货郎问题

货郎问题的定义

- 输入

  有穷个城市的集合 $C=\{c_1,c_2,\ldots,c_n\}$,

  距离 $d(c_i,c_j)=d(c_j,c_i)\in Z^+,\ 1\le i<j\le n$

- 解：$1,2,\ldots,n$ 的排列 $k_1,k_2,\ldots,k_n$ 使得：

$$\min\{\sum_{i=1}^{n-1} d(c_{k_i},c_{k_{i+1}})+d(c_{k_n},c_{k_1})\}$$

算法设计

解向量为 $<1, i_1, i_2, \ldots, i_{n-1}>$, 其中 $i_1, i_2, \ldots, i_{n-1}$ 为 $\{2,3,\ldots,n\}$ 的排列.

搜索空间为排列树，结点 $<i_1, i_2, \ldots, i_k>$ 表示得到 k 步路线 .

约束条件：令 $B = \{ i_1, i_2, \ldots, i_k \}$, 则

$$i_{k+1} \in \{ 2, \ldots, n \} - B$$

即每个结点只能访问一次 .

代价函数与界

界：当前得到的最短巡回路线长度

代价函数：设顶点 c_i 出发的最短边长度为 l_i，d_j 为选定巡回路线中第 j 段的长度

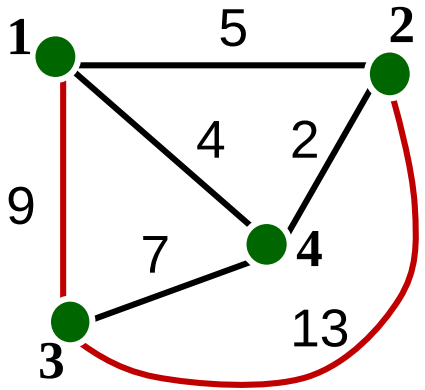
$$L = \sum_{j=1}^{k} d_j + l_{i_k} + \sum_{i_j \notin B} l_{i_j}$$

已走过路径长

剩余长度下界

代价函数

$$L=\sum_{j=1}^{k} d_j + l_{i_k} + \sum_{i_j \notin B} l_{i_j}$$



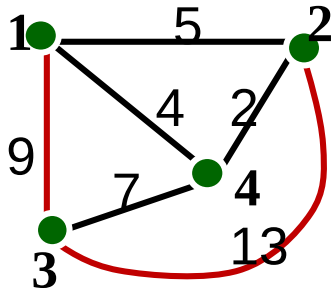
部分路线<1,3,2>

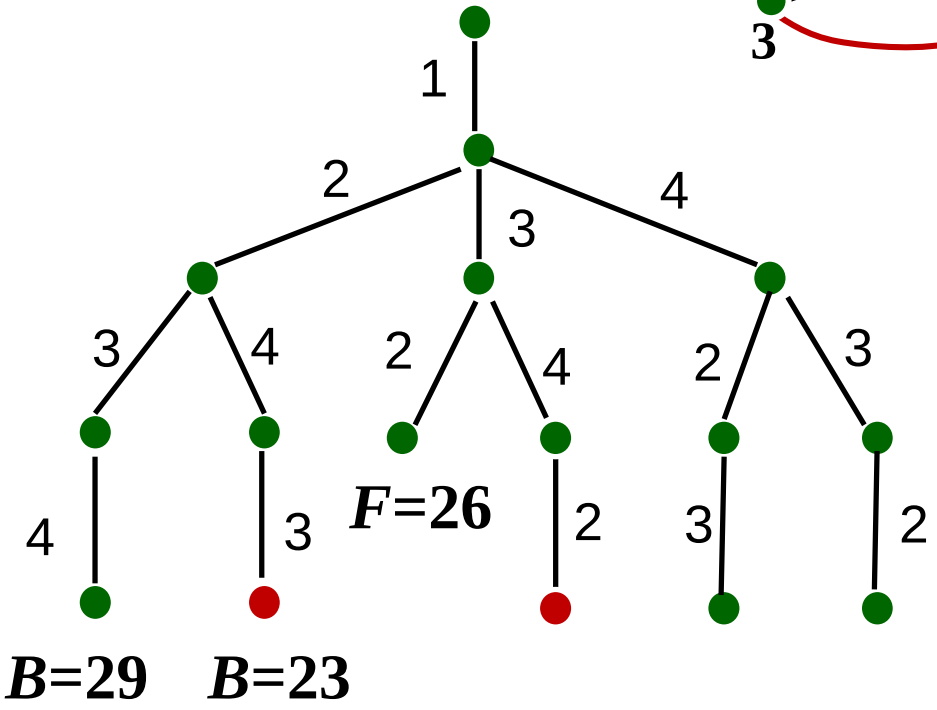
L=9+13+2+2=26

9+13为走过的路径长度

后两项分别为从结点2及4出发的最短边长

搜索树





实例运行

深度优先遍历搜索树

- 第一个界: <1,2,3,4>，长度为 29
- 第二个界: <1,2,4,3>，长度为 23
- 结点 <1,3,2>: 代价函数值 26>23，不再搜索, 返回<1,3>,右子树向下
- 结点<1,3,4>，代价函数值 9+7+2+2=20，继续，得到可行解 <1,3,4,2>，长度 23.
- 回溯到结点<1>，沿<1,4>向下...

最优解<1,2,4,3>或<1,3,4,2>，长度23

算法分析

- 搜索树的树叶个数：$O((n-1)!)$，每片树叶对应 1 条路径，每条路径有 n 个结点.
- 每个结点代价函数计算时间$O(1)$，每条路径的计算时间为 $O(n)$
- 最坏情况下算法的时间 $O(n!)$

小结

- 货郎问题的分支限界算法：
  约束条件：只能选没有走过的结点
  代价函数：走过长度+后续长度的
  下界

- 时间复杂度：$O(n!)$