# Homework 7

PB17111623 范睿

2019 年 11 月 25 日

## 1

### 1.1 给定一个有向无环图G=(V,E),边权重为实数，给定图中两个顶点s和t。设计动态规划算法，求从s到t的最长加权简单路径。

设weight[s][t]为s与t间的最长加权路径的权重，有递归式：

$$weight[s][t] = \max_{p \in s.S}(weight[p][t] + w[s][t])$$

其中，s.S表示所有s指向的结点的集合，w[s][t]表示st这条边的边权。

**Algorithm 1** Weight

**Require:** 图G，顶点s，顶点t (weight数组初始均为-1)

**Ensure:** 最长边权w

**if** s is t **then**
    $weight[s][t] \leftarrow 0$
**return** 0
**end if**
**for** $u \in V$ **do**
    **for** $v \in V$ **do**
        **for** $p \ \in \ V \ and \ pq \ \in E$ **do**
            **if** $weight[u][p] \ is \ -1$ **then**
                $weight[u][p] \leftarrow Weight(G, u, p)$
            **end if**
            $weight[u][v] = \max(path[u][j], w[p][j] + weight[i][p])$
        **end for**
    **end for**
**end for**
**return** weight[s][t]

### 1.2 设定动态规划算法求解 0-1 背包问题，要求运行时间为 O(nW), n 为商品数量，W 是小偷能放进背包的最大商品总重量。

**Algorithm 2**

**Require:** 商品数量n，最大重量W，每个商品的重量和价值$w_i, v_i, 1 \le i \le n$
**Ensure:** 最大价值

let A[0...W][1...n+1] be a new array
**for** i = 0 to W **do**
  $A[i][n+1] \leftarrow 0$
**end for**
**for** i = n down to 1 **do**
  **for** j = 0 to W **do**
    $A[i][j] \leftarrow 0$
    **if** $j < w_i$ **then**
      $A[i][j] \leftarrow A[i][j+1]$
    **else**
      $value \leftarrow A[i-w_i][j+1] + v_j$
      **if** $value \ge A[i][j]$ **then**
        $A[i][j] \leftarrow value$
      **end if**
    **end if**
  **end for**
**end forreturn** A[W][1]

### 1.3 一位公司主席正在向 Stewart 教授咨询公司聚会方案。公司的内部结构关系是层次化的，即员工按主管-下属关系构成一棵树，根结点为公司主席。人事部按“宴会交际能力”为每个员工打分，分值为实数。为了使所有参加聚会的员工都感到愉快，主席不希望员工及其直接主管同时出席。公司主席向 Stewart 教授提供公司结构树，采用左孩子右兄弟表示法（参见课本 10.4 节）描述。每个节点除了保存指针外，还保存员工的名字和宴会交际评分。设计算法，求宴会交际评分之和最大的宾客名单。分析算法复杂度。

寻找子结构：
令score(T)表示以T为根的树的评分最大值，有递归式:

$$score(T) = max(T.score + \textstyle\sum_{p \in T'sgrandson} score(p), \sum_{p \in T'schildren} score(p))$$

**Algorithm 3**

**Require:** 树T

**Ensure:** 最大评分scoremax

将树T转换成堆的形式存在数组A[1...n]中,n为总人数
let B[1...n] be a new array
**for** i = 1 to n **do**
  $B[i] \leftarrow 0$
**end for**
**for** i = n downto 1 **do**
  **if** $lchild(i) > n$ **then**
    $B[i] \leftarrow score_i$
  **else**
    **if** $grandson(i) > n$ **then**
      $value \leftarrow score_i + score_{rchild(i)} + score_{rchild(rchild(i))} \cdots$
      $B[i] \leftarrow value > score_{lchild(i)} ? value : score_{lchild(i)}$
    **else**
      $value1 \leftarrow score_i + score_{lchild(lchild(i))}$
      $value2 \leftarrow score_{lchild(i)}$
      $B[i] \leftarrow value1 > value2 ? value1 : value2$
    **end if**
  **end if**
**end forreturn** B[1]

复杂度为$\mathcal{O}(n)$n为人数

## 1.4 ．设计一个高效的算法，对实数线上给定的一个点集 x1，x2,...xn，求一个单位长度闭区间的集合，包含所有给定的点，并要求此集合最小。证明你的算法是正确的。

先找到最左边的点，将单位区间的左端点与之重合，用单位长度区间覆盖掉，然后把所有被覆盖掉的点去掉，再找最左边的点，重复，直到所有点全部被覆盖掉。证明是最小：
1.最左边的点一定会被某一个区间覆盖
2. 如果从最左边的点的左边开始放置区间，则会将区间左边点至最左边的区域浪费掉，有更优解，即提出算法。

## 1.5 考虑用最少的硬币找 n 美分零钱的问题。假定每种硬币的面额都是整数。设计贪心算法求解找零问题，假定有 25 美分、10 美分、5 美分和 1 美分四种面额的硬币。证明你的算法能找到最优解。

先找最多的25美分，再找最多的10美分，再找最多的5美分，剩下的全用1美分找。
1. 同样价格的前，大面值美分的个数比小面值美分的个数少。 2. 如果将本算法中的大面值用小面值替换，将会得到更多的硬币，不是最优解。