ACM 程序设计

计算机学院 刘春英

今天，

你 AC了吗？

每周一星（3）：

05059127陈谦益

第四讲

动态规划(1）（Dynamic programming）

先热身一下——

（1466）计算直线的交点数

问题描述：

平面上有n条直线，且无三线共点，问这些直线能有多少种不同交点数。

输入：n（n<=20）

输出：每个测试实例对应一行输出，从小到大列出所有相交方案，其中每个数为可能的交点数。

样例输入

4

样例输出

0 3 4 5 6

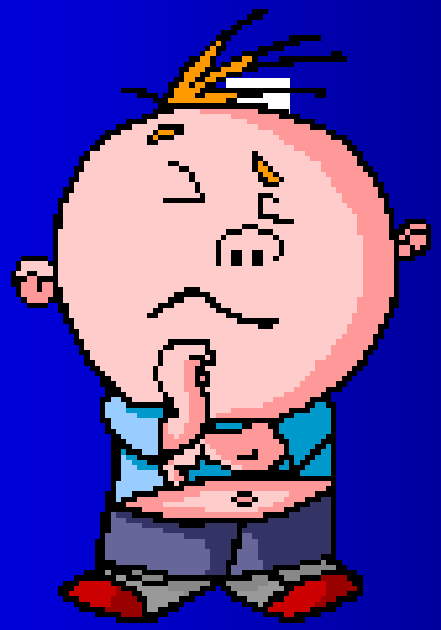
初步分析:

我们知道:

n条直线互不平行且无三线共点的最多交点数max=1+2+……(n-1)=n(n-1)/2,

但本题不这么简单，因为问题问的是：这些直线有多少种不同的交点数？

思考2分钟:如何解决?



容易列举出N=1,2,3的情况：

0

0，1

0，2，3

如果已知<N的情况，我们来分析加入第N条直线的情况(这里N=4)：

1、第四条与其余直线全部平行 => 无交点；

2、第四条与其中两条平行，交点数为(n-1)*1+0=3;

3、第四条与其中一条平行，这两条平行直线和另外两点直线的交点数为(n-2)*2=4,而另外两条直线既可能平行也可能相交，因此可能交点数为：

（n-2）*2+0=4　或者　(n-2)*2+1=5

4、 第四条直线不与任何一条直线平行，交点数为：

(n-3)*3+0=3　或者 (n-3)*3+2=5　或者 (n-3)*3+3=6

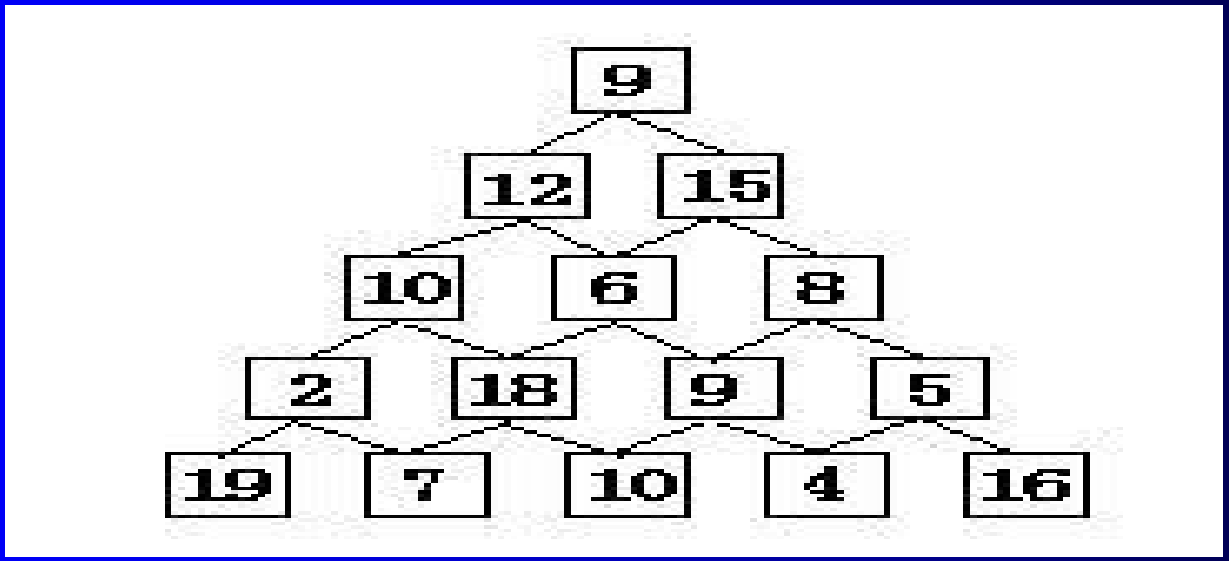
即n=4时，有0个，3个，4个，5个，6个不同交点数。

从上述n=4的分析过程中，我们发现：

m条直线的交点方案数
=（m-r）条平行线与r条直线交叉的交点数
+ r条直线本身的交点方案
=（m-r）*r+r条之间本身的交点方案数
（1<=r<=m）

一、数塔问题

有形如下图所示的数塔，从顶部出发，在每一结点可以选择向左走或是向右走，一直走到底层，要求找出一条路径，使路径上的值最大。



用暴力的方法，可以吗？

试想一下：

这道题如果用枚举法（暴力思想），在数塔层数稍大的情况下（如31），则需要列举出的路径条数将是一个非常庞大的数目（2^30= 1024^3 > 10^9=10亿）。

拒绝*暴力*，倡导*和谐*~

考虑一下：

从顶点出发时到底向左走还是向右走应取决于是从左走能取到最大值还是从右走能取到最大值，只要左右两道路径上的最大值求出来了才能作出决策。

同样，下一层的走向又要取决于再下一层上的最大值是否已经求出才能决策。这样一层一层推下去，直到倒数第二层时就非常明了。

如数字2，只要选择它下面较大值的结点19前进就可以了。所以实际求解时，可从底层开始，层层递进，最后得到最大值。

结论：自顶向下的分析，自底向上的计算。

Understand?

二、思考题：最长有序子序列

I	0	1	2	3	4	5	6	7	8
Num[I]	1	4	7	2	5	8	3	6	9

请回答：

穷举（暴力）方法的时间复杂度是多少？

解决方案：

I	0	1	2	3	4	5	6	7	8
Num[I]	1	4	7	2	5	8	3	6	9

F[I]	1	2	3	2	3	4	3	4	5

三、HDOJ_1160 FatMouse's Speed

- 题目链接

Sample Input

```
6008  1300
6000  2100
500   2000
1000  4000
1100  3000
6000  2000
8000  1400
6000  1200
2000  1900
```

Sample Output

```
4
4
5
9
7
```

题目分析：

- 设Mice[i].W表示第i只老鼠的重量，Mice[i].S表示第i只老鼠的速度。我们先对Mice进行排序，以W为第一关键字，从小到大，S为第二关键字，从大到小。
- 设f[i]为Mice[i]至Mice[n]最长的序列长度。考虑某一个f[i]，则有：
- f[i] = max(f[i], f[j]+1) (1<=j<i，且Mice[i].W> Mice[j].W，Mice[i].S < Mice[j].S)
- 其中，初始条件为f[i]=1 (i=1, 2, ..., n)。

Qestion:

两个问题有本质区别吗？

思考（期末考试题）：

Super Jumping!
Jumping! Jumping!

解题思路？

四、HDOJ_1159 Common Subsequence

- 题目链接

Sample Input

abcfbc abfcab
programming contest
abcd mnp

Sample Output

4
2
0

请先计算暴力算法的时间复杂度：
（当然是指最坏情况！）

辅助空间变化示意图

	a	b	c	f	b	c
a	1	1	1	1	1	1
b	1	2	2	2	2	2
f	1	2	2	3	3	3
c	1	2	3	3	3	4
a	1	2	3	3	3	4
b	1	2	3	3	4	4

子结构特征：

- $$f(i,j)=\begin{cases} f(i-1,j-1)+1 & (a[i]==b[j]) \\ \max(f(i-1,j),f(i,j-1)) & (a[i]!=b[j]) \end{cases}$$

- 由于f(i,j)只和f(i-1,j-1), f(i-1,j)和f(i,j-1)有关, 而在计算f(i,j)时, 只要选择一个合适的顺序, 就可以保证这三项都已经计算出来了, 这样就可以计算出f(i,j). 这样一直推到f(len(a),len(b))就得到所要求的解了.

理论总结

一、动态规划的基本思想

如果各个子问题不是独立的，不同的子问题的个数只是多项式量级，如果我们能够保存已经解决的子问题的答案，而在需要的时候再找出已求得的答案，这样就可以避免大量的重复计算。由此而来的基本思路是，用一个表记录所有已解决的子问题的答案，不管该问题以后是否被用到，只要它被计算过，就将其结果填入表中。

二、动态规划的基本步骤

动态规划算法通常用于求解具有某种最优性质的问题。在这类问题中，可能会有许多可行解。每一个解都对应于一个值，我们希望找到具有最优值（最大值或最小值）的那个解。设计一个动态规划算法，通常可以按以下几个步骤进行：

（1）找出最优解的性质，并刻画其结构特征。
（2）递归地定义最优值。
（3）以自底向上的方式计算出最优值。
（4）根据计算最优值时得到的信息，构造一个最优解。

其中（1）——（3）步是动态规划算法的基本步骤。在只需要求出最优值的情形，步骤（4）可以省去。若需要求出问题的一个最优解，则必须执行步骤（4）。此时，在步骤（3）中计算最优值时，通常需记录更多的信息，以便在步骤（4）中，根据所记录的信息，快速构造出一个最优解。

三、动态规划问题的特征

动态规划算法的有效性依赖于问题本身所具有的两个重要性质：

1、最优子结构：当问题的最优解包含了其子问题的最优解时，称该问题具有最优子结构性质。

2、重叠子问题：在用递归算法自顶向下解问题时，每次产生的子问题并不总是新问题，有些子问题被反复计算多次。动态规划算法正是利用了这种子问题的重叠性质，对每一个子问题只解一次，而后将其解保存在一个表格中，在以后尽可能多地利用这些子问题的解。

思考：免费馅饼

如何解决？

请发表见解 ☺

Any Question?

附：课后作业

一. 《ACM Programming》Exercise（4）

二. DP入门练习题目：

- 1003 、 1466 、1087、1159、1176
- 1058、1069、2059、2084

ACM, 天天见！