算法设计与分析 Week 7

第七次作业-solution

Lecturer: 杨启哲 Last modified: 2025 年 11 月 16 日

1. (最大生成树) 请给出一个算法，求出一个无向图的最大生成树，即权重最大的生成树。

   解答. 注意到 Prim 算法可以处理负权重的边，因此我们可以利用这一点来设计求最大生成树的算法:

   (1) 将图中所有边的权重取相反数。
   (2) 运行 Prim 算法，得到该图的最小生成树，即为原图的最大生成树。

   □

2. Alice 想要举办一个舞会，为此需要决定邀请什么人参加。目前共有 n 个人可供选择，Alice 根据他们之间是否相互认识列出一个相互配对的列表。她希望在邀请的人能在满足每个人都最少可以找到 5 个认识的和 5 个不认识的人的前提下，尽可能多地邀请人参加舞会。请设计一个高效的算法，帮助 Alice 决定邀请哪些人参加舞会。

   算法的输入是一个 n 个人的列表，以及相识的配对列表，输出的是最优的邀请人列表。

   解答. 我们将问题转换成图论，构造一个无向图 $G = (V, E)$，其中每个顶点表示一个人，如果两个人相识，则在它们之间连一条边。现在我们需要找出图中的一个最大顶点集 S，使得对于任意 $v \in S$，$5 \leqslant \deg(v) \leqslant n - 5$。我们有如下观察:

   Lemma 0.1

   若存在点 v，使得 $\deg(v) < 5$，则任意一个满足要求的 S 都不会包含 v。

   Lemma 0.2

   若存在点 v，使得 $\deg(v) > n - 5$，则任意一个满足要求的 S 都不会包含 v。

   引理0.1的证明是容易的，因为删去任何一个点不会增加 v 的度数，因此如果 v 的度数小于 5，则无论如何删去其他点，v 的度数都不会大于等于 5。引理0.2的证明则略微更精妙些，但只需注意到图的总点数也会随着减少，因此删去一个点也并不会增加 v 的度数和途中总点数的值。

   因此如下的算法是显然的，我们只需要不断删去度数小于 5 或大于 $n - 5$ 的点，直到图中不存在这样的点为止。此时剩余的图中的每个点都满足度数在 5 到 $n - 5$ 之间，因此剩余的图中的所有点构成了一个合法的邀请列表。算法简单描述如下:

算法: 舞会邀请

(1) 构造图 $G = (V, E)$，其中每个顶点表示一个人，如果两个人相识，则在它们之间连一条边。

(2) 重复以下操作直到图中不存在度数小于 5 或大于 $n - 5$ 的点为止:

(i) 如果存在点 v，使得 $\deg(v) < 5$，则删去 v。

(ii) 如果存在点 v，使得 $\deg(v) > n - 5$，则删去 v。

(3) 返回剩余图中的所有顶点作为邀请列表。

算法的正确性由引理0.1和引理0.2保证。现在来考虑其时间，注意到最多进行 n 次删去操作，每次删去一个点需要更新其邻接点的度数，这个过程是 $O(n)$ 的，因此整个算法的时间复杂度为 $O(n^2)$。 □

3. 给定一个无向图 G 和其中的一条边 e，请设计一个线性时间算法，判断图中是否存在一个包含 e 的环。

解答. 基本思路在于假设 e 的两个端点为 u 和 v，如果图中删去 e，u 和 v 不连通，则 e 不在环中；否则，e 在环中。算法简单描述如下:

(1) 构造图 $G' = (V, E')$ 满足 $E' = E \setminus \{e\}$。

(2) 在图 G' 从 u 出发进行 DFS，如果 DFS 能访问到 v，则 e 在环中；否则，e 不在环中。

□

4. 我们考虑一个新的方法来得到最小生成树:

(1) 请证明如下性质：选择图中任何一个环，删除其中权重最大的边，得到的图的最小生成树仍然是原图的最小生成树。

(2) 考虑如下的算法:

算法:MST-DeleteCycle

(i) 将图中的边按照权重从大到小排序。

(ii) 按排好的顺序考虑每一条边，如果这条边在图中形成了环，则删除这条边。

(iii) 直到图中没有环为止。返回剩下的图。

请证明算法的正确性。

解答.

(1) 反设结论不成立。即存在一个圈 φ 删去其权重最大的边 e 以后，得到的最小生成树不相同。令删掉 e 后的图为 G'，相应的最小生成树为 T, T'。注意到 G 和 G' 具有相同的顶点，从而 T' 也是 G 的生成树，从而 $e \in T$, 否则 T 不是最小生成树。

现在考虑 $T-\{e\}$，由最小生成树的性质，其必然是不连通的，并且其将 G 划分成了 V_1, V_2 两个点集，满足 e 是其中一条横跨的边。由于 φ 是一个包含 e 的圈，因此 φ 中必然还存在一条横跨 V_1, V_2 的边 e'，由假设 e' 的权重小于 e，从而 $T-\{e\}\cup\{e'\}$ 是一个更小的生成树这与 T 是最小生成树矛盾，因此结论成立。

(2) 令每一轮后剩余的图为 G_i，由上一问的结论我们有：

Lemma 0.3

G_i 的最小生成树是 G_{i-1} 的最小生成树。

注意到算法最终返回的也是 G 的生成树，因此算法的正确性得证。

(3) 令 m 表示边的个数，n 表示顶点的个数。整个算法的时间如下：

(i) 排序的时间复杂度为 $O(m\log m)$。

(ii) 每次判断是否形成圈的时间复杂度为 $O(m)$。循环至多执行了 $m-n+1$ 次。

因此整个算法的时间复杂度为 $O(m^2)$。

□

5. (最长回文子序列) 如果一个子序列从左向右和从右向左读都一样，则称之为回文。例如，序列：

$$A, C, G, T, G, T, C, A, A, A, A, T, C, G$$

有很多回文子序列，如 A, C, G, C, A 和 A, A, A, A。请设计一个 $O(n^2)$ 时间的算法，对于输入一个长度为 n 的序列，找出其最长回文子序列的长度。

解答. 我们可以利用动态规划的思想来解决这个问题。x 需要注意的是，因为是回文子序列，所以并不一定连续。因此记 $L[i][j]$ 为序列 $A[i], \ldots, A[j]$ 的最长回文子序列的长度，则我们有如下递推关系：

$$L[i][j] = \begin{cases} 1 & i = j \\ 2 & i = j-1, A[i] = A[j] \\ L[i+1][j-1]+2 & i < j-1, A[i] = A[j] \\ \max\{L[i+1][j], L[i][j-1]\} & i < j, A[i] \neq A[j] \end{cases}$$

根据上述递推关系可以构造如下算法：

(1) 初始化 $L[i][i] = 1$ 和所有的 $L[i][i+1]$。

(2) 根据 $L[i][j]$ 中 $j-i$ 的大小从 2 到 $n-1$ 开始循环，依次根据上述关系更新每组的 $L[i][j]$。

简单的伪代码如下：

最长回文子序列

输入: 序列 $A[1, \ldots, n]$

输出: 最长回文子序列的长度

```
for i = 1 to n do
    L[i][i] ← 1
end for
for i = 1 to n − 1 do
    if A[i] = A[i + 1] then
        L[i][i + 1] ← 2
    else
        L[i][i + 1] ← 1
    end if
end for
for k = 2 to n − 1 do
    for i = 1 to n − k do
        j ← i + k
        if A[i] = A[j] then
            L[i][j] ← L[i + 1][j − 1] + 2
        else
            L[i][j] ← max{L[i + 1][j], L[i][j − 1]}
        end if
    end for
end for
return L[1][n]
```

算法需要进行 $n \times n$ 的一个循环，每次循环的操作次数是常数次，所以时间复杂度为 $O(n^2)$。 □