知识精炼（三）

主讲人：邓哲也

POJ 2486 Apple Tree

苹果树是一个树形的结构，在节点处长有苹果。

一个人从 1 号节点开始，只能走 K 步。

只要访问到一个节点，就一定会吃光这个节点所有的苹果。

问最多能吃到几个苹果。

$1 \leqslant N \leqslant 100$，$1 \leqslant K \leqslant 200$

POJ 2486 Apple Tree

设计状态：用 f[i][j] 表示以 i 为根的子树中最多走 j 步能获得的苹果数。

那我们只要把 j 步分配给 i 的子树就行。

但需要注意一点，需要对是否走回 i 进行讨论。

因此再加一维，表示是否回到 i。

POJ 2486 Apple Tree

f[i][j][0] 表示在子树 i中最多走 j步最后还回到 i 能得到的最大苹果数。

f[i][j][1] 表示在子树 i中最多走 j步最后不回到 i 能得到的最大苹果数。

POJ 2486 Apple Tree

```
f[i][j][0] = max(f[i][j][0], f[i][j-k][0] + f[v][k-
2][0]);

f[i][j][1] = max(f[i][j][1],
                 max(f[i][j-k][0] + f[v][k-1][1],
                     f[i][j-k][1] + f[v][k-
2][0]));
```

第一种是先走其他儿子回到 i 再走 v

第二种是先走 v 回到 i 再走其他儿子

POJ 2486 Apple Tree

```
void dfs(int u){
        for (int i = 0;i <= K;i ++)
                f[u][i][0] = f[u][i][1] = w[i];
        for (int i = h[u]; i != -1; i = e[i].next){
                int v =e[i].v;
                dfs(v);
                for (int j = K; j >= 0;j --){
                        for(int k = 1;k <= j;k ++){
                                f[u][j][1] = max(f[u][j][1], f[u][j-k][0]
+ f[v][k-1][1]);
                                ...
```

POJ 2486 Apple Tree

```
        …
        if (k % 2 == 0){
                f[u][j][1] = max(f[u][j][1], f[u][j-k][1] + f[v][k-2][0]);
                f[u][j][0] = max(f[u][j][0], f[u][j-k][0] + f[v][k-2][0]);
        }
}
```

时间复杂度 $O(NK^2)$

HDU 3586 Information Disturbing

给出 n 个节点的一棵树，每条边有权值。

现在要求切段其中的一些边，使得任意一个叶子没有走到祖先的路。

要求切断的边的总权值不能超过 m。求所有方案中切断的最大的边权的最小值。

$n \leqslant 1000$，$m \leqslant 10^6$

Sample Input

```
5 5
1 3 2
1 4 3
3 5 5
4 2 6
0 0
```

Sample Output

```
3
```

HDU 3586 Information Disturbing

首先看到求最大值的最小值就知道可以二分。

二分最大值 max。

然后统计最小花费，和 m 进行比较。

统计的时候边权大于 max 的边不能切断。

用 f[i] 表示 i 与所有叶子节点断开的最小花费。

HDU 3586 Information Disturbing

枚举 i 的子节点 j。

第一种：直接切掉（i, j）这条边，花费是 w(i, j)

第二种：子树 j 中的叶节点已经与 j 断开，花费是 f[j]

HDU 3586 Information Disturbing

```
if (w(i, j) <= max)
        f[i] += min(f[j], w(i, j));
else
        f[i] += f[j]
```

HDU 3586 Information Disturbing

```
void dfs(int u, int fa){
        f[u] = 0;
        int flag = 1;
        for (int i = h[u]; i != -1;i = e[i].next){
                int v = e[i].v;
                if (v != fa){
                        flag = 0;
                        dfs(v, u);
                        if (e[i].w <= limit) f[i] +=
min(f[j], e[i].w);
                        else f[i] += f[j];
                }
        }
        if (flag) f[u] = 1e9;
}
```

下节课再见