Homework 7

PB17000297 罗晏宸

November 12 2019

1 Problem 15-1 Longest simple path in a directed acyclic graph

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。

解 由于 G 是有向无环图，且算法要求的是一条简单路径。因此不需要考虑回路，记 $W(uv)$ 是边 uv 的权重，记 $L(G,p,q)$ 是在图 G 中从 p 到 q 的最长加权简单路径的长度，给出转移方程如下

$$L(G,p,q)=\begin{cases}\max\limits_{pp'\in E}\{W(pp')+L\left(G\setminus\{p\},p',q\right)\}, & p\neq q\\ 0, & p=q\end{cases} \tag{1}$$

记录算法中每一次最大值计算对应的路径，即可得到最长加权简单路径。

2 Exercise 16.2-2

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。

解 设 $F[i,w]$ 表示前 i 件物品恰放入一个容量为 w 的背包可以获得的最大价值，给出状态转移方程：

$$F[i,w]=\max\left\{F[i-1,w],\,F[i-1,w-w_i]+v_i\right\} \tag{2}$$

其中 w_i 与 v_i 分别是第 i 件物品的重量和价值。

据此给出算法伪代码如下：

0-1 KNAPSACK(n, W, w, v)

$F[0, 0..W] = 0$
for $i = 1$ **to** n
    for $w = w_i$ **to** W
        $F[i, w] = \max\{F[i-1, w],\, F[i-1, w-w_i] + v_i\}$
return $F[n, W]$

3 Problem 15-6 Planning a company party

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。

解 由于主席不希望员工及其直接主管同时出席，对于结构树上的一个结点，若将其加入宾客名单，则不能将其左孩子及其左孩子的全部右兄弟加入名单。设 T 表示公司结构树，其属性 $root$ 对应公司主席、$number$ 对应全公司人数。$C[i]$ 表示结构树上以结点 i 为根的子树的最大宴会交际评分之和，其中 i 是结点在 T 中的后序遍历序号，记 $GuestList[i]$ 是 $C[i]$ 对应的宾客名单。

```
COMPANY PARTY PLAN(T)
for i = 1 to T.number
    if i.left-child == NIL
        C[i] = i.conviviality
        GuestList[i] = {i.name}
    else
        C[i] = max{ Σ_{j.parent=i} C[j], Σ_{j.parent.parent=i} C[j] + i.conviviality }
        add arg max{ {j|j.parent = i}, {j|j.parent.parent = i} } into
        GuestList[i]
return GuestList[T.root]
```

由于在循环中需要向前寻找 i 的孩子和孙子，所以算法的时间复杂度是 $O(n^2)$ 的。

4 Exercise 16.2-5

设计一个高效的算法，对实数线上给定的一个点集 $x_1, x_2, \cdots, x_n$ ，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。

解 本题可以用贪心的思想解决，对于给定的点集，设其中最小的点是 x_{min}，则在给出的单位长度闭区间集合 $s_1, s_2, \cdots, s_m$（假设按左端点从小到大排列）中，在一个最优解中一定有 $x_{min} \in s_1$，且 $s_1 = [x_{min}, x_{min}+1]$，因为若 $\nexists x \in (x_{min}, x_{min}+1]$，则在 $[x_{min}-1, x_{min}+1]$ 中的任意单位长度闭区间或 s_1 对于最后的集合元素个数都没有影响，否则 s_1 包含的 $[x_{min}, x_{min}+1]$ 中的点一定不少于任何单位长度闭区间。则问题可以递归的转化为逐点贪心的过程，得到的集合是最优的。设 $\mathcal{I}[X]$ 表示包含点集 X 中所有点的最小单位长度闭区间个数，则有：

$$\mathcal{I}[X] = \begin{cases} 1 + \mathcal{I}\Big[X \setminus \big\{x_i | x_i \in [x_{min}, x_{min}+1], i = 1, \cdots, n\big\}\Big], & X \neq \varnothing \\ 0, & X = \varnothing \end{cases} \tag{3}$$

据此给出算法伪代码如下：

Unit-Length Closed Intervals Contain Set(X)

```
sort X
while X ≠ ∅
    add [X[0], X[0] + 1] to the set
    X = X \ {x|x ∈ [X[0], X[0] + 1]}
return the set
```

5 Problem 16-1 a. Coin changing

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。

解 认为最优解即使得找零的硬币最少，为达到这一目的，每次寻找能找的最大面额的硬币，算法用伪代码可表述如下：

Coin Changing(n)

```
quarter = 0
dime = 0
nickel = 0
penny = 0
while n ≥ 25
    quarter = quarter + 1
    n = n − 25
while n ≥ 10
    dime = dime + 1
    n = n − 10
while n ≥ 5
    nickel = nickel + 1
    n = n − 5
penny = n
```

假设这不是最优解，即存在这样的解 $quarter', dime', nickel', penny'$，使得

$$
\begin{aligned}
&n \\
=&25quarter' + 10dime' + 5nickel' + penny' \\
=&25quarter + 10dime + 5nickel + penny \\
&quarter' + dime' + nickel' + penny' < quarter + dime + nickel + penny
\end{aligned}
$$

记 $\Delta q = quarter' - quarter, \Delta d = dime' - dime, \Delta n = nickel' - nickel, \Delta p = penny' - penny$，则有

$$
\begin{cases}
\Delta q + \Delta d + \Delta n + \Delta p < 0 \\
25\Delta q + 10\Delta d + 5\Delta n + \Delta p = 0
\end{cases}
$$

若 $\Delta q < 0$ 则必有 $10\Delta d + 5\Delta n + \Delta p > 25|\Delta q| > 0$，同理若 $\Delta d < 0$ 则必有 $5\Delta n + \Delta p > 10|\Delta d| > 0$，若 $\Delta n < 0$ 则必有 $\Delta p > 5|\Delta n| > 0$，因此 $\Delta q = \Delta d = \Delta n = \Delta p = 0$，即两解相同，矛盾，因此算法得到的是最优解。