做男人不容易系列:是男人就过8题
--LouTiancheng题

PKU 1737-1744

部分引用TimGreen大牛去年的ppt

Connected Graph

- 求N个顶点的连通图的个数。N<=50,每个顶点看成是不同的。
- 方法是显然要Dp了。

方法一

- S[x, y]表示一个已连通的x个点的团和y个孤立点组成连通图的方案数。
- F[N] = S[1, N - 1];
- 对S[x, y]用记忆化搜索。转移时枚举有几个y直接连向x。
- 只是跑的太慢最多只能打表交了....
- O(N^3*高精)

方法二

- 记F[N]就是答案,G[N]是2^(N*(N-1)/2)-F[N];
- 我们这么计算G[N]。枚举和第一个点连通的有多少个点,余下的点任意。
- 所以Sum{C(i-1,N-1)*F[i]*2^((N-j)*(N-j-1)/2),i=1…N-1}
- O(N^2*高精)

An old Stone Game

- 经典的石子合并问题 每次合并代价为两堆石子数的和 求总代价的最小值
- 单纯贪心的反例：5 3 4 5

方法一

- 圆方贪心
- 开始认为是N个圆。
- 每次合并两个和最小的且中间没有圆形物品的物品，变成一个方的物品。
- 合并所有相邻的方。
- 全局用Winner Tree取最小（Winner Tree的相关内容可以看黄劲松的论文）

合并相邻方所采用的数据结构

- （1）fib堆 O(NLogN)（可以参考龙凡的ppt）
- （2）二项堆 O(NLogN)
- （3）左偏树 O(NLogN)（可以参考黄源河的ppt）
- （4）普通堆+启发式合并 O(N(LogN)^2)

比较

- 编程复杂度（1）>（2）>（3）>（4）
- 运行速度（不包括（1））
- （3）>（2）>（4）

方法二

- Knuth的方法
- 从左往右扫描，第一次遇到a, b, c且a > b, c > a，则将a, b合并

Tony's Tour

- 求从左下走到右下角的哈密尔顿路的数量
- 与HNOI04－Day1的一道题目相似
- 搜索很难通过，只能DP

TimGreen大牛的解法

- 状态压缩的Dp。
- 状态是一行(或一列) 的连通性(用最小表示)。
- 如010122表示第2个和第4连通了,第5个和第6个连通了。第1个和第3个没有向下走。
- 每个点的走法有6种(─│ ┐ ┌ └ ┘)
- 然后一行行Dp下去(Search每个点的走法,有些烦)。
- 中间因为不是所有的状态都是合法的,所以每一层的状态数不是很多。
- 再一点要注意的是最后一行　起点和终点上都只能是(─│)　连通性只能是10..001

A New Stone Game

- 开始给出N堆石子,每一次可以选一堆石子取走至少一个,然后可以任意的将这一堆余下的任意多个分配到其它堆里。问两个人都使用最优策略的情况下, 是不是先手胜。

结论

- 会输只有一种情况“N是偶数且每个数出现偶数次”

证明方法

- 证明有点繁,大致是这样。
- 定义上面所说的输的状态全部属于T。
- 定义所有不属于T的状态属于S。
- 首先先证明对于T中一个状态执行一步后一定会属于S。
- 再证明对于S中的每一个状态一定有一种方法可以使它转移到T中。
- 最后注意到全空这个输的状态属于T。
- O(1)

Tree

- 求一棵树中距离不超过给定值的点对数

- 对于一个树，去掉一个结点，最分散的每颗子树分别求解，然后用O(NLogN)的方法合并结果。
- 一般排序 $O(N(LogN)^2)$
- 基数排序 O(NLogN)

Coins

- 给出N种硬币和个数,问可以取到1->M中的多少个值。

- 经典的01背包 复杂度O(NMC)　超时！
- 下面介绍来自Lee.MaRS大牛笔记的两种可以AC的方法

方法一

- 将1...ci的coin看面1,2,4,..2^x,ci-(2^{x+1}-1)的组合。
- 例如15个1与1 2 4 8是等价的
- 复杂度降为O(NMlogC)
- 将多个bool压成int（Pascal 32个bool压成longint，C＋＋ 直接使用bitset）

方法二

- 剩余类优化的动态规划算法
- 状态仍然是F[i,j]表示用前i种钱币是否能拼出面值j。考虑在计算第i阶段时，面值为d[i]，数量为n[i]。从状态转移方程中，我们发现F[i,j]所依赖的所有状态，都属于模d[i]的一个剩余类[j mod d[i]]，即不同剩余类内的状态不相互影响。于是，我们可以将第i个阶段的状态按剩余类划分，每次只对一个剩余类的状态进行更新。
- 复杂度O(NM)

Musical Theme

- 给出一个数列，将数列相邻两项做差，形成新数列，求数列中的最长重复子串（不可相交）

方法一

- 后缀数组+二分答案（后缀数组相关内容可以看许智磊的论文）
- 假如二分得到答案L，如何知道它是可行的呢？
- 因为对于排序后的后缀，Lcp (Suffix (List [i]) , Suffix (List [i - 1]))
- 是所有与Suffix (List [i])的LCP值中最大的一个。
- 因为 Height [i] 表示的是排序后后缀数组中第i个后缀和第i-1个后缀的LCP值。
- 那么对于后缀数组中的一段 L - R , 若 Height [L + 1] ~ Height [R] 全部大于等于L，那么就等价于第L到第R个后缀中任意两个后缀的LCP值都大于等于L。
- 那么只要取这里面相隔最远的两个后缀，若他们相距大于L，那么就是可行的。
- （ 为什么不是等于L呢 ？ 因为我们取的关键字是 S[i]-S[i-1] ， 若相距等于L，那么两段里面的首尾相连了，是不符合条件的）
- P.S. LCP = 最长公共前缀

方法二－TimGreen大牛的方法

- 先坐出原数列差数列。对差数列建后缀树。
- 如果不要求不相交的话。因为每一个中间结点以下的子树至少有两个叶子。所以这个结点到根行成的单词一定是重复子串。那么只要对后缀树中和每一个中间结点看不看长度,找出最大的就是答案。
- 现在考虑相交的情况。
- 对于一个中间结点,它到根和单词可以是不相交和重复子串，它以下的叶子结点中有两个的长度差>=这个重复子串的长。
- 所以我们从下到上树形Dp,O(N)计算出每一个中间结点下的叶子结点长度的最大值和最小值。
- O(N)

Elevator Stopping Plan

- 给出N个人要去的楼层。电梯4s每层,人20s每层,电梯若要在一层停留就要停留10s。求最迟到的人的最早能到达时间。

O(NlogN)

- 对于每个给定的时间t，我们可以使用贪心法确定是否可以在时间t内让所有人都到达目的层。显然，每一次电梯都尽量往上开。
- 比如说现在第i层有人要下，电梯应该在哪一层停靠呢？假设电梯已经停靠了n次，那么我们让电梯在第j=[(t-10*n+20*i+4)/24]层停靠即可。注意此时若j<i，那么在t时间内不可能让所有人都到达所在层。对t枚举时可以采用二分法，加快速度。
- 注意：1。可以直接走楼梯。
- 2．电梯在第j层停靠以后，不能直接继续考虑第2*j-i+1层，而是考虑第(t-10*n+16*j+4)/20+1层。