

IOI2014国家集训队作业 ACM/ICPC World Finals题解

湖南师大附中 胡泽聪
完成情况: 97/156

Contents

| | | |
|----------|--|-----------|
| 1 | ACM/ICPC World Finals 2013 | 1 |
| 1.1 | A - Self-Assembly | 1 |
| 1.2 | C - Surely You Congest | 2 |
| 1.3 | E - Harvard | 3 |
| 1.4 | F - Low Power | 4 |
| 1.5 | H - Matryoshka | 5 |
| 1.6 | I - Pirate Chest | 6 |
| 1.7 | J - Pollution Solution | 7 |
| 1.8 | K - Up a Tree | 7 |
| | | |
| 2 | ACM/ICPC World Finals 2012 | 9 |
| 2.1 | B - Curvy Little Bottles | 9 |
| 2.2 | C - Bus Tour | 10 |
| 2.3 | E - Infiltration | 11 |
| 2.4 | G - Minimum Cost Flow | 12 |
| 2.5 | K - Stacking Plates | 13 |
| 2.6 | L - Takeover Wars | 14 |
| | | |
| 3 | ACM/ICPC World Finals 2011 | 15 |
| 3.1 | A - To Add or to Multiply | 15 |
| 3.2 | C - Ancient Messages | 16 |
| 3.3 | E - Coffee Central | 17 |
| 3.4 | H - Mining Your Own Business | 18 |
| 3.5 | J - Pyramids | 18 |
| 3.6 | K - Trash Removal | 19 |
| | | |
| 4 | ACM/ICPC World Finals 2010 | 20 |
| 4.1 | B - Barcode | 20 |
| 4.2 | D - Castles | 21 |
| 4.3 | F - Contour Mapping | 22 |
| 4.4 | G - The Islands | 23 |
| 4.5 | J - Sharing Chocolate | 24 |

| | | |
|----------|--|-----------|
| 5 | ACM/ICPC World Finals 2009 | 26 |
| 5.1 | A - A Careful Approach | 26 |
| 5.2 | B - My Bad | 26 |
| 5.3 | E - Fare and Balanced | 27 |
| 5.4 | F - Deer-Proof Fence | 28 |
| 5.5 | G - House of Cards | 29 |
| 5.6 | H - The Ministers' Major Mess | 29 |
| 6 | ACM/ICPC World Finals 2008 | 31 |
| 6.1 | A - Air Conditioning Machinery | 31 |
| 6.2 | B - Always an Integer | 32 |
| 6.3 | D - The Hare and the Hound | 33 |
| 6.4 | F - Glenbow Museum | 33 |
| 6.5 | J - The Sky is the Limit | 36 |
| 6.6 | K - Steam Roller | 37 |
| 7 | ACM/ICPC World Finals 2007 | 39 |
| 7.1 | A - Consanguine Calculations | 39 |
| 7.2 | C - Grand Prix | 39 |
| 7.3 | E - Collecting Luggage | 40 |
| 7.4 | G - Network | 41 |
| 7.5 | I - Water Tanks | 42 |
| 7.6 | J - Tunnels | 44 |
| 8 | ACM/ICPC World Finals 2006 | 46 |
| 8.1 | A - Low Cost Air Travel | 46 |
| 8.2 | B - Remember the A La Mode! | 47 |
| 8.3 | D - Bipartite Numbers | 48 |
| 8.4 | G - Pilgrimage | 49 |
| 8.5 | I - Degrees of Separation | 50 |
| 8.6 | J - Routing | 50 |
| 9 | ACM/ICPC World Finals 2005 | 53 |
| 9.1 | B - Simplified GSM Network | 53 |
| 9.2 | C - The Traveling Judges Problem | 54 |
| 9.3 | E - Lots of Sunshine | 55 |
| 9.4 | G - Tiling the Plane | 55 |
| 9.5 | H - The Great Wall Game | 56 |
| 9.6 | I - Workshops | 57 |
| 9.7 | J - Zones | 58 |

| | |
|--|-----------|
| 10 ACM/ICPC World Finals 2004 | 60 |
| 10.1 D - Insecure in Prague | 60 |
| 10.2 E - Intersecting Dates | 61 |
| 10.3 F - Merging Maps | 61 |
| 10.4 H - Tree-Lined Streets | 62 |
| 10.5 I - Suspense! | 63 |
| 11 ACM/ICPC World Finals 2003 | 66 |
| 11.1 A - Building Bridges | 66 |
| 11.2 B - Light Bulbs | 67 |
| 11.3 D - Eurodiffusion | 67 |
| 11.4 F - Combining Images | 68 |
| 11.5 G - A Linking Loader | 69 |
| 11.6 H - A Spy in the Metro | 69 |
| 11.7 I - The Solar System | 70 |
| 11.8 J - Toll | 71 |
| 12 ACM/ICPC World Finals 2002 | 72 |
| 12.1 A - Balloons in a Box | 72 |
| 12.2 C - Crossing the Desert | 72 |
| 12.3 E - Island Hopping | 73 |
| 12.4 H - Silly Sort | 74 |
| 13 ACM/ICPC World Finals 2001 | 75 |
| 13.1 A - Airport Configuration | 75 |
| 13.2 B - Say Cheese | 75 |
| 13.3 E - The Geoduck GUI | 76 |
| 13.4 F - A Major Problem | 77 |
| 13.5 G - Fixed Partition Memory Management | 77 |
| 13.6 H - Professor Monotonic's Network | 78 |
| 13.7 I - A Vexing Problem | 79 |
| 14 ACM/ICPC World Finals 2000 | 81 |
| 14.1 A - Abbott's Revenge | 81 |
| 14.2 B - According to Bartjens | 81 |
| 14.3 C - Cutting Chains | 82 |
| 14.4 E - Internet Bandwidth | 83 |
| 14.5 F - Page Hopping | 83 |
| 15 ACM/ICPC World Finals 1999 | 85 |
| 15.1 A - Bee Breeding | 85 |
| 15.2 C - A Dicey Problem | 86 |
| 15.3 D - The Fortified Forest | 86 |

| | | |
|-----------|--|-----------|
| 15.4 | E - Trade on Verweggistan | 87 |
| 15.5 | G - The Letter Carrier's Rounds | 88 |
| 15.6 | H - Flooded! | 88 |
| 16 | ACM/ICPC World Finals 1998 | 90 |
| 16.1 | A - Crystal Clear | 90 |
| 16.2 | B - Flight Planning | 91 |
| 16.3 | C - Lead or Gold | 91 |
| 16.4 | D - Page Selection by Keyword Matching | 92 |
| 16.5 | E - Petri Net Simulation | 92 |
| 16.6 | G - Spatial Structures | 93 |

Chapter 1

ACM/ICPC World Finals 2013

1.1 A - Self-Assembly

1.1.1 Description

在一个平面上有一些正方形的分子，每条边有一个类型。类型都是一个大写字母加上“+”或者“-”，或者是“00”。两个正方形可以相邻，当且仅当靠在一起的边可以兼容。两条边可以兼容当且仅当两条边的类型都不是“00”，字母相同，而且符号相反。比如“A+”可以和“A-”兼容，但不能和“A+”、“B-”或“00”兼容。

一共有 n 种分子，每种有无限个，而且可以旋转和翻转。求这些分子能不能结合成一个无限大的结构体。

$$n \leq 40000。$$

1.1.2 Solution

假设得到了一个无限长的分子的序列 p_1, p_2, \dots ，如何判断这个序列能否排列在平面上？答案是一定可以，因为分子可以翻转和旋转，我们一定可以将序列的第 $i+1$ 个分子排在第 i 个分子的上面或者右边。故无需在意能否排列，只要存在无限长的分子序列，答案就是“unbounded”。

如果 n 足够小，我们可以对每种分子设点，枚举每对分子，如果可以相邻就连一条无向边，判断是否有环即可。不过这里 n 达到了40000，这个做法是会超时的。但是注意到边的种类一共只有52种，我们可以对每种边设点。

令 $inv(x)$ 为与类型为 x 的边匹配的边，如 $inv(A+) = A-$ 。对于在同一个分子上的两条边 x 和 y ，连有向边 $x \rightarrow inv(y)$ 和 $y \rightarrow inv(x)$ 。只要判断这个图是否有环即可。

由于点数只有52，可以枚举一个起点，然后从这个点BFS，看是否能回到起点。当然用Tarjan算法缩环也是可以的。

1.1.3 Complexity

令 $m = 52$ 。

时间复杂度： $O(n + m^3)$ 。

空间复杂度： $O(m^2)$ 。

难度：2/5。

1.2 C - Surely You Congest

1.2.1 Description

给定一个 n 个点 m 条边的无向图，有 c 个人在图中的某些点处。每个人都会选择一条到1号点的最短路径行走。如果两个人在一条边上同时同向行走，则会造成拥堵。每个人都不愿意在到达1号点之前停下。求在不造成拥堵的前提下，最多有多少人可以到达1号点。

$n \leq 25000$, $m \leq 50000$, $c \leq 1000$ 。

1.2.2 Solution

以1号点为起点做最短路。设 i 号点到1号点的最短路距离为 $d(i)$ ，建一个新图，对于原图中的一条边 (i, j) ，如果 $d(i) + w(i, j) = d(j)$ ，那么在新图中添加有向边 $j \rightarrow i$ 。构出的新图是以1号点为起点的最短路图，是一个DAG。

可以发现，如果两个人所在的点到1号点的最短路距离不同，这两个人是不会造成拥堵的，因为他们不会同时到达一个顶点，也就不会同时通过某条边。因此我们可以把人按照到1号点的距离分组，每次只考虑距离相同的一组人。而一组人的解实际上就是最大流问题。向图中添加一个源点 s ，令1号点为汇点。新图中的边的容量均为1，从 s 向每个人所在的点连一条容量为1的边。这个网络的最大流就是一组人最多能到达1号点的人数。答案就是所有组的最大流之和。

1.2.3 Complexity

时间复杂度：上界为 $O((n + m) \log n + c(n + m + \text{MaxFlow}(n, m)))$ 。

空间复杂度： $O(n + m)$ 。

难度: 1.5/5。

1.3 E - Harvard

1.3.1 Description

给定一个包含 n 个命令的程序。命令一共三种:

1. **Vx**, 访问第 x 个变量。
2. **Rn**, 标示计数循环的开始, 循环 n 遍。循环可以嵌套。 $n \leq 10^6$ 。
3. **E**, 标示计数循环的结束。

运行程序的计算机有 b 个内存池, 每个内存池可以存 s 个变量。内存池编号为 $0 \sim b-1$, 访问0号内存池的变量需要1单位时间, 而访问其他内存池需要设置访问指针。假设要访问 i 号内存池, 如果访问指针没有指向 i 号内存池, 需要先花费1单位时间改变指针, 然后再花费1单位时间访问。程序会访问 p 个不同的变量, 你需要将这些变量映射到内存池中, 使得总访问时间最小, 并求出这个时间。

$1 \leq b, s \leq 13, p \leq \min(b \times s, 13), n \leq 1000$, **Vx**命令的总执行次数不超过 10^{12} 次。时限10s。

1.3.2 Solution

0号内存池的访问不依赖于指针, 先枚举哪些变量放入0号内存池。后面的过程中我们直接忽略所有0号池中的变量。

对于不在0号内存池内的任意两个变量 i 和 j , 求出有多少次对 j 的访问直接出现在对 i 的访问之后, 设这个值为 $a[i][j]$ 。如果 i 和 j 不在同一个内存池中, 那么答案就要加上 $a[i][j] + a[j][i]$ 。而 a 数组可以通过一些简单的模拟得到。

剩下的就是确定变量的归属。注意到分组的方案数不超过第 p 个贝尔数 B_p , 即将 p 个有标号物品分为若干组的方案数。我们可以用搜索枚举所有方案。枚举所有不在0号内存池的变量, 对于每个变量, 要么和之前某个变量放在同一个内存池(前提是内存池未滿), 要么放入一个新的内存池。这样搜索的复杂度不会超过 $p \cdot B_p$ 。在搜索的同时, 用之前求出的 a 数组计算并更新答案。

1.3.3 Complexity

记 B_i 为第 i 个贝尔数。

时间复杂度：整个搜索部分的复杂度上界为 $O(p \cdot B_p)$ ，单次预处理部分的复杂度为 $O(n)$ 。总复杂度的上界为 $O(p \cdot B_p + 2^p \cdot n)$ 。

空间复杂度： $(n + p^2)$ 。

难度：3/5。

1.4 F - Low Power

1.4.1 Description

有 n 台机器，每台有两组芯片，每组有 k 块芯片。这 $2nk$ 块芯片中的第 i 块的功率为 p_i ，一组芯片的功率为其中芯片的最小功率。设计一个将芯片分到每台机器的方案，使得所有机器中两组芯片功率之差的绝对值的最大值最小。

$2nk \leq 10^6$ ， $p_i \leq 10^9$ 。

1.4.2 Solution

首先二分答案。假设二分得到的答案为 d ，现在我们需要判断答案是否可行。

将 $p[]$ 从小到大排序，一台机器的两组芯片中的最小值应该是相邻的两个数。那么每次考虑相邻的两个整数能否被选择（即其差的绝对值是否不大于 d ）。如果不能选出 n 对，那么答案不可行。

假设选出了 n 对，由于选的是一组的最小值，那么应该存在一种方法将剩下的芯片分给每组，使得每组都能分到 $2(k-1)$ 块功率不小于选出的最小值的芯片。首先在选出 n 对的时候就应贪心地选择尽量小的数字，之后从大到小扫描 $p[]$ ，统计有多少个未被选择的数字。如果对于任意 $1 \leq i \leq n$ 都有，不小于第 i 大的一对数的未被选择的数字不少于 $i \cdot 2(k-1)$ 个，那么答案可行。否则不可行。

1.4.3 Complexity

记 $W = 10^9$ 。

时间复杂度： $O(n \log n + n \log W)$ 。

空间复杂度： $O(n)$ 。

难度：1/5。

1.5 H - Matryoshka

1.5.1 Description

有 n 个套娃排成一排。每个套娃有个大小 a_i ，一个套娃只能放进大小严格大于自己的大小的套娃里。一组完整的套娃从内到外应该是大小为 $1, 2, \dots, m$ 的套娃， m 可以是任意正整数。每次可以合并相邻的两组套娃，代价为需要打开的套娃数。求最少需要花费的代价，使得每组套娃都是完整的，或指出问题无解。

$$n \leq 500, a_i \leq 500。$$

1.5.2 Solution

令 $f[l][r]$ 表示序列中 $[l, r]$ 一段合并成一个套娃的最小代价。转移时枚举分界点 k ，合并两组套娃 $[l, k]$ 和 $[k + 1, r]$ 。方程如下：

$$f[i][j] = \min_{k=i}^{j-1} \{f[i][k] + f[k+1][j] + combine([i, k], [k+1, j])\}$$

考虑合并两组套娃 A 和 B 的代价 $combine(A, B)$ 。 B 中大小大于 A 中最小套娃的所有套娃都应被打开， A 中的套娃也同理。故合并 $[l, r]$ 和 $[i, j]$ 两段套娃的代价为， $[l, r]$ 中大于 $\min\{a[i..j]\}$ 的套娃个数，加上 $[i, j]$ 中大于 $\min\{a[l..r]\}$ 的套娃个数。

直接转移的复杂度达到了 $O(n^4)$ ，需要优化。注意到 $a_i \leq 500$ ，所以可以预处理区间最值以及小于某数的个数的前缀和。

求出 f 之后，令 $g[i]$ 表示前 i 个套娃形成若干完整的组的最小代价。则有：

$$g[i] = \min_{j=0}^{i-1} \{g[j] + f[j+1][i]\}$$

注意 j 能作为 i 的决策点当且仅当 $[j+1, i]$ 形成一个完整的组。

1.5.3 Complexity

令 $W = \max\{a[i]\}$ 。

时间复杂度： $O(n^3 + n^2W)$ 。

空间复杂度： $O(n^2 + nW)$ 。

难度：1.5/5。

1.6 I - Pirate Chest

1.6.1 Description

有一块被划分成 $n \times m$ 网格的水域，第 i 行第 j 列的深度为 $d_{i,j}$ 。你可以造一个长宽不超过 a 和 b ，且长宽高都是整数的箱子，并选择一个紧贴网格的位置把箱子沉下去。箱子沉下去的时候会排开等体积的水，因此水面会上升。箱子的高度必须小于水面高度。求箱子的最大体积。

$$n, m \leq 500。$$

1.6.2 Solution

假设确定了箱子的底面积 S 和下沉位置的最浅深度 h ，设箱子的高度为 x ，可以列出下面的方程：

$$Sx + nmh > nm x$$

解之得

$$x < \frac{nmh}{nm - S}$$

可知当 S 固定时， h 越大 x 也就越大，体积也就越大。因此为了表述简单，直接认为我们要求 $S \cdot h$ 的最大值。

先考虑一个一维的问题，即选定一个长度不超过 l 的区间，并最大化区间长度乘区间最小值。我们可以考虑每个数作为区间最小值时，可以向左向右拓展的最大长度。这个问题可以用单调栈在 $O(n)$ 的时间内解决。我们左右扫描每个数，并维护一个单调不减的单调栈。当一个数出栈的时候，假设这个数在序列中的位置为 j ，使它出栈的数的位置为 i ，它在栈中前一个数的位置为 k （如果它是第一个数那么 $k = 0$ ）。可知， a_k 是 j 之前第一个比 a_j 小的数， a_i 是 j 之后第一个比 a_j 小的数，那么当 a_j 作为区间最小值时，区间长度最长为 $\min(l, i - k - 1)$ 。这样我们可以求出所有极长区间，进而求出答案。

现在考虑原问题。我们可以枚举箱子的长度 c 以及下沉位置的左上角的横坐标 l 。构造一个新序列 p ，满足

$$p_i = \min_{l \leq j < l+c} \{d_{j,i}\}$$

问题就转化为了上面的一维问题。只需要再枚举一下 a 和 b 哪个是长哪个是高，再用上面的公式计算实际高度即可。

1.6.3 Complexity

由于 n 和 m 同阶，故用 n 代替 m 。

时间复杂度： $O(n^3)$ 。

空间复杂度： $O(n^2)$ 。

难度：2.5/5。

1.7 J - Pollution Solution

1.7.1 Description

给定一个 n 个顶点的简单多边形，求这个多边形和一个以原点为圆心、半径为 r 、在 x 轴上方的半圆的交的面积。多边形的所有顶点也在 x 轴上方。

$n \leq 100$, $r \leq 1000$, 坐标的绝对值不超过1500。

1.7.2 Solution

首先求出多边形每条边与半圆的所有交点。交点可以通过公式求得，也可以直接按交点个数分类讨论然后二分和三分。将所有的交点也加入多边形。

我们用类似计算多边形面积的向量法，枚举所有由多边形上相邻两个点和原点构成的三角形。可以发现，一个三角形内部的交的区域，要么是三角形本身，要么是一个扇形。而由于所计算的是有向面积，在多边形外部的部分会被减去。故答案就是所有三角形内部交的区域的有向面积之和。

1.7.3 Complexity

时间复杂度： $O(n)$ （用二分法求交点则为 $O(n \log n)$ ）。

空间复杂度： $O(n)$ 。

难度：2/5。

1.8 K - Up a Tree

1.8.1 Description

有如下三段代码：

```
void Pre(node t) {
    print(t.value);
    if (t.l != null)
        ____ (t.l);
    if (t.r != null)
        ____ (t.r);
}
```

```
void In(node t) {
    if (t.l != null)
        ____ (t.l);
    print(t.value);
    if (t.r != null)
        ____ (t.r);
}
```

```
void Post(node t) {
    if (t.l != null)
        ____ (t.l);
    if (t.r != null)
        ____ (t.r);
    print(t.value);
}
```

代码中的下划线则为Pre、In和Post三个函数中的一个，而且在这六个空中，每个函数恰好出现两次。

给定由这样的三段程序输出的一棵二叉树的前序、中序、后序遍历序列，求程序中下划线分别可以填入哪些函数，并对于每种方案，输出字典序最小的符合条件的二叉树的正确遍历序列。

序列长度 $n \leq 26$ 。

1.8.2 Solution

由于程序方案最多只有90种，可以直接枚举程序方案。那么问题就在于如何判断在给定程序下是否存在符合三个序列的合法二叉树。

如果知道前序或者后序序列，就能知道根对应的字母。如果还知道中序序列，就还能知道左右子树的大小。但是根据程序的不同，可能会出现到深度较深的节点时，只知道两个甚至一个序列的情况。这个时候就需要枚举根对应的字母，以及左子树的大小。

得到了左子树和右子树的序列之后，就要检查是否合法。如果非空序列包含的字母集合不同，那么必然不合法。如果合法，就可以递归处理子树了。如果左右子树都递归得到了合法的二叉树，那么这个子树也是合法的。注意有多个方案时取字典序最小的。

另一个要注意的是，我们需要存下每次的结果，即记忆化搜索。这样状态数最多达到 $O(n^4)$ 级别。记忆化可以直接用map，也可以存在数组中，通过序列的长度，和三个序列的开始下标来访问。代码实现方面，还有一些其他不合法的情况需要判断，较为繁琐。

1.8.3 Complexity

时间复杂度：上界为 $O(n^4)$ （如果使用map则为 $O(n^4 \log n)$ ）。

空间复杂度：上界为 $O(n^5)$ 。

难度：4/5。

Chapter 2

ACM/ICPC World Finals 2012

2.1 B - Curvy Little Bottles

2.1.1 Description

给定一个 n 次的多项式，截取多项式图像在 $x \in [l, r]$ 的部分，绕 x 轴旋转一周，并封住 $x = l$ 的一端，得到一个容器。求出容器的体积，以及如果要刻等体积线，需要刻在离底端多远的位置。

$n \leq 10$ ，多项式所有系数的绝对值不超过100。

2.1.2 Solution

令多项式为 $f(x)$ 。容器的体积 V 可以如下计算：

$$V = \int_l^r f(x)^2 dx$$

令 $g(x) = f(x)^2$ ， $g(x)$ 为一个 $2n$ 次的多项式。令 a_i 为其 x_i 次项的系数，那么有：

$$V = \int_l^r g(x) dx = \sum_{i=0}^{2n} \frac{a_i}{i+1} (r^{i+1} - l^{i+1})$$

而等体积线的位置则可以通过二分法求出。

2.1.3 Complexity

令 h 为二分次数。

时间复杂度： $O(n^2 + nh)$ 。

空间复杂度： $O(n)$ 。

难度：0/5。

2.2 C - Bus Tour

2.2.1 Description

给定一个 n 个点 m 条边的图，点的编号为 $0 \sim n-1$ ，边有边权。0号点是起点， $n-1$ 号点是终点， $1 \sim n-2$ 号点是旅馆。求一条从起点出发，访问每个旅馆各一次，再到终点的路径，以及一条从终点出发，访问每个旅馆各一次，再回到起点的路径。要求在第一条路径中访问的前 $\lfloor \frac{n-2}{2} \rfloor$ 个旅馆，也是第二条路径中访问的前 $\lfloor \frac{n-2}{2} \rfloor$ 个旅馆。可以经过一个旅馆而不访问。求最短的路径长度和。

$n \leq 20$, $m \geq 2$ 。时限10s。

2.2.2 Solution

先求出任意两点间的最短路长度。记 $d[x][y]$ 代表 x 到 y 的最短路长度。

令 $f[s][t][x]$ 代表从 s 出发到达 t ，中途访问了集合 x 中的点的最短路径长度。用二进制表示集合，可以通过状压DP在 $O(2^n n^3)$ 的时间复杂度内求出 f 数组。假设访问的前 $\lfloor \frac{n-2}{2} \rfloor$ 个旅馆的集合为 S ，剩下的旅馆的集合为 T ，路径可以表示为

$$d[0][s] + f[s][t][S] + d[t][x] + f[x][y][T] + d[y][n-1]$$

s 和 t 都是 S 中的点， x 和 y 都是 T 中的点。而返程的路径可以类似表示，而且两条路径是不相关的。这样可以在 $O(2^n n^3)$ 的复杂度内解决问题。由于时限有10s，可以通过。

但是我们可以进一步优化。我们其实可以只把一条路径分为三个部分，从起点出发访问前 $\lfloor \frac{n-2}{2} \rfloor$ 个旅馆并到达某个点 x ，以及从终点出发访问剩下的旅馆并到达某个点 y ，以及 $d[x][y]$ 这三个部分。因此可以令 $f[t][x]$ 代表从起点出发，访问了集合 x 中的点并到达 t 的最短路径长度。类似地令 $g[t][x]$ 代表从终点出发的最短路径长度。路径就可以表示为

$$f[x][S] + d[x][y] + g[y][T]$$

这样只需要 $O(2^n n^2)$ 就可以解决问题。注意在实现时由于 g 的起点是 $n-1$ 号点，可以把所有点的编号倒过来处理。

2.2.3 Complexity

时间复杂度： $O(2^n n^2)$ 。

空间复杂度: $O(n2^n)$ 。

难度: 2/5。

2.3 E - Infiltration

2.3.1 Description

给定一个 n 个点的有向图, 任意两点之间都有且仅有一条边。求这个图的最小覆盖集。

$n \leq 75$ 。

2.3.2 Solution

给定的图实际上是竞赛图。而关于竞赛图有一些性质:

性质. 在 n 个点的竞赛图中, 一定存在一个节点的出度为 $\lceil \frac{n-1}{2} \rceil$ 。

证明: 证明其实很简单。令 $d(u)$ 为点 u 的出度, 有

$$\sum d(u) = \binom{n}{2} = \frac{n(n-1)}{2}$$

那么根据鸽巢原理, 必然会存在一个点的出度不小于 $\lceil \frac{1}{n} \cdot \frac{n(n-1)}{2} \rceil = \lceil \frac{n-1}{2} \rceil$ 。 \square

由此我们可以得到下面的结论:

结论. n 个点的竞赛图的最小覆盖集的大小一定不会大于 $\lfloor \log_2 n \rfloor$ 。

证明: 考虑一个贪心的算法: 每次选择覆盖了出度最大的点。由上面的性质可得, 出度最大的点的出度不小于 $\lceil \frac{n-1}{2} \rceil$ 。也就是说, 每次最少可以删掉 $\lfloor \frac{n}{2} \rfloor + 1$ 个点, 而删除之后的图是一个不超过 $\lfloor \frac{n-1}{2} \rfloor$ 个点的竞赛图。故 n 个点的竞赛图的最小覆盖集的大小一定不会大于 $\lfloor \log_2 n \rfloor$ 。 \square

而在本题中, n 最大为75, 也即最小覆盖集的大小不会超过6。那么我们可以暴力枚举所有大小不超过5的集合并判断是否是覆盖集。如果不存在大小不超过5的覆盖集, 就用贪心算法构造一个大小为6的覆盖集。由于本题中 $n \leq 75$, 大小不超过5的集合个数为 $\sum_{i=1}^5 \binom{n}{i} \leq 18,545,215$, 是可以承受的。不过在判断是否是覆盖集时需要用二进制压位优化。

2.3.3 Complexity

令 $S = \sum_{i=1}^5 \binom{75}{i} = 18,545,215$ 。

时间复杂度：上界为 $O(Sn^2/32)$ 。

空间复杂度： $O(n^2/32)$ 。

难度：2.5/5。

2.4 G - Minimum Cost Flow

2.4.1 Description

给定三维空间中的 n 个枢纽的坐标，有一些枢纽上有洞，第 i 个枢纽有 k_i 个洞。存在 m 条已有的管道连接了一些枢纽。现在你需要从源点向汇点输水，你可以控制水压，即设定一个高度，使得水无法到达 z 轴坐标大于这个高度的枢纽。但是如果水流经过了一个有洞的枢纽，就会导致漏水。填补一个洞的代价为 0.5，在不属于同一个枢纽的两个洞之间连一个新的管子的代价是两个枢纽的欧氏距离。求从源点向汇点输水，并保证不漏水的的核心代价。

$n \leq 400$, $m \leq 50000$ 。

2.4.2 Solution

我们首先枚举输水的高度，假设为 h 。我们可以直接忽略所有 z 轴坐标大于 h 的点，只考虑剩下的点。把所有连通块缩成一个点，并令这样的连通块中洞的个数之和为 p_i 。

容易发现，我们只有在迫不得已的情况下才会选择建新的管子，因为堵上两个洞的代价不会比连接两个洞的代价大。而所谓迫不得已的情况就是源点和汇点并不连通。那么现在问题就变成了，需要找一条从源点所在连通块到汇点所在连通块的路径，使得代价最小。

记源点和汇点所在的连通块分别为 S 和 T ，如果 $S = T$ ，则代价为 $0.5p_S$ 。否则，必须有 $p_S > 0$ 且 $p_T > 0$ ，不然无法连接出去。对于途经的其他连通块 i ，应当满足 $p_i \geq 2$ ，且 i 中存在至少两个有洞的点。一定不会在同一个点上连两根新管道。这样我们可以建出一个新图，在这个图上用类似 SPFA 的转移就可以求出到每个连通块的最小花费。在所有枚举的高度中取最小值即可。

2.4.3 Complexity

时间复杂度: $O(n^3 + nm)$ 。

空间复杂度: $O(n^2 + m)$ 。

难度: 1.5/5。

2.5 K - Stacking Plates

2.5.1 Description

有 n 个序列, 第 i 个序列有 h_i 个数, 每个序列中的数按照非降序排列。现在要把这些序列合并成一个序列, 有两种可用操作:

1. 把一个序列从中间断开, 拆成两个。
2. 把两个序列合为一个, 要求一个序列末尾的数不大于另一个序列开头的数。

求最少操作次数。

$$n, h_i \leq 50。$$

2.5.2 Solution

我们给原来每个序列中的元素染上一种颜色。问题可以转化成, 将所有数排成非降序, 使得相邻的异色数对数最少。令 x 为相邻的异色数对数, 那么答案为 $x - n + 1$ 。

如果没有相同的数, x 可以直接统计。而如果有相同的数, 可以DP解决。令 $f[i][j]$ 表示排好了值不超过 i 的数, 末尾的数颜色为 j 时的最小的 x 。转移时枚举上一段和这一段末尾的颜色, 根据这一段不同颜色的数的个数进行转移。可以用滚动数组优化。

2.5.3 Complexity

令 $m = \sum h_i$, 即数的总数。

时间复杂度: $O(mn^2)$ 。

空间复杂度: $O(n)$ 。

难度: 1/5。

2.6 L - Takeover Wars

2.6.1 Description

有A和B两个人，A有一个长度为 n 的序列 $a[]$ ，B有一个长度为 m 的序列 $b[]$ 。A先手，双方轮流行动。每次有两种操作：

1. Takeover，从自己序列中选择一个数 x ，并删掉对方序列中一个小于 x 的数。
2. Merge，从自己序列中删除两个数 x 和 y ，并加入一个数 $x + y$ 。

先无法操作的一方输。求先手是否必胜。

$n, m \leq 100000$ 。

2.6.2 Solution

稍作分析，可以得出下面的结论：

1. 如果要进行一次Takeover，一定会删去对方序列中最大的数。如果没法删去最大的数，就不会进行Takeover。
2. 如果要进行一次Merge，一定会将两个最大的数合并。
3. 如果上一轮已经被对方Takeover，那么这一轮一定会Merge。
4. 如果对方在Merge之后，我方还可以进行Takeover，那么我方必胜。

通过这些结论可知，确定了第一步之后，之后的步骤都已经确定。所以只需枚举第一步并模拟即可。

2.6.3 Complexity

时间复杂度： $O(n + m)$ 。

空间复杂度： $O(n + m)$ 。

难度：0.5/5。

Chapter 3

ACM/ICPC World Finals 2011

3.1 A - To Add or to Multiply

3.1.1 Description

定义操作序列为一个只包含A和M的字符串。程序首先读入一个数，然后逐步执行操作序列。A代表给数字加上 a ，M代表给数字乘上 m 。执行完毕后会输出得到的数。

给定 a, m ，保证输入的数字属于 $[p, q]$ ，求一个最短的操作序列，使得输出的数属于 $[r, s]$ 。如果有多个最短的操作序列，输出字典序最小的。

$$1 \leq a, m, p, q, r, s \leq 10^9。$$

3.1.2 Solution

我们先考虑几种特殊的情况：

1. $q > s$ ，答案是impossible。
2. $p \geq r$ 且 $q \leq s$ ，答案是empty。
3. $m = 1$ ，如果 $r - p > s - q$ ，答案是impossible。否则令 $d = \lceil \frac{r-p}{a} \rceil$ ，如果 $p + ad \geq r$ 且 $q + ad \leq s$ ，那么序列为 d 个A，否则还是impossible。

后面我们只讨论 $m > 1$ 、 $p < r$ 且 $q \leq s$ 的一般情况。

设序列中有 k 个M，显然 $k \leq \lfloor \log_m (s/q) \rfloor$ 。我们可以枚举 k 的所有可能值。

我们令 $b_0, b_1, b_2, \dots, b_k$ 代表在第一个M之前、第一和第二个M之间、第二和第三个M之间……第 k 个M之后有几个A。令 $f(x)$ 代表输入为 x 时输出的数，则有：

$$f(x) = xm^k + a \sum_{i=0}^k b_i m^{k-i}$$

显然 $f(x)$ 单调递增。那么我们需要得到一个 b 序列，使得 $f(p) \geq r$ 且 $f(q) \leq s$ 。序列长度就是 $k + \sum b_i$ 。

不难得出，给 $b_i + 1$ 加上 m 不如给 b_i 加上1。设 $c = \sum_{i=0}^k b_i m^{k-i}$ ，有：

$$\left\lceil \frac{r - pm^k}{a} \right\rceil \leq c \leq \left\lfloor \frac{s - qm^k}{a} \right\rfloor$$

如果 c 的取值范围为空集则不合法。

那么令 b_k, b_{k-1}, \dots 为 $\left\lceil \frac{r - pm^k}{a} \right\rceil$ 分解为 m 进制的低位到高位。假设一共 cnt 位。如果 $cnt > k + 1$ ，就只分解到 $k + 1$ 位为止，更高的位都赋给 b_0 。

称一次操作为令 $b_i = 0$ ，并令 b_{i-1} 加1。如果 $b_i > 0$ ，而且操作后 $c \leq \left\lfloor \frac{s - qm^k}{a} \right\rfloor$ ，那么进行这样的操作必然会令答案更优。于是我们从后往前考虑除了最高位的每一位，如果当前位可以操作则操作。最后得到的序列就是含有 k 个 m 的最短且字典序最小的序列。答案也就是枚举的所有 k 中最短的序列，如果有多个长度相同的最短序列，那么 k 较小的序列字典序必然更小。

3.1.3 Complexity

时间复杂度： $O(\log^2 s)$ 。







空间复杂度： $O(\log s)$ 。

难度：2.5/5。

3.2 C - Ancient Messages

3.2.1 Description

给定一个 $n \times m$ 的二进制图像，求下面六种象形文字分别出现了多少次。一个象形文字的所有黑色像素四连通。

| | | | | | |
|---|---|---|---|---|---|
|  |  |  |  |  |  |
| Ankh | Wedjat | Djed | Scarab | Was | Akeht |

$n, m \leq 200$ 。

3.2.2 Solution

观察可以发现，每个图形的“洞”的个数都不相同，分别是0 ~ 5个。那么只需要对外圈的白色像素做一遍Floodfill，求出每个文字的轮廓，然后再对每个图形内部的白色像素Floodfill，求出洞的个数。最后统计一下输出答案即可。

需要注意的是，可能有图形将整张图分成了多个部分。可以通过在最外面加一圈白色像素来处理这种情况。

3.2.3 Complexity

时间复杂度： $O(nm)$ 。

空间复杂度： $O(nm)$ 。

难度：1/5。

3.3 E - Coffee Central

3.3.1 Description

在一个 $a \times b$ 的网格图上有 n 个特殊点。定义两点间的距离为曼哈顿距离，有 q 次询问，每次给定 d ，求图中的一个格点，使得到格点距离不超过 d 的特殊点个数最多。

$a, b \leq 1000$, $0 \leq n \leq 500000$, $q \leq 20$ 。

3.3.2 Solution

我们称到一个特殊点距离不超过 d 的所有格点为这个特殊点的可行域。可以发现，所有点的可行域都是以该点为中心的一个正菱形。我们考虑把坐标系顺时针旋转45度，并重新定义坐标系。原来的点 (x, y) 会变成 $(x + y, y - x)$ ，可行域也成了正方形。那么问题就转化成了，给定若干矩形，求一个点使得它被最多的矩形覆盖。

这是一个经典问题，用一些高级数据结构显然可做，但是还有更简单的做法。注意到所有的查询都在修改之后，所以可以直接对修改做二维的差分，全部修改完之后求一遍前缀和就可以得到每个点的覆盖次数了。

另外有两个需要注意的地方。一是旋转之后坐标范围会变为原来的两倍，而且纵坐标可能为负；二是 n 可能为0。

3.3.3 Complexity

时间复杂度: $O(q(ab + n))$ 。

空间复杂度: $O(n + ab)$ 。

难度: 1/5。

3.4 H - Mining Your Own Business

3.4.1 Description

给定一个 n 个点 m 条边的无向连通图，从中选出一些点打上标记，要求无论删去图中哪个点，剩下的点都能够到达至少一个标记点。求最少需要标记多少点，并求出选择的方案数。

$m \leq 50000$ 。

3.4.2 Solution

首先求出图中所有的割点，并从图中删去。将剩下的每个连通块缩成一个点，再把割点加回来，得到一个新图。可知新图是一棵树。那么只需要在每个叶子节点对应的连通块中都标记一个点即可，方案数就是叶子节点对应连通块的大小的乘积。

注意有一个特殊情况，即当整个图中不存在割点时，需要标记两个点，因为被标记的点也可以被删掉。

3.4.3 Complexity

时间复杂度: $O(n + m)$ 。

空间复杂度: $O(n + m)$ 。

难度: 2/5。

3.5 J - Pyramids

3.5.1 Description

给定正整数 n ，将 n 拆分成尽量少的数，使得每个数是某个合法金字塔的石头数。求一个拆分方案。有多种方案时，输出从大到小排序后字典序最大的方案。

$n \leq 10^6$ 。时限7s。

3.5.2 Solution

在数据范围内，合法的金字塔最多只有320种，我们可以考虑直接做0-1背包。枚举时按照石头数从小到大枚举每个金字塔，更新时只要不劣于原有方案就更新，这样才能保证字典序。至于方案，在数据范围内，如果有解，那么拆分出的数字不会超过6个。直接记下整个方案，转移时维护一下即可。

3.5.3 Complexity

令 $k = 320$ 。

时间复杂度： $O(nk)$ 。

空间复杂度： $O(n)$ 。

难度：1/5。

3.6 K - Trash Removal

3.6.1 Description

给定一个 n 个顶点的简单多边形，求经过旋转之后，这个多边形最小可能的宽度。

$n \leq 100$ 。

3.6.2 Solution

先对多边形求凸包，得到的凸包和原多边形是等价的。最小的宽度一定是多边形某条边恰好垂直于地面时取到的，那么可以枚举这条边，然后求其它点到这条边的最大值。所有最大值中的最小值就是答案。这实际上就是经典的旋转卡壳算法。

3.6.3 Complexity

时间复杂度： $O(n^2)$ （可以做到 $O(n \log n)$ ）。

空间复杂度： $O(n)$ 。

难度：0.5/5。

Chapter 4

ACM/ICPC World Finals 2010

4.1 B - Barcode

4.1.1 Description

条形码序列可以被描述为一系列由宽条和窄条构成的序列，宽条的宽度为窄条的两倍。序列中每个元素代表一个条的宽度，可能与实际宽度有5%的误差。条形码序列中5个连续的条代表一个字符（原题附有字符表），相邻的字符之间用一个窄条隔开。一个特殊的字符是起止符，序列最开始和最末尾的一个字符应为起止符。序列中还有两位校验码，可以根据公式计算。

给定长度为 n 的条形码序列，判断是否合法，并解码。注意序列可能是从左至右的，也可能是从右至左的。

$n \leq 200$ 。

4.1.2 Solution

首先判断出哪些条是宽条，哪些是窄条。由于起止符中含有窄条，整个序列中宽度最小的元素必然为窄条。取其1.5倍为阈值，大于这个值的都判为宽条。找出最宽和最窄的宽条，判断最窄宽度与最宽宽度的比值是否不小于95/105，如果小于则说明存在宽条的宽度不在误差范围内，为不合法序列。窄条类似判断。之后再判断窄条和宽条之间的比例关系，如果不是在误差允许范围内的两倍，同样为不合法序列。

由于起止符不是回文串，所以必然只有一个顺序是可行的。剩下的就是简单的模拟了，校验码直接根据公式计算即可。另外再判断一下序列的长度是否模6余5，否则一样不合法。

4.1.3 Complexity

时间复杂度: $O(n)$ 。

空间复杂度: $O(n)$ 。

难度: 1.5/5。

4.2 D - Castles

4.2.1 Description

有 n 座城堡形成一棵树，攻打第 i 座城堡需要 a_i 名士兵，战斗中会牺牲 m_i 名士兵，占领之后需要 h_i 名士兵把守。可以任意选择第一座攻打的城堡，之后可以攻打与已占领的城堡相邻的其他城堡。军队在同一方向上经过同一条路最多一次。求占领所有城堡最少需要多少名士兵。

$n \leq 100$ 。

4.2.2 Solution

占领城堡需要的士兵数 $f_i = \max(a_i, m_i + h_i)$ ，占领城堡消耗的士兵数 $g_i = m_i + h_i$ ，因此占领了一座城堡之后还能继续战斗的士兵数就是 $f_i - g_i$ 。如果可以以任意顺序攻打城堡，就可以按照 $f_i - g_i$ 贪心，优先攻打 $f_i - g_i$ 较大的城堡。

而题目中，如果我们确定了第一座攻打的城堡 r ，攻打城堡的顺序实际上就是以 r 为根的树的某个DFS序。攻打了一座城堡之后，必然会先打完其所有的儿子，再去攻打别的城堡。因此可以令 tf_i 和 tg_i 分别代表占领以 i 为根的子树需要的士兵数和消耗的士兵数，然后树形DP。处理一座城堡时，按照儿子的 $tf_i - tg_i$ 贪心。

4.2.3 Complexity

时间复杂度: $O(n^2 \log n)$ 。

空间复杂度: $O(n)$ 。

难度: 1/5。

4.3 F - Contour Mapping

4.3.1 Description

有 n 排测高点，奇数排有 m 个点，偶数排有 $m + 1$ 个点，点与点之间的位置关系与连边如下图所示：

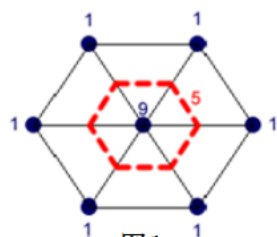


图1

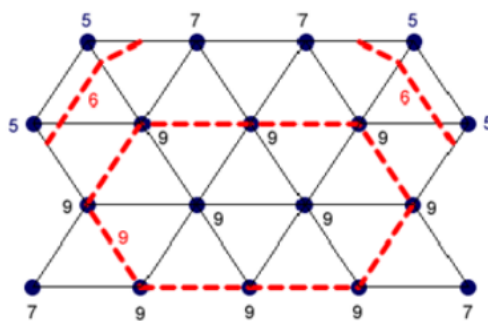


图2

图中每个三角形都是等边三角形。

每个测高点有一个海拔高度。定义海拔为 h 的等高线如下：取出图中所有海拔为 h 的点（不一定是测高点，可以是平面上的任意点），这些点分为很多个连通块，每个部分要么是孤立点，要么形成一条线段，要么形成一个多边形。取出形成的所有线段以及多边形的所有边，其并即为海拔为 h 的等高线。上图中红色虚线为等高线。

求这个地图中所有海拔为 h 的非负整数倍的等高线的长度之和。

$n, m \leq 100$, $1 \leq h \leq 1000$ 。

4.3.2 Solution

等高线只有两种情况，一是在三角形内部，二是在三角形的某条边上。先考虑第一种情况。

我们可以分开处理每个三角形。一共有三种三角形：有一个、两个、三个顶点的海拔相等的，分别称为一类、二类和三类三角形。这里不会有三类三角形。对于一类三角形，假设三个顶点的海拔分别为 a, b, c 且 $a < b < c$ ，在海拔 a 与 c 的点的连边上找到海拔为 b 的点，将三角形分成两个二类三角形。

现在只有二类三角形了。不难发现，在一个二类三角形内的所有等高线的长度呈等差数列。于是这部分可以用公式解决。注意如果三角形边上有等高线，不能算进来。

剩下的就是等高线在三角形边上的情况了。有等高线的边的两个顶点海拔必

须相同，而且都是 h 的非负整数倍。那么枚举每条边，如果这条边只在一个三角形上，那么它一定是边界，这条边上也就会有等高线。否则如果这条边所在的两个三角形中至少有一个是二类三角形，那么这条边上也有等高线。这样就能处理在三角形一边上的情况了。

两部分之和就是答案。坐标处理有些麻烦，需要细心。

4.3.3 Complexity

时间复杂度： $O(nm)$ 。

空间复杂度： $O(nm)$ 。

难度：2/5。

4.4 G - The Islands

4.4.1 Description

有 n 座岛屿，岛屿的横坐标互不相同，并按照横坐标从小到大编号为 $0 \sim n-1$ 。船从0号岛出发，经过一些岛到达 $n-1$ 号岛，再经过剩下的所有岛返回0号岛。在从左到右的途中，必须一直向右；在从右到左的途中，必须一直向左。有两座特殊的岛，在从左至右的途中必须访问其中一座，从右至左的途中访问另一座。求整个路径的最短长度，并输出方案。

$n \leq 100$ 。

4.4.2 Solution

我们把从右至左的路径反过来，视为两条从0号岛出发的除起点和终点外不相交的路径。先不考虑特殊的岛，我们可以得到一个DP的做法。令 $f[i][j][k]$ 表示，已经走遍了前 i 座岛，此时第一条路径的终点为 j ，第二条路径的终点为 k 。转移时枚举当前的岛是在第一条路径中还是第二条路径中。

而对于特殊岛，只要枚举一下两座岛的“分配”方式，在转移的时候特判一下，只允许添加到指定的路径中。最后答案就是两种分配方式中

$$\max(\max\{f[n][i][n]\}, \max\{f[n][n][i]\})$$

的较大值。方案则记录一下转移即可。需要注意的是题目要求方案中第二个访问的是1号点，所以可能需把求出的整条路径反转。

另外，目前的状态表示是可以优化掉一维的。因为 j 和 k 中必定有一个是 i ，所以记录另外那个数和哪一个是 i 就可以了。但是对于题目的数据范围来说，不优化也足以通过。

4.4.3 Complexity

时间复杂度： $O(n^3)$ （优化状态表示后则为 $O(n^2)$ ）。

空间复杂度： $O(n^3)$ （优化状态表示后则为 $O(n^2)$ ）。

难度：0.5/5。

4.5 J - Sharing Chocolate

4.5.1 Description

有一块 $r \times c$ 的巧克力，需要切成 n 块，使得第 i 块的大小为 a_i 。每次只能将一块巧克力横着或者竖着切成两块。求是否可行。

$n \leq 15$, $r, c \leq 100$ 。

4.5.2 Solution

不难得到下面这个状压DP的做法。令 $f[i][j][k]$ 表示集合 i 中的若干块巧克力能否由长度 k 宽度 j 的巧克力切出。令 $sum[i]$ 表示集合 i 中巧克力的大小之和，显然有长度 $k = \frac{sum[i]}{j}$ ，因此不用记录 k 。转移时，枚举 i 的一个子集 x ，并考虑是横着切还是竖着切。如果可以横着切，那么从

$$f[x][\min(j, sum[x]/j)] \text{ and } f[i - x][\min(j, sum[i - x]/j)]$$

转移，竖着切类似。令 all 为全集，最后如果 $f[all][\min(x, y)]$ 为真，那么可行，否则不可行。

这个DP的状态数为 $O(2^n r)$ ，转移总复杂度可以达到 $O(3^n r)$ ，可能超时。但是可以发现，可以转移到最终状态的状态并不多，因此直接从最终状态开始记忆化搜索，就可以通过此题。

4.5.3 Complexity

时间复杂度：最坏情况下 $O(3^n r)$ 。

空间复杂度： $O(2^n r)$ 。

难度: 1/5。

Chapter 5

ACM/ICPC World Finals 2009

5.1 A - A Careful Approach

5.1.1 Description

有 n 架飞机，每架飞机都有一个安全着陆时间区间，飞机必须在这个时间区间内着陆。求飞机着陆最小间隔的最大值。

$n \leq 8$ 。

5.1.2 Solution

答案满足二分的性质，故我们二分答案。判断是否可行时，枚举所有着陆的顺序，在满足顺序要求和答案要求的前提下令每架飞机尽量早地着陆，如果存在一种着陆方案则可行，否则不可行。

5.1.3 Complexity

令 W 为答案的范围。

时间复杂度： $O(n! \log W)$ 。

空间复杂度： $O(n)$ 。

难度：0/5。

5.2 B - My Bad

5.2.1 Description

一个电路中有 n 个输入， g 个门，和 u 个输出。这其中有最多一个门出了问

题，问题要么是输出总与正确输出相反，要么是总是输出0或1。现在对电路进行了 b 次测试，判断是否得到了与预期不同的输出，并找到出错的门，或指出有不止一种可能性。

$$n \leq 8, g, u \leq 19。$$

5.2.2 Solution

门之间的关系构成一个拓扑图，先构出这个拓扑图，并求出拓扑序。之后就可以直接按照拓扑序模拟。如果得到了错误的输出，就枚举出错的门，以及错误的种类，再模拟验证即可。

5.2.3 Complexity

时间复杂度： $O(bg^2 + g(n + u))$ 。

空间复杂度： $O(b(n + u) + g)$ 。

难度：0/5。

5.3 E - Fare and Balanced

5.3.1 Description

给定一个 n 个点 m 条边的有向无环图，每个点都在从1到 n 的路径上，每条边有边权。现在要增加一些边的边权，使得任意一条从1到 n 的路径上的边权和都相同且最小，而且任意一条从1到 n 的路径上增加了边权的边不能超过一条。构造一个方案，或指出无解。

$$n, m \leq 50000。$$

5.3.2 Solution

首先，如果有解，那么路径边权和一定是原图中的最长路。证明显然。下面考虑怎么构造方案。

求出从1到 i 的每条路径是否长度都相同，记为布尔值 $p[i]$ 。考虑一条边 $i \rightarrow j$ ，边权为 w 。如果 $p[j]$ 为真，那么不能增加这条边的边权。可以简单证明如下：

1. 如果 $i \rightarrow j$ 在一条最长路上，那么显然不能增加边权，因为这会导致最长路长度增加。

2. 否则, 在任意一条经过 $i \rightarrow j$ 的路径上, 这条边之后一定存在一条边 $x \rightarrow y$ 满足 $p[x]$ 为真且 $p[y]$ 为假, 修改 $x \rightarrow y$ 的效果和修改 $i \rightarrow j$ 的效果相同。

那么我们只在满足 $p[i]$ 为真且 $p[j]$ 为假的边 $i \rightarrow j$ 上增加权值。由于一条路径上这种边最多只有一条, 故只要存在方案, 就一定合法。

令 $d[i]$ 为从 1 到 i 的最长路长度, $val[i]$ 为点 i 必须增加的权值, 那么边 $i \rightarrow j$ 应增加的权值为 $d[j] - d[i] + w + val[j]$ 。我们按拓扑序逆序, 从 n 倒推到 1。对于一条边 $i \rightarrow j$, 如果 $p[i]$ 为真, 那么给这条边增加权值。否则把这个权值赋给 $val[i]$, 如果此时 $val[i]$ 已有值, 而且与当前权值不同, 则无解。

5.3.3 Complexity

时间复杂度: $O(n + m)$ 。

空间复杂度: $O(n + m)$ 。

难度: 2/5。

5.4 F - Deer-Proof Fence

5.4.1 Description

平面上有 n 个点, 把它们分为若干组, 分别用围栏围起来。围栏上任意一点到围栏内点的距离不得小于 d 。求围栏长度和的最小值。

$n \leq 9$, $d > 0$ 。最多 100 组测试数据。

5.4.2 Solution

先枚举分组方案, 方案数不会超过第 9 个贝尔数 $B_9 = 21147$ 。对于一组内的点, 最优且合法的围栏一定是这些点的凸包外扩 d 个单位长度, 围栏长度也就是凸包周长 + 半径为 d 的圆的周长。那么对每组计算长度再求和即可。不需要判断是否有围栏会相交, 因为相交的方案一定不如把相交的两组合并更优。

一个小优化是, 在枚举方案之前先对所有点排序。这样在之后计算的时候就不用排序了。

5.4.3 Complexity

记 B_i 为第 i 个贝尔数。

时间复杂度: 单组数据 $O(nB_n + n \log n)$ 。

空间复杂度: $O(n)$ 。

难度: 0.5/5。

5.5 G - House of Cards

5.5.1 Description

定义一种游戏，一共有 $m - 4$ 回合。每回合双方轮流抽取下一张牌，然后操作，并根据规则计算得分。具体规则这里略去。所有牌都正面向上，即双方都知道自己和对方会抽到哪些牌。给出所有牌，并指定先手，假设双方都按最优策略操作，问其中指定的一人与另一人的分差的最大值为多少。

$5 \leq m \leq 13$ 。

5.5.2 Solution

游戏最多只会进行9个回合，而可行的状态数其实并不多。因此我们直接使用极小化极大算法搜索，并加入Alpha-Beta剪枝，就可以通过这题。

5.5.3 Complexity

时间复杂度: 无法给出准确复杂度。

空间复杂度: $O(m)$ 。

难度: 1.5/5。

5.6 H - The Ministers' Major Mess

5.6.1 Description

有 n 个议案和 m 位大臣。每位大臣有 $1 \sim 4$ 个投票，每个投票是对某一议案的赞成或者反对。请找出一个通过议案的方案，使得每位大臣有超过一半的投票被满足。如果有解，输出每个议案是必须通过、必须不通过，或者皆有可能。

$n \leq 100, m \leq 500$ 。

5.6.2 Solution

首先有一个观察：如果一位大臣有不超2个投票，那么他的投票全部都要被满足。如果一位大臣有2个以上的投票，那么至多一个投票可以不满足。

我们可以构建一个2-SAT的模型。对每个议案建两个点，一个代表通过，一个代表不通过。对于不超过2个投票的大臣，直接选择投票对应的点。对于2个以上投票的大臣，枚举不满足的投票，并与其他投票对应的点连边。

构完图之后，对于每个尚未被选择的点，DFS判断它是否必须被选择。如果存在一个议案，通过的点和不通过的点都被选择了，则无解。否则如果两个点中有一个被选择了，就可以唯一确定这个点的状态；否则皆有可能。

5.6.3 Complexity

时间复杂度： $O(nm)$ 。

空间复杂度： $O(n + m)$ 。

难度：1.5/5。

Chapter 6

ACM/ICPC World Finals 2008

6.1 A - Air Conditioning Machinery

6.1.1 Description

给定一个 $a \times b \times c$ 的盒子，需要在盒子内部铺设管道。盒子上有两个开口，而可用的管道部件只有6个图1中的弯管。弯管占用恰好4个单位立方体。弯管不能放在盒子外部，或者穿过盒子。两个弯管不能重合，而且开口必须对接。例如，对于图3中的盒子，一个可行的管道铺设方案如图2所示。

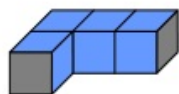


Figure 1

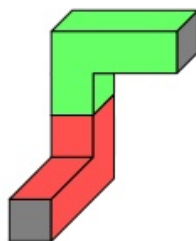


Figure 2

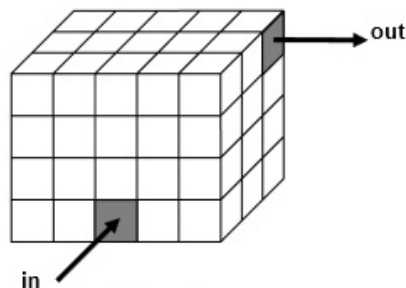


Figure 3

求在给定的盒子内铺设管道最少需要几个弯管，或指出无法用不超过6个弯管铺设管道。

$$a, b, c \leq 20.$$

6.1.2 Solution

由于最多使用6个弯管，而确定了对接面之后，弯管只有8种摆放方式，故可以直接枚举答案，并暴力枚举所有方案。总方案数不超过 $\sum_{i=1}^6 8^i = 299,592$ 。枚举了一种方案之后只要判断管道是否会穿过盒子或者重合即可。

6.1.3 Complexity

令 $S = \sum_{i=1}^6 8^i = 299,592$ 。

时间复杂度：上界为 $O(S \cdot abc)$

空间复杂度： $O(abc)$ 。

难度：0.5/5。

6.2 B - Always an Integer

6.2.1 Description

给定最高次数为 n 的多项式 $f(x)$ 以及正整数 d ，求当 x 取任意正整数时， $f(x)/d$ 是否恒为整数。

$n \leq 100$ ，所有系数和 d 都在 `int` 范围内。

6.2.2 Solution

结论. 对于最高次数为 n 的多项式 $f(x)$ 和正整数 d ，如果对于 $1 \leq x \leq n+1$ 都有 $f(x)/d$ 为整数，那么当 x 取任意正整数时， $f(x)/d$ 恒为整数。

证明： 用数学归纳法证明。

对于最高次数为 0 的多项式，显然判断当 $x = 1$ 是否成立即可。

假设已经证明对于最高次数为 $n-1$ 的多项式，只需判断 $1 \leq x \leq n$ 时是否成立即可。考虑最高次数为 n 的多项式 $f(x)$ ， $f(x)/d$ 恒为整数等价于， $f(1)/d$ 为整数且对于任意 $x \geq 1$ 有 $(f(x+1) - f(x))/d$ 为整数。令 $g(x) = f(x+1) - f(x)$ ，可知 $g(x)$ 为最高次数不超过 $n-1$ 的多项式，因此只需判断 $1 \leq x \leq n$ 时 $g(x)/d$ 是否为整数，也即判断当 $1 \leq x \leq n+1$ 时 $f(x)/d$ 是否为整数。 \square

运算过程中，为了避免高精度计算，可以只求对 d 取模的结果。

6.2.3 Complexity

时间复杂度： $O(n^2)$ 。

空间复杂度： $O(n)$ 。

难度：1/5。

6.3 D - The Hare and the Hound

6.3.1 Description

题意较为复杂，这里略去。

6.3.2 Solution

直接按照所述模拟即可。有几个需要注意的地方：

- 道路并不一定是笔直的，出道路的角度和进入道路的角度并不一定相同，如样例。
- 在没有违反主要道路规则时，兔子如果遇到死胡同会原路返回。
- 起点可能和终点相同，这个时候路径长度不为0，仍旧会从起点出发，绕一圈再回来。

6.3.3 Complexity

时间复杂度：无法给出准确复杂度。

空间复杂度： $O(360ncp + nroad + ncm)$ 。

难度：2.5/5。

6.4 F - Glenbow Museum

6.4.1 Description

对于一个由 n 条平行于坐标轴的边组成的简单多边形，记其顶点序列为从一个顶点出发，逆时针绕多边形一圈，用R表示一个90度的内角，O表示一个270度的内角，如此得到的序列。由于没有指定边长，所以一个顶点序列可以对应多个多边形。称一个顶点序列合法，当其对应的至少一个多边形内部存在一个点可以看到整个多边形内部。求有多少个长度为 n 的顶点序列合法。

$n \leq 1000$ 。

6.4.2 Solution

设长度为 n 的顶点序列中存在 x 个R。由多边形的内角和公式得

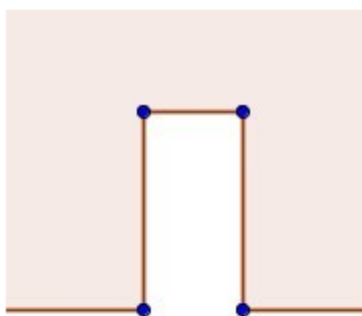
$$90x + 270(n - x) = 180(n - 2)$$

解得 $x = \frac{n+4}{2}$ 。因此当 n 为奇数时方案数为 0。接下来就是考虑怎样的序列才会合法。先给出结论：

结论. 一个顶点序列是合法的，当且仅当序列中不存在两个相邻的 0。首尾亦视作相邻。

证明： 证明从两方面入手，一是当序列中存在两个相邻的 0 时，序列一定不合法；二时候当序列中不存在两个相邻的 0 时，序列一定合法。

先证明第一点。如果序列中存在两个相邻的 0，那么多边形中必然会存在下面的情况（有色的部分是多边形内部）：



那么从多边形中的任意一个点观察，下方的两个蓝点中至少有一个看不见。

接下来证明第二点。过多边形的每个顶点作一条垂直于 x 轴的直线，将多边形划分成一些竖着的长方形。用数组 a 表示这些长方形从左到右的上边界的高度，数组 b 表示其下边界的高度。如果序列中不存在两个相邻的 0，那么 a 数组只有下面三种情况：

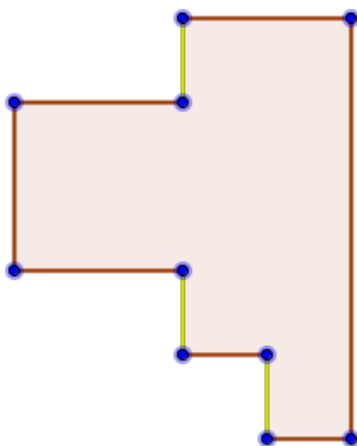
1. a 数组中的元素单调不增；
2. a 数组中的元素单调不减；
3. a 数组中的元素先单调不减再单调不增。

类似地， b 数组也只有下面三种情况：

1. b 数组中的元素单调不增；
2. b 数组中的元素单调不减；
3. b 数组中的元素先单调不增再单调不减。

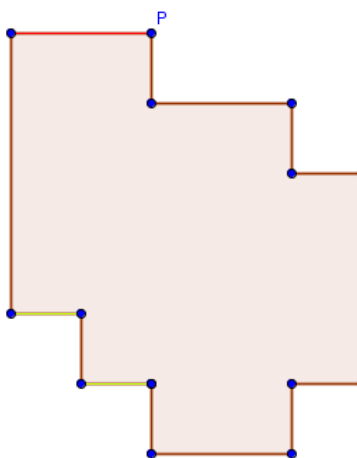
剩下的就是分类讨论了。注意多边形的各边边长可以任意调整。

1. 如果 a 为情况 1，且 b 为情况 2；或者 a 为情况 2，且 b 为情况 1。如下图所示：



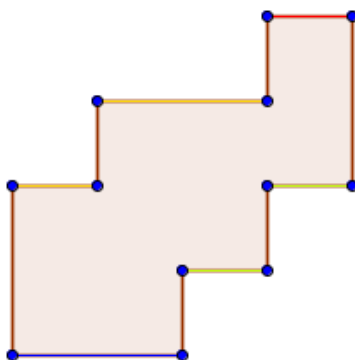
我们可以将绿色的线段长度缩成无限小，然后就可以在左右边中较长的边上看到整个多边形内部。

2. 如果 a 和 b 中只有一种为情况3。如下图所示：



我们可以将绿色的线段长度缩小，使得其总长度不超过红色线段的长度。在过点P的竖线上就可以看到整个多边形内部，其左侧和右侧化为了情况1。

3. 如果 a 和 b 都是情况3。我们仍然可以通过调整水平边的长度，使得多边形可以分为两个情况1的多边形。
4. 如果 a 和 b 同为情况1或2。如下图所示：



我们延长蓝色与红色的线段，缩短橙色和绿色的线段，使得绿色线段长度之和不超过红色线段的长度，且橙色线段长度之和不超过蓝色线段的长度，而且红色和蓝色线段在 x 轴上的投影相交。这样在相交处的竖线上就可以看到整个多边形内部。

证毕。

□

剩下的计算就很简单了。由组合数学的知识，可以得出答案为 $\binom{x}{4} + \binom{x-1}{4}$ 。也可以用动态规划求解。

6.4.3 Complexity

时间复杂度： $O(1)$ （动态规划做法则为 $O(n^2)$ ）。

空间复杂度： $O(1)$ （动态规划做法则为 $O(n^2)$ ）。

难度：2.5/5。结论并不难猜，难度在于证明。

6.5 J - The Sky is the Limit

6.5.1 Description

给定 n 个在平面直角坐标系中，底边在 x 轴上，最高点在 x 轴上方的等腰三角形，求这些三角形形成的图形的上轮廓线的长度。

$n \leq 100$ 。

6.5.2 Solution

我们只需要知道上轮廓线上的所有端点就可以知道上轮廓线的长度。可以枚举两个三角形，求出其所有交点，并判断每个交点是不是严格在某个三角形内部

(除底边外的边界不算)。如果一个交点不在三角形内部,那么必然在轮廓线上。求出所有在轮廓线上的交点之后,按横坐标排序,答案就是所有相邻点的距离和。

6.5.3 Complexity

时间复杂度: $O(n^3)$ 。

空间复杂度: $O(n^2)$ 。

难度: 1/5。

6.6 K - Steam Roller

6.6.1 Description

有一个 $n \times m$ 的网格图,每条边或者有正的通行时间,或者无法通行。现在要驾驶拖拉机从给定起点到终点。如果一条经过的边的一个端点是起点或终点,或者要在一个端点处转弯,则这条边的通行时间会变为原来的两倍。求一条拖拉机的最短路,或指出无法到达。

$n, m \leq 100$ 。

6.6.2 Solution

一个比较直接的想法是,对于每个点连 $O(n + m)$ 条边,代表直接行驶到同行和同列的每个点的代价,这个图上起点到终点的最短路就是答案。但是这个图的边数达到了 $O(n^3)$,会超时。

我们可以换个方法建模。我们想象在网格图上面有层“高速路”,通行的代价是原代价,而下面这层的代价是原代价的两倍。称一个在下层的点为 $l(x)$,在上层的点为 $u(x)$ 。对于两个相邻的点 x 和 y ,设代价为 w ,连边如下:

1. $l(x) \rightarrow l(y)$, 边权为 $2w$ 。
2. $u(x) \rightarrow l(y)$, 边权为 $2w$ 。
3. $l(x) \rightarrow u(y)$, 边权为 $2w$ 。
4. $u(x) \rightarrow u(y)$, 边权为 w 。

上述的边都是有向边。需要注意的是,上下左右四个方向需要各建一层点。

这个图的点数为 $5nm$,边数是 $O(nm)$ 级别的,可以承受最短路的复杂度。答案就是 $l(\text{起点})$ 到 $l(\text{终点})$ 的最短路。

6.6.3 Complexity

时间复杂度: $O(\text{ShortestPath}(5nm, 8nm))$ 。

空间复杂度: $O(nm)$ 。

难度: 2/5。

Chapter 7

ACM/ICPC World Finals 2007

7.1 A - Consanguine Calculations

7.1.1 Description

给出一家人的血型，其中有一人的血型未知。求出血型未知的人的所有可能血型，或判断不可能。

7.1.2 Solution

只有当父母中有一人血型未知时才会出现不可能的情况。具体来说，只有当孩子为O型血，父母一方为AB型血；或者孩子为AB型血，父母一方为O型血时才不可能。对于剩下的情况，分类讨论已知血型两人的血型即可。RH血型系统可以和ABO血型系统分开讨论。

7.1.3 Complexity

时间复杂度： $O(1)$ 。

空间复杂度： $O(1)$ 。

难度： 0/5。

7.2 C - Grand Prix

7.2.1 Description

有一条在斜坡上的赛道。设山坡和水平面的夹角为 θ ，以山坡的平面和水平面的相交线为 y 轴，高度增加的方向为 x 轴正方向建立坐标系。赛道的起点在

原点，接着依次有 n 个拐点，第1条路的起点和终点分别为原点和第1个拐点，第 i ($i > 1$)条路的起点和终点分别为第 $i - 1$ 和第 i 个拐点。

判断是否存在一条下坡的路，即是否存在一条路的终点的高度比起点的高度低。如果存在，是否可以通过绕原点旋转整条赛道从而使得不存在下坡的路。如果可以，输出最小旋转角度。

$n \leq 10000$ 。

7.2.2 Solution

先特判掉 $\theta = 0$ 的情况。用一个向量从路的起点指向终点，我们可以得到 n 个向量，问题转化成，能否通过旋转使得向量在 x 轴的分量均非负。

我们可以求出所有向量的极角并排序。如果所有极角都在 $[-\frac{\pi}{2}, \frac{\pi}{2}]$ 内，那么不需要旋转。否则一定是把某个向量旋转到恰好 $-\frac{\pi}{2}$ 或者 $\frac{\pi}{2}$ 的角度上。那么我们可以枚举这个向量，然后将其他的向量旋转，并判断是否可行。在所有可行方案中选一个最小角度即可。

7.2.3 Complexity

时间复杂度： $O(n^2)$ 。

空间复杂度： $O(n)$ 。

难度：0.5/5。

7.3 E - Collecting Luggage

7.3.1 Description

传送带是一个 n 个点的简单多边形，初始时，行李在传送带的第一个点上，按逆时针方向匀速在传送带上移动。乘客在传送带外的一个点 S 处，要去拿行李。乘客以匀速移动，不能穿过传送带。设行李的速度为 v_l ，乘客的速度为 v_p ，总有 $v_l < v_p$ 。求乘客拿到行李的最短时间。

$n \leq 100$, $0 < v_l < v_p \leq 10000$ 且 v_l, v_p 为整数，所有坐标均为绝对值不超过10000的整数。

7.3.2 Solution

题目中一个关键的条件是 $v_l < v_p$ ，这意味着答案满足可二分的性质。那么我

们二分答案，并求出此时行李所在的位置 T ，问题就变成了，求一条从 S 到 T 的最短路径，使得路径不穿过一个简单多边形。

如果 S 到 T 的直线段不经过多边形内部，那么 $\text{dist}(S, T)$ 就是答案。否则，最优路径一定是从 S 走到多边形的某个顶点，再经过若干个顶点，最后到达 T 。我们可以先求出哪些顶点之间可以互达，并用Floyd算法预处理出顶点间的最短路。每次二分判断时，枚举在多边形上经过的第一个和最后一个顶点，判断是否可行并更新答案即可。

剩下的就是计算几何的问题了。我们需要判断一条线段 ab 是否穿过一个简单多边形。这里有几种情况：

1. 与多边形在非 a 、 b 的位置上相交。我们认为 ab 穿过了多边形。
2. 与多边形在 a 或 b 的位置上相交。 ab 可能完全在多边形内部，也可能完全在多边形外部，因此我们可以取 ab 的中点 p ，并判断 p 是否在多边形内部。如果在则 ab 穿过了多边形。判断点是否在多边形内部用射线法。
3. 与多边形不相交。由于 a 和 b 中至少有一个在多边形上，这种情况不存在。

7.3.3 Complexity

令 W 为最大答案。

时间复杂度： $O(n^3 + n^2 \log W)$

空间复杂度： $O(n^2)$ 。

难度：1.5/5。

7.4 G - Network

7.4.1 Description

有 n 条信息需要传输，第 i 条信息的字节数为 s_i 。这些信息被分割为 m 个包，第 i 个包包含了第 a_i 条信息的第 l_i 到 r_i 字节。包按照顺序被接受，可以直接将其输出或者存入缓冲区。缓冲区中的包可以随时输出。在输出一个包 i 时，第 a_i 条信息的前 l_i 字节必须已经被输出，而且直到第 a_i 条信息的所有字节都被输出为止，都不能输出其它信息的字节。求缓冲区的最小大小。

$$n \leq 5, m \leq 1000, \sum s_i \leq 64000。$$

7.4.2 Solution

首先枚举信息输出的顺序，接下来只要按照题目的要求模拟即可。在接受一个包时，如果它可以被输出就直接输出，不然就存入缓冲区。在输出一个包之后，判断一下缓冲区中是否有可以输出的包。所有顺序得到的缓冲区大小的最小值就是答案。

在实现时，我使用了`set`来维护缓冲区中的包，当然也可以直接用数组记录下每个信息有哪些字节在缓冲区中。

7.4.3 Complexity

令 $W = \sum s_i$ 。

时间复杂度： $O(n!m \log m)$ （数组记录的方法则为 $O(n!(m + W))$ ）。

空间复杂度： $O(n + m)$ （数组记录的方法则为 $O(n + m + W)$ ）。

难度：0/5。

7.5 I - Water Tanks

7.5.1 Description

有 n 个横截面积为1立方米的圆柱形水箱，从左到右的高度分别为 $v_1 \sim v_n$ 。相邻的两个水箱之间有尺寸可以忽略的横管相连，连接水箱 i 和水箱 $i + 1$ 的横管高度为 p_i ，而且 $p_i > p_{i-1}$ 。水箱1是打开的，水可以从顶部流入，空气也可以进出，其他水箱都是封闭的，水和空气只能从横管中进出。

水箱中原本都是空气。现在不断向水箱1中注水，直到无法再继续注水为止。求此刻所有水箱中的水的体积和。

$n \leq 10$, $v_i \leq 10^6$ 。

题目中会用到下面的物理原理：

- 与外界联通的区域中，气压等于一个大气压。
- 空气可以被压缩，水不能被压缩。
- 封闭空间中的气压处处相同，而且其体积与气压成反比。
- 在一个连通的水体中，水的压强在每个高度上都相同。
- 在一个空气层下的水体，在水面以下 d 米处的水压等于在表面的气压再加上 $0.097 \cdot d$ 个大气压。

7.5.2 Solution

以下的解答思路来自2008年集训队员方戈的论文《浅析信息学竞赛中一类与物理有关的问题》。

令 h_i 表示水箱 i 的水深。

先只考虑前两个水箱。当 $h_1 = p_1$ 时，水开始注入水箱2中。当 $h_2 = p_1$ 时，横管被没过，水箱2以及整个右边部分的空气都与外界隔绝了，里面的气体总体积不会再改变。

而对于水箱 i ，当水箱 i 中的水没过了两侧的横管时，水箱 i 上部的空气也与之后的水箱隔绝，故可以单独考虑了。而由于 $p_i > p_{i-1}$ ，故在水箱 i 中空气与之后的水箱隔绝之前，之前的水箱的空气已经与水箱 i 隔绝了。也就是说，可以从左往右依次考虑。

我们可以给注水的过程划分阶段。一个阶段包括三个部分，水箱 i 的水位由 p_{i-1} 上升到 p_i 、水箱 $i+1$ 的水位从0上升到 p_i ，以及水箱 i 的水位继续上升直到无法再上升为止。如果一个阶段无法被完成，之后的水箱中也无法再继续注入进水。

我们直接考虑最后的情况。在最后， h_1 必然等于 v_1 ，对于和水箱1中水体相连的水箱 i 中的水体，设其表面气压为 P ，其表面水压为 $P(1 + 0.097(v_1 - h_i))$ 个大气压。

初始化 $h_1 = v_1$ 、 $h_2 = p_1$ 。之后从水箱2开始依次考虑每个水箱。假设一个阶段的一个部分中，水箱 i 中的水位上升 x 米后无法继续上升，那么应该满足：

$$1 + 0.097(v_1 - h_i - x) = V \times P / (V - x)$$

其中 V 和 P 为上升前水箱上方空气层的体积和压力。变形之后可以得到一个关于 x 的二次方程，得到的最小非负解就是上升的水位。如果解不在该阶段的该部分上升水位范围内，说明这之后还可以继续注水；否则说明注水结束。

注水结束之后， $\sum h_i$ 的值就是答案。

7.5.3 Complexity

时间复杂度： $O(n)$ 。

空间复杂度： $O(n)$ 。

难度：3/5。根据物理知识的掌握程度，难度会有所不同。

7.6 J - Tunnels

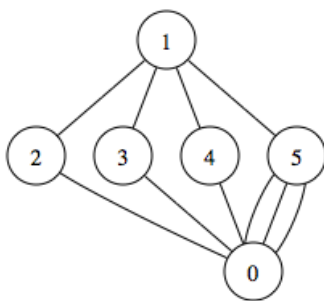
7.6.1 Description

给定一个 $n + 1$ 个点 m 条边的图，点的编号为 $0 \sim n$ 。间谍从点1出发，要逃到点0。间谍通过一条边的时间可以视为0。你可以监视间谍的所在位置，并在任何时刻摧毁任意的边。求在最坏情况下，至少要摧毁多少条边，才能使间谍无法到达点0。

$$n \leq 50, m \leq 1000.$$

7.6.2 Solution

令 $MinCut(u, v)$ 表示点 u 和点 v 之间的最小割容量。第一眼看上去，答案似乎是 $MinCut(1, 0)$ ，或者是 $\min\{MinCut(i, 0)\}$ 之类的东西。但实际上不是的。考虑下面这张图：



$MinCut(1, 0) = 4$, $\min\{MinCut(i, 0)\} = 2$ ，但是这里需要摧毁3条边。当间谍还在点1时，先摧毁点1和点5之间的边。等间谍移动到点2、3、4中的任意一个时，摧毁和所在点相连的两条边。一共摧毁3条边。可以证明这是最优方案。

之所以答案不是单纯的最小割，是因为我们可以一边观察间谍的移动一边行动。也就是说，最优方案应该是把间谍逼到某些特定的点中，然后再切断此时最小割中的边。我们可以通过逆向思维得到下面的基于最小割的算法：

算法. 令 $x_i = MinCut(i, 0)$ 。找出所有 x_i 中的最小值 $minX$ ，并删掉所有 $x_i = minX$ 的点 i 。对于尚未被删掉的点 j ，更新 x_j 为 $\min(x_j, MinCut(j, 0) + minX)$ 。如此重复直到点1被删掉。答案就是 x_1 。

我们可以这样证明这个算法的正确性。如果在某个时刻点 i 被删去了，就说明在剩下的点中，间谍处于点 i 时需要摧毁的边是最少的之一。那么如果间谍走到了一个这次还未被删去的点 j ，有两种情况。如果间谍直接走到了这回被删去的点，那么可以用 $minX$ 的代价困住他；否则，为了阻止他通过未被删去的点到达点0，

我们再摧毁此时的最小割。换句话说，在任意时刻，间谍如果到了点 i ，都可以用不超过那个时刻的 x_i 的代价阻止间谍到达点0。

而算法的最优性可以用上面提到的最优方案证明。令 S 为某次被删去的点的集合。如果某个 x_i 这次被更新了，就说明间谍走到点 i 时的最优方案变成了把间谍逼到集合 S 中的点处，然后再用 S 中点的最优方案处理。这里没有办法给出严谨的证明，但是通过感性的理解可以得出上面提到的最优方案的确是最优的，而算法所有的就是这样的最优方案，故算法得出的解是最优的。

7.6.3 Complexity

时间复杂度：上界为 $O(n^2(n+m)MaxFlow(n,m))$ 。

空间复杂度： $O(n+m)$ 。

难度：4/5。

Chapter 8

ACM/ICPC World Finals 2006

8.1 A - Low Cost Air Travel

8.1.1 Description

有 n 条航线，第 i 条航线沿途经过 l_i 个城市（包括起点和终点），花费为 w_i 。要搭乘第 i 条航线，必须从起点出发，按顺序经过若干城市，并在航线沿途的任意一个城市结束行程。在行程结束前经过的每个城市，可以选择停靠或者不停靠。

现在有 m 条旅游路线，第 i 条路线长度为 t_i ，路线上第 j 个城市为 $a_{i,j}$ 。对于一条旅游路线，要求从路线上的第一个城市出发，按顺序访问路线上的每个城市，最后到达终点。在访问一个城市之前可以经过其他城市。求每条旅游路线最小的花费，并输出方案。保证有解且解唯一。

$$n, m \leq 20, 2 \leq l_i, t_i \leq 10.$$

8.1.2 Solution

我们在访问下一个城市时有两种策略：

1. 从当前城市出发，通过某条航线访问接下来的若干个城市，并结束行程。
2. 通过在多条航线之间转机，来访问下一个城市。

那么我们令 $f[i][j]$ 表示访问了路线上前 i 个城市，且当前在城市 j 的最小花费。考虑第一种策略，枚举一条从 j 出发的航线 x ，对于航线上的城市 k ，用 $f[i][j] + w_x$ 来更新 $f[i + l][k]$ 的值，其中 l 为在访问城市 k 之前最多能顺路访问的城市数。而对于第二种策略，其实就相当于从 $f[i][j]$ 转移到 $f[i][k]$ 。由于转移有环，所以直接用SPFA处理。最后答案就是 $f[t_i][a_{i,t_i}]$ 。转移时记录方案即可。

8.1.3 Complexity

令 p 为城市数, $a = \sum l_i$, $b = \sum t_i$, $c = \max\{t_i\}$ 。

时间复杂度: $O(b(a + SPFA(p, p^2)))$

空间复杂度: $O(a + p^2 + cp)$

难度: 2/5。

8.2 B - Remember the A La Mode!

8.2.1 Description

有 n 种饼和 m 种冰淇淋, 每种分别有 a_i 和 b_i 个, 并且有 $\sum a_i = \sum b_i$ 。每块饼应当搭配恰好一份冰淇淋。已知不同搭配的收益 (或是某两种饼和冰淇淋之间不能搭配), 求在全部搭配完的前提下, 可以获得的最大和最小的收益。

$n, m \leq 50$ 。

8.2.2 Solution

我们可以建出一个费用流的模型。对每种饼和冰淇淋设点, 连边如下:

1. 从源向饼的点连边, 容量为饼的个数, 费用为0。
2. 从冰淇淋的点向汇连边, 容量为冰淇淋的个数, 费用为0。
3. 在可以搭配的饼和冰淇淋之间连边, 容量为无穷大, 费用为搭配的收益。

这个模型的最小费用最大流和最大费用最大流分别是答案。求的时候有两个细节需要注意, 一个是收益可以为0, 另一个是, 在SPFA时转移时应判断精度误差, 不然可能会出现转移出环的情况。

8.2.3 Complexity

时间复杂度: $O(CostFlow(n + m, nm))$ 。

空间复杂度: $O(nm)$ 。

难度: 1/5。

8.3 D - Bipartite Numbers

8.3.1 Description

定义一个二段数为，恰好包含两种不同的十进制数 s 和 t ， $s \neq 0$ ，且 s 均出现在 t 前面的正整数。可以用一个四元组 (a, b, c, d) 描述一个二段数，即 a 个 b 后面接 c 个 d 。

给定一个正整数 x ，求一个最小的二段数 p ，使得 p 是 x 的倍数而且 $p > x$ 。

$1 \leq x \leq 99999$ 。最多200组测试数据。

8.3.2 Solution

一个直观的感觉是答案的二段数长度（即 $a + c$ ）不会特别大。如果当真如此的话，我们可以枚举 a, b, c, d 来确定一个二段数。判断时可以预处理 n 个1和 10^n 分别对 x 取模的值，记为 $f[n]$ 和 $g[n]$ ，那么 $p \bmod x$ 的值就是 $b \times f[a] \times g[c] + d \times f[c]$ 。事实上长度的确不会特别大。在数据范围内最长的二段数是 $x = 99670$ 时的答案，长度为9967。但是这个长度对于直接枚举还是无法承受。

我们可以进一步发现，并不是所有二段数都这么长。实际上，在数据范围内，不是10的倍数也不是25的倍数的 x ，答案的二段数长度不会超过997。而当 x 为10的倍数时， d 必然为0，而且 c 不会超过11。当 x 为25的倍数时， c 不会超过6。因此只需要分情况讨论一下，然后直接枚举即可。因为要求答案最小，枚举时要注意枚举的顺序，或者枚举长度之后枚举对应长度的所有数字，再取最小值。需要注意的是有些 x 本身就是二段数，枚举的时候需要判断得到的二段数是不是比 x 大。

8.3.3 Complexity

令 $W = 997$ 。

时间复杂度：单组数据最坏情况下约为 $O(100W^2)$ 。

空间复杂度： $O(W)$ 。

难度：1.5/5。

8.4 G - Pilgrimage

8.4.1 Description

一个队伍里初始有若干人和若干元钱的公款。之后有如下4种共 n 次操作:

1. COLLECT k , 向队伍中的每个人收取 k 元加入公款。
2. IN k , 队伍中加入了 k 人, 每个人需缴纳(公款/加入前人数)的钱。
3. OUT k , k 人从队伍离开了, 每个人得到(公款/离开前人数)的钱。
4. PAY k , 从公款中支付 k 元。

此外, 还有两个要求:

1. 在IN和OUT操作发生时, 公款必须恰好是人数的倍数。
2. 任意时刻队伍中的人数都不少于1。

求初始时队伍的人数的所有方案。如果方案有无限种, 输出其最小值。

$n \leq 50$, $k \leq 2000$ 。

8.4.2 Solution

我们分别考虑4种操作对两个要求的影响:

1. COLLECT操作

显然COLLECT对第二个要求没有影响, 而由于向每个人收的钱都相同, 所以对第一个要求也没有影响。因此我们可以直接忽略所有的COLLECT操作。

2. IN和OUT操作

这两个操作本质相同。易知如果IN和OUT操作之前公款是人数的倍数, 操作之后仍然是人数的倍数, 故对第一个要求没有影响。而对于第二个要求, 需要记录下每个时刻较初始状态的人数之差, 取最小值, 设为 m 。答案不能小于 $1 - m$ 。

3. PAY操作

首先, 在所有IN和OUT操作之前和之后的PAY操作都是无关紧要的。因为可以自由确定初始时的公款, 而之后由于没有IN和OUT操作, 对公款和人数的关系也没有限制了。

那么我们只需要考虑中间的操作。我们可以把相邻的PAY操作都合并成一个, 这和原来是等价的。可以得出如下结论: 当前人数必须是PAY操作的钱数的因子。而且, 由于因子个数是有限的, 只要存在PAY操作, 方案的数量

就有限。

由上述分析我们得到了以下的算法：忽略所有COLLECT操作以及所有IN和OUT操作之前和之后的PAY操作，并将相邻的PAY操作合成一个。如果此时没有PAY操作，则方案有无限种，最小值为 $1 - m$ 。否则，处理每个PAY操作，并求出满足所有PAY操作的数，即为方案。注意要除去所有小于 $1 - m$ 的方案。

8.4.3 Complexity

时间复杂度： $O(n\sqrt{k})$ 。

空间复杂度： $O(n + \sqrt{k})$ 。

难度：2/5。

8.5 I - Degrees of Separation

8.5.1 Description

定义无向图的分离度为所有点对之间最短路的最大值。给定一个 n 个点 m 条边的无向图，判断其是否连通，若连通，求其分离度。

$n \leq 50$ 。

8.5.2 Solution

用Floyd算法处理出任意两点间的最短路即可。注意判断不连通的情况。

8.5.3 Complexity

时间复杂度： $O(n^3)$ 。

空间复杂度： $O(n^2)$ 。

难度：0/5。

8.6 J - Routing

8.6.1 Description

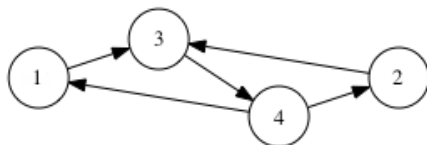
给定一个 n 个点 m 条边的有向图，求出一条从1号点到2号点再回到1号点的路径，使得路径上不同的点的个数最少。

$n \leq 100$, $m \leq 1000$ 。

8.6.2 Solution

有一个比较直接的想法：令 $f[i]$ 表示从1到*i*再回到1的路径上最少的不同点个数，转移时枚举另一个点，用两点间的最短路长度更新。答案就是 $f[2]$ 。

这个算法乍一看是对的，但是在下面这个图上会得出错误的答案：



用这个算法会算出来5，但图中只有4个点。原因在于，上面算法的正确性基于这个结论：最终的路径一定是，从1出发按顺序经过若干重复点到2，再按逆序经过这些重复点回到1。换言之，我们可以这样分解路径：

$$1 \rightarrow a \rightarrow A \rightarrow b \rightarrow 2$$

$$1 \leftarrow d \leftarrow B \leftarrow c \leftarrow 2$$

其中 a, b, c, d 为一些子路径， A 为一个重复点。结论等价于， a 和 d 、 b 和 c 中可以有相同的点，但是 a 和 c 、 b 和 d 中不能有相同的点。这个结论显然是错的，反例就是上面的图。

我们来考虑反例。反例即存在这样的路径：

$$1 \rightarrow a_1 \rightarrow B \rightarrow a_2 \rightarrow A \rightarrow b \rightarrow 2$$

$$1 \leftarrow d \leftarrow A \leftarrow c_2 \leftarrow B \leftarrow c_1 \leftarrow 2$$

其中 a_1, a_2, c_1, c_2 为子路径， B 为重复点。可以发现 a_2 与 c_2 都是从 B 到 A 的路径，最有情况下应该有 a_2 和 c_2 均为 B 到 A 的最短路。而如果还存在重复点，可以继续类似分解。

我们考虑这样的DP。令 $f[x][y]$ 表示一条从1到*x*的路径和一条从*y*到1的路径的最少不同点个数。转移是有下面几种情况：

1. 从*x*往下走一步到*i*，*i*不是重复点，即 $i \neq y$ 。有

$$f[i][y] = \min(f[i][y], f[x][y] + 1)$$

2. 从*x*往下走一步到*i*，*i*是重复点，即 $i = y$ 。有

$$f[y][y] = \min(f[y][y], f[x][y])$$

3. 从 y 往回走一步，与上面两种类似。
4. x 和 y 都是重复点，类似上面 B 和 A 的情况。有

$$f[y][x] = \min(f[y][x], f[x][y] + d[x][y] - 1)$$

其中 $d[x][y]$ 为 x 到 y 的最短路。

可以证明这样覆盖了所有的状况。转移时类似最短路，可以用堆优化。答案即为 $f[2][2]$ 。

8.6.3 Complexity

时间复杂度: $O(n^2 \log n)$ 。

空间复杂度: $O(n^2)$ 。

难度: 2/5。

Chapter 9

ACM/ICPC World Finals 2005

9.1 B - Simplified GSM Network

9.1.1 Description

在平面上有 p 座信号塔，一个点被距离它最近的信号塔管辖，一个信号塔的管辖范围也就是其管辖的所有点构成的多边形。经过两个不同管辖范围的边缘时需要一次切换。平面上还有 n 个城市和 m 条路，一条路是连接两个城市的直线段。有 q 个询问，每个询问给定起点与终点，求一条起点到终点的路径，使得切换次数最少。

$p, n \leq 50$, $m \leq 2500$ （注意清橙上的数据范围和原题不同）， $q \leq 10$ 。

9.1.2 Solution

先求出所有信号塔的管辖范围。只考虑两个信号塔，它们的管辖范围分别是它们中垂线两侧的半平面。那么对于一个信号塔而言，它的管辖范围就是它与其他所有信号塔的中垂线各自构成的半平面的交。所求的实际上就是平面Voronoi图。

求出每条边与几个边缘相交，作为这条边的边权，并建图。对于每个询问，直接在图上从起点开始求最短路即可。

9.1.3 Complexity

时间复杂度： $O(qnm + mp^2)$ 。

空间复杂度： $O(n + m + p^2)$ 。

难度：1.5/5。

9.2 C - The Traveling Judges Problem

9.2.1 Description

给定一个 n 个点 m 条边的无向图，边带权。有 k 个人在其中一些点上。这 k 个人可以沿着图中的边行走，所有人都要到达点 T 。行走的代价为这 k 个人经过的所有边的边权和，如果有多个人经过了同一条边，这条边的权只被计算一次。求出一组行走路径的方案，使得行走代价最小。如果有多组符合条件的方案，输出经过总点数最少的。如果仍有多组方案，输出所有经过点按编号升序排序后，字典序最小的。

$n \leq 20$, $k \leq 10$ 。保证图连通。

9.2.2 Solution

这是一个斯坦纳树模型的直接应用。

首先不难得出，所有人的行走路径之并一定是一棵树，树根为点 T 。我们称有人的点为特殊点。我们考虑动态规划，令 $f[u][i]$ 表示当前子树的根为 u ，子树中包含的特殊点的集合为 i 。有两种转移：

$$f[u][i] = \min_{(u,v) \in E} \{f[v][i] + w[u][v]\}$$

其中 $w[u][v]$ 为边权。另一种转移是：

$$f[u][i] = \min_{j \subset i} \{f[u][j] + f[u][i - j]\}$$

答案即为 $f[T][S]$ ， S 为所有特殊点的集合。为了构造方案，可以额外记录每个状态是从哪个状态转移过来的。在实现时，可以用SPFA来实现第一种转移，第二种转移则在第一种转移之间完成。

而至于经过点数最少以及字典序最小的要求，可以通过改变权值实现。给所有边权都乘上 10^9 ，额外给每个点赋点权，点 i 的点权为 $10^7 + 2^{i-1}$ 。这样求出的解就满足点数最少且字典序最小。在第二种转移时注意减去重复计算的点权即可。

9.2.3 Complexity

时间复杂度： $O(3^k n^2)$ 。

空间复杂度： $O(2^k n)$ 。

难度：2/5。

9.3 E - Lots of Sunshine

9.3.1 Description

n 栋楼房从东往西直线排列。一间公寓高 h 米，东西向宽 w 米。第 i 栋楼房有 m_i 层，每层都是一间公寓。第 i 栋和第 $i+1$ 栋楼之间的距离为 d_i 。给定日出和日落的时间，有 q 次询问，每次询问某一间公寓的太阳直射时间的起止时刻。一间公寓被直射当且仅当其东侧或西侧墙壁完全被直射，或者太阳在其正上方。

$n < 100$, $m_i < 100$, $w, h, d_i \leq 100$, $q \leq 1000$ 。

9.3.2 Solution

不难得出，起始时刻就是公寓所在楼栋东侧的楼栋最高的阴影恰好到公寓东墙的底部的时刻。枚举东侧的所有楼栋，计算出阴影恰好照射在东墙底部时太阳的直射角（不考虑被其他楼栋遮挡）。起始时刻即为最大直射角对应的时刻。终止时刻可以类似求出。

由于需要比较的角度都在90度之内，可以用向量表示，用叉积判断大小，最后再转换成角度。这样程序常数更小，而且判断时不存在精度问题。

9.3.3 Complexity

时间复杂度： $O(nq)$ 。

空间复杂度： $O(n)$ 。

难度：1/5。

9.4 G - Tiling the Plane

9.4.1 Description

给定一个各边都平行于坐标轴的简单多边形，求这个多边形能否无缝平铺一个无限的二维平面。

令多边形的边数为 n ，周长为 L 。 $n, L \leq 50$ 。最多50组测试数据。

题目中会用到下面的性质：

只有两种本质不同的铺满平面的情况：使用正四边形铺满平面（棋盘覆盖），或使用正六边形铺满平面（蜂巢覆盖）。一个多边形当且仅当满足以下两个条件中至少一个时可以铺满平面：

1. 在凸多边形边界上顺次存在四个点A、B、C、D（不一定是多边形的顶点），使得A到B的边界与D到C的边界重合，B到C的边界与A到D的边界重合。这表明这个多边形可以用棋盘覆盖的方式铺满平面。
2. 在凸多边形边界上顺次存在六个点A、B、C、D、E、F（不一定是多边形的顶点），使得A到B的边界与E到D的边界重合，B到C的边界与F到E的边界重合，C到D的边界与A到F的边界重合。这表明这个多边形可以用蜂巢覆盖的方式铺满平面。

9.4.2 Solution

我们用一个四进制串来描述多边形，串中的一位代表多边形某条单位长度的边的方向。那么如果两段边界能重合，它们对应的两个四进制串也应该相同。如果两段边界的方向不同，应将其中一个串取反。

显然性质中边界上的若干个点是整点。对于棋盘覆盖的情况，枚举A和B的位置，C和D的位置就确定了。而对于蜂巢覆盖的情况，枚举A、B和C的位置，D、E和F的位置也确定了。如果得到的若干段边界是否能对应重合，这个多边形就能平铺平面。只需要对四进制串哈希判断即可。实现上可以用断环为链的方法处理。

9.4.3 Complexity

时间复杂度：单组数据 $O(L^3)$ 。

空间复杂度： $O(L)$ 。

难度：1/5。

9.5 H - The Great Wall Game

9.5.1 Description

一个 $n \times n$ 的棋盘上有 n 枚棋子，一个格子上最多有一枚棋子。给定这些棋子的初始摆放，目标摆放中所有棋子必须在同一行、同一列或者同一斜线上。一次操作可以将一个棋子移动到上下左右相邻的格子中，求移至目标状态的最少操作次数。

$n \leq 15$ 。最多500组数据。

9.5.2 Solution

容易得出，只要得出了一个初始状态下棋子与目标状态下棋子的对应关系，就一定存在一种移动方法，只要花费每对棋子曼哈顿距离之和的代价。换种说法就是，一个格子上最多有一枚棋子的条件是可以忽略的。证明略去。那么一个比较显然的做法就是枚举目标状态是哪一种，然后建一个最小权匹配模型，用费用流求解。这个做法虽然复杂度略高，但是可以通过。

另一个做法是，只对于目标状态为斜线的用费用流求解。对于同一行或者同一列的，可以利用贪心求解。以行为例，将横纵坐标分开，纵坐标上的代价就是到这行的距离和，而横坐标上代价的最优方案一定是把所有横坐标排序后，从左到右依次匹配。这个贪心的正确性显然，复杂度相比上一个算法也低了不少。

9.5.3 Complexity

时间复杂度：单组数据 $O(n^2 \log n + CostFlow(2n, n^2))$ 。

空间复杂度： $O(n^2)$ 。

难度：1/5。

9.6 I - Workshops

9.6.1 Description

有 n 场讲座和 m 个房间，每场讲座有参加人数与持续时间，每个房间有能容纳的人数上限和空闲时间。一场讲座能在一个房间中举行，当且仅当参加讲座的人数不超过房间能容纳的人数上限，而且讲座的持续时间不超过房间的空闲时间。一场讲座只能在一个房间举行，一个房间也只能举行一场讲座。求最多能举办多少场讲座，以及不能举办的讲座的参加人数之和最少是多少。

$n, m \leq 1000$ 。

9.6.2 Solution

由于讲座和房间有一一对应关系，不难想到一个费用流的模型。对所有讲座和房间设点，连边如下：

- 从源向每场讲座连边，容量为1，费用为参加人数；
- 从每场讲座向能举办这场讲座的房间连边，容量为1，费用为0；
- 从每个房间向汇连边，容量为1，费用为0。

这个网络的最大费用最大流的流量就是最多能举办的讲座场数，总参与人数减去最大费用就是不能举办的讲座的参加人数的最小值。求解时，最多增广 $O(n)$ 次，单次增广的复杂度约为 $O(|V| + |E|)$ 。但是模型中 $|V|$ 可以达到 $O(nm)$ 级别，费用流的算法是会超时的。

考虑连续最短路算法。在这个网络中，每次会选择可增广的参加人数最多（即费用最大）的一场讲座进行增广。我们可以想到一个贪心算法：按参加人数从大到小考虑每场讲座，选择所有能举办这场讲座的房间中空闲时间最短的房间举办这场讲座。如果没法举办则不举办这场讲座。

这个贪心算法的正确性可以分两部分证明。按参加人数从大到小考虑能选就选的正确性可以用连续最短路算法的正确性证明；而选择举办讲座的房间部分的正确性在于，之后考虑的讲座参加人数肯定不大于当前的这场，故房间的人数限制不用考虑，而选择空闲时间最短的房间对后面的影响一定比选择空闲时间较大的房间更小。

9.6.3 Complexity

时间复杂度： $O(nm)$ 。

空间复杂度： $O(n + m)$ 。

难度：1.5/5。

9.7 J - Zones

9.7.1 Description

有 n 个信号塔，第 i 个信号塔覆盖了 a_i 个用户。一些信号塔的覆盖范围有交集，一共 m 个不相交的部分被多个信号塔覆盖，第 i 个部分有 b_i 个用户，被集合 S_i 中的信号塔覆盖。现从中选出 k 个信号塔，求这 k 个信号塔覆盖范围的并集中最多有多少用户。

$n \leq 20, m \leq 10$ 。

9.7.2 Solution

枚举选择哪些信号塔，其覆盖范围并集内的用户数即为，每个信号塔能覆盖到的用户之和减去重复计算的部分。如果一块区域被选出的信号塔中的 p 个

($p > 0$) 覆盖, 就被重复计算了 $p - 1$ 次, 减去即可。答案即为所有选择方法中覆盖的最大值。

9.7.3 Complexity

时间复杂度: $O(2^n nm)$ 。

空间复杂度: $O(n + m)$ 。

难度: 0.5/5。

Chapter 10

ACM/ICPC World Finals 2004

10.1 D - Insecure in Prague

10.1.1 Description

对于一个长度为 n 的字符串 p ，有下面的一种加密方法。选择一个密文 c 长度 m ，满足 $m \geq 2n$ 。同时选择四个整数 s, t, x, y 满足 $0 \leq s, t, x, y < n$ 且 $x < y$ 。 p 的第一个字符放在 c 的第 s 个位置。如果 s 不是空位则跳到 s 之后的第一个空位。 p 的第 k ($k \geq 2$)个字符放在第 $k-1$ 个字符的位置之后的第 $x+1$ 个空位内。如果跳到了末尾，则跳回开头。之后用 t 替换 s 、 y 替换 x 重新放置一遍。对于剩下的空位，往里面填入随机的字符。

给定一个长度为 n 的密文串，求一个最长的原文，使得原文加密后可以得到这个密文串。如果有多组解则输出对应的信息。保证有解。

$n \leq 40$ 。

10.1.2 Solution

先考虑一个比较暴力的方法。枚举原文的长度和 s, t, x, y 的值。判断一次的复杂度可以达到 $O(n^3)$ ，整个算法是 $O(n^8)$ 的。

枚举部分似乎没法优化，我们从判断入手。最坏情况下，跳一次需要 $O(n)$ 的时间，而一次定位的跳跃次数也可以达到 $O(n)$ 。但是这里有很多重复运算。我们可以预处理 $pos[i][j][k]$ 代表长度为 i 的字符串，每次跳 j 步时得到的第 k 个字符的位置。和主过程的复杂度相比，预处理的复杂度可以忽略不计。那么从 s 开始跳得到的第 k 个字符的位置就是 $(pos[n][x][k] + s) \bmod n$ ，第一遍放置就可以在 $O(n)$ 的时间内完成。而第二遍可以将密文串中第一遍已经选择的字符删去，得到一个新的

字符串，并把这个串当做原密文串放置，这样复杂度也是 $O(n)$ 的。

整个算法的复杂度为 $O(n^6)$ ，但是常数很小，可以通过此题。

10.1.3 Complexity

时间复杂度： $O(n^6)$ 。

空间复杂度： $O(n^3)$ 。

难度：2/5。

10.2 E - Intersecting Dates

10.2.1 Description

集合 A 中有 n 个日期区间，集合 B 中有 m 个日期区间，求集合 $B - A$ 中所有的日期区间。

$n, m \leq 100$ ，日期范围在1700/1/1到2100/12/31之间。

10.2.2 Solution

先把所有给定的区间改为左闭右开区间，并拆成左端点和右端点排序。由于在限定的日期范围内只有146452天，故可以枚举每一天，然后判断这天是否在 B 集合中且不在 A 集合中。最后把相邻的日期合并成一个区间输出即可。

10.2.3 Complexity

令 $W = 146562$ 。

时间复杂度： $O((n + m) \log n + W)$ 。

空间复杂度： $O(n + m + W)$ 。

难度：0.5/5。

10.3 F - Merging Maps

10.3.1 Description

有 n 张地图，每张地图可以用一个二维字符数组表示，每一位是一个特征。定义两个地图的得分为以某种offset重叠后对应位置相同的特征数，如果得分大

于0，则可以合并。在所有地图中，每次选择得分最高的两张地图合并，不断重复直到无法合并。输出得到的若干张地图。

$n \leq 10$ ，地图的长和宽不超过10。

10.3.2 Solution

令 m 表示地图尺寸。一个直接的想法是每次合并时枚举两张地图以及offset，之后暴力计算得分。这样一次合并的复杂度可以达到 $O(n^2m^4)$ 。由于合并最多有 n 次，总复杂度可以达到 $O(n^3m^4)$ 。虽然复杂度看上去比较吓人，但是由于常数较小，实际上是可以通过的。

不过算法还有优化的余地。我们没必要每次合并都计算一次，只需要最开始计算 n 张地图间两两合并的得分，以及在每次合并出一张新地图时计算和所有已有地图之间的得分即可。这样复杂度降到了 $O(n^2m^4)$ 。

有一个特别坑爹的地方：输入文件的每行后面可能有多余的字符，读入时应该忽略。

10.3.3 Complexity

令 m 表示地图尺寸。

时间复杂度： $O(n^2m^4)$ 。

空间复杂度： $O(nm^2)$ 。

难度：0.5/5。

10.4 H - Tree-Lined Streets

10.4.1 Description

平面上有 n 条线段，不存在两条线段在端点处相交，也不存在三条线段相交于同一点。现在要在线段上种树，一条线段上的任意两棵树之间的距离不得小于50个单位长度，一棵树到线段上的任意交点的距离不得小于25个单位长度。求树上最多能种多少棵树。

$n \leq 100$ 。

10.4.2 Solution

枚举每条线段，求出其他所有线段与这条线段的交点，并把交点左右的25个

单位长度的范围都标为不可行。那么一段可行线段上种数的最优方案必然是从一个端点开始每隔50个单位长度种一棵树。枚举相邻的两个交点，以及最左与最右的交点与线段端点构成的段并计算即可。

10.4.3 Complexity

时间复杂度: $O(n^2)$ 。

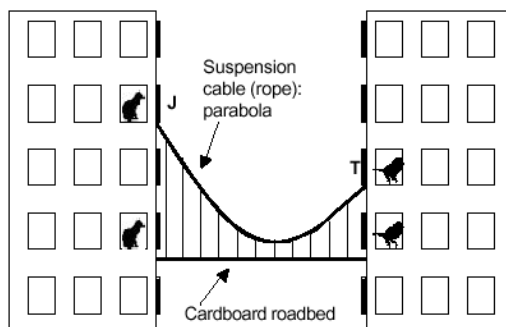
空间复杂度: $O(n)$ 。

难度: 0/5。

10.5 I - Suspense!

10.5.1 Description

有两栋大楼，之间相隔 d 米。大楼的每层楼高3米，窗户高1.5米且比该层地面高1米，层与层之间的高度忽略不计。甲和乙分别住在两栋楼的 n 层和 m 层。现在他们准备在两栋楼之间架一座吊桥，以两条相同长度的绳索为缆绳。由物理原理可得悬挂的绳索形成了抛物线，如下图所示：

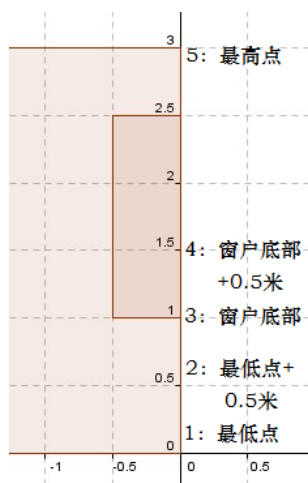


桥面恰比缆绳的最低点低1米，而且桥面要比地面高至少1米，比甲和乙的窗户底部低至少2米。有些人养了猫或者鸟。猫无法向上跳0.5米，也无法向下跳3米。吊桥要保证猫无法抓到鸟。求是否可以造吊桥，以及绳子的长度的最大值。

$2 \leq n, m \leq 25$, $1 \leq d \leq 25$ 。最多100000组数据。

10.5.2 Solution

桥面的高度越低，绳子的长度就越长。我们考虑第 i 层楼，并按照下图划分成若干区间：



我们用 $[l, r]$ 一类符号来描述区间，其中 l 和 r 对应图中的点。下面分类讨论：

- 如果桥面高度在 $[1, 3]$ ，第 i 层的猫可以跳到桥面上。
- 如果桥面高度在 $(3, 4)$ ，第 i 和 $i + 1$ 层的猫可以跳到桥面上。
- 如果桥面高度在 $[4, 5]$ ，第 $i + 1$ 层的猫可以跳到桥面上。
- 如果桥面高度在 $[1, 2]$ ，桥面上的猫可以抓到第 $i - 1$ 层的鸟。
- 如果桥面高度在 $(2, 3)$ ，桥面上的猫可以抓到第 i 和 $i - 1$ 层的鸟。
- 如果桥面高度在 $[3, 5]$ ，桥面上的猫可以抓到第 i 层的鸟。

可以发现，如果区间 $(j, j + 1)$ 合法，那么点 j 也会合法。因此只有图中的点才可能成为最优点。事实上，只有点 $2 \sim 4$ 可能成为最优点。这样我们就能得出抛物线的最低点。

设抛物线为 $y = a(x - b)^2 + c$ ，易得 c 为最低点高度。令 l 和 r 为甲和乙的窗户底部高度，有 $l = 3n - 2$ ， $r = 3m - 2$ 。设抛物线过点 $(0, l)$ 以及 (d, r) ，可得

$$\begin{aligned} l &= ab^2 + c \\ r &= a(d - b)^2 + c \end{aligned}$$

如果 $l = r$ ，那么 $b = \frac{d}{2}$ 。否则令两式相除，可得：

$$\frac{l - c}{r - c} = \frac{b^2}{(d - b)^2}$$

化简可以得到一个一元二次方程，其在 $[0, d]$ 内的最小非负解就是 b 的值。求出 b 之后，进而可以求出 a ，从而求出抛物线的解析式。

由于抛物线是轴对称图形，我们可以沿最低点分成两部分分别求长度。把抛物线平移，使得最低点与原点重合，则抛物线解析式为 $y = ax^2$ 。可以推导出如下

关于长度 L 的积分公式:

$$\begin{aligned} L = \int_{t=0}^d (x')^2 t + (y')^2 t &= \int_{t=0}^d \sqrt{1 + 4a^2 t^2} \\ &= \frac{x}{2} \sqrt{4a^2 x^2 + 1} + \frac{1}{4a} \ln(2ax + \sqrt{4a^2 x^2 + 1}) \end{aligned}$$

答案就是两部分的长度和。

10.5.3 Complexity

时间复杂度: 单组数据 $O(n + m)$ 。

空间复杂度: $O(n + m)$ 。

难度: 2/5。

Chapter 11

ACM/ICPC World Finals 2003

11.1 A - Building Bridges

11.1.1 Description

一块地图被划分成 $n \times m$ 个格子，每个格子是空地或者建筑物。如果两座建筑物八连通，那么它们之间可以通行。现在要在格子的边上建一些桥，每座桥连接两座建筑物，必须水平或者竖直，而且不能穿过其他建筑物。求连接所有建筑物的一个方案，如果无法连接所有建筑物，则使得剩下的连通块个数最少。在此基础上，使得建的桥的数量最少，同时桥的总长最小。

$n, m \leq 50$ 。有较多组测试数据。

11.1.2 Solution

先求出每个建筑物属于哪个连通块，再枚举每个格点，找到其向右和向下可以建的桥。最后求一遍最小生成森林即可。

在实现上稍微有些地方需要注意。找桥时，遇到一座建筑物就应当停下，这样才能保证得到的边数是 $O(nm)$ 级别的。同时，由于连通块数可以达到 $O(nm)$ 级别，不能用邻接矩阵处理。

11.1.3 Complexity

由于 n 和 m 同阶，故用 n 代替 m 。

时间复杂度： $O(n^2 \log n)$ （使用Prim算法求最小生成森林可以达到 $O(n^2)$ ）。

空间复杂度： $O(n^2)$ 。

难度：0/5。

11.2 B - Light Bulbs

11.2.1 Description

有一排 n 个灯泡和 n 个开关，第1个开关控制1、2号灯泡，第 n 个开关控制 $n - 1$ 、 n 号灯泡，第 $1 < i < n$ 个开关控制 $i - 1$ 、 i 和 $i + 1$ 号灯泡。改变开关的状态会改变这个开关控制的所有灯泡的状态。给定初始状态和目标状态，求最少需要改变几个开关的状态使得可以从初始状态转换成目标状态。输入输出均为以十进制描述的01串，其长度为 len 。

$len \leq 100$ 。

11.2.2 Solution

一个开关的状态只会改变至多一次。可以发现，只要确定了第一个开关是否改变，其他开关的状态可以直接推出来。只需要枚举第一个开关的状态并顺推其他开关的状态，最后判断最后一盏灯能否到目标状态即可。

11.2.3 Complexity

时间复杂度： $O(len^2 + n)$ 。

空间复杂度： $O(n + len)$ 。

难度：0.5/5。

11.3 D - Eurodiffusion

11.3.1 Description

在一个 10×10 的地图上有 n 个国家，每个国家都是一个整点矩形，矩形内的每个整点都是一个城市。任意两个国家不相交。每个国家都有自己的货币，初始时每个城市有 10^6 个自己国家的货币。在每一天，设一个城市有 x 个某种货币，则它会给与它四连通的每个城市 $\lfloor \frac{x}{1000} \rfloor$ 个这种货币。求经过最少多少天，每个城市都有 n 种货币。

11.3.2 Solution

当一个城市得到了一种货币之后，这个城市的这种货币的数量就不会再变为0。经过实测，答案不会超过大概40000，所以按照题目要求模拟即可。一个常

数优化是，枚举时跳过所有不是城市的整点。

11.3.3 Complexity

令 $m = 10$ ，即地图大小。令 W 为答案的最大值。

时间复杂度： $O(Wnm^2)$ 。

空间复杂度： $O(nm^2)$ 。

难度：0/5。

11.4 F - Combining Images

11.4.1 Description

给定两个相同大小图像的四分树编码，求其交的四分树编码。

对于一个 $n \times n$ ，且 n 为2的整数幂的黑白图像，定义其四分树编码如下：

- 如果整个图像相同，那么其四分树编码以1开始，然后是像素的颜色。
- 否则，其四分树编码以0开始，然后依次是其左上、右上、左下、右下象限的四分树编码。

关于十六进制的四分树编码还有其它的定义。

十六进制编码长度不超过100。

11.4.2 Solution

递归处理四分树编码。对于某一层，假设两个编码分别为 a 和 b ，有三种情况：

1. a 和 b 都以1开始。如果颜色都为1则交的颜色为1，否则为0。
2. a 以1开始， b 以0开始（或相反）。如果 a 的颜色为0，则直接为0；否则直接为 b 。
3. a 和 b 都以0开始。递归处理。

这样可以得出交的四分树编码，但是这个编码不一定是简的。比如编码010111110和011101011的交应为10，而直接求会得到010101010，因此还要对最后得到的编码化简。

11.4.3 Complexity

令二进制编码长度为 l 。

时间复杂度: $O(l)$ (用C++的`string`实现则可能达到 $O(l^2)$)。

空间复杂度: $O(l)$ 。

难度: 1/5。

11.5 G - A Linking Loader

11.5.1 Description

题意较为复杂, 这里略去。

11.5.2 Solution

本题的难点在于看懂题。看懂之后, 直接模拟即可。如果不理解题意, 请结合原题看下面这段:

- 第一个模块的起始地址为0100, 之后每个模块的起始地址为上一个模块最后一个地址的下一个地址。
- 每个模块中的地址只有由c操作写入的地址。
- D和E操作只定义和引用符号, 并不会写内存。

11.5.3 Complexity

令 n 为内存地址最大值。

时间复杂度: $O(n)$ 。

空间复杂度: $O(n)$ 。

难度: 0/5。

11.6 H - A Spy in the Metro

11.6.1 Description

有一条双向通行的地铁, 一共经过 n 个地铁站。间谍从1号站出发, 要恰好在时刻 T 到达 n 号站。已知相邻两站之间地铁的运行时间, 以及分别从首站和末站出发的 m_1 和 m_2 趟地铁的出发时间, 假设上下车以及换乘不需要时间, 求在站台上的最短等待时间。

$$n, m_1, m_2 \leq 50, T \leq 200。$$

11.6.2 Solution

先预处理出每个时刻在每个站点是否有上行和下行的地铁。令 $f[i][j]$ 代表第 j 个时刻在第 i 个站点时的最少等候时间，直接BFS转移即可。

11.6.3 Complexity

时间复杂度： $O((m_1 + m_2)nT)$ 。

空间复杂度： $O(nT)$ 。

难度：0.5/5。

11.7 I - The Solar System

11.7.1 Description

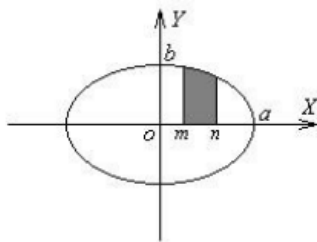
给定太阳系中两颗行星的描述，包括第一颗行星轨道的半长轴、半短轴长度和运行周期，以及第二颗行星轨道的半长轴、半短轴长度，假设太阳在第二颗行星轨道在 x 轴非负的焦点处，初始时第二颗行星在轨道上 x 轴坐标最大处，并以逆时针运行，求其在 t 个单位时间后的坐标。

所有数据皆为整数且在`int`范围内。输出精确到三位小数。

11.7.2 Solution

由开普勒第三定律可以求出第二颗行星的运行周期，设为 p ，并令 t 对 p 取模。分 $t \leq \frac{1}{2}p$ 和 $t > \frac{1}{2}p$ 两种情况讨论。下面只考虑小于等于的情况，大于等于的情况类似。

由开普勒第二定律可知，运行时间与行星与太阳的连线扫过的面积成正比。我们可以把扫过的面积分为椭圆一部分的面积加上一个三角形的有符号面积。椭圆任意一部分的面积可以用如下的公式求：



$$S = \int_m^n \frac{b}{a} \sqrt{a^2 - x^2} dx = \frac{b}{a} \left(\frac{x}{2} \sqrt{a^2 - x^2} + \frac{a^2}{2} \arcsin \left(\frac{x}{a} \right) \right) \Big|_m^n$$

二分横坐标之后用上述公式计算判断即可。

11.7.3 Complexity

令 W 为坐标的取值范围。

时间复杂度: $O(\log W)$ 。

空间复杂度: $O(1)$ 。

难度: 1.5/5。

11.8 J - Toll

11.8.1 Description

给定一个 n 个点的无向图，需要从起点向终点运送货物。到达一个点时，需要缴纳费用。有两种点，一种的费用为1，另一种的费用和携带的物品数量有关，如果携带了 x 件，则费用为 $\lceil \frac{x}{20} \rceil$ 。求运送 p 件货物到终点，至少需要携带多少件货物从起点出发。

$n \leq 52, 0 \leq p \leq 1000$ 。

11.8.2 Solution

二分答案，考虑如何检验。令 $d[i]$ 表示到达点 i 时最多能携带的货物，可以类似SPFA地转移。

11.8.3 Complexity

时间复杂度: $O(SPFA(n, m) \log p)$ 。

空间复杂度: $O(n^2)$ 。

难度: 0.5/5。

Chapter 12

ACM/ICPC World Finals 2002

12.1 A - Balloons in a Box

12.1.1 Description

在三维空间中有一个箱子，给出其两个相对的顶点坐标。空间中还有 n 个点，可以在点上放置气球。气球会一直膨胀，直到碰到箱子边缘或者一个已经膨胀的气球。如果点在一个已膨胀的气球内部则不能放置气球。你可以以任意顺序放置气球，求箱子中未被气球占据的最小体积。

$n \leq 6$ 。

12.1.2 Solution

由于数据范围非常小，可以直接枚举放置的顺序。对于一个顺序，枚举之前的所有气球，计算距离即可。

12.1.3 Complexity

时间复杂度： $O(n!n^2)$ 。

空间复杂度： $O(n)$ 。

难度：0/5。

12.2 C - Crossing the Desert

12.2.1 Description

平面上有 n 个点，1号点为起点， n 号点为终点，两点之间的距离为欧氏距

离。你有一个容量为 w 的背包，可以装食物和水。起点可以买到无限的食物，任意点都可以获得无限的水并存储无限的食物。1单位的食物需要花费1的代价，水则免费。行走 x 单位长度的路程需要消耗 x 个单位的食物和水。求从起点到终点至少要花费多少代价，或说明无解。如果答案大于 10^6 也算作无解。

$n \leq 20$ 。

12.2.2 Solution

首先二分答案，设答案为 x 。令 $f[i][j]$ 表示，经过了 i 个点，到了点 j 时，点 j 最多有多少单位的食物。那么 $f[0][1] = x$ 。考虑如何转移。

假设点 a 和 b 之间的距离为 d ，点 a 处的食物量为 r 。有下面三种情况：

1. $w < 2d$ ，两点之间无法通行。
2. $2d \leq w < 3d$ ，只能从 a 点到 b 点，无法返回。可搬运的食物量为 $\min(w - 2d, r - d)$ 。
3. $w \geq 3d$ ，可以从 a 到 b 再回到 a ，也即可以反复搬运食物。单次可搬运的食物量为 $w - 3d$ 。如果到某个时刻 $r \leq 3d$ ，则转为情况2。

分类讨论进行转移。如果可以到达终点，则答案可行。否则不可行。

12.2.3 Complexity

令 $W = 10^6$ ，即答案上界。

时间复杂度： $O(n^3 \log W)$ 。

空间复杂度： $O(n^2)$ 。

难度：2/5。

12.3 E - Island Hopping

12.3.1 Description

给定 n 个岛屿，第 i 个岛屿上有 m_i 个人。要在岛屿之间连 $n - 1$ 条边，边权为两点之间的欧氏距离，使得任意两个岛屿之间可互达，并最小化边权和。所有边的建设同时开始，完成时间即为边权。定义一个人的连通时间为他所在岛屿与1号岛屿连通的最早时间，求所有人的连通时间的平均值。

$n \leq 50$ 。

12.3.2 Solution

处理出所有边，按边权排序，用Kruskal算法求MST。在Kruskal的过程中，要加入一条边时，处理出有哪些岛屿在加入前与1不连通，加入后变为连通，这些岛屿上人的连通时间即为这条边的边权。

12.3.3 Complexity

时间复杂度： $O(n^3)$ 。

空间复杂度： $O(n^2)$ 。

难度：0.5/5。

12.4 H - Silly Sort

12.4.1 Description

给定一个长度为 n 的序列 $a[]$ ，任意两个元素不同。每次可以交换任意两个元素，代价为两个元素之和。求最少要花费多少代价才能将序列变为升序。

$n \leq 1000$ 。

12.4.2 Solution

先求出每个元素在排序后序列中的位置，记这个序列为 $b[]$ 。可以发现， $b[]$ 中每个循环是独立的，即只需要在循环内交换即可使得序列变为升序。

我们单独考虑一个循环，设循环中有 cnt 个元素，其和为 sum ，最小值为 k 。我们只需要 $cnt - 1$ 次交换，而每次都可以将最小值与其他元素交换，即代价为 $sum + (cnt - 2)k$ 。

但是这不一定是最优解。我们还可以从循环外选一个元素代替循环内的最小值完成所有的交换。显然应该选全局最小值，记其为 t ，那么这种情况下的代价为 $(cnt + 1)t + sum + k$ 。答案即为两种情况的较小值之和。

12.4.3 Complexity

时间复杂度： $O(n \log n)$ 。

空间复杂度： $O(n)$ 。

难度：2/5。

Chapter 13

ACM/ICPC World Finals 2001

13.1 A - Airport Configuration

13.1.1 Description

机场有两排门，到达门与出发门，各 n 扇。每个到达门和出发门都对应一个城市。两个门之间的客流指数为流量乘上距离，距离为两个门编号之差的绝对值+1。给定一个到达门和出发门对应城市的方案，求客流指数之和。

$n \leq 25$ 。

13.1.2 Solution

按题目中的要求计算即可。

13.1.3 Complexity

时间复杂度： $O(n^2)$ 。

空间复杂度： $O(n^2)$ 。

难度：0/5。

13.2 B - Say Cheese

13.2.1 Description

给定三维空间中的两个点 S 和 T 的坐标，和 n 个球的半径及球心坐标。一只虫子从 S 出发到 T ，每前进1个单位长度需要10秒，但是在球内部穿行不花费时间。求从 S 到 T 的最短耗时。注意球与球之间不一定相离。

$n \leq 100$ 。

13.2.2 Solution

球到球的最短距离是在其球心之间所连直线上行走，点到球的最短距离也是在点与球心之间所连直线上行走。因此求出 n 个球之间的最短距离，以及 n 个球与两个点之间的最短距离，求最短路即可。注意如果两个球不相离，即其球心之间的距离小于两个球半径之和，则其最短距离为0。

13.2.3 Complexity

时间复杂度： $O(n^2)$ 。

空间复杂度： $O(n^2)$ 。

难度：0.5/5。

13.3 E - The Geoduck GUI

13.3.1 Description

给定 k 个向量，定义一个向量的移动序列如下：将向量起点置于网格图中一个网格的中心，记录下向量沿途穿过网格边缘的方向，如果穿过的格点则先横向移动再纵向移动，记录下的方向即为这个向量的移动序列。现需选出一对向量，使得两个点在一个 $n \times m$ 的网格中按照两个向量的移动序列移动，能覆盖的格子最多，在此基础上使得所需时间最短。两个点同时移动，如果某次移动两个点重合，或者移动到了已访问过的点，则停止。输出所有满足上述条件的向量对。

$n, m \leq 50, k \leq 10$ 。

13.3.2 Solution

如果求出了移动序列，只需枚举选择的两个向量，再模拟即可。而移动序列可以用各种近似乱搞的方法求出来，这里就不多说了。注意精度误差。

13.3.3 Complexity

时间复杂度： $O(nmk^2)$ 。

空间复杂度： $O(nmk)$ 。

难度：0.5/5。

13.4 F - A Major Problem

13.4.1 Description

给定一串音符、这串音符所属的大调音阶，和目标大调音阶，输出转换之后的音符。

13.4.2 Solution

一共有15个大调音阶，全部打表存下来即可。维基百科上有全部的大调音阶。

打表时有一点需要注意，如果是下面这种写法：

```
string t[] = {"C", "D", "E", "F", "G", "A", "B"};
```

有可能会因为漏打逗号而使相邻两项合成一个字符串。一种比较好的方法如下：

```
string A = "A";  
string Ab = "Ab";  
.....  
string t[7];  
t[0] = A, t[1] = B, ...
```

可以另外通过一些其他方法来减少代码量。

13.4.3 Complexity

令 n 为音符数量。

时间复杂度： $O(n)$ 。

空间复杂度： $O(1)$ 。

难度：0/5。

13.5 G - Fixed Partition Memory Management

13.5.1 Description

有 n 块内存和 m 个程序，每块内存有容量大小，每个程序有最低内存要求。给定每个程序运行时间与内存的关系，求一个程序运行的方案（程序在哪块内存、何时开始执行），使得程序结束运行的时间的平均值最小。保证有解。

$n \leq 10, m \leq 50$ 。

13.5.2 Solution

建一个费用流模型，对每个程序设点，同时令 (i, j) 表示第 i 块内存倒数第 j 个运行的程序的一个空位，并对这 nm 个空位设点。连边如下：

1. 对于每个程序，从源向其连一条容量为1，费用为0的边。
2. 对于每个空位，从其向汇连一条容量为1，费用为0的边。
3. 对于每个程序和每个空位 (i, j) ，从程序对应的点向空位对应的点连一条容量为1，费用为 $w \cdot j$ 的边，其中 w 为该程序在内存 i 上运行所需时间。

这个模型的最小费用最大流的费用就是最小的结束运行时间和。方案只需要看每个程序对应的点的流量流向了哪个空位即可。

13.5.3 Complexity

时间复杂度： $O(nm^2 + CostFlow(nm, nm^2))$ 。

空间复杂度： $O(nm^2)$ 。

难度：2/5。

13.6 H - Professor Monotonic's Network

13.6.1 Description

给定一个有 n 个输入端和 k 个比较器的比较网络，判断是否是排序网络，并求出网络的最小运行时间。

$n \leq 12, k \leq 150$ 。

13.6.2 Solution

结论. 如果任意01序列都可以通过比较网络排序，这个网络就是排序网络。

证明: 我们可以通过证明这个命题的逆否命题来证明它。假设一个网络不是排序网络，那么一定存在一个序列 a 不能被排序。设 a 通过网络得到的序列为 b ，则 b 中存在至少一个逆序对，设这个逆序对为 (x, y) 。我们构造一个01序列，如果 a 中第 i 位不小于 x ，那么01序列的第 i 位为1，否则为0。可知，这个01序列同样无法被排序，故逆否命题成立，故原命题成立。□

这个结论又叫做01原则。至于最小运行时间，只要按照比较器的拓扑序扫一遍即可求出。

13.6.3 Complexity

时间复杂度: $O(2^n m)$

空间复杂度: $O(n + m)$

难度: 1/5。

13.7 I - A Vexing Problem

13.7.1 Description

给定一个 $n \times m$ 的地图，从上往下标为第 $0 \sim n - 1$ 行，从左到右标为第 $0 \sim m - 1$ 列。第 0 列、第 $m - 1$ 列和第 $n - 1$ 行都是墙壁，其他的各个格子可以是墙壁、标有字母的石块，或者空地。每次操作可以选择一个石块移动到其左侧或者右侧的空地上。如果某个石块下面是空地，它就会掉落下去。待所有石块都静止后，如果有两个或者以上字母相同的石块连在了一起，它们就会被消掉。如此重复直到不再有石块掉落。求最少需要几步消除所有石块，并给出方案。

$n, m \leq 9$ 。保证有解，而且步数不多于11步。时限10s。

13.7.2 Solution

直接从初始状态开始广搜，并对局面哈希进行判重。有两个比较关键的剪枝：

1. 如果某种字母的石块只剩一块，局面一定无解，故剪枝。
2. 如果某种字母的石块只剩两块，记两块石头所在列分别为 a 和 b ，那么消掉这两块石头的代价一定不小于 $\max(0, |a - b| - 1)$ 。如果当前步数加上所有代价大于11，则剪枝。

虽说无法保证这样搜索访问到的状态数足够少，但是由于数据弱，所以可以通过。下面给出一个无法在时限内得出解的数据：

```
9 9 KILLER
#-----#
#-----#
#F-A----#
#E-B-E-A#
#D-C-D-B#
```

```
#C-D-C-C#  
#B-E-B-D#  
#A-F-A-E#  
#####
```

另外有一个需要注意的地方是，关卡名字中可能含有空格，而且开头可能含有空格，开头的空格不记入名字中。

13.7.3 Complexity

时间复杂度：无法给出准确复杂度。

空间复杂度：无法给出准确复杂度。

难度：无法给出准确难度。

Chapter 14

ACM/ICPC World Finals 2000

14.1 A - Abbott's Revenge

14.1.1 Description

给定一个最大 9×9 的四连通网格图，以及从每个方向到达每个点时可以采取的行动（向左、向前、向右），求从指定起点以指定方向出发，到指定终点的最短距离，并输出方案。

14.1.2 Solution

将每个点拆成4个，分别代表从不同方向到达这个点。之后建图直接BFS求最短路即可。

14.1.3 Complexity

令 n 为网格图中的点数。

时间复杂度： $O(n)$ 。

空间复杂度： $O(n)$ 。

难度：0.5/5。

14.2 B - According to Bartjens

14.2.1 Description

给定一个长度为 n 的数字，要求在其中插入至少一个运算符，使得计算出来的值为2000。运算符只包括+、-和 \times ，不能使用括号，被运算符隔开的数字不得

含有前导零，并且 $-$ 只能作为双目运算符。输出所有方案。

$$n \leq 9。$$

14.2.2 Solution

由于两个数字之间最多有一个符号，可以直接枚举是否插入，以及插入什么符号。之后模拟计算一下，看答案是否为2000即可。

14.2.3 Complexity

时间复杂度： $O(n4^n)$ 。

空间复杂度： $O(n)$ 。

难度：0/5。

14.3 C - Cutting Chains

14.3.1 Description

有 n 个铁环，其中一些铁环被套在了一起。现在要把所有铁环连成一条链，求至少要打开再合上几个铁环。

$$n \leq 15。$$

14.3.2 Solution

我们可以把问题转成图论模型。把每个铁环视为一个点，套在一起的铁环之间连边，我们需要通过最少的操作次数使得图成为一条链。需要弄清楚的一点是一次操作（即打开再合上一个铁环）可以干什么。事实上，一次操作可以任意改变一个点的连边状态，即可以断开任意条与其相连的边，并与任意个其它的点连边。

由于数据范围很小，我们不妨枚举操作哪些点。考虑怎么判断合法。从图中删去所有选择的点，图应该满足下面的两个条件：

1. 剩下的每个连通块都是一条链。
2. 假设选择了 k 个点，那么连通块的个数不能大于 $k + 1$ 。

而判断一个连通块是不是链等价于判断是否无环且每个点的度数不超过2。

14.3.3 Complexity

时间复杂度: $O(n^2 2^n)$ 。

空间复杂度: $O(n^2)$ 。

难度: 1.5/5。

14.4 E - Internet Bandwidth

14.4.1 Description

给定一张含有 n 个点 m 条边无向网络, 求从给定源到给定汇的最大流。

$n \leq 100$, $m \leq \frac{n(n-1)}{2}$ 。

14.4.2 Solution

无向边只需要把反边的容量赋为和正边一样即可。直接用最大流算法解决。

14.4.3 Complexity

时间复杂度: $O(\text{MaxFlow}(n, m))$ 。

空间复杂度: $O(n + m)$ 。

难度: 0/5。

14.5 F - Page Hopping

14.5.1 Description

给定一张含有 n 个点的有向强连通图, 边权均为1, 求任意两点之间的最短路的平均值。

$n \leq 100$ 。

14.5.2 Solution

由于边权均为1, 从每个点开始BFS即可。

14.5.3 Complexity

时间复杂度: $O(n^3)$ 。

空间复杂度: $O(n^2)$ 。

难度: 0/5。

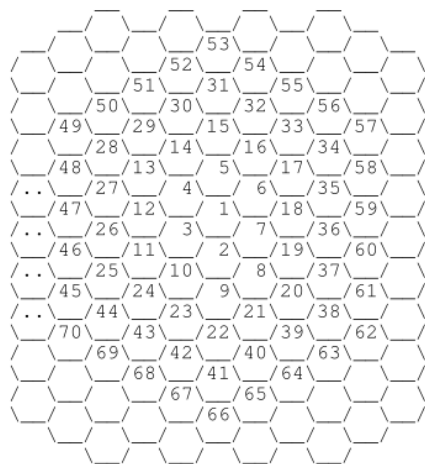
Chapter 15

ACM/ICPC World Finals 1999

15.1 A - Bee Breeding

15.1.1 Description

用正六边形镶嵌一块无限的区域，任选一个正六边形编号为1号，然后将其它的正六边形按照下图方式顺时针编号：



从一个正六边形出发，到达与其共边的正六边形需要一步。给定两个正六边形的编号 a 与 b ，问它们的距离是多少。

$a, b \leq 10000$ 。多组测试数据。

15.1.2 Solution

此处正六边形的数量不多，我们完全可以建出整个图来。通过观察，不难得出建图的方法。对于一组询问，直接从一个点开始BFS即可。

15.1.3 Complexity

记 $n = 10000$ ，即 a 和 b 的最大取值。

时间复杂度：预处理 $O(n)$ ，单组数据 $O(\max(a, b))$ 。

空间复杂度： $O(n)$ 。

难度：0.5/5。

15.2 C - A Dicey Problem

15.2.1 Description

给定一个 $n \times m$ 的平面，平面上每一格要么是骰子的一面，要么是一颗五角星。一个骰子被放在某一个格子上。骰子可以往四个方向滚动，但不能滚出边界。骰子能够放在一个格子上，当格子的图案是五角星，或者格子的图案和骰子底面的图案相同。求一条路径，使得骰子从起点出发，最后又回到起点。骰子不能往回滚。保证存在至多一条合法的路径。

$n, m \leq 10$ 。

15.2.2 Solution

骰子一共有24种摆放方式。我们用一个三元组来表示状态：横纵坐标，以及骰子的摆放方式。由于保证最多存在一条合法路径，可以直接从初始状态开始BFS，如果回到了起点则说明有解。而至于骰子的滚动，可以用比较暴力的方式模拟，也可以直接打出一个转移表。

15.2.3 Complexity

时间复杂度： $O(24nm)$ 。

空间复杂度： $O(24nm)$ 。

难度：1/5。

15.3 D - The Fortified Forest

15.3.1 Description

平面上有 n 棵树，给定这些树的坐标、价值以及高度，现在要砍倒一些树做围栏，要能围住剩下没被砍倒的树。在这个前提下，最小化树的价值。输出要砍

哪些树。

$$n \leq 15。$$

15.3.2 Solution

先枚举砍哪些树，然后对剩下的树求一个凸包。如果砍倒的树的高度和不小于凸包的周长，那么可行，否则不可行。注意特判只剩一个点和两个点的情况。

15.3.3 Complexity

时间复杂度： $O(2^n n \log n)$ 。

空间复杂度： $O(n)$ 。

难度： 0.5/5。

15.4 E - Trade on Verwegistan

15.4.1 Description

有 n 堆物品，第 i 堆有 p_i 件物品，从上往下的价值分别为 $a_{i,1} \sim a_{i,p_i}$ 。如果要购买某堆中的某件物品，必须购买堆中在其上方的所有物品。求最大收益，以及达到最大收益可以购买的物品件数的所有方案。

$$n \leq 50, p_i \leq 20。$$

15.4.2 Solution

直接令第 i 堆的第 j 件物品的收益 $v[i][j] = \sum_{k=1}^{j-1} a_{i,k}$ 。令 $f[i][j]$ 表示前 i 堆买 j 件物品的最大收益，有

$$f[i][j] = \max_{0 \leq k \leq \min(p_i, j)} \{f[i-1][j-k] + v[i][k]\}$$

令 $sum = \sum p_i$ ，答案即为

$$\max_{0 \leq i \leq sum} \{f[n][i]\}$$

方案则为所有达到答案的 $f[n][i]$ 的 i 。

15.4.3 Complexity

令 $m = \max \{p_i\} \leq 20$ 。

时间复杂度: $O(nm \cdot \text{sum})$ 。

空间复杂度: $O(n \cdot \text{sum})$ 。

难度: 0.5/5。

15.5 G - The Letter Carrier's Rounds

15.5.1 Description

要求模拟邮件发送过程中邮件传送代理之间的通信。用户数量、消息条数都不超过 n 。

$n \leq 100000$ 。

15.5.2 Solution

按要求模拟即可。C++中可使用map、set和vector等容器来简化代码。

15.5.3 Complexity

时间复杂度: $O(n \log n)$ 。

空间复杂度: $O(n)$ 。

难度: 0.5/5。

15.6 H - Flooded!

15.6.1 Description

一块区域被划分成 $n \times m$ 块，给定每块的高度以及区域内的总积水体积。地势较高的区域的积水总是能流到地势较低的区域，并且不会流出整块区域。求积水的海拔高度。

$n, m \leq 30$ ，所有数字目测不超过int。

15.6.2 Solution

由于题目奇葩的流水方式，直接对所有高度排个序后，枚举积水高度在哪个区间内，计算即可。

比较奇怪的是，直接二分答案会得0分，而枚举时倒着枚举也只有85分。具体原因未知。

15.6.3 Complexity

时间复杂度： $O(nm)$ 。

空间复杂度： $O(nm)$ 。

难度： 0/5（如果你在无数次WA之后仍旧能保持平常心）。

Chapter 16

ACM/ICPC World Finals 1998

16.1 A - Crystal Clear

16.1.1 Description

在二维平面上，以每个整点为圆心，0.5个单位长度为半径作圆。给定一个 n 个点的简单多边形，称一个圆合法当多边形的边不过这个圆，或者与圆相交的所有边都过圆心。求所有合法圆在多边形内部的面积和。

$n \leq 25$ ，坐标的绝对值不超过250。

16.1.2 Solution

由于坐标范围比较小，我们可以枚举范围内的所有整点。一个圆对答案有贡献，前提是圆心在多边形内部。具体来说有三种情况：

1. 圆心在多边形顶点上。圆合法，对答案的贡献是被两条边所夹的扇形面积。
由于多边形不一定凸，扇形的角度可能大于 π 。
2. 圆心在多边形边上。圆合法，对答案的贡献是 $\frac{1}{8}\pi$ 。
3. 圆心在多边形严格的内部。需要判断每条线段和圆是否有交，若有交则不合法。若合法，对答案的贡献是 $\frac{1}{4}\pi$ 。

判断点是否在多边形内部时用射线法。

16.1.3 Complexity

令 $W = 500$ ，即一维的坐标范围。

时间复杂度： $O(nW^2)$ 。

空间复杂度： $O(n)$ 。

难度: 1.5/5。

16.2 B - Flight Planning

16.2.1 Description

给定一个飞机飞行高度油耗的计算公式，以及一条航线 n 个航段的数据，求一个飞行高度的方案使得总油耗最小。如果存在多种方案，求字典序最小的。

$n \leq 100$ ，飞行高度只有20种。

16.2.2 Solution

由于要求字典序最小，故我们倒着求。令 $f[i][j]$ 表示完成后 i 段航线，且飞行高度为 j 时的最小油耗，转移时枚举上一段的飞行高度，并记录方案。

16.2.3 Complexity

令 $k = 20$ 。

时间复杂度: $O(nk^2)$ 。

空间复杂度: $O(nk)$ 。

难度: 0/5。

16.3 C - Lead or Gold

16.3.1 Description

三种金属构成了 n 种合金，每种合金含有的三种金属的量的比例确定。给定一种待合成合金，问能否用已有的这些合金合成出待合成的合金。

$n \leq 100$ 。

16.3.2 Solution

对于一种合金，设三种金属占总量的百分比为 a 、 b 和 c ，由于 $a + b + c = 1$ ，我们可以用平面上的一个点 (a, b) 来表示一种合金。考虑只有两种合金的特殊情况，由向量的合成可得，只有当待合成合金对应的点在已有合金对应点的连线上时才可能。将结论推广到 n 个点，即当点在已有合金对应点的凸包内时才可能。

因此我们只要求凸包然后判断一个点是否在凸包内即可。当然由于数据范围小，直接枚举三个点然后判断应该也是能通过的。

16.3.3 Complexity

时间复杂度: $O(n \log n)$ 。

空间复杂度: $O(n)$ 。

难度: 1.5/5。

16.4 D - Page Selection by Keyword Matching

16.4.1 Description

给定 n 个页面，每个页面有不超过8个关键词。有 q 次查询，每次查询不超过8个关键词。给定了一个计算相关度的公式，输出最相关的不超过5个页面编号。

$n, q \leq 25$ 。

16.4.2 Solution

直接模拟即可。注意UVa上的数据有多余的空格。

16.4.3 Complexity

令 $k = 8$ 。

时间复杂度: $O(nqk^2)$

空间复杂度: $O((n + q)k)$ 。

难度: 0/5。

16.5 E - Petri Net Simulation

16.5.1 Description

一个Petri网中有 n 个“库”和 m 个“变迁”，每个库中可以有若干令牌。每个变迁有若干入库与若干出库。一可变迁可行当且仅当每个入库含有至少一个令牌。特殊地，一个变迁可能有若干相同入库，假设有 k 个入库 i ，则变迁可行要求入库 i 中有至少 k 个令牌。一回合中，任意选择一个可行的变迁发生，所有入库中的令牌

减1, 所有出库中的令牌加1。如果出库 i 有 k 个, 则出库 i 中增加 k 个令牌。如果不存在可行变迁, Petri网被认为死亡。给定Petri网, 模拟 T 个回合, 判断 T 个回合后Petri网是否仍未死亡, 并输出有哪些库中有令牌。

$n, m < 100, T < 1000$ 。

16.5.2 Solution

直接模拟即可。注意可能第0个回合死亡（即初始状态中就没有可行变迁），但不会在恰好 T 个回合后死亡。

16.5.3 Complexity

时间复杂度: $O(Tnm)$ 。

空间复杂度: $O(nm)$ 。

难度: 0/5。

16.6 G - Spatial Structures

16.6.1 Description

对于一个 $n \times n$ 的01矩阵, 可以构造出与其一一对应的四分树, 而我们可以用四分树中每个黑色节点到根的路径表示一棵四分树。实现01矩阵与四分树之间的相互转换。

$n \leq 64$, 且 n 为2的幂。

16.6.2 Solution

直接模拟即可。

16.6.3 Complexity

时间复杂度: $O(n^3)$ 。

空间复杂度: $O(n^2)$ 。

难度: 0/5。