，其中 n 为城市数，边 v_iv_j 的权为相应的城市 i 和 j 之间的距离
  - 求 K_n 的最小生成树 T
  - 将 T 中的每条边变为两条有相同起点和终点，权相同的边，得到一**Euler**图 G_1
  - 求 G_1 的一条**Euler**环游 C_1
  - 用“抄近路”的方法将 C_1 改造为一条**TSP**环游

最小生成树加倍算法

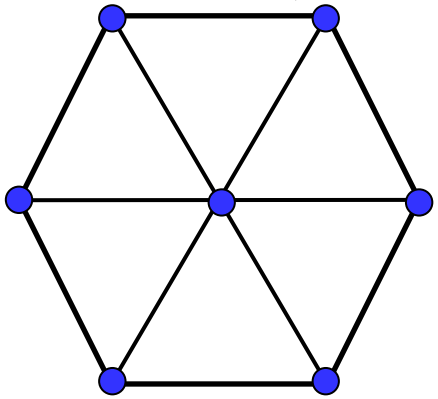
- 最小生成树加倍算法的最坏情况界不超过 **2**
  - 记 l_T 为 K_n 的最小生成树长度，则 $C^*(I) \geq l_T$，

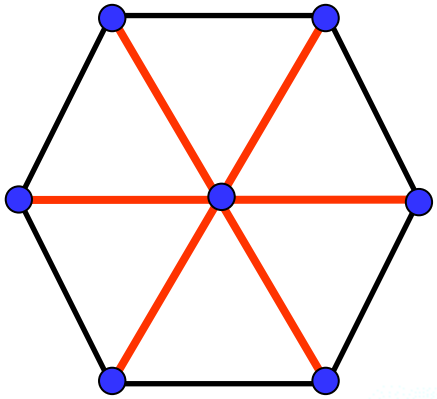
$$C^{DM}(I) \leq 2l_T \leq 2C^*(I)$$

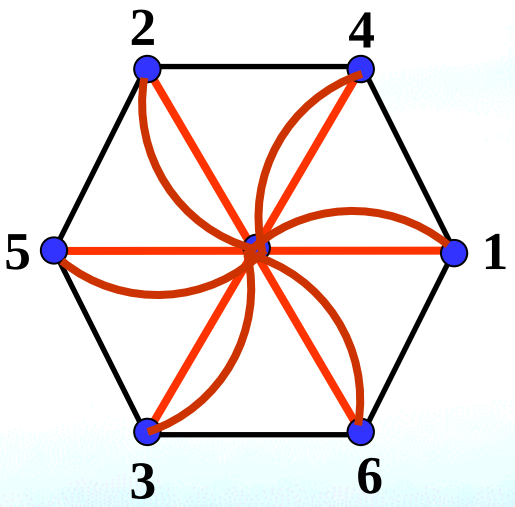


最小生成树加倍算法

- 最小生成树加倍算法的最坏情况界至少为 **2**





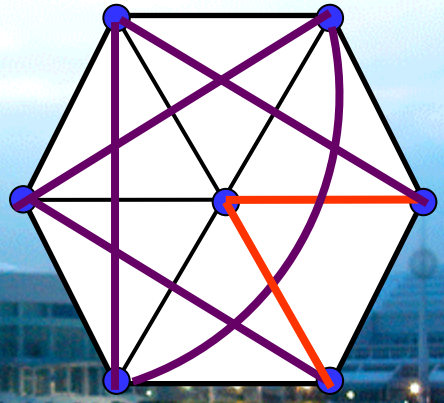


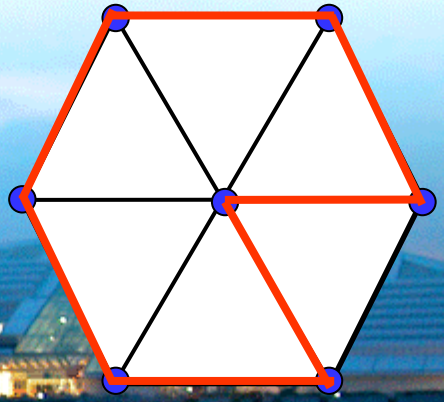
城市间距离在图中显示的为 **1**，未显示的为 **2**

$$C^{DM}(I)=10 \qquad C^{DM}(I_n)=2n-2$$

$$C^{*}(I)=7 \qquad C^{*}(I_n)=n$$

$$\frac{C^{DM}(I_n)}{C^{*}(I_n)}=\frac{2n-2}{n}\to 2\,(n\to\infty)$$





最小权完美匹配

组合优化

- 为将 K_n 的最小生成树 T 改造为**Euler**图，只需将 T 的奇度顶点两两相连
- 匹配
  - 给定图 $G=(V,E)$，E 的非空子集 M 称为 G 的一个匹配（**matching**），若 M 中任意两条边在 G 中均没有公共端点
  - 若 G 中所有顶点都与匹配 M 中某条边关联，则称 M 为完美匹配（**perfect matching**）
  - 偶数个顶点的完全图必存在完美匹配，任一完美匹配将图中顶点两两相连
  - 赋权图中总权和最小的完美匹配称为最小权完美匹配
- 任意图的最小权完美匹配 $\in \mathcal{P}$

Edmonds, J., Paths, trees and flowers. *Canadian Journal of Mathematics*, 17, 449–467, 1965.

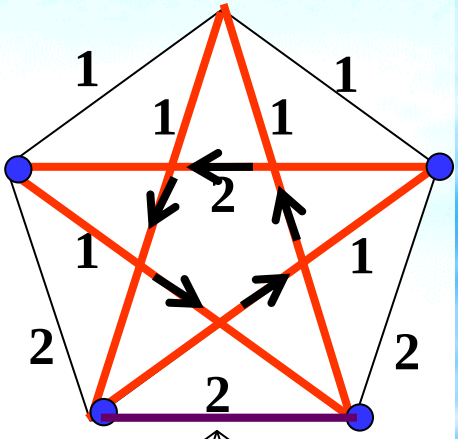
Edmonds, J., Maximum matching and a polyhedron with (0, 1) vertices. *Journal of Research of the National Bureau of Standards* B, 69, 125–130, 1965.

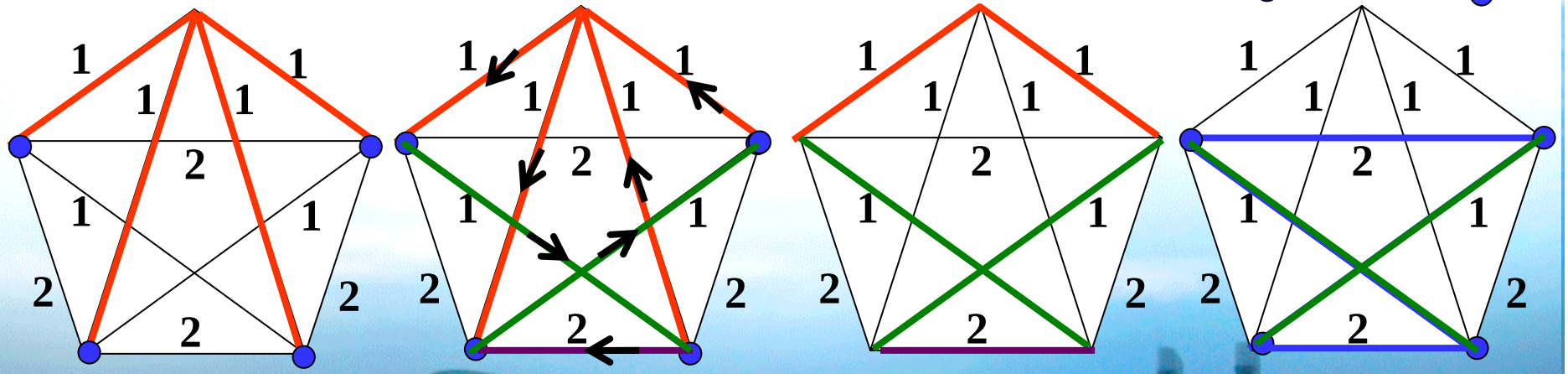
Christofides算法

- **Christofides算法**
  - 构造赋权完全图 K_n，其中 n 为城市数，边 v_iv_j 的权为相应的城市 i 和 j 之间的距离
  - 求 K_n 的最小生成树 T
  - 求以 T 的奇度顶点集为顶点集，边的权与 K_n 中对应边相同的赋权完全图 H 的最小权完美匹配，将该匹配中的边加入 T，得到一**Euler**图 G_2
  - 求 G_2 的一条**Euler**环游 C_2
  - 用“抄近路”的方法将 C_2 改造为一条**TSP**环游

Christofides算法

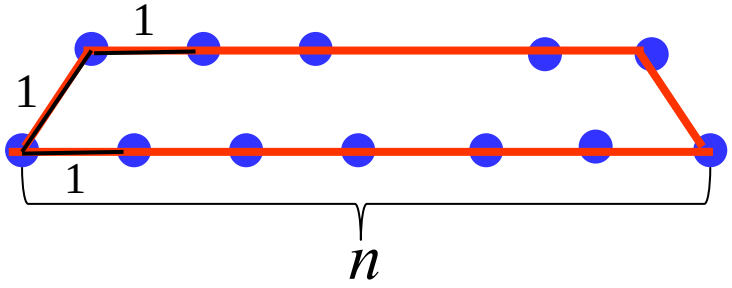
- **Christofides算法的最坏情况界不超过** $\frac{3}{2}$
  - 记 l_M 为 H 的最小权完美匹配的总权和，则 $C^*(I) \geq 2l_M$
  - $C^{Chr}(I) \leq l_M + l_T \leq \frac{3}{2} C^*(I)$



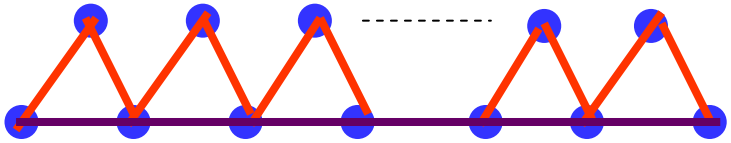


Christofides算法

- **Christofides算法的最坏情况界不小于** $\frac{3}{2}$



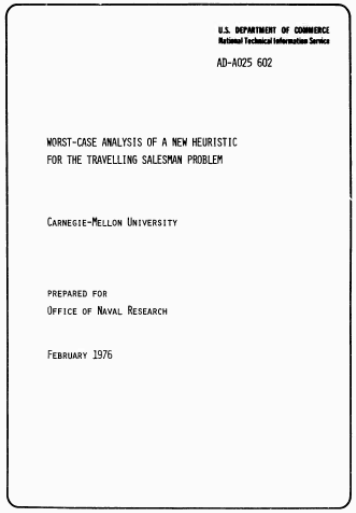
$$C^* = 2n+1$$



$$C^{Chr} = 3n$$

$$\frac{C^{Chr}(I_n)}{C^*(I_n)} = \frac{3n}{2n+1} \to \frac{3}{2}(n \to \infty)$$

Christofides算法求解Euclidean TSP的最坏情况界也为 3/2。它仍是至目前为止，度量TSP最坏情况界最小的多项式时间近似算法


U.S. DEPARTMENT OF COMMERCE
National Technical Information Service
AD-A025 602

WORST-CASE ANALYSIS OF A NEW HEURISTIC
FOR THE TRAVELLING SALESMAN PROBLEM

CARNEGIE-MELLON UNIVERSITY

PREPARED FOR
OFFICE OF NAVAL RESEARCH

FEBRUARY 1976


Management Sciences Research Report No. 388

WORST-CASE ANALYSIS OF A NEW HEURISTIC
FOR THE TRAVELLING SALESMAN PROBLEM

by

Nicos Christofides*

February 1976

This research was prepared as part of the activities of the Management Sciences Research Group, Carnegie-Mellon University. Reproduction in whole or in part is permitted for any purpose of the U.S. Government.

Management Science Research Group
Graduate School of Industrial Administration
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213

*Carnegie-Mellon University; on leave from Imperial College, London.

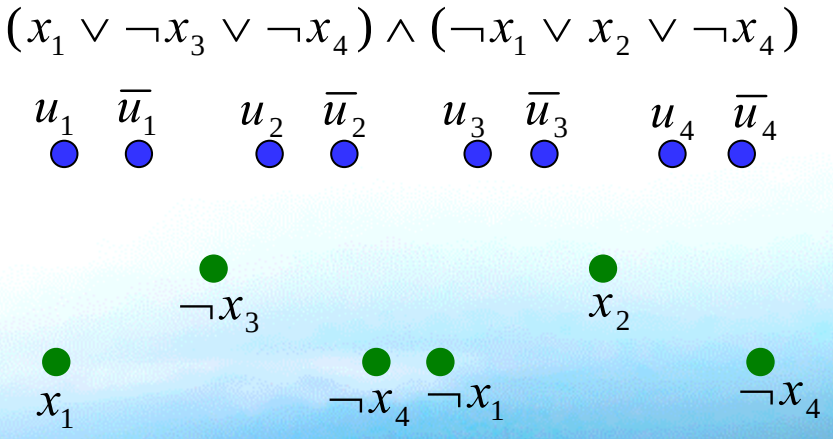
Christofides N., Worst-case analysis of a new heuristic for the travelling salesman problem. Report No. 388. Management Sciences Research Group, Graduate School of Industrial Administration, Carnegie-Mellon University, 1976.

顶点覆盖

- 给定图 $G=(V,E)$，顶点集 V 的子集 V' 称为 G 的顶点覆盖（**vertex cover**），若 E 中每条边至少有一个端点在 V' 中
- 最小权顶点覆盖问题（**WVC**）：给定图 $G=(V,E)$，其中顶点 v_j 的权为 $w_j>0$，求 G 的总权和最小的顶点覆盖
  - 最小顶点覆盖问题（**VC**）：给定图 $G=(V,E)$，求 G 的最小顶点覆盖，即包含顶点数最少的顶点覆盖
- **VC** $\in \mathcal{NP}$-难

$3SAT \leq_m^p VC$

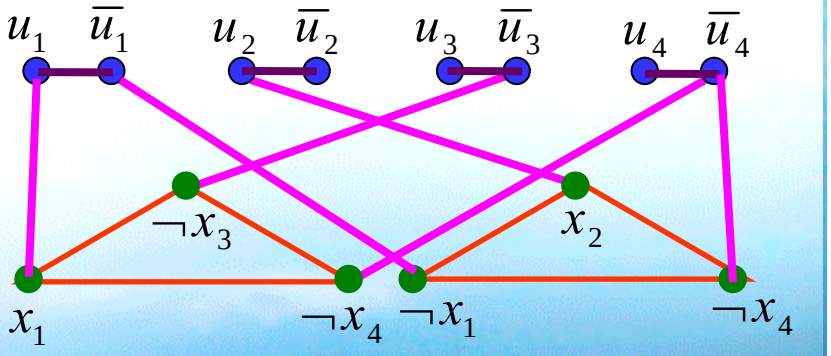
- 任取**3SAT**问题实例 I_{3SAT}：n 个**Boolean**变量 $x_1, x_2, \cdots x_n$ 的 **CNF** $\bigwedge_{i=1}^{m} c_i$，其中 $c_i = l_{i_1} \vee l_{i_2} \vee l_{i_3}$
- 构造**VC**判定问题实例 I_{VC}：图 $G = (V, E)$，阈值 $M = n + 2m$
  - $V = V_1 \cup V_2, E = E_1 \cup E_2 \cup E_3$
  - $V_1 = \bigcup_{j=1}^{n} \{u_j, \overline{u}_j\}$，其中 $u_j, \overline{u}_j$ 分别对应文字 $x_j, \neg x_j$
  - $V_2 = \bigcup_{i=1}^{m} \{w_{i_1}, w_{i_2}, w_{i_3}\}$，其中 $w_{i_1}, w_{i_2}, w_{i_3}$ 分别对应 c_i 中的 **3**个文字 $l_{i_1}, l_{i_2}, l_{i_3}$



$3SAT \leq_m^p VC$

- 图 $G=(V,E)$，$E=E_1 \cup E_2 \cup E_3$
  - $E_1=\{e_1,e_2,\cdots,e_n\}$，其中 e_j 连接 V_1 中两个顶点 $u_j,\overline{u}_j$
  - $E_2=\bigcup_{j=1}^{m} K_i$，其中 $K_i=\bigcup_{j=1}^{m}\{w_{i_1}w_{i_2},w_{i_2}w_{i_3},w_{i_3}w_{i_1}\}$
  - $E_3=\bigcup_{j=1}^{m} F_i$，其中 $F_i=\{f_{i_1},f_{i_2},f_{i_3}\}$，$(x_1 \vee \neg x_3 \vee \neg x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_4)$

    f_{i_k} 连接 V_2 中顶点 w_{i_k} 与 V_1 中对应相同文字的顶点



- G 含有 $2n+3m$ 个顶点和 $n+6m$ 条边，构造可在多项式时间内完成

$3SAT \leq_m^p VC$

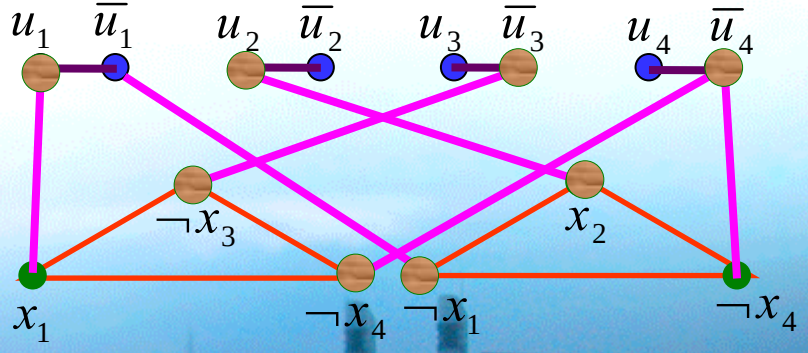
- 若 x_j 取真，则令 $u_j \in V'$，若 x_j 取假，则令 $\overline{u}_j \in V'$
- $\bigwedge_{i=1}^{m} c_i$ 可满足，c_j 可满足，c_j 中至少一文字为真，不妨设 l_{i_1} 为真，则令 $w_{i_2}, w_{i_3} \in V'$
- $u_j, \overline{u}_j$ 中至少有一个顶点在 V' 中，故 e_j 可覆盖
- 由于 $w_{i_2}, w_{i_3} \in V'$, K_i 中任一边可覆盖
- f_{i_1} 在 V_1 中的端点在 V'中，f_{i_2}, f_{i_3} 在 V_2 中的端点 w_{i_2}, w_{i_3} 在 V'中

若 I_{3SAT} 答案为“是”，则 I_{VC} 答案也为“是”

若存在变量的一组赋值，使 $\bigwedge_{i=1}^{m} c_i$ 可满足，则 G 存在一顶点覆盖 V'，且 $|V'| \leq n+2m$

$(x_1 \vee \neg x_3 \vee \neg x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_4)$

1 $x_1=1,\ x_2=1,\ x_3=0,\ x_4=0$ 1



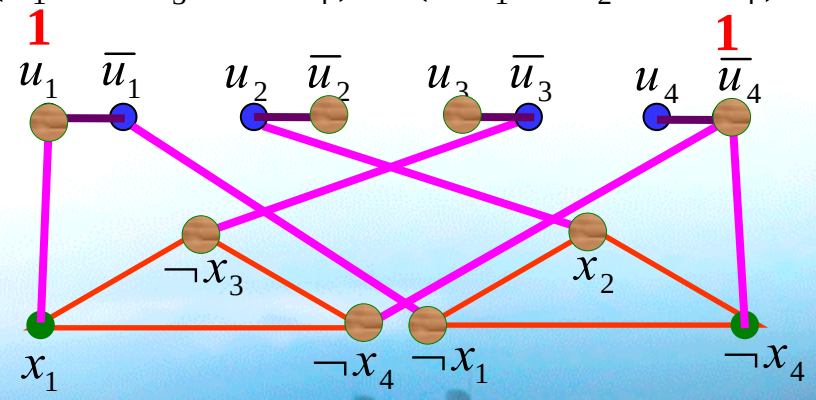
$3SAT \leq_m^p VC$

- E_1 中任一边 e_j 可覆盖，e_j 两端点 $u_j, \overline{u}_j$ 至少之一在 V' 中
- E_2 子集 K_i 中边均可覆盖，$w_{i_1}, w_{i_2}, w_{i_3}$ 三个顶点中至少两个在 V' 中
- 由于 $|V'| \leq n+2m$，故 $|V' \cap V_1| = n$，$|V' \cap V_2| = 2m$
- E_3 子集 F_i 的三条边中，至多有两条边被 $V' \cap V_2$ 中顶点覆盖，至少有一条边被 $V' \cap V_1$ 中顶点覆盖
- 若 $u_j \in V'$，则令 x_j 取真，若 $\overline{u}_j \in V'$，则令 x_j 取假
- F_i 中被 $V' \cap V_1$ 中顶点覆盖的边的端点对应文字取真，子句 c_i 中至少有一个文字取真，c_i 可满足，$\bigwedge_{i=1}^{m} c_i$ 可满足

若 I_{VC} 答案为“是”，则 I_{3SAT} 答案也为“是”

若 G 存在一顶点覆盖 V'，且 $|V'| \leq n+2m$，则存在变量的一组赋值，使 $\bigwedge_{i=1}^{m} c_i$ 可满足

$(x_1 \vee \neg x_3 \vee \neg x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_4)$



$x_1=1,\ x_2=0,\ x_3=1,\ x_4=0$

整数规划

- 最小权顶点覆盖问题的整数规划
  - 决策变量 $x_i = \begin{cases} 1, & \text{若} v_i \in V' \\ 0, & \text{其它} \end{cases}$
  - 整数规划（**IP**）

$$\min \sum_{i=1}^{n} w_i x_i$$

$$s.t. \quad x_j + x_k \geq 1, \quad \forall v_j v_k \in E$$

$$0 \leq x_i \leq 1$$

  - 松弛线性规划（**LP**）

最小权顶点覆盖的最优值
即为（IP）的最优值

E 中任一条边的两个端
点中至少有一个在 V' 中

线性规划松弛算法

- 记（IP）的最优解为 $\mathbf{x}^* = (x_1^*, x_2^*, \cdots, x_n^*)^T$，（LP）的最优解为 $\mathbf{x}^{LP} = (x_1^{LP}, x_2^{LP}, \cdots, x_n^{LP})^T$
- 线性规划松弛算法
  - 求解（LP）并令 $V' = \left\{ v_i \mid x_i^{LP} \geq \frac{1}{2} \right\}$
  - 线性规划松弛算法的最坏情况界不超过2
  - V'是G的一个顶点覆盖
    - 对E中任意边$v_j v_k$，若 $v_j \notin V', v_k \notin V'$，则 $x_j^{LP} < \frac{1}{2}, x_k^{LP} < \frac{1}{2}$，故 $x_j^{LP} + x_k^{LP} < 1$，$\mathbf{x}^{LP}$ 不是（LP）的可行解
  - $$\sum_{v_i \in V'} w_i \leq 2 \sum_{v_i \in V'} w_i x_i^{LP} \leq 2 \sum_{i=1}^{n} w_i x_i^{LP} \leq 2 \sum_{i=1}^{n} w_i x_i^*$$

V'定义　　松弛性质

可近似性与难近似性

- **WVC的算法**
  - **Local ratio**算法：$2-\frac{\log_2\log_2|V|}{2\log_2|V|}$
  - **SDP**（**semidefinite programming**）松弛算法：$2-\Theta\left(\frac{1}{\sqrt{\log_2|V|}}\right)$
- **VC的难近似性**
  - 若 $\mathcal{P}\neq\mathcal{NP}$，不存在最坏情况界小于 $10\sqrt{5}-21\approx1.3606$ 的多项式时间近似算法

Bar-Yehuda R, Even S. A local-ratio theorem for approximating the weighted vertex cover problem. *Annals of Discrete Mathematics*, 25, 27-45, 1985.
Bar-Yehuda R, Bendel K, Freund A, Rawitz D. Local ratio: A unified framework for approximation algorithms, in memoriam: Shimon even 1935-2004. *ACM Computing Surveys*, 2004, 36, 422-463.
Karakostas G. A better approximation ratio for the vertex cover problem. *ACM Transactions on Algorithms*, 2009, 5, Article No. 41.
Dinur I., Safra S., On the hardness of approximating vertex cover, *Annals of Mathematics*, 162, 439-485, 2005.

近似方案

- 算法族$\{A_\varepsilon\}$称为多项式时间近似方案（**Polynomial Time Approximation Scheme，PTAS**），若对任意给定的ε，算法A_ε的最坏情况界为$1+\varepsilon$，且A_ε的时间复杂性为$f(n)=O(p(n))$，这里$p(x)$为一多项式
- 算法族$\{A_\varepsilon\}$称为完全多项式时间近似方案（**Fully Polynomial Time Approximation Scheme，FPTAS**），若对任意给定的ε，算法A_ε的最坏情况界为$1+\varepsilon$，且A_ε的时间复杂性为$f(n,\varepsilon)=O\left(p\left(n,\frac{1}{\varepsilon}\right)\right)$，这里$p(x,y)$为一二元多项式
  - 时间复杂性为$O\left(\frac{n}{\varepsilon^2}\right)$的算法族可以成为一**FPTAS**，时间复杂性为$O\left(n^{\frac{1}{\varepsilon}}\right)$的算法族可以成为一**PTAS**，但不能成为一**FPTAS**
- 近似方案具有几乎最好的近似性能，但算法实现通常较为复杂，实际运算时间可能很长

PTAS

- 背包问题的**PTAS** P_ε
  - 记 $\mathbb{J}$ 为满足以下要求的物品子集全体
    - 子集中物品数不超过 k 个，其中 $k=\left\lceil\frac{1}{\varepsilon}\right\rceil$　　数量有限
    - 子集中物品大小之和不超过 C　　可行解
  - 对任意 $\mathcal{J}_o\in\mathbb{J}$，按如下步骤得到一可行解 $\mathcal{J}_o^f$
    - 将 $\mathcal{J}_o$ 中的物品放入背包　　初始物品全部放入
    - 按价值密度非增的顺序逐个考虑剩余物品，若物品大小不超过当前背包剩余容量，则放入该物品　　贪婪思想
  - 在 $\{\mathcal{J}_o^f \mid \mathcal{J}_o\in\mathbb{J}\}$ 中，选择放入物品价值之和最大的输出　　复合思想

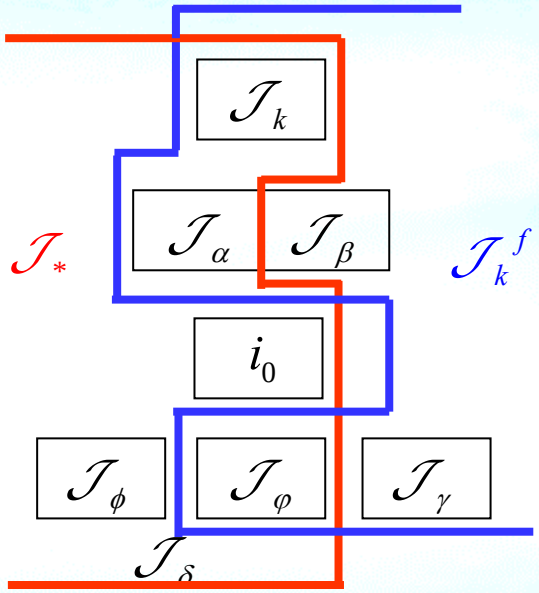
$$|\mathbb{J}|\le\sum_{j=0}^{k}\binom{n}{j}\le kn^k \qquad O(kn^{k+1})=O\left(\frac{1}{\varepsilon}n^{\frac{1}{\varepsilon}+1}\right)$$

PTAS

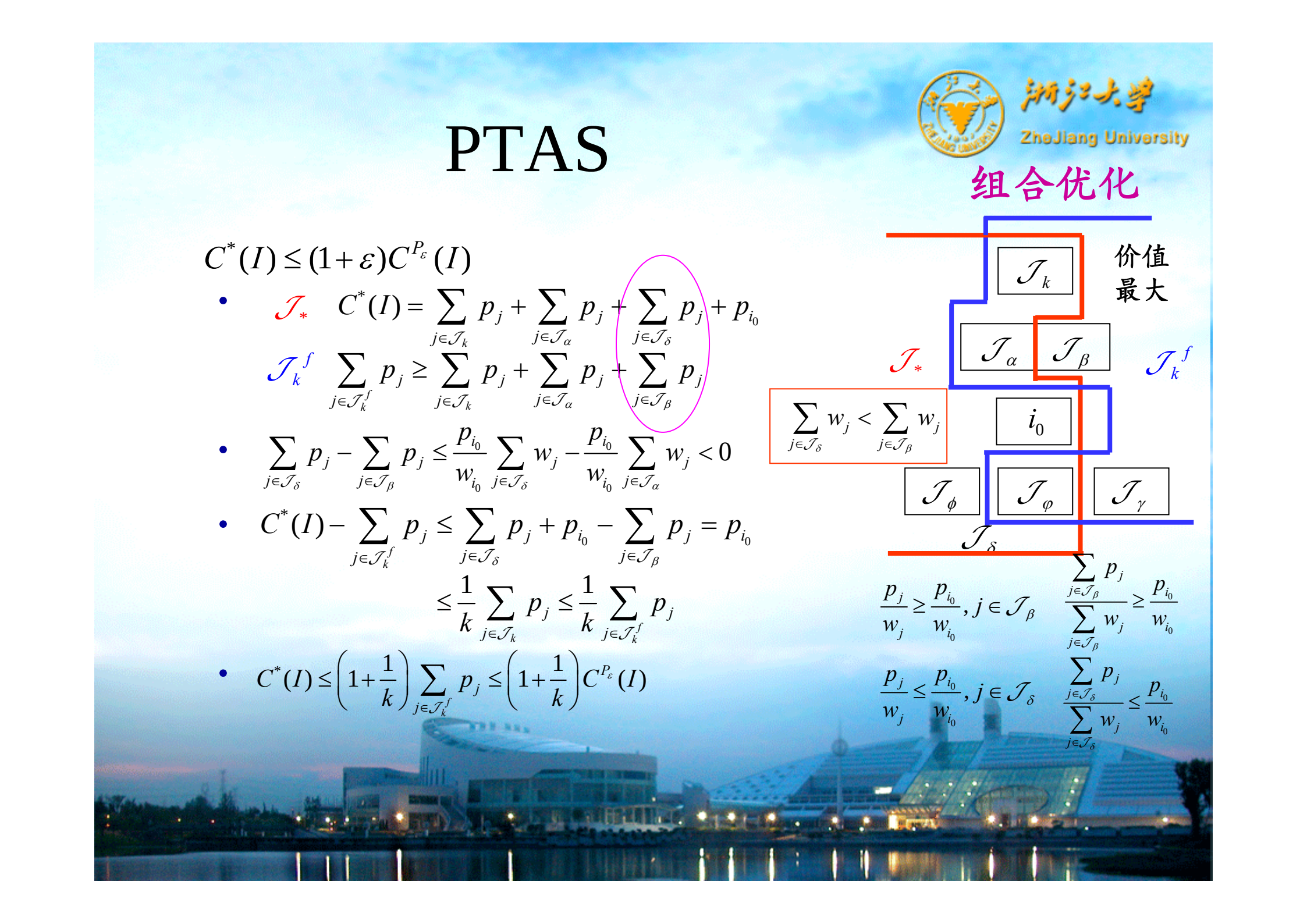
- $C^*(I) \le (1+\varepsilon)C^{P_\varepsilon}(I)$
  - 记最优解中放入背包的物品集为 $\mathcal{J}_*$，若 $|\mathcal{J}_*| \le k$，则 $\mathcal{J}_* \in \mathbb{J}$，$\mathcal{J}_*^f$也为最优解，算法求得最优解
  - 若$|\mathcal{J}_*| > k$，记由 $\mathcal{J}_*$ 中价值最大的 k 个物品组成的集合为 $\mathcal{J}_k$，则 $\mathcal{J}_k \in \mathbb{J}$
  - 若 $\mathcal{J}_* \subseteq \mathcal{J}_k^f$，算法求得最优解。若不然，设 i_0 是 $\mathcal{J}_* \setminus \mathcal{J}_k^f$ 中价值密度最大的物品
    - $\mathcal{J}_\alpha$：物品 i_0 之前放入，也在 $\mathcal{J}_*$中物品
    - $\mathcal{J}_\beta$：物品 i_0 之前放入，不在 $\mathcal{J}_*$中物品
    - $\mathcal{J}_\delta$：物品 i_0 之后所有 $\mathcal{J}_*$ 中物品
  - $\mathcal{J}_*$ $\sum_{j\in\mathcal{J}_k} w_j + \sum_{j\in\mathcal{J}_\alpha} w_j + \sum_{j\in\mathcal{J}_\delta} w_j + w_{i_0} \le C$

    $\mathcal{J}_k^f$ $\sum_{j\in\mathcal{J}_k} w_j + \sum_{j\in\mathcal{J}_\alpha} w_j + \sum_{j\in\mathcal{J}_\beta} w_j + w_{i_0} > C$

    $\Rightarrow \sum_{j\in\mathcal{J}_\delta} w_j < \sum_{j\in\mathcal{J}_\beta} w_j$



PTAS



浙江大学 ZheJiang University

组合优化

$$C^*(I) \le (1+\varepsilon) C^{P_\varepsilon}(I)$$

- $\mathcal{J}_*$ $\quad C^*(I) = \sum_{j\in\mathcal{J}_k} p_j + \sum_{j\in\mathcal{J}_\alpha} p_j + \sum_{j\in\mathcal{J}_\delta} p_j + p_{i_0}$

  $\mathcal{J}_k^f$ $\quad \sum_{j\in\mathcal{J}_k^f} p_j \ge \sum_{j\in\mathcal{J}_k} p_j + \sum_{j\in\mathcal{J}_\alpha} p_j + \sum_{j\in\mathcal{J}_\beta} p_j$

- $\sum_{j\in\mathcal{J}_\delta} p_j - \sum_{j\in\mathcal{J}_\beta} p_j \le \frac{p_{i_0}}{w_{i_0}} \sum_{j\in\mathcal{J}_\delta} w_j - \frac{p_{i_0}}{w_{i_0}} \sum_{j\in\mathcal{J}_\alpha} w_j < 0$

- $C^*(I) - \sum_{j\in\mathcal{J}_k^f} p_j \le \sum_{j\in\mathcal{J}_\delta} p_j + p_{i_0} - \sum_{j\in\mathcal{J}_\beta} p_j = p_{i_0}$

  $\le \frac{1}{k} \sum_{j\in\mathcal{J}_k} p_j \le \frac{1}{k} \sum_{j\in\mathcal{J}_k^f} p_j$

- $C^*(I) \le \left(1+\frac{1}{k}\right) \sum_{j\in\mathcal{J}_k^f} p_j \le \left(1+\frac{1}{k}\right) C^{P_\varepsilon}(I)$

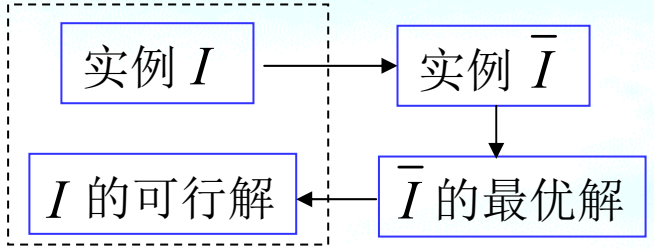
$\frac{p_j}{w_j} \ge \frac{p_{i_0}}{w_{i_0}}, j \in \mathcal{J}_\beta$ $\quad \frac{\sum_{j\in\mathcal{J}_\beta} p_j}{\sum_{j\in\mathcal{J}_\beta} w_j} \ge \frac{p_{i_0}}{w_{i_0}}$

$\frac{p_j}{w_j} \le \frac{p_{i_0}}{w_{i_0}}, j \in \mathcal{J}_\delta$ $\quad \frac{\sum_{j\in\mathcal{J}_\delta} p_j}{\sum_{j\in\mathcal{J}_\delta} w_j} \le \frac{p_{i_0}}{w_{i_0}}$

FPTAS

- 背包问题的**FPTAS** FP_ε
  - （缩微）构造背包的一个新实例 $\bar{I}$：背包容量仍为 C，物品数仍为 n，物品 j 的价值和大小分别为 $\bar{p}_j = \left\lfloor \frac{p_j}{2^t} \right\rfloor$ 和 $\bar{w}_j = w_j$，其中 $t = \left\lfloor \log_2 \left(\frac{\varepsilon}{1+\varepsilon} \cdot \frac{p_{\max}}{n} \right) \right\rfloor$，$p_{\max} = \max_{1 \le i \le n} p_i$
  - （求解）用动态规划**DPV**求解实例 $\bar{I}$。在 $\bar{I}$ 的最优解中，标号在 $\bar{J}^*$ 中的物品被放入背包
  - （反馈）将标号在 $\bar{J}^*$ 中的物品放入背包，得到原实例 I 的一个近似解

$$\sum_{j=1}^{n} \bar{p}_j \le \sum_{j=1}^{n} \frac{p_j}{2^t} \le \frac{n p_{\max}}{2^t} \le 2n^2 \frac{1+\varepsilon}{\varepsilon} \qquad O\left(n \sum_{i=1}^{n} \bar{p}_j \right) = O\left(n^3 \frac{1}{\varepsilon} \right)$$



动态规划**DPV**

$O\left(n \sum_{i=1}^{n} p_j \right)$

优点：最优算法

缺点：非多项式时间算法

优点：伪多项式时间算法

适合求解价值小的实例

为什么不用**DPS** $O(nC)$

$\bar{C} = \left\lfloor \frac{C}{2^t} \right\rfloor$ $\bar{w}_j = \left\lfloor \frac{w_j}{2^t} \right\rfloor$

$O\left(n \sum_{i=1}^{n} \bar{p}_j \right)$

FPTAS

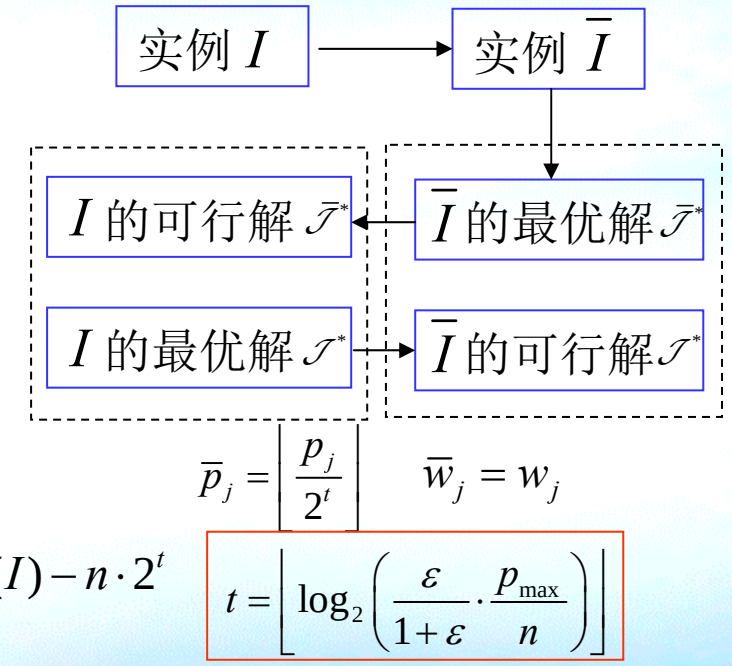
- $C^*(I) \le (1+\varepsilon) C^{FP_\varepsilon}(I)$
  - $\sum_{j\in\mathcal{T}^*} \left\lfloor \frac{p_j}{2^t} \right\rfloor = \sum_{j\in\mathcal{T}^*} \bar{p}_j \le \sum_{j\in\bar{\mathcal{T}}^*} \bar{p}_j = \sum_{j\in\bar{\mathcal{T}}^*} \left\lfloor \frac{p_j}{2^t} \right\rfloor$

    $\bar{\mathcal{T}}^*$ 是 $\bar{I}$ 的最优解
  - $C^{FP_\varepsilon}(I) = \sum_{j\in\bar{\mathcal{T}}^*} p_j = 2^t \sum_{j\in\bar{\mathcal{T}}^*} \frac{p_j}{2^t} \ge 2^t \sum_{j\in\bar{\mathcal{T}}^*} \left\lfloor \frac{p_j}{2^t} \right\rfloor$

    $\ge 2^t \sum_{j\in\mathcal{T}^*} \left\lfloor \frac{p_j}{2^t} \right\rfloor \ge 2^t \sum_{j\in\mathcal{T}^*} \left(\frac{p_j}{2^t} - 1 \right)$

    $= \sum_{j\in\mathcal{T}^*} p_j - 2^t \sum_{j\in\mathcal{T}^*} 1 \ge \sum_{j\in\mathcal{T}^*} p_j - n\cdot 2^t = C^*(I) - n\cdot 2^t$
  - $\frac{C^*(I)}{C^{FP_\varepsilon}(I)} \le \frac{C^*(I)}{C^*(I) - n2^t} = \frac{1}{1 - \frac{n\cdot 2^t}{C^*(I)}} \le \frac{1}{1 - \frac{n\cdot 2^t}{p_{\max}}} = \frac{p_{\max}}{p_{\max} - n\cdot 2^t} \le \frac{p_{\max}}{p_{\max} - n\frac{\varepsilon}{1+\varepsilon}\frac{p_{\max}}{n}} = 1+\varepsilon$

    $C^*(I) \ge p_{\max}$



背包问题的近似方案

浙江大学 ZheJiang University

组合优化

近似方案	PTAS	FPTAS
设计思路	有限枚举	动态规划求解缩微实例
误差来源	从有限到全部	以缩微实例最优解为近似解
时间复杂性	$O\left(\frac{1}{\varepsilon}n^{\frac{1}{\varepsilon}+1}\right)$ $O\left(n^{\frac{1}{\varepsilon}-2}\right)$	$O\left(n^3\frac{1}{\varepsilon}\right)$ $O\left(n\min\left\{\ln n,\ln\frac{1}{\varepsilon}\right\}+\frac{1}{\varepsilon^2}\ln\frac{1}{\varepsilon}\min\left\{n,\frac{1}{\varepsilon}\ln\frac{1}{\varepsilon}\right\}\right)$

Caprara A, Kellerer H, Pferschy U, Pisinger D. Approximation algorithms for knapsack problems with cardinality constraints. *European Journal of Operational Research*, 123, 333-345, 2000.

Kellerer H, Pferschy U. A new fully polynomial time approximation scheme for the knapsack problem. *Journal of Combinatorial Optimization*, 3, 59-71, 1999.

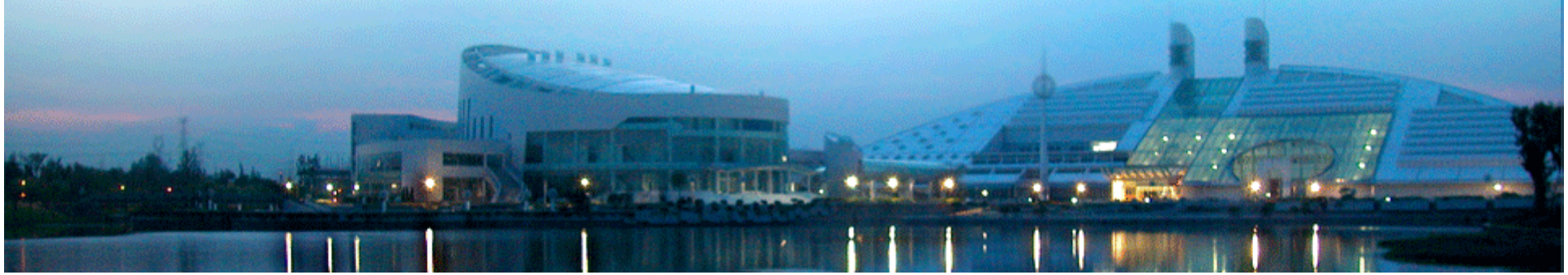
Kellerer H, Pferschy U. Improved dynamic programming in connection with an FPTAS for the knapsack problem. *Journal of Combinatorial Optimization*, 8, 5-11, 2004.

近似方案的存在性

- 设 Π 为（极大化）强$\mathcal{NP}$ −难问题，且存在二元多项式 $q(x,y)$，使得 $C^*(I) \le q(\text{size}(I), \text{Max}(I))$ ，则 Π 不存在**FPTAS**，除非 $\mathcal{P}=\mathcal{NP}$
  - 设 A_ε 为 Π 的一**FPTAS**，对 Π 的任一实例 I ，取 $\varepsilon_0 = \dfrac{1}{q(\text{size}(I), \text{Max}(I))+1}$
    - A_{ε_0} 的时间复杂性为

$$p\left(\text{size}(I), \frac{1}{\varepsilon_0}\right) = p(\text{size}(I), q(\text{size}(I), \text{Max}(I))+1) = p'(\text{size}(I), \text{Max}(I))$$

    - $C^*(I) \le (1+\varepsilon_0) C^{A_{\varepsilon_0}}(I)$
  - $$C^*(I) - C^{A_{\varepsilon_0}}(I) = \varepsilon_0 C^{A_{\varepsilon_0}}(I) \le \varepsilon_0 C^*(I) \le \frac{q(\text{size}(I), \text{Max}(I))}{q(\text{size}(I), \text{Max}(I))+1} < 1$$
    - 由于 $C^*(I)$ 与 $C^{A_{\varepsilon_0}}(I)$ 均为整数，故 $C^*(I) = C^{A_{\varepsilon_0}}(I)$
  - A_ε 可成为求解 Π 的伪多项式时间算法　矛盾



积划分

- 积划分（**Product Partition**）问题：给定正整数集 $A=\{a_1,a_2,\cdots,a_n\}$，问是否存在子集 $A'\subseteq A$，使得 $\prod_{i\in A'} a_i = \prod_{i\in A\setminus A'} a_i$
- 极小化积划分（**Product Partition**）问题：给定正整数集 $A=\{a_1,a_2,\cdots,a_n\}$，求子集 $A'\subseteq A$，使得 $\max\left\{\prod_{i\in A'} a_i, \prod_{i\in A\setminus A'} a_i\right\}$ 最小
- 积划分问题是强$\mathcal{NP}$-完全的；极小化积划分问题是强$\mathcal{NP}$-难的，但存在**FPTAS**
  - 极小化积划分问题实例规模为 $n+\log \max_{i=1,\cdots,n} a_i$，但 $C^*(I)\geq\left(\prod_{i=1}^{n} a_i\right)^{\frac{1}{2}}$

Ng CT, Barketau MS, Cheng TCE, Kovalyov MY. "Product Partition" and related problems of scheduling and systems reliability: Computational complexity and approximation. *European Journal of Operational Research*, 207, 601-604, 2010.

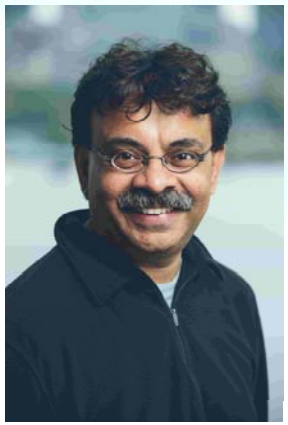
度量TSP的难近似性

浙江大学 ZheJiang University

组合优化

- **1992年，Arora等人提出了概率可验证明（Probabilistically Checkable Proofs，PCP）的新技术，并证明了度量TSP不存在PTAS**
- **度量TSP不存在最坏情况界小于 $\frac{123}{122}$ 的多项式时间近似算法**

Sanjeev Arora
印裔美籍
计算机科学家
(1968-)



Madhu Sudan
印裔美籍
计算机科学家
(1966-)

2002年国际数学家大会（IMU）Rolf Nevanlinna Prize得主

Arora S, Safra S. Probabilistic checking of proofs: A new characterization of NP. *Journal of the ACM*, 45, 70-122, 1998.
Arora, S., Lund, C., Motwani, R., Sudan, M., Szegedy, M. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45, 501-555, 1998.
Karpinski M, Lampis M, Schmied R. New inapproximability bounds for TSP, *Proceedings of the 24th International Symposium on Algorithms and Computation, Lecture Notes in Computer Science*, 8283, 568-578, 2013.

Euclidean TSP的PTAS

- **1996年，Arora和Joseph S.B. Mitchell独立给出了Euclidean TSP的PTAS**

History. The current paper evolved out of preliminary results obtained in January 1996, culminating in a submission to *IEEE FOCS 1996* in April 1996 [Arora 1996]. A few weeks later, Mitchell [1996/1998] independently discovered an $n^{O(c)}$ time approximation scheme for points in $\mathcal{R}^2$. His algorithm used ideas from his earlier constant-factor approximation algorithm for k-MST [Mitchell 1996]. It relies on the geometry of the plane and does not seem to generalize to higher dimensions. In January 1997, the author discovered the nearly-linear-time algorithm described in this paper. The key ingredient of this algorithm is Theorem 2.1.2, which the author had originally conjectured to be false. He is grateful to Karen Wang, whose inability [1996] to construct any counterexample to Theorem 2.1.2 motivated him to abandon his conjecture.

Arora S. Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems. *Journal of the ACM*, 45, 753-782, 1998.

Euclidean TSP的PTAS

浙江大学 ZheJiang University

组合优化

In an exciting recent development, Arora [1] announced that he had found a PTAS for the Euclidean TSP, as well as the other problems considered in this paper, thereby achieving essentially the same results that we report here, using decomposition schemes that are somewhat similar to our own. Arora's remarkable results predate this paper by several weeks. Arora has generalized his method also to higher dimensions, obtaining a running time of $n^{O(\log^{d-2} n)/\epsilon^{d-1}}$ in d dimensions. For the two-dimensional TSP, Arora estimates that his analysis yields a time bound of roughly $n^{100/\epsilon}$; he adds that a more careful analysis should yield roughly $n^{30/\epsilon}$. In this paper, we give analysis giving an explicit exponent: time $O(n^{20m+5})$ to get within factor $(1+\frac{2\sqrt{2}}{m})$ of optimal in the Euclidean planar TSP; in terms of ϵ, the time bound is $O(n^{\frac{40\sqrt{2}}{\epsilon}+5})$.

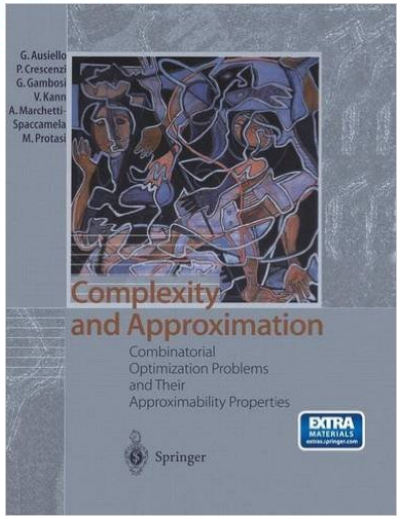
Mitchell JSB. Guillotine subdivisions approximate polygonal subdivisions: A simple polynomial-time approximation scheme for geometric TSP, k-MST, and related problems. *SIAM Journal on Computing*, 28, 1298-1309, 1999.

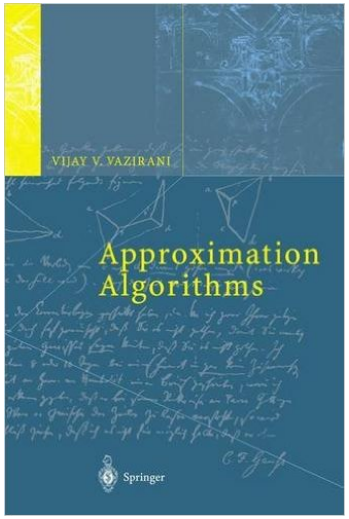
2010年Arora和Mitchell共同获得Godel奖，该奖项由European Association for Theoretical Computer Science和the Special Interest Group on Algorithms and Computation Theory of ACM颁发，授予"outstanding papers in the area of theoretical computer science "

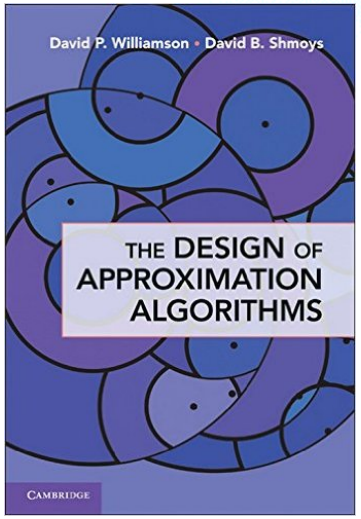
参考资料

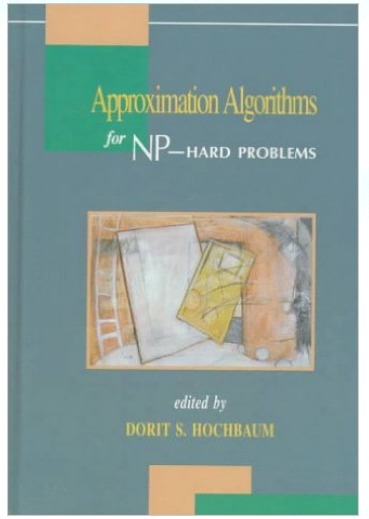
浙江大学 ZheJiang University

组合优化









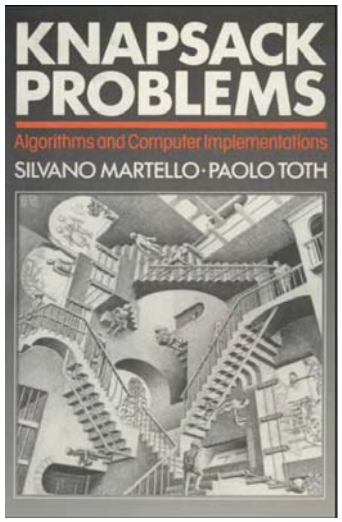
Ausiello G, Crescenzi P, Gambosi G, Kann V, Marchetti-Spaccamela A, Protasi, M. ***Complexity and approximation: Combinatorial optimization problems and their approximability properties*****. Springer, 2003.**

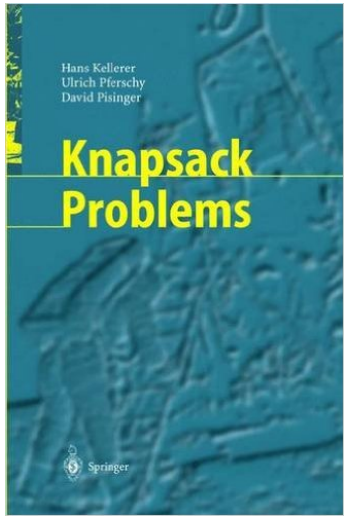
Vazirani VV. ***Approximation Algorithms*****, Springer, 2004.**

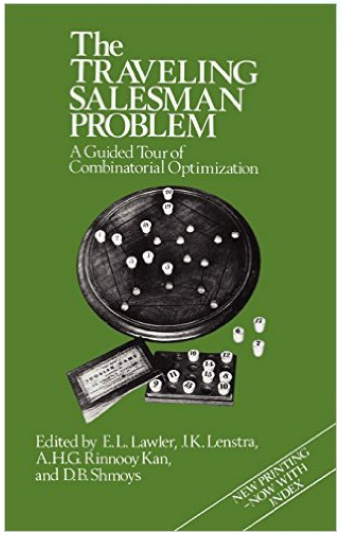
Williamson DP, Shmoys DB. ***The Design of Approximation Algorithms*****, Cambridge University Press, 2011.**

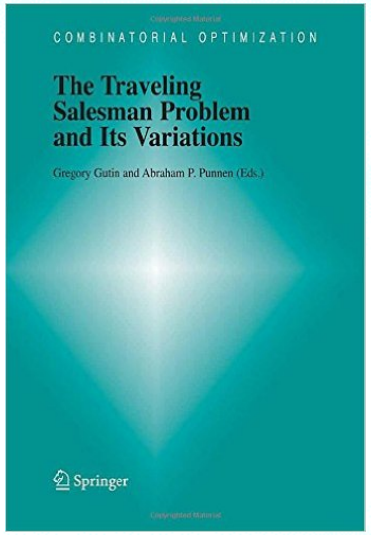
Hochbaum DS (Eds.). ***Approximation Algorithms for NP-Hard Problems*****. PWS Publishing, 1997.**

参考资料









Martello S, Toth P. *Knapsack Problems: Algorithms and Computer Implementations*, Wiley, 1990.

Kellerer H, Pferschy U, Pisinger D. *Knapsack Problems*. Springer, 2004.

Lawler EL, Lenstra JK, Rinnooy Kan AHG, Shmoys DB. *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, Wiley, 1985

Gutin G, Punnen AP (Eds.). *The Traveling Salesman Problem and Its Variations*, Springer, 2002.


浙江大学
ZheJiang University
谢　谢