动态规划

北京 提高组基础班

张若天 me@zrt.io

2018 年 2 月 10 日

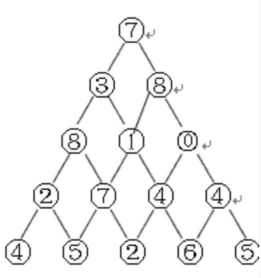
清华大学 交叉信息研究院

大家好!

基本概念

数字三角形

如图所示的数字三角形，从顶部出发，在每一结点可以选择向左走或得向右走，一直走到底层，要求找出一条路径，使路径上的值最大。



Naïve 想法

- 枚举所有路径，取最大值。

Naïve 想法

- 枚举所有路径，取最大值。
- n 层的三角形的路径数量是 2^n。

```
int f(int x,int y){
    if(x==n){
        return a[x][y];
    }
    return a[x][y]+max(f(x+1,y),f(x+1,y+1));
}
int ans=f(1,1);
```

复杂度 2^n。

```
int f(int x,int y){
    if(x==n){
        return a[x][y];
    }
    return a[x][y]+max(f(x+1,y),f(x+1,y+1));
}
int ans=f(1,1);
```

复杂度 2^n。

显然对于确定的 x,y，f(x,y) 的返回值是一定的，与前面如何走到 x,y 无关。

```
int f(int x,int y){
    if(x==n){
        return a[x][y];
    }
    return a[x][y]+max(f(x+1,y),f(x+1,y+1));
}
int ans=f(1,1);
```

复杂度 2^n。

显然对于确定的 x,y，f(x,y) 的返回值是一定的，与前面如何走到 x,y 无关。

所以考虑能否把所有的 f(x,y) 只计算一次。

```
int f(int x,int y){
    if(vis[x][y]) return g[x][y];
    else vis[x][y]=1;
    if(x==n){
        return g[x][y]=a[x][y];
    }
    return g[x][y]=a[x][y]+max(f(x+1,y),f(x+1,y+1));
}
int ans=f(1,1);
```

复杂度 n^2。

```
int f(int x,int y){
    if(vis[x][y]) return g[x][y];
    else vis[x][y]=1;
    if(x==n){
        return g[x][y]=a[x][y];
    }
    return g[x][y]=a[x][y]+max(f(x+1,y),f(x+1,y+1));
}
int ans=f(1,1);
```

复杂度 n^2。

这种方法也叫作记忆化搜索。

```
int f(int x,int y){
    if(vis[x][y]) return g[x][y];
    else vis[x][y]=1;
    if(x==n){
        return g[x][y]=a[x][y];
    }
    return g[x][y]=a[x][y]+max(f(x+1,y),f(x+1,y+1));
}
int ans=f(1,1);
```

复杂度 n^2。

这种方法也叫作记忆化搜索。

记忆化搜索在满足 1) 函数返回结果只与参数有关 2) 同样参数的函数会被多次调用的情况下可以优化算法。

```
int f(int x,int y){
    if(vis[x][y]) return g[x][y];
    else vis[x][y]=1;
    if(x==n){
        return g[x][y]=a[x][y];
    }
    return g[x][y]=a[x][y]+max(f(x+1,y),f(x+1,y+1));
}
int ans=f(1,1);
```

复杂度 n^2。

这种方法也叫作记忆化搜索。

记忆化搜索在满足 1) 函数返回结果只与参数有关 2) 同样参数的函数会被多次调用的情况下可以优化算法。

是一种用空间换时间的方式。

上面函数的递归的作用只是确定了计算顺序。

为什么不只用数组呢?

```
int f(int x,int y){
    if(vis[x][y]) return g[x][y];
    else vis[x][y]=1;
    if(x==n){
        return g[x][y]=a[x][y];
    }
    return g[x][y]=a[x][y]+max(f(x+1,y),f(x+1,y+1));
}
int ans=f(1,1);
------------------      --------------------------
int f[1005][1005];
for(int i=1;i<=n;i++) f[n][i]=a[n][i];
for(int i=n-1;i>=1;i--){
    for(int j=1;j<=i;j++){
        f[i][j]=a[i][j]+max(f[i+1][j],f[i+1][j+1]);
    }
}
ans=f[1][1]
```

- 上面这两种优化方式叫做动态规划。
- 动态规划的两种实现方式：记忆化搜索，递推。
- 上面函数的参数叫做动态规划中的状态。
- 从一个状态计算出另一个状态叫做状态转移。

注意：一般情况下有明确的计算顺序才可以使用递推方法。

滑雪

你在一个 n*n 的带权值的网格上滑雪，权值表示每个格子的高度，每一秒可以决定往上下左右四个方向滑，但是只能滑向更低的地方。求从哪个格子开始滑雪可以滑的时间最长。

关于状态

- 处理动态规划题目设计状态十分重要。
- 状态既要能完全表示当前状态，又不能有冗余信息。
- 还要求有最优子结构、无后效性的性质。

关于最优子结构

最优子结构：把原问题化到规模更小的问题后，原问题的最优解一定能从规模更小问题的最优解推出。

数字三角形 W

数字三角形
要求走到最后和 mod 100 最大。

在 mod 100 这个条件下，刚刚的状态就没有了最优子结构。

关于最优子结构

最优子结构：把原问题化到规模更小的问题后，原问题的最优解一定能从规模更小问题的最优解推出。

数字三角形 W

数字三角形
要求走到最后和 mod 100 最大。

在 mod 100 这个条件下，刚刚的状态就没有了最优子结构。

一般可以通过加状态维数来解决。

- 状态
- 状态转移方程
- 阶段
- 决策

动态规划实际上是将本质相同 (后续转移决策相同) 的大量信息压缩成“单一状态”，将本质相同的大量转移压缩成“单一转移”。

数字三角形 WWW

数字三角形必须经过某一个点，使之走的路程和最大。

传球游戏

n 个同学站成一个圆圈，其中的一个同学手里拿着一个球，当老师吹哨子时开始传球，每个同学可以把球传给自己左右的两个同学中的一个（左右任意），当老师再次吹哨子时，传球停止，此时，拿着球没传出去的那个同学就是败者，要给大家表演一个节目。

聪明的小蛮提出一个有趣的问题：有多少种不同的传球方法可以使得从小蛮手里开始传的球，传了 m 次以后，又回到小蛮手里。两种传球的方法被视作不同的方法，当且仅当这两种方法中，接到球的同学按接球顺序组成的序列是不同的。比如有 3 个同学 1 号、2 号、3 号，并假设小蛮为 1 号，球传了 3 次回到小蛮手里的方式有 1->2->3->1 和 1->3->2->1，共 2 种。

tyvj1008

最长上升子序列 (LIS)

给一个数组 $a_1, a_2, \ldots, a_n$，找到最长的上升子序列 $a_{b_1} < a_{b_2} < \ldots < a_{b_k}$，其中 $b_1 < b_2 < \ldots < b_k$。

优化。

最长上升子序列 (LIS)

给一个数组 $a_1, a_2, \ldots, a_n$，找到最长的上升子序列 $a_{b_1} < a_{b_2} < \ldots < a_{b_k}$，其中 $b_1 < b_2 < \ldots < b_k$。

优化。

练习：poj 1050 最大子矩形,tyvj1124 花店橱窗布置,tyvj1015 公路乘车,codevs2098 化工厂装箱员。

最长公共子序列 (LCS)

给两个数组 $a_1, a_2, \ldots, a_n$，$b_1, b_2, \ldots, b_n$ 找到最长的公共子序列 $a_{c_1}, a_{c_2}, \ldots, a_{c_k}$，和 $b_{d_1}, b_{d_2}, \ldots, b_{d_k}$，其中 $c_1 < c_2 < \ldots < c_k$，$d_1 < d_2 < \ldots < d_k$, 满足 $a_{c_1} = b_{d_1}, a_{c_2} = b_{d_2}, \ldots, a_{c_k} = b_{d_k}$。

尼克的任务

尼克每天上班之前都连接上英特网，接收他的上司发来的邮件，这些邮件包含了尼克主管的部门当天要完成的全部任务，每个任务由一个开始时刻与一个持续时间构成。

尼克的一个工作日为 N 分钟，从第一分钟开始到第 N 分钟结束。当尼克到达单位后他就开始干活。如果在同一时刻有多个任务需要完成，尼克可以任选其中的一个来做，而其余的则由他的同事完成，反之如果只有一个任务，则该任务必需由尼克去完成，假如某些任务开始时刻尼克正在工作，则这些任务也由尼克的同事完成。如果某任务于第 P 分钟开始，持续时间为 T 分钟，则该任务将在第 P+T-1 分钟结束。

写一个程序计算尼克应该如何选取任务，才能获得最大的空暇时间。

tyvj1034

区间动态规划

沙子合并

设有 N 堆沙子排成一排，其编号为 1，2，3，...，N（N<=300）。每堆沙子有一定的数量，可以用一个整数来描述，现在要将这 N 堆沙子合并成为一堆，每次只能合并相邻的两堆，合并的代价为这两堆沙子的数量之和，合并后与这两堆沙子相邻的沙子将和新堆相邻，合并时由于选择的顺序不同，合并的总代价也不相同，如有 4 堆沙子分别为 1 3 5 2 我们可以先合并 1、2 堆，代价为 4，得到 4 5 2 又合并 1，2 堆，代价为 9，得到 9 2 ，再合并得到 11，总代价为 4+9+11=24，如果第二步是先合并 2，3 堆，则代价为 7，得到 4 7，最后一次合并代价为 11，总代价为 4+7+11=22；问题是：找出一种合理的方法，使总的代价最小。输出最小代价。

tyvj1055

环形情况?

常用技巧：将序列加倍。

能量项链

在 Mars 星球上，每个 Mars 人都随身佩带着一串能量项链。在项链上有 N 颗能量珠。能量珠是一颗有头标记与尾标记的珠子，这些标记对应着某个正整数。并且，对于相邻的两颗珠子，前一颗珠子的尾标记一定等于后一颗珠子的头标记。因为只有这样，通过吸盘（吸盘是 Mars 人吸收能量的一种器官）的作用，这两颗珠子才能聚合成一颗珠子，同时释放出可以被吸盘吸收的能量。如果前一颗能量珠的头标记为 m，尾标记为 r，后一颗能量珠的头标记为 r，尾标记为 n，则聚合后释放的能量为（Mars 单位），新产生的珠子的头标记为 m，尾标记为 n。需要时，Mars 人就用吸盘夹住相邻的两颗珠子，通过聚合得到能量，直到项链上只剩下一颗珠子为止。显然，不同的聚合顺序得到的总能量是不同的，请你设计一个聚合顺序，使一串项链释放出的总能量最大。

例如：设 N=4，4 颗珠子的头标记与尾标记依次为 (2，3) (3，5) (5，10) (10，2)。我们用记号 @ 表示两颗珠子的聚合操作，(j@k) 表示第 j，k 两颗珠子聚合后所释放的能量。则第 4、1 两颗珠子聚合后释放的能量为：(4@1)=10*2*3=60。这一串项链可以得到最优值的一个聚合顺序所释放的总能量为 ((4@1)@2)@3）=10*2*3+10*3*5+10*5*10=710。

tyvj1056

能量项链

在 Mars 星球上，每个 Mars 人都随身佩带着一串能量项链。在项链上有 N 颗能量珠。能量珠是一颗有头标记与尾标记的珠子，这些标记对应着某个正整数。并且，对于相邻的两颗珠子，前一颗珠子的尾标记一定等于后一颗珠子的头标记。因为只有这样，通过吸盘（吸盘是 Mars 人吸收能量的一种器官）的作用，这两颗珠子才能聚合成一颗珠子，同时释放出可以被吸盘吸收的能量。如果前一颗能量珠的头标记为 m，尾标记为 r，后一颗能量珠的头标记为 r，尾标记为 n，则聚合后释放的能量为（Mars 单位），新产生的珠子的头标记为 m，尾标记为 n。需要时，Mars 人就用吸盘夹住相邻的两颗珠子，通过聚合得到能量，直到项链上只剩下一颗珠子为止。显然，不同的聚合顺序得到的总能量是不同的，请你设计一个聚合顺序，使一串项链释放出的总能量最大。

例如：设 $N=4$，4 颗珠子的头标记与尾标记依次为 (2，3) (3，5) (5，10) (10，2)。我们用记号 @ 表示两颗珠子的聚合操作，(j@k) 表示第 j，k 两颗珠子聚合后所释放的能量。则第 4、1 两颗珠子聚合后释放的能量为：$(4@1)=10*2*3=60$。这一串项链可以得到最优值的一个聚合顺序所释放的总能量为 ((4@1)@2)@3）$=10*2*3+10*3*5+10*5*10=710$。

tyvj1056

玩具取名

某人有一套玩具，并想法给玩具命名。首先他选择 WING 四个字母中的任意一个字母作为玩具的基本名字。然后他会根据自己的喜好，将名字中任意一个字母用“WING”中的某两个字母代替，使得自己的名字能够扩充得很长。现在，他想请你猜猜某一个很长的名字，最初可能是由哪个字母变形过来的。

Len $\leq$ 200, 每个字母可替换的两个字母组数 $\leq$ 16

bzoj1055

逆序对数列

对于一个数列 $\{a_i\}$，如果有 i < j 且 $a_i > a_j$，那么我们称 a_i 与 a_j 为一对逆序对数。若对于任意一个由 1∼n 自然数组成的数列，可以很容易求出有多少个逆序对数。那么逆序对数为 k 的这样自然数数列到底有多少个？由于这个数可能很大，你只需输出该数对 10000 求余数后的结果。n ≤ 1000, k ≤ 1000

bzoj2431

逆序对数列

对于一个数列 $\{a_i\}$，如果有 i < j 且 $a_i > a_j$，那么我们称 a_i 与 a_j 为一对逆序对数。若对于任意一个由 1∼n 自然数组成的数列，可以很容易求出有多少个逆序对数。那么逆序对数为 k 的这样自然数数列到底有多少个？由于这个数可能很大，你只需输出该数对 10000 求余数后的结果。n ≤ 1000, k ≤ 1000

bzoj2431

F[i][j] 表示 1-i 的排列有 j 个逆序对有多少个。前缀和优化。

逆序对数列

对于一个数列 $\{a_i\}$，如果有 i $<$ j 且 $a_i > a_j$，那么我们称 a_i 与 a_j 为一对逆序对数。若对于任意一个由 1∼n 自然数组成的数列，可以很容易求出有多少个逆序对数。那么逆序对数为 k 的这样自然数数列到底有多少个？由于这个数可能很大，你只需输出该数对 10000 求余数后的结果。n $\leq$ 1000, k $\leq$ 1000

bzoj2431

F[i][j] 表示 1-i 的排列有 j 个逆序对有多少个。前缀和优化。

练习：poj1821

传纸条

在一个矩阵内找出两条从 1,1 到 m,n 的路径（一条从 1，1 到 m,n 一条从 m,n 到 1，1），并且路径上的权值之和最大。

noip2008

背包问题

部分背包

有 n 个物品，每个物品有体积 v_i 和价值 w_i，你有一个体积为 V 的背包，问最多能装多少价值的物品?

其中：1. 物品可以拿任意实数份，2. 物品可以拿 0-1 内任意实数份。

部分背包

有 n 个物品，每个物品有体积 v_i 和价值 w_i，你有一个体积为 V 的背包，问最多能装多少价值的物品？

其中：1. 物品可以拿任意实数份，2. 物品可以拿 0-1 内任意实数份。

贪心

完全背包

有 n 个物品，每个物品有体积 v_i 和价值 w_i，你有一个体积为 V 的背包，问最多能装多少价值的物品？

其中：物品可以拿任意整数份。

01 背包

有 n 个物品，每个物品有体积 v_i 和价值 w_i，你有一个体积为 V 的背包，问最多能装多少价值的物品？

其中：物品可以拿任意 0 份或 1 份。

多重背包

有 n 个物品，每个物品有体积 v_i 和价值 w_i，你有一个体积为 V 的背包，问最多能装多少价值的物品?

其中：第 i 物品可以拿 a_i 份 (整数)。

树上的动态规划

树的直径

给一棵树，求树上的最长链。

树的重心

给一棵树，求树的重心，即删掉某个点后剩下的最大连通块大小最小。

没有上司的舞会

Ural 大学有 N 个职员，编号为 1-N。他们有从属关系，也就是说他们的关系就像一棵以校长为根的树，父结点就是子结点的直接上司。每个职员有一个快乐指数。现在有个周年庆宴会，要求与会职员的快乐指数最大。但是，没有职员愿和直接上司一起与会。

tyvj1052

没有上司的舞会

Ural 大学有 N 个职员，编号为 1-N。他们有从属关系，也就是说他们的关系就像一棵以校长为根的树，父结点就是子结点的直接上司。每个职员有一个快乐指数。现在有个周年庆宴会，要求与会职员的快乐指数最大。但是，没有职员愿和直接上司一起与会。

tyvj1052

练习: luogu2014 选课

Questions?

谢谢大家！

Email: me@zrt.io

QQ: 401794301

https://zrt.io

LaTeX