HomeWork 7

BY YANGZHE PB17000025

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。

   $L(s,t)=1+\max\{L(s',t)\}, s'\in V-\{s\}, s$和$s'$连通

   即s到t的最长路径，一定由某条$s\to s'$的边和$s'\to t$的路径拼在一起，且$s'\to t$最长。

   若$s=t$，则$L(s,t)=0.$

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。

   引入辅助数组$F[n+1][W+1]$

   $F[0,j]=0,\ F[i,0]=0$

   $F[i,j]=F[i-1,j],\ \text{if}\ w_i>j$

   $F[i,j]=\max\{F[i-1,j], F[i-1,j-w[i]]+v[i]\},\ \text{if}\ w_i\leqslant j$

   其中i表示在前i件物品中取, j表示取出物品的总重量为j。

   对于每一个物品都有两种选择，选或不选，如果选了那么$F[i,j]=F[i-1,j-w[i]]+v[i]$,

   如果没选，那么$F[i,j]=F[i-1,j]$,我们取两者的最大值。

   如果当前的物品重量大于j，则不可能装入背包，故直接取$F[i-1,j]$.

   这个算法的时间复杂度为$O(n\,W)$， 因为我们只需要把辅助数组$F[n+1][W+1]$计算出来， 那么01背包问题的结果就是$F[n][W]$.

   $T(n)=O(n)+O((n+1)(W+1))=O(n+nW+n+W+1)=O(nW)$

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。

   引入辅助数组M

   因为不希望员工和其直接主管同时出席，所以一个根节点r如果出席，那么此时的值是

   $V[r]+\sum_j M[j], \text{for}\ j\ \text{is}\ r's\ \text{grandchild}$

   如果根节点r不出席，那么此时的值是：

   $\sum_j M[j], \text{for}\ j\ \text{is}\ r's\ \text{child}$

   故$M[r]=\max\{V[r]+\sum_j M[j], \text{for}\ j\ \text{is}\ r's\ \text{grandchild}, \sum_j M[j], \text{for}\ j\ \text{is}\ r's\ \text{child}\}$

   若r是叶子节点，那么$M[r]=V[r]$

   因为这个是左孩子右兄弟表示法，所以孩子都在左子树，而孙子在左子树的左子树。

   这个算法的目标是计算出来$M[\text{root}]$， 使用自底向上的方法， 则只需要遍历一遍所有节点， 即可算出$M[\text{root}]$，故时间复杂度为$O(n)$.

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

   找到这n个点的最小值，把单位闭区间的左端设为它，并删去在这个区间内的所有点。

   然后找出剩余区间的最小值，将下一个单位闭区间的左端设为它，并删去在这个区间内的所有点。

   如此重复，当所有的点都被删去时，输出区间集合。

   证明最优：我们证明这种做法满足“greedy choice property”

   对于第一个最小的点$x_{(1)}$，我们选择区间$I = [x_{(1)}, x_{(1)} + 1]$，假设最优的选择是$I_{\text{Opt}} = [p, p + 1]$，且$x_{(1)} \in I_{\text{Opt}}$.

   因为$x_{(1)} \in I_{\text{Opt}}$，所以$p \leqslant x_{(1)}$，假设$p < x_{(1)}$，那么$[p, x_{(1)})$就被浪费了，不满足最优的条件。

   所以，如果是最优的选择，那么只能是$p = x_{(1)}$。

   同样的，我们可以证明$x_{(2)}, x_{(3)}, ..., x_{(n)}$对应区间的选择都是最优的，故整体是最优的。

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。

   根据n的值选择最开始的零钱x_1，

$$\begin{cases} 25 & n \geqslant 25 \\ 10 & 10 \leqslant n < 25 \\ 5 & 5 \leqslant n < 10 \\ 1 & 1 \leqslant n < 5 \end{cases}$$

   选择$\text{floor}(n / x_1)$个x_1,此时剩余的$n < x_1$， 所以我们选择仅次于x_1的零钱x_2， 重复上述操作， 直至n减至0.

   首先一定能找到解，因为最小的零钱是1 美分，而1可以整除任何非零数。

   其次证明解是最优解，因为25可以由10和5线性组成，而10正好是5的整数倍。

   所以能使用25时， 使用10和5， 就会产生更多的硬币，$25 = 2 \times 10 + 5 = 5 \times 5 = 25 \times 1$， 所以如果不选择25会多出2个、4个或24个硬币。能使用10的时候,$10 = 2 \times 5 = 10 \times 1$， 不使用10会多1个或9个硬币。能使用5时，$5 = 5 \times 1$， 不使用5会多出4个硬币。