杂题选

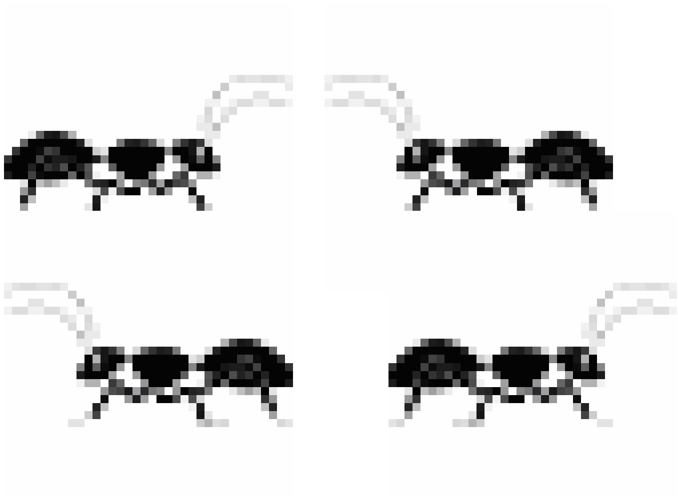
刘汝佳

蚂蚁

- 一些蚂蚁以1cm/s的速度在长度为Lcm的线段上爬行，爬到线段端点就会掉下去。当两只蚂蚁相遇，就会立刻掉头返回。已知L和一开始每只蚂蚁的位置，但不知道它们的方向，求它们最早何时全部掉落，最迟何时全部掉落。
- 最多1,000,000只蚂蚁。

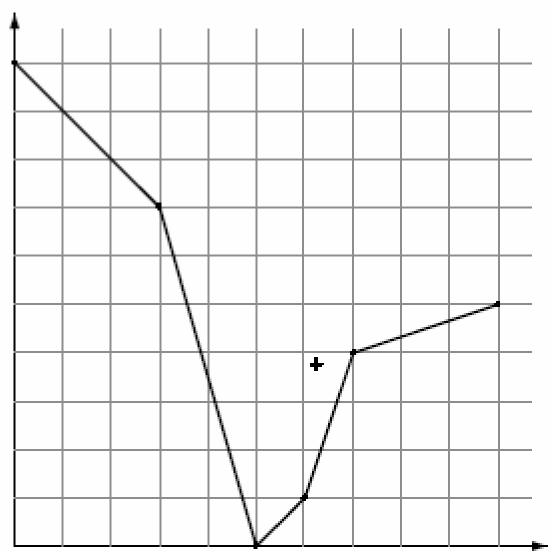
分析

- 相遇??



照明

- 给出x坐标单调, y坐标为凸的折线，表示一个洞穴的地面，求一个最低点，在此高度的某一个位置放一盏灯，就能照亮所有地面，一直线上也算照亮



蛇

- 一块正方形土地，从(0,0)到(1000,1000)，上面有n条蛇（被看成点），每条蛇都有一个攻击距离，问一个人能不能在不被蛇咬的情况下从西侧x=0走到东侧x=1000。
- 如果可以，给出尽量靠北（即纵坐标尽量大）的起点和终点。

Get Out!

- 一艘船（形状是圆形）被困在群岛中（每个岛都是圆形）。一共n(n<=300)个岛，给定它们的坐标和半径，又给定船的坐标和半径，问船能不能开出来。

Sumset

- 求整数N写成若干个2的整数幂的和的不同形式的总数（的最后9位数字）
- 例如：7 = 1+1+1+1+1+1+1
  - = 1+1+1+1+1+2
  - = 1+1+1+2+2
  - = 1+1+1+4
  - = 1+2+2+2
  - = 1+2+4
- 所以Ans(7)=6

方法一

- 令f[i,j]表示将i写成若干个2的整数幂的和，且其中最大的一个不超过2^j的总数
- 则$f[i,j]=f[i,j-1]+f[i-2^j,j]$
- 时间复杂度为O(NlogN)，如果使用了滚动数组，空间复杂度为O(N)

方法二

- 不难得到：当i为奇数时，**f[i]=f[i-1]**
- 如果i是偶数，我们考虑i的表达式中是否含有1：
  - 如果有，则i的表达式对应了一个i-1的表达式；
  - 如果没有，则我们可以将i的表达式的每一项除以2，就对应了一个i/2的表达式。
- 由此我们又得到：当i为偶数时，**f[i]=f[i-1]+f[i/2]**
- 状态总数只有N个，时间复杂度也随之变为O(N)

施工队

- 有一条道路，上面有N个坏的地方，现在政府准备修理这些坏的地方。政府将整条道路分为若干大小为M的段，对于每一段如果有坏处的话，就聘请一个施工队去修理。分段的方法如下：如果第1个段开始于K，那么第L个段的范围即为K+(L-1)*M到K+L*M（从K开始连续分配）。现在要求你确定这个K，使得聘请的施工队最少。

分析

- 枚举K, 每次移动窗口时…
- 所有坏处按模M分类, 每个恰好被处理一次
- 总时间复杂度为O(n)

Fiber Network

- 给定一张有向图，每条弧上标有至少一个字母。每次查询一对起点和终点，要求求出所有的满足以下条件的字母：从起点出发只沿标有该字母的弧可以到达终点

The Rotation Game

- 如图形状的棋盘上分别有8个1、2、3，要往A~H方向旋转棋盘，使中间8个方格数字相同

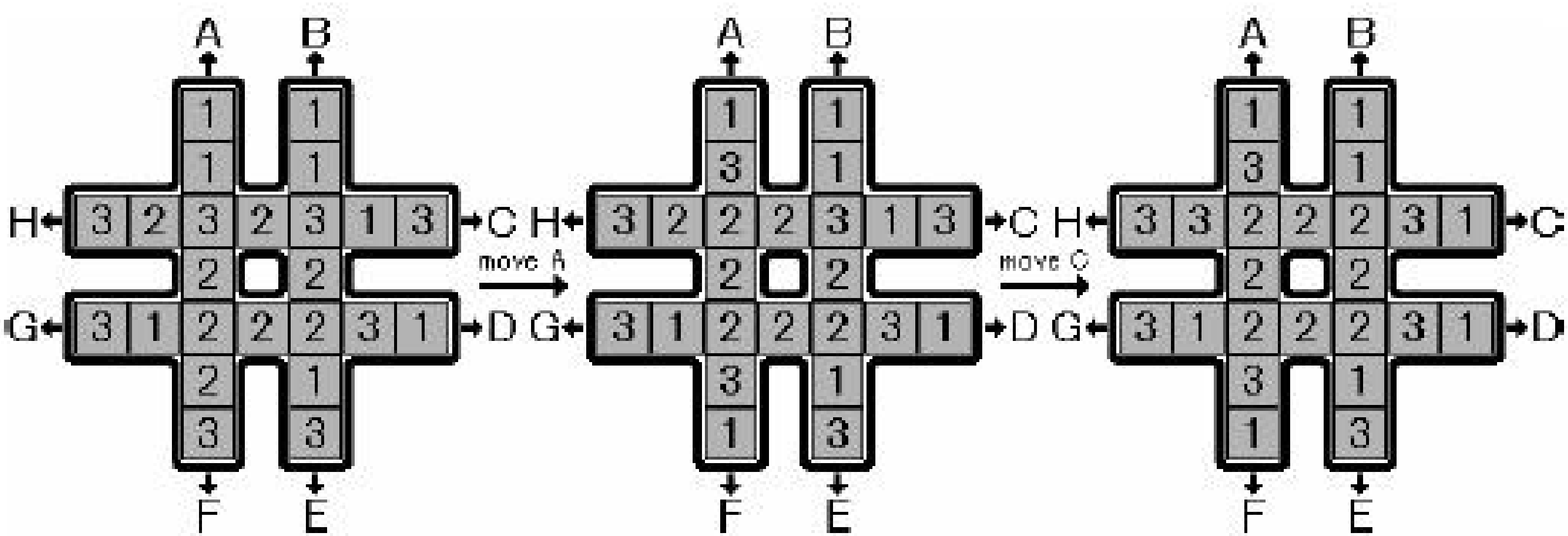


Fig.1

Tournament

- 每次选择两个程序比赛, 败者淘汰出局
- 已知一些程序之间的强弱关系, 求出所有可能得冠军的程序
- n<=100,000, m<=1,000,000

分析

- 不知道关系的程序, 相互都可能战胜对方, 需要连两条边u→v和v→u, 因此新图的边是$O(n^2)$的, 需要降为$O(m)$
- 方法一: 先求无向图补图的连通分量, 然后求原图的SCC
- 方法二: 随便生成一个方案, 找到一个强连通分量中的点，然后扩展. 理由: 基源只有一个!

Dept

- 教授分别欠了A,B,C三个人x,y,z ($<=10^5$)元钱，他决定用手头的N($<=10^5$)块水晶来偿还这些债务。一块小水晶值1元，大水晶则值2元。但由于各人的评判标准不同，对于同一块水晶，有人认为它是大水晶，有的却只认为那是小水晶
- 如何分配水晶以偿还所有债务? 也就是使得每个人得到的水晶，按照他自己的标准，价值都不少于教授拖欠的债务。

分析

- 先考虑BSS, SBS, SSB, 如果A已经还清则BSS等于SSS
- 枚举BBS给A的数量, 则其他给B. 如果A还没还完则BSB尽量给A, SBB先给B后给C
- 再分配BBB, 最后是SSS

Evil Straw Warts Live

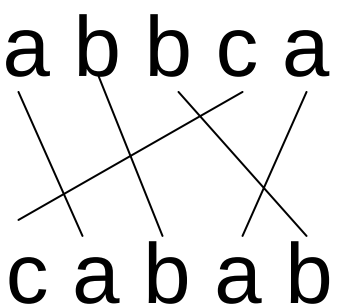
- 给定一个字符串，问通过多次交换相邻的两个字符能否将它变为一个回文串（即从左边和右边读起来一样）。如果能，给出最少交换次数

分析

- 有解当且仅当出现奇数次的字母最多1个
- 特殊情况：前一半和后一半中所含的每种字符的个数都一样（假设出现次数为奇数的字母不存在或者恰好在“前一半”和“后一半”中间）
- 将“后一半”倒过来，然后与“前一半”的字母一一对应起来。我们把“前一半”的字母编号，那么就可以由“后一半”得到一个序列

分析

- **逆序对个数!** 出现两个相同的字母（如前面例子里的两个a和两个b）怎么建立对应关系？回答也很简单：最靠左边的与最靠左边的相连，第二靠左边的与第二靠左边的相连……最靠右边的与最靠右边的相连即可。假如不这样做，逆序对的个数会比原来的大



分析

- 交换相当于在不改变其他字符的相对位置的情况下，将一个字符“推移”。假设左边多了个a，则把左边的最后一个a推到右边第一个a的左边。
- 为什么不推其他的a而是推最后一个呢？又为什么只需要推到右边第一个a的左面呢？因为我们后面要进行的计算：求逆序对的个数，相当于求出了最优解。假设内部多推几步（把最后一个a以外的a推到右边相当于把它推到最后一个a的左面，然后再把最后一个a推到右边，然后再浪费一步）后能得到一个更优的解，与逆序对最优矛盾

Polymania

- 给出N个点的权值R_i。要求将这些点安排在平面上，使得其中任意三个点组成的三角形面积都恰好等于这三个点的权值和。所有权值为正（其中，$3 \leqslant N \leqslant 8$，$R_{N-1}=R_N$）

分析

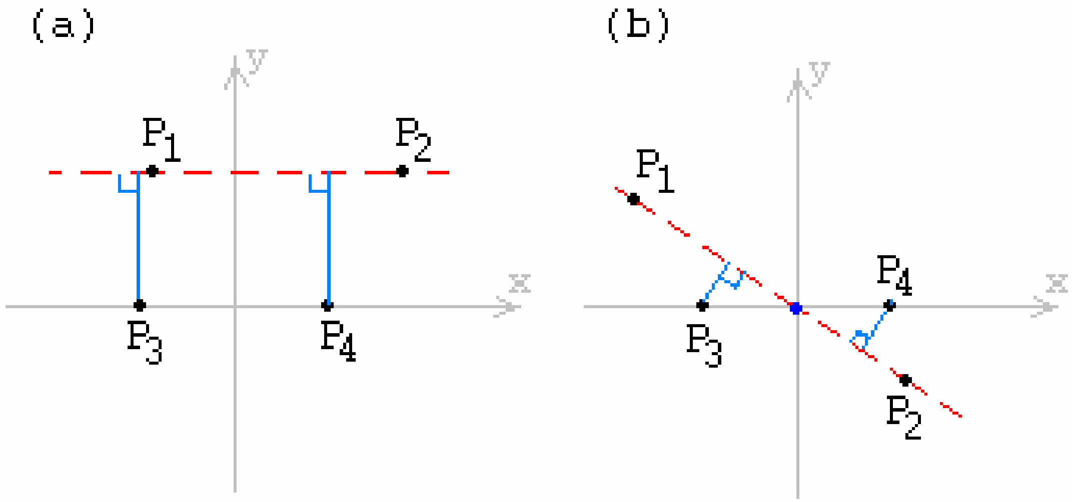
- 由于最后可行的安排方案不是唯一的，通过任意的旋转、翻转，仍然可以保证方案的正确性。因此，我们不妨将最后两个点安排在数轴上，设它们的坐标P_{N-1}与P_N分别为$(-x_0,0)$与$(x_0,0)$, 其中$x_0>0$, 最后两个点的距离$|P_{N-1}P_N|$等于$2x_0$
- 对于一个包含有*N*个点的合法的安排方案，去掉任意一个点，剩下的部分将仍然是一个包含有*N*-1个点的合法方案。因此可以由较小规模的的问题开始分析，找出规律，进而解决问题

N=3

- 三角形$P_iP_{N-1}P_N$的面积为$R_i+R_{N-1}+R_N(1<=i<=N-2)$，因此$P_i$到x轴的距离为$2(R_i+R_{N-1}+R_N)/2x_0=(R_i+R_{N-1}+R_N)/x_0$，这也就是第i个点的纵坐标的绝对值$|y_i|$
- N=3时只存在一个三角形，因此只要第一个点的纵坐标的绝对值符合上文所分析的情况，这样的安排就是合法的。令$x_0=1$，则$|y_1|=R1+R2+R3$，得到了一种安排方案，三个点依次为$(0,R_1+R_2+R_3)$、$(-1,0)$和$(1,0)$

N=4

- 由于$R_3=R_4$，因此三角形$P_1P_2P_3$和三角形$P_1P_2P_4$的面积相同，即P_3与P_4到直线P_1P_2的距离相同。只存在两种情况：一是直线P_1P_2与P_3P_4平行；二是直线P_1P_2经过线段P_3P_4的中点(0,0)



N=5

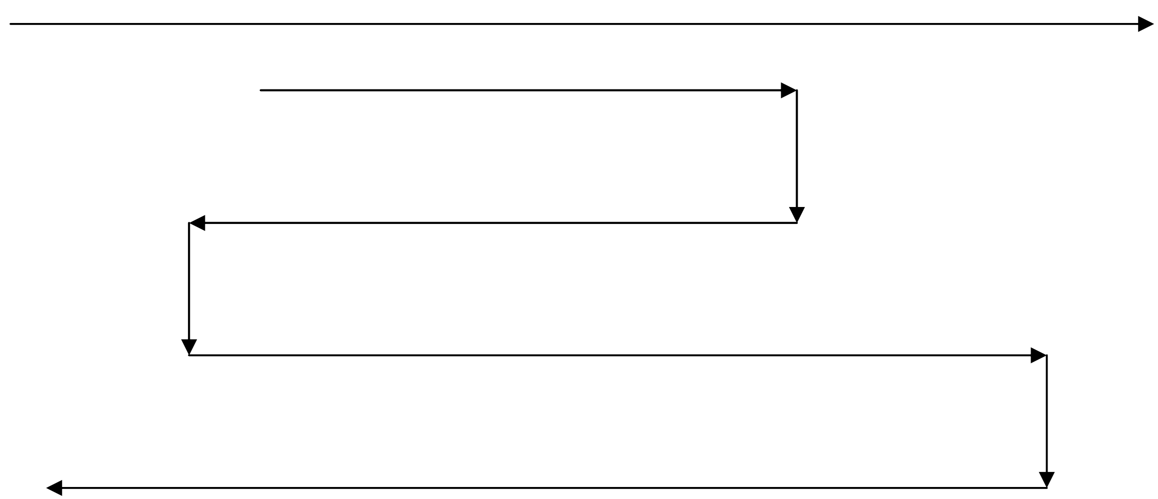
- P_1与P_2、P_1与P_3、P_2与P_3之间的位置关系都应当分别符合刚才的两种情况之一，至少会有两组或两组以上的位置关系符合同一种情况，那么就会推出P_1、P_2与P_3三点共线，即三角形$P_1P_2P_3$的面积为0，与点权为正整数矛盾
- 结论: N>=5时不存在合法的安排方案

灭火

- 在X轴上有N个城市，它们的坐标都是非负整数。有一天，这N个城市同时着火了。每个城市都有一个初始损失值C_i与增加损失值D_i.设开始着火的时刻为0，则如果第i个城市在时刻t被扑灭，则它的损失值为C_i+t*D_i.不幸的是，总共只有一台灭火车。它的初始位置为s.灭火车的速度为1单位长度/单位时间。它的神奇之处在于，在它到达某个城市的瞬间，那个城市的火就被扑灭。
- 给定N,s,每个城市的位置X_i以及C_i,D_i，要求最小的总损失值。

分析

- 被扑灭的区间是连续的
- d[i,j,t,d]表示时间为t, 方向为d…
- 时间复杂度O(n^2t), 无法承受

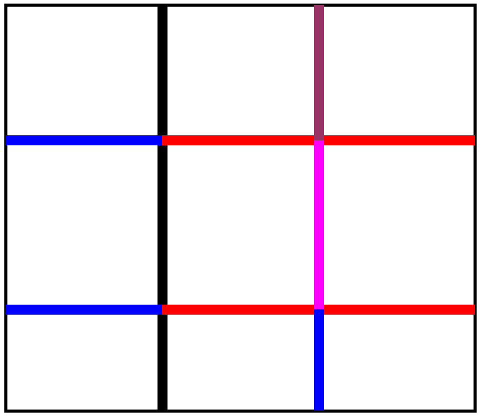


巧克力

- 有一块m*n的巧克力（可以想象成一个网格矩阵），现在需要把它切成m*n个大小相等的小块。沿着某一竖线切与沿着某一横线切都有一定的花费。沿着竖线切的花费依次为$x_1,x_2,\ldots,x_{m-1}$，沿着横线切的花费依次为$y_1,y_2,\ldots y_{n-1}$.在切的过程中只许一次对一块巧克力进行操作，且必须将它一刀切成两半。无论此刀的切痕有多长，都按照一整刀计算。现在要求出将巧克力切成m*n个小块所需的最小花费。

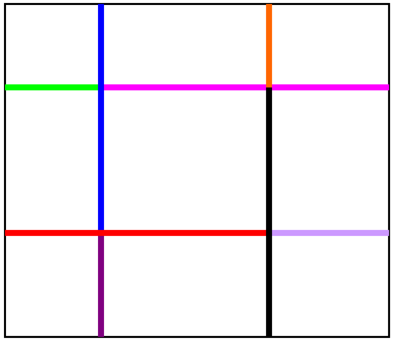
分析

- 切割过程：不断地选择切割的线并按照这条线完整地切下来，如果在与其垂直的方向上已经切割了i条线，则总代价就加上这条线的代价*(i+1)
- 如图，先选x1,接着选y2，然后选y1，最后选了x2。代价就是1*x1+2*y2+2*y1+3*x2



拓扑序?

- 拓扑序是存在的
- 新问题: 为$x_1,x_2,\ldots\ldots,x_{m-1},y_1,y_2,\ldots,y_{n-1}$安排顺序，总代价=∑(每一项的代价*排在它前面的与其垂直的线所对应的项的个数+1)，求最小总代价



dp vs greedy

- 解法一: x和y分别从大到小排序, 然后dp
- 解法二: x和y一起排序
- 相等值无影响

过桥

- 有N(1<=N<=16)个人想过桥，每个人都有重量W_i及独自过桥所需时间T_i. 为了尽快过桥，若干个人可以一起过。他们过桥的时间为其中最慢的那个人所需的时间。必须一组人完全过桥后，下一组人才可以出发。但是，桥本身也有能够承受的最大重量W. 所以规定一起过桥的一组人的重量之和不得超过W. 现在要求所有人都过桥的最短时间

Survival

- 在KOF中，你需要扮演一个角色。为了打通关，你必须战胜N个对手(1<=N<=20), 包括太阳神。你可以任意安排对手的顺序，唯一的要求是太阳神必须是最后一个
- 整个过关过程就是你依次与安排的对手交战，每次HP值减去所需的HP值，加上所恢复的HP值（超过100取100）。要求是任意时刻你的HP值必须非负。问否可能打通关

电梯

- 给定N个用户(1<=N<=10)乘坐电梯的起始层A_i与终止层B_i以及他们到达电梯口的时间T_i. 电梯初始时位于第1层。电梯从第i层运行到第j层需要|i-j|单位时间。每个时刻电梯可以选择上升、下降或停止不动。用户上下电梯的时间忽略不计
- 要求从初始时刻0到所有用户都到达各自终止层所需最短时间

分析

- 状态量: 已到达人集合、当前层(离散值), 电梯里的人集合
- 转移
  - 上升一层
  - 下降一层
  - 等待: 转移细化
- dp? dijkstra

Game

- 给定L1,L2，分别代表两个数列的长。这两个数列都是正整数序列。
- 每次可以从序列1的尾部选取K1个，序列2的尾部选取K2个。设序列1尾部的K1个的和为S1，类似地定义S2。这次将K1+K2个数删掉的费用定义为(S1-K1)*(S2-K2)。
- 注意，每次删除后，两个序列不能一个为空，另一个非空！求将所有元素删除，所需要的最小费用

分析

- 最先的想法是容易想到的，就是将所有元素减一，费用被重新定义为S1*S2。
- 对某次操作，如K1>1且K2>1，那么它一定可分解为多次K1=1的操作，费用更小
- 设d[s,I,j]表示对于序列1的前i个数，以及序列2的前j个数。若s=0，那么表示当前正在处理K1=1的操作，否则是K2=1的操作。

Unique

- 给定N<=2000001个数，其中只有一个数只出现一次，其它数至少2次。
- 要求找到那个数并且输出

Joke with Turtles

- 有n只乌龟描述一次赛跑中报告自己的成绩（多少比自己快，多少比自己慢，余下的任何自己成绩相同）。有的乌龟说谎，试求一个最大的全说真话的集合。

分析

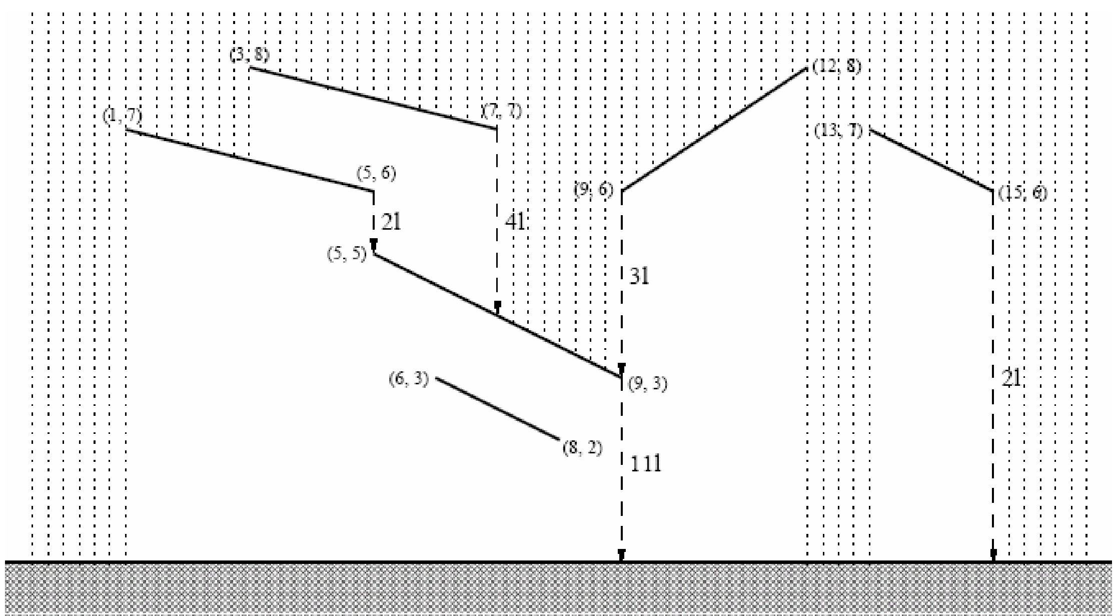
- 有*N*个区间，每个区间都有某个特定的权值，要求选择一些不相交的区间，且权值最大。而这，在为区间的节点进行O(*N*log*N*)的排序之后，完全可以用一个简单的O(*N*)的DP来完成。如果硬要追求复杂度的话，O(*N*)的基数排序，也是可以完成的，因此本题算法的时间复杂度仅为：O(*N*)

Fun Game

- 一个01组成的圈，已知其中的n段弧，每段弧可能是顺时针或者逆时针。请还原这个圈，使圈是所有可能的情况中最小的一个

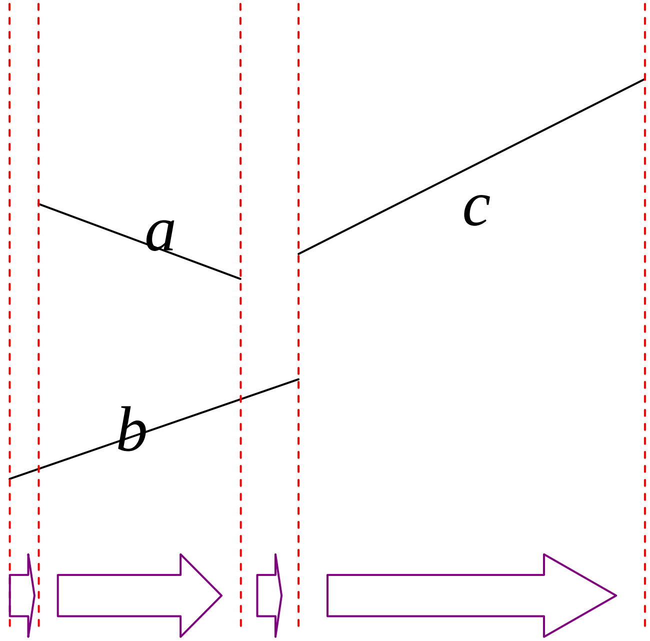
Rain

- 在单位时间单位长度降雨量为1单位的时候，每条“屋顶”都有一个单位时间落下的水量(就是单位时间从屋顶较低的那一侧落下去的水量). 对于每个屋顶，算出落水量



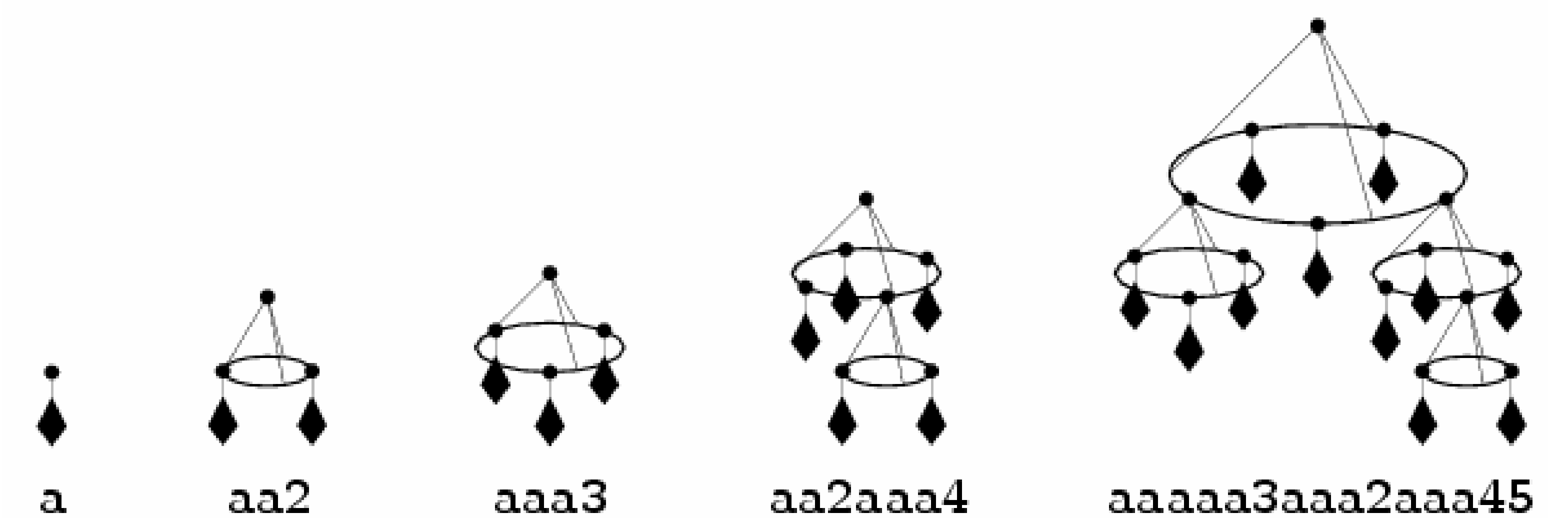
主算法

- 用扫描法构图, 然后BFS统计
- x坐标相同的事件点，我们要先插入，再检测落雨点，最后删除，才能满足题设条件



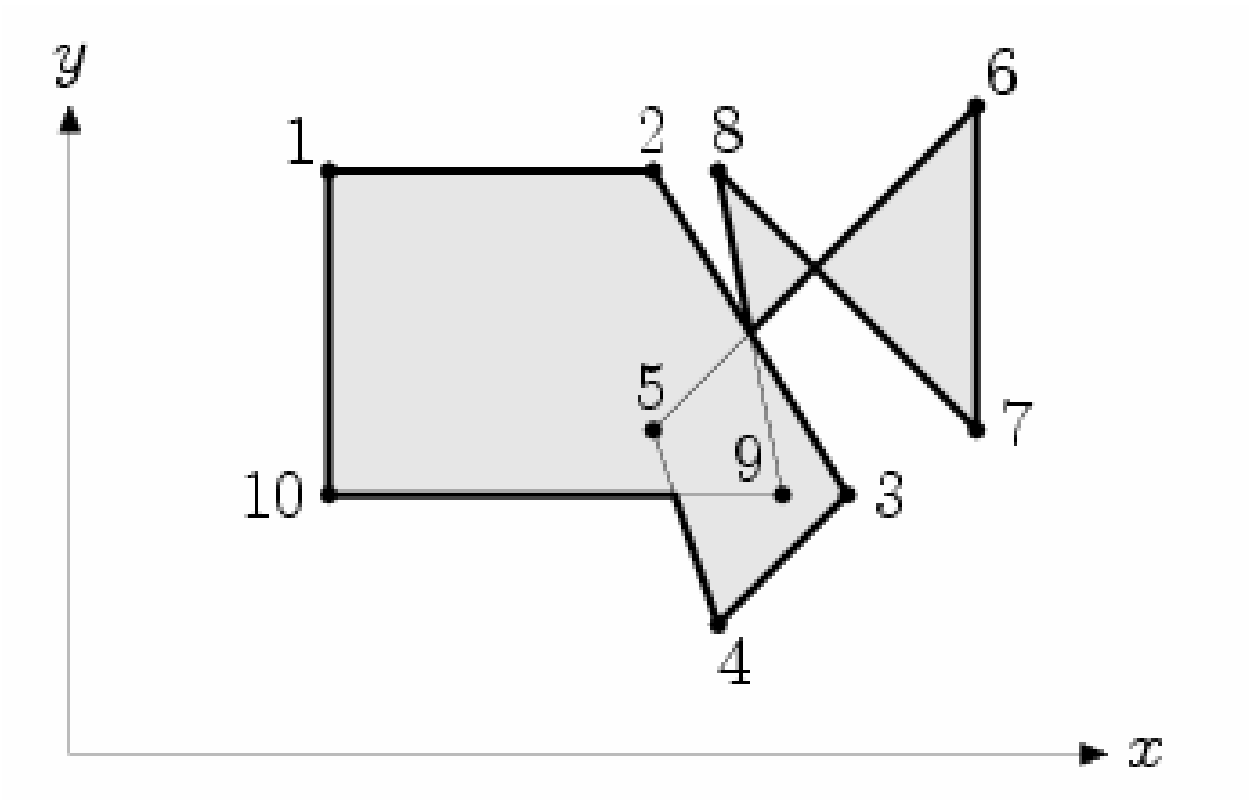
Chandelier

- 寻找与已知串等价的、需要栈深度最小的串. 每个ring上可以旋转
- 下图的最优解为6. 方案aaa3aaa2aaa4aa5



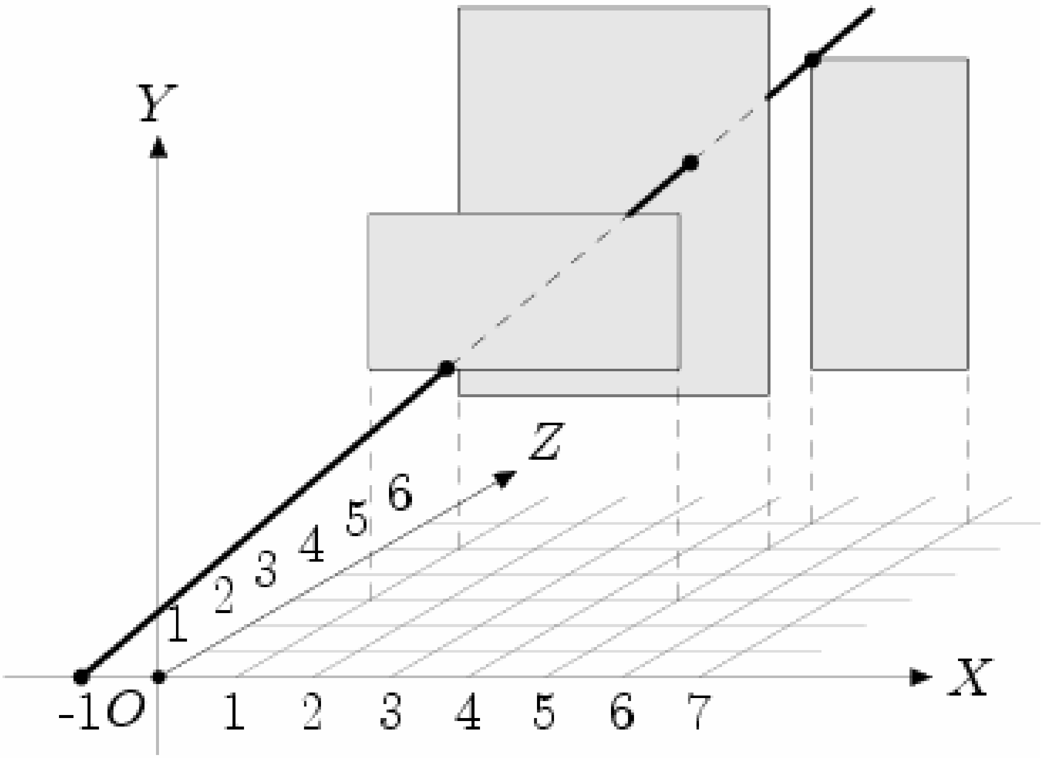
Find the Border

- 求轮廓线



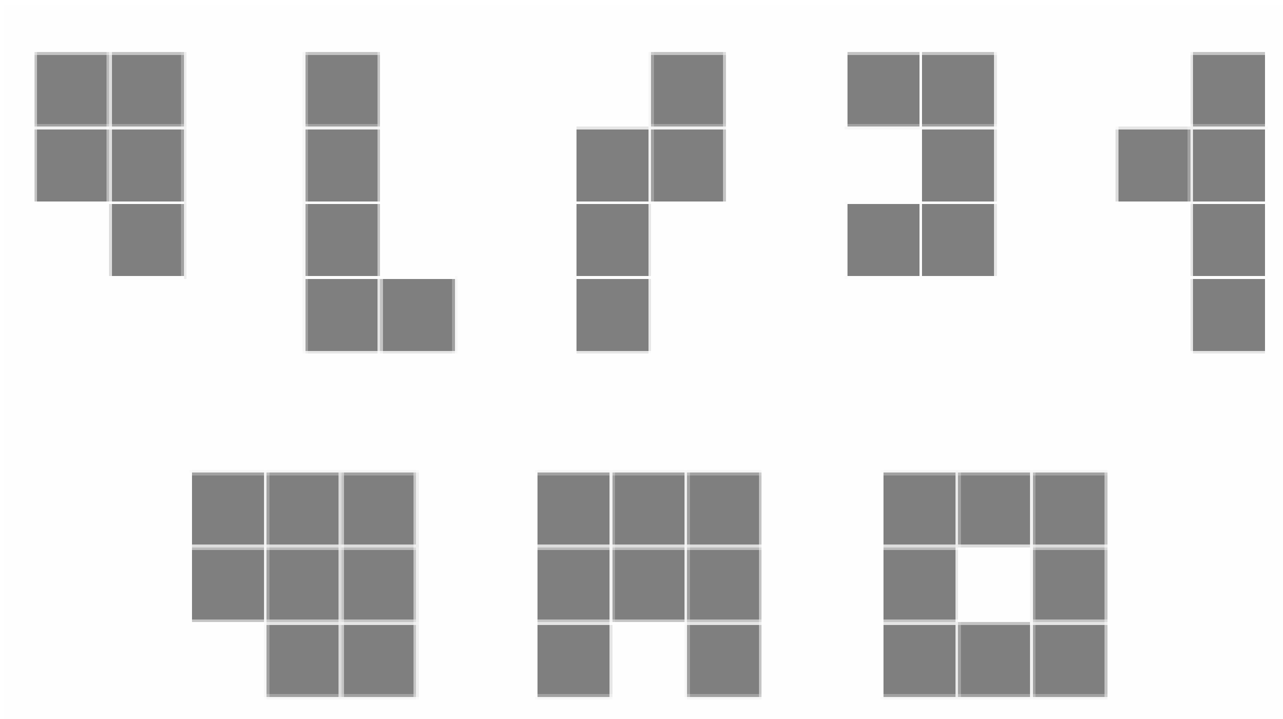
Gunman

- 如何发射一棵子弹, 穿过尽量多的矩形



Lattice Animals

- 计算w*h的格子里选n个组成的连通图形的个数(旋转、翻转不能重复)



Irrelevant Elements

- 给n个数, 每次求相邻两个数的和得到新数列, 最后变成一个数, 求它mod m的值
- 问这个值和最初的哪些数无关

Barbarian Invasion

- 给出n个点, 删除一个点使得其他点的凸包周长尽量小

BTP

- N头奶牛(N<=65536, 总是2的幂次)参加网球比赛。在第一轮，N/2对选手进行比赛，获胜的N/2个选手进入下一轮。同样，下面的每轮比赛中，都是获胜的一半进入下一轮，直到只剩一头牛
- 如果参赛的两头牛排名之间的差距大于一个给定的常数K(0<=K<=N-1)，那么排名较高的奶牛总是会赢得比赛的胜利。计算最后有可能夺冠的牛中排名最低的牛的排名，并且给出一种能使这头牛获胜的场次安排。

分析

- 二分最后牛的排名X, 进行模拟
  - 决赛是X胜X+K
  - 每次让每头牛胜尽量强的牛
- 朴素方法: 从X+K开始检查X+K, X+K-1, …, 直到有一个没输过
- 优化: 并查集

Fence

- 有M个油漆工和顺序放置的N个木板。第i个油漆工坐在木板S_i前，手臂可以伸长L_i，也就是说，他只能对S_i-L_i+1~S_i+L_i-1的木板漆油漆。每个油漆工所漆的木板，必须是一个连续木板序列，并且包含S_i。第i个油漆工漆每个木板的费用为P_i
- 求使得总费用最大的一个安排方案，每块木板最多只能由一个油漆工上漆

Friends

- 有N个人，分成了若干个组。每次可以询问一个序列的人，问他们所在的组的所有的人是哪些
- 例如: N=8，3组{1,2,3,4},{5,6,7},{8}
  - 你如果询问{1}，会得到{1,2,3,4}的答案
  - 如果询问{7，8}，会得到{5,6,7,8}的答案
- 必须用不超过log2(N)+1的询问次数

Race

- 给定平面上N<=2000个点的坐标。另外有M<=25000条无向边。
- 现在要求从1号点开始绕一个圈，每次只能左转（转角[0,180)度范围内）并且使路径上的最短的边，尽可能长。

Empodia

- 生物序列是一个包含0到M-1的一个排列，第一个数是0，最后一个数是M-1。任意数都不会比前一个数恰好大1。片段是生物序列的连续子序列, 帧是特殊的片段，它的第一个数和最后一个数分别是整个片段中最小的和最大的，而且夹在它们中间的数都在这个片段中出现。empodio是特殊的帧，它里面不包含其他的帧
- 例如(0, 3, 5, 4, 6, 2, 1, 7)本身是帧, 但包含另一个帧(3, 5, 4, 6)，所以它不是empodio。(3, 5, 4, 6)不包含帧，所以它是empodio。找出一个生物序列的所有empodio。

分析

- (i,j)为帧的充要条件
  - j-i-1 = A[j]-A[i]-1, 因此**A[i]-i=A[j]-j**
  - bigger(i)为i左边第一个大于A[i]的数的下标, smaller(i)类似, 则**smaller(i)>j, bigger(j)<i**
- 主算法: 先找出一些帧, 再排除重复
- 方法: 扫描的同时计算smaller, bigger并维护可能的左端点集合, 以**A[i]-i**的值分类

分析

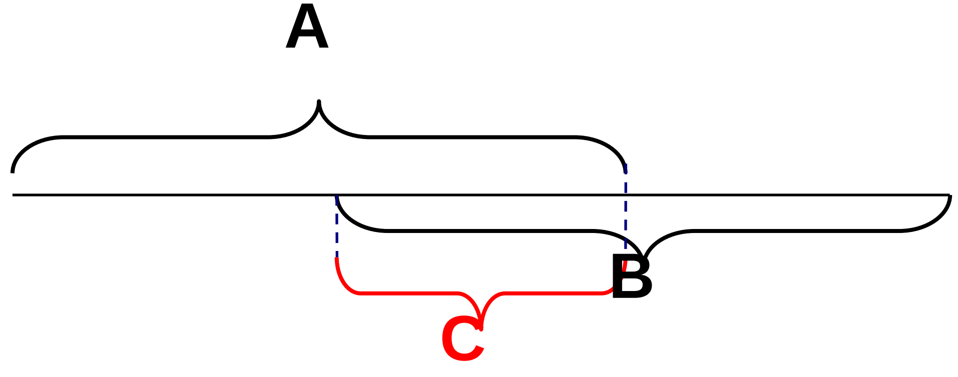
- 栈Down存放递减序列. 每次检查下一点，如果该点满足递减关系，直接将其加入；否则，逐步退去栈顶元素，直至满足递减关系，并将新点加入。当栈维护到j并要将j加入时，此时的栈顶元素就是bigger(j)
- 链表List存放候选左端点集合. 每一个链List[k]存放所有A[i]-i=k且可能成为某个片段的左端点的点
- 栈Up存放递增的序列. 当栈维护到j并要将i退栈时，就表示Smaller(i)=j, 同时也表示i所代表的点已经不可能再成为某个右端点在j或j右侧的片段的左端点，将i在List[A[i]-i]中删除

分析

- 从小到大枚举片段的右端点j，同时维护栈Down和Up，并检查List[A[j]-j]链中最后一个点i（即最近加入的点）。
  - **i>bigger(j)**, 则(i,j)是满足要求的片段，不需要尝试List[A[j]-j]中的其它点，因为即使也满足要求，但内部已经包含了一条更小的片段(i,j)
  - 否则(i,j)不满足要求，则List中比i小的点就更不可能满足要求。但对于更大的j，这些点有可能满足要求，因此仍将它们留在List中作为候选点。
- 每个元素都只处理一次，总时间O(N)

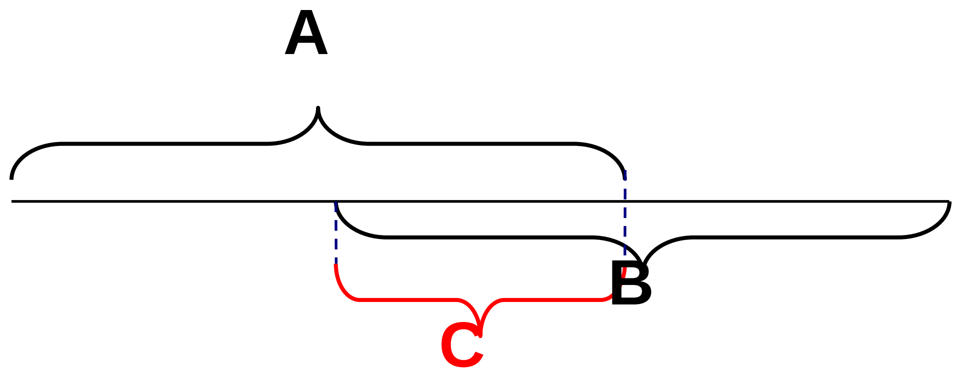
A和B的交

- 设A为(l, j), B为(l', j'), 则C为(l', j)
- 满足$S_{l'}$<x<S_j的数一定在A中也在B中，因此在C中, 且不可能有其他数(不满足A和B的范围限制)
- 结论: 有A有B→有C



分析

- 第一步求出的片段中仍然可能有相互包含的情况, 但不可能出现相交但不包含的情况，因为A有B→有C→无A）
- 删除那些包含其它片段的片段，剩下的就是满足题目要求的所有片段，这一部分也可以在O(N)时间内完成



The Book Shelver’s Problem

- 有N本书，每本高h_i，宽w_i。需要将N本书按顺序放入一宽为W的书橱中。放置时可以将若干本书放在同一层中，该层高度h为这些书中最高的高度。将整个书橱至少需要多高才可放下所有书？

画矩形

- 平面上给出N个点，以这些点为左上角画N个矩形，矩形的边平行于坐标轴，每个矩形的大小完全一样，且长（横向）正好是宽的3倍。所有矩形都不能相交，但边或顶点可以重合。

Pigs

- 有M个锁着的猪圈，每位顾客都有一些钥匙
- 买卖的过程是这样的：一个顾客到来后打开所有他可以打开的猪圈, 然后你从这些猪圈里牵出固定数目的猪卖给顾客（最多只能和顾客需要数相等），并可以重新安排这些开着的猪圈中的猪。每个猪圈可以存放任意数目的猪
- 已知所有顾客有的钥匙(每把钥匙只能开一个特定的锁)和他们需要买猪的数量在事先都告诉了你，你要使卖出去的猪最多

Traffic

- 在一个数轴上一共有N辆汽车，每辆汽车都在以固定的速度行驶，给出它们在0时刻的位置S_0以及它们的速度v。
- 在某一时间t，位置S_t最大的那辆汽车为领先汽车，求在时间段$[a, b]$内每一辆汽车的领先时间。
- 规模：N不大于10^5，输入数据按照v从大到小给出，没有任意两辆汽车速度相同。