树型动态规划

2016年5月31日

主要内容

- 树型DP特征
- 树型DP例题讲解
- 总结

什么是树型动态规划

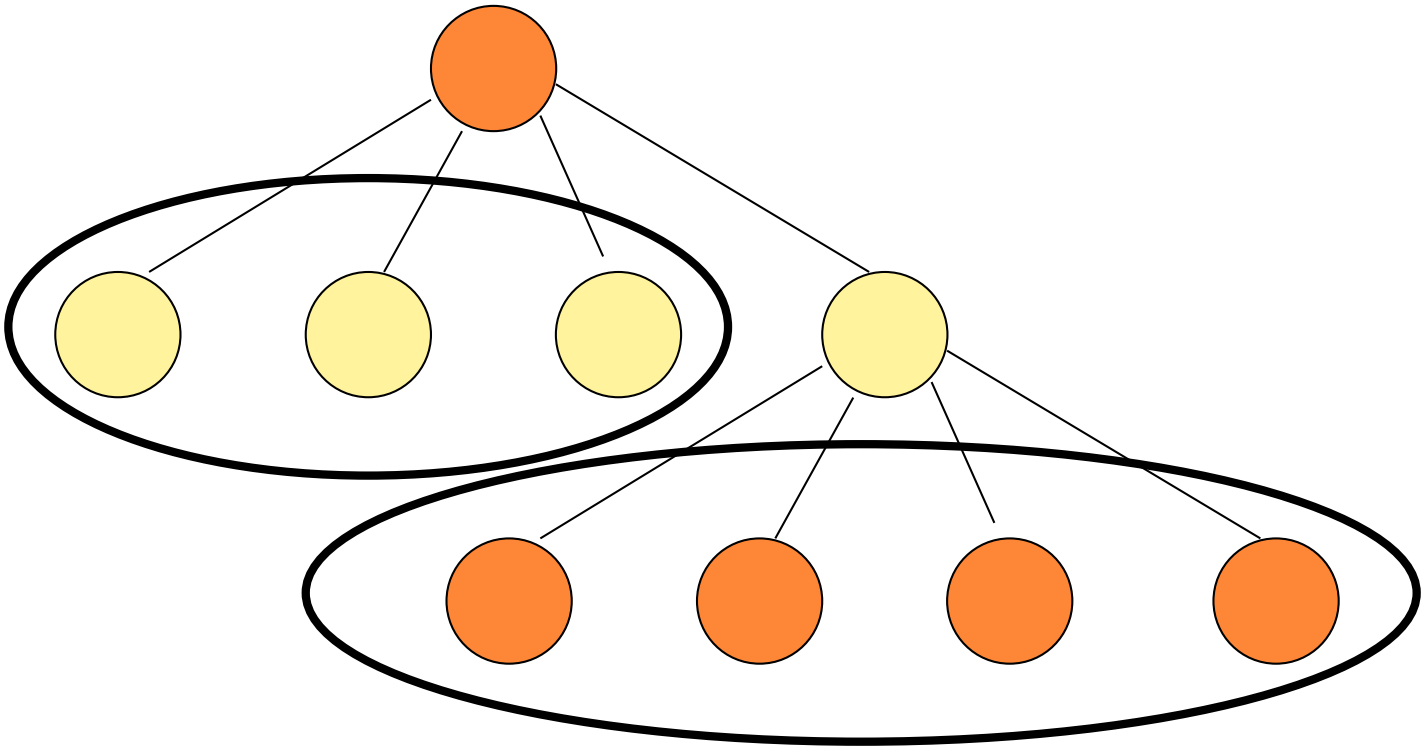
- 顾名思义，树型动态规划就是在“树”的数据结构上的动态规划，平时作的动态规划都是线性的或者是建立在图上的，线性的动态规划有二种方向既向前和向后，相应的线性的动态规划有二种方法既顺推与逆推，而树型动态规划是建立在树上的，所以也相应的有二个方向：
  - 根—>叶：不过这种动态规划在实际的问题中运用的不多，也没有比较明显的例题，所以不在今天讨论的范围之内。
  - 叶—>根：既根的子节点传递有用的信息给根，完后根得出最优解的过程。这类的习题比较的多，下面就介绍一些这类题目和它们的一般解法。

例题一：HDU 2412 PARTY AT HALI-BULA

- 题目大意：
- n个人形成一个关系树，每个节点代表一个人，节点的根表示这个人的唯一的直接上司，只有根没有上司。要求选取一部分人出来，使得每2个人之间不能有直接的上下级的关系，求最多能选多少个人出来，并且求出获得最大人数的选人方案是否唯一。
- 这是一个经典的树型动态规划。
- 状态？
- 转移？

1.2 Party at Hali-Bula

- 简单的染色统计是不正确的



1.3 Party at Hali-Bula

- 人之间的关系形成树型结构
- DP数组设计：维数？意义？
- DP, 用dp[i][0]表示不选择i点时，i点及其子树能选出的最多人数，dp[i][1]表示选择i点时，i点及其子树的最多人数。

1.4 Party at Hali-Bula

- 状态转移方程：
  - 对于叶子节点 k
  - dp[k][0] = 0, dp[k][1] = 1
  - 对于非叶子节点 i
  - dp[i][0] = ∑max(dp[j][0], dp[j][1]) (j是i的儿子)
  - dp[i][1] = 1 + ∑dp[j][0] (j是i的儿子)
- 最多人数：
- max(dp[0][0], dp[0][1])
- 如何判断最优解是否唯一?

1.5 Party at Hali-Bula

- 新加一个状态dup[i][j]，表示相应的dp[i][j]是否是唯一方案。
- 对于叶子结点，
- dup[k][0] = dup[k][1] = 1.
- 对于非叶子结点，
  - 对于i的任一儿子j，若(dp[j][0] > dp[j][1] 且 dup[j][0] == 0) 或 (dp[j][0] < dp[j][1] 且 dup[j][1] == 0) 或 (dp[j][0] == dp[j][1])，则dup[i][0] = 0
  - 对于i的任一儿子j有dup[j][0] = 0, 则dup[i][1] = 0

例题二：HDU 1054 STRATEGIC GAME

- 题目大意：
- 一城堡的所有的道路形成一个n个节点的树，如果在一个节点上放上一个士兵，那么和这个节点相连的边就会被看守住，问把所有边看守住最少需要放多少士兵。

- 典型的树型动态规划

- 状态？
- 转移？

2.2 STRATEGIC GAME

- dproot[i]表示以i为根的子树，在i上放置一个士兵，看守住整个子树需要多少士兵。

- all[i]表示看守住整个以i为根的子树需要多少士兵。

2.3 STRATEGIC GAME

- 状态转移方程：
- 叶子节点：

  dproot[k] =1； all[k] = 0；
- 非叶子节点：

  dproot[i] = 1 + ∑all[j](j是i的儿子)；

  all[i] = min(dproot[i], ∑dproot[j](j是i的儿子));

- 这个题目还是比较简单的，如果把题目中看守边变成看守相邻的点呢？留给你来思考^_^

例题三： ZXA AND LEAF

- 链接

例题三 加分二叉树

- 设一个n个节点的二叉树tree的中序遍历为（l,2,3,…,n），其中数字1,2,3,…,n为节点编号。每个节点都有一个分数（均为正整数），记第i个节点的分数为di，tree及它的每个子树都有一个加分，任一棵子树subtree（也包含tree本身）的加分计算方法如下：
- subtree的左子树的加分× subtree的右子树的加分＋subtree的根的分数
- 若某个子树为空，规定其加分为1，叶子的加分就是叶节点本身的分数。不考虑它的空子树。
- 试求一棵符合中序遍历为（1,2,3,…,n）且加分最高的二叉树tree。

3.2 基础回顾

- 树的中序遍历
- 若二叉树为空则结束返回，
- 否则：
- （1）中序遍历左子树。
- （2）访问根结点。
- （3）中序遍历右子树。

- 树的前序遍历
- 若二叉树为空则结束返回，否则：
- （1）访问根结点.
- （2）前序遍历左子树.
- （3）前序遍历右子树 .

3.3 样例

- 【输入格式】
- 第1行：一个整数n（n＜30），为节点个数。
- 第2行：n个用空格隔开的整数，为每个节点的分数（分数＜100）。
-
- 【输出格式】
- 第1行：一个整数，为最高加分（结果不会超过4,000,000,000）。
- 第2行：n个用空格隔开的整数，为该树的前序遍历。
-
- 【输入样例】
- 5
- 5 7 1 2 10
-
- 【输出样例】
- 145
- 3 1 2 4 5

3.4 分析

- 本题适合用动态规划来解。如果用数组value[i,j]表示从节点i到节点j所组成的二叉树的最大加分，则动态方程可以表示如下：
- value[i,j]=max{value[i,i]+value[i+1,j],
  value[k,k]+value[i,k]*value[k+1,j] | i<k<j,
  value[i,j-1] + value[j,j] };

- 第一项 为 左子树为空， 根为i，右子树[i+1,j]。
- 第二项 为 左子树为i,根为i+1,右子树为[i+2,j].

树型动态规划总结

- 必要条件：子树之间不可以相互干扰，如果本来是相互干扰的，那么我们必须添加变量使得他们不相互干扰。

- 树形动态规划通常从叶节点（边界）开始逐步向上一层的节点（即父节点）进行状态方程的转移，直到根节点。

Thank You
for your time!