



CSP考点整理

Amagi_Yukisaki

关于这篇课件

- 关于CSP常见考点的一些梳理
- （虽然我也只刷掉了从NOIP存在以来到2016年的提高组全部题目，AK过NOIP2018D1）
- 算法按照我认为的常考程度/重要程度排序。（本课件共34题）
- 希望个人的汇总能让大家看到一些隐藏在题目背后的普遍规律吧。
- 也希望我的微薄努力，能幻化成一点星光，照亮前路，让大家不再迷茫，一路勇往无前。
- 以梦为码，逐鹿天涯。
- 背景Pixiv71471901P3。

打表找规律

- 打表找规律，就是通过计算机的计算，求出一定范围内的答案，然后通过人类智慧找出答案之间的关系。
- 这里最经典的就是NOIP2017：小凯的疑惑
- 答案就是 $a * b - a - b$ ，暴力背包+打表很容易观察。
- 然而我选择了手推 $exgcd$ ，浪费了不少时间。
- 遇到这种题目不妨在手玩之前先让机器帮你玩一下。
- 不要让计算机下岗嘛。
- （单隐层神经网络在规模足够大的情况下可以以无限逼近精度拟合任何连续函数。你的大脑可比这东西复杂多了）

模拟

- 顾名思义，就是按照题意进行操作。利用计算机的算力，快速得出答案。
- 不要小看模拟题，真的很重要。别人都AC的题你丢分了，那你基本上就能考虑回哪个班学文化课了。
- NOIP2016：玩具谜题
- 直接按照方向模拟转圈即可。
- NOIP2017：时间复杂度
- 写一个解释器，计算时间复杂度？（这题我一个函数少打了 `return`，然后就爆零了……）

模拟

- *NOIP2003*: 侦探推理
- 算法十分显然: 枚举今天是星期几, 然后枚举谁是罪犯。
- 判断是否有解, 解是否唯一。
- 难度在于字符串处理!
- 建议这种简单的字符串处理不要用传统字符串数组, 用string类。
- 无论是空格还是换行符的处理都要简单得多。
- (我当时*NOIP2017*: 时间复杂度就是用传统字符串写的, 特别麻烦, 根本调不出来。当然也不排除是我太菜了)

模拟

- *NOIP2005*: 等价表达式
- 思路十分显然，枚举几个数，带进去，看数值是否相等即可。
- 如果是`python`的话，直接`eval()`即可。
- 然而`C++`需要自己写多项式解析，维护两个栈即可。
- 这种东西一定要熟练！别人分分钟写完你浪费半天最后也会凉凉。

模拟

- 最后欢迎大家来挑战一下这道题：
- <http://www.ntoj.top/problem.php?id=1067>
- *NTOJ*1067: [*RYOI*2018]人造情感
- 真的是非常简单的题目啦！代码也就写了10K吧。
- 还没有大作业的*UCT*长。

动态规划

- *NOIP*非常喜欢考动态规划，几乎是每年必考。
- *CSP*估计也会这样吧。
- *NOIP*阶段的*DP*题目一般都不是太难。
- 合理设计状态，列出转移方程，写出暴力，60分就到手了。
- 之后考虑数据结构/方程性质优化（决策单调性，斜率优化）等。
- 如果实在想不到怎么优化，就暴力走人。不要浪费太多时间。
- 注意不要把一些看起来很像找规律的*DP*数学题当成找规律题做。那样真的找不出来规律。

动态规划

- 常见考点：
- 按照优化方式：
 - 暴力 DP ，数据结构优化 DP ，斜率优化 DP ，决策单调性优化 DP ，矩阵优化 DP 。
- 按照表现形式：
 - 序列 DP ，树上 DP ，图上 DP ，状态压缩 DP ，数位 DP ，组合计数。
- 按照实现方式：
 - 普通迭代 DP ，记忆化搜索 DP ，分治实现 DP ，动态 DP 。（注意记忆化和迭代在某些迭代难以实现的题目上有奇效）

动态规划

- *NOIP2010*: 引水入城
- 显然每个蓄水厂能供水的干旱区是一个区间。对第一行每个点搜索出其能覆盖的区间，然后跑区间覆盖即可。
- 可以记忆化一下避免每次重新进行整个搜索。

动态规划

- *NOIP2016*: 换教室
- 暴力预处理点对间最短路。
- $f[i][j][0/1]$ 表示考虑前 i 节课, 换了 j 节, 第 i 节是否换, 最小期望。
- 转移显然。



动态规划

- *NOIP2016*: 愤怒的小鸟
- 状态压缩DP，枚举过三个点的每一条抛物线，看能打到那些猪。预处理一次能打到的所有集合。
- 转移显然。



动态规划

- *NOIP2006*: 2^k 进制数
- 先求出答案最多有多少个 2^k 进制位, 记录为 a , 恰好有 a 位时最高位最大数字, 记录为 b , 然后考虑 *DP*。
- $f[i][j]$ 表示含 i 个 2^k 进制位, 最低位 $\leq j$ 的数字个数。
- 显然这个数字内不能有任何一位为 0。我们可以将每一位转化为 2^k 减去这一位的值。这样就变成了左边每一位比右边大。
- 每次向从高到低数的第一位添加数字。 $f[i][j] = f[i][j-1] + f[i-1][j-1]$ 。
- 答案为 $\sum_{i=2}^{a-1} f[i][2^k-1] + \sum_{i=1}^b f[a-1][2^k-1-i]$
- 前面是枚举长度, 后面是枚举最高位的值。
- 然后写一个高精度就好啦。

动态规划

- *NOIP2017*: 宝藏
- ~~枚举每个点的父节点能拿到70分？摸了摸了。~~
- 考虑状压暴搜，枚举一个起点，在搜索的过程中动态维护每个点的深度信息。对于一个状态，枚举已经在树中的一个点，加入一个没有在树中的点。只在更新后更优的情况下递归搜索。（复杂度？反正不是 $n!$ ）

动态规划

- *NOIP2017*: 逛公园
- 先把距离为0的块缩起来。从1, n 分别跑最短路。
- 如果最短路经过了块, 则答案为inf。
- 否则令 $f[i][j]$ 表示从1开始到点 i , 多走长度为 j , 方案数。显然 j 不能大于50, 且转移中 j 单调不减。
- 跑出最短路图, 拓扑排序转移即可。
- 最后别忘了取模! (70变30的故事)

动态规划

- *NOIP2018*: 货币系统
- 按照面额从小到大暴力背包。显然能被更小的面额表出的面额是没有用的；不能被表出的面额必须用，否则它自己就不能被表出。
- 听说*unsigned char*比*bool*快？



动态规划

- *NOIP2018*: 保卫王国
- 显然的DP, $f[i][0/1]$ 表示点 i 有/无军队, 子树合法的最小代价。
- $f[i][0] = \sum_{j|son_i} f[j][1]$
- $f[i][1] = val[i] + \sum_{j|son_i} \min\{f[j][0], f[j][1]\}$
- 钦定一个点? 把转移写成矩阵形式, 倍增维护一下就好。点 i 的结果矩阵为一个初始矩阵和子树结果矩阵的连乘积)
- 不要写树剖+线段树维护动态DP, 我当时写了没调出来(这有何难.jpg)

贪心

- 贪心算法是指：在当前的每个位置选择最优解，一定能得出全局最优解。
- 和动态规划的区别为：动态规划的局部最优方案不一定包含于全局最优，而贪心算法的局部最优一定包含在全局最优当中。
- 贪心的常考程度与动态规划不相上下。
- ~~但是普遍代码难度小于动态规划？~~
- 考点：不可整理（依据题目性质分析解题策略）

贪心

- *NOIP2018*: 铺设道路
- 我们记录答案为更新完前 i 个点后的最优解。
- 考虑第 i 个点和第 $i - 1$ 个点的高度，如果第 i 个点更小，就在填第 $i - 1$ 个点的时候顺便把它填了，否则需要多填的次数就是两者的深度之差。
- 这题考场上10分钟A不掉你就没时间扫雷了。

贪心

- NOIP2018：赛道修建
- 最小值最大，一眼二分答案。（注意限制一下二分边界防止卡常数）
- 考虑如何验证。我们令 $f[i]$ 表示从 i 点上面向下，所能达到的最长长度。
- 如何计算在满足限制情况下最多能组成多少条赛道。
- 将点 i 的孩子的 $f[]$ 按照放进一个 set ，能单独出来的提前删去。
- 然后从小到大单独试图为每一个进行配对，配对成功删去。
- 将 set 中剩下的最大值作为 $f[i]$ 。
- 正确性？因为每个孩子的 f 最多用一次，所以在 i 这里先进行配对显然是不劣的。（这题30分钟A不掉的话，你依旧没时间扫雷）

贪心

- *NOIP*2018：旅行（从这里我们可以看出*NOIP*2018考了三道贪心）
- 图中最多有一个环。
- 将每个点的相邻点按照从小到大排序。
- 枚举断掉哪条边，从1开始暴搜即可。
- 正确性显然。
- 注意常数。
- *vector* 存边+每次重构+每次 *sort* 的我被卡了12分（老年选手无所畏惧）

贪心

- *NOIP2008*: 双栈排序
- 首先两个栈中的元素都应该是单调不增的。
- 对于每次操作, 先判断能否弹栈。因为没有相同数, 所以可以弹栈时一定要弹。
- 对于能压入 a 栈的数 t , 找到它后面第一个同时比它和 b 栈栈顶大的数 c 。
- 在 c 后面, 不能有比 t 更大的数 f 。
- 因为 c 如果压入 a 栈, 则必须弹出 t , f 会把 t 卡住。
- 而 c 如果压入 b 栈, 当前 b 栈栈顶 $\geq a$ 栈栈顶, 所以也能用 t 判断

图论

- 图论题目在 $NOIP$ 中也多次出现。
- 但是一般都不是单独考察图论算法及相关内容，而是结合其他考点（如 DP ，贪心，数据结构）进行考察。



图论

- 主要考点:
- 图的连通性 (*tarjan*: 点双、边双、强连通)
- 最短路 (*SPFA*已经死了)。
- 生成树 (以及计数)
- 拓扑排序
- 网络流: 最大流*Dinic*, 最小费用最大流。(会打板子即可)
- 费用流有时候对你懒得思考的*DP*题有奇效。

图论

- *NOIP2013*: 华容道
- 棋盘上除了关键棋子和空白格子以外所有东西都是无差别的。
- 定义 $d[i][j][0 \dots 3][0 \dots 3]$ 表示关键棋子在 (i, j) , 空白格子相对关键棋子的位置, 关键棋子要去的位置相对关键棋子的位置。
- *bfs* 预处理一下。
- 每次询问, 先把空白格子移动到关键棋子旁边, 然后 *spfa*。
- 这种题目考场上建议写暴力吧。

图论

- *NOIP2015*: 运输计划
- 最大值最小，一眼二分答案。
- 如何验证？考虑所有长度超过当前答案的路径，它们的交集上面一定有一条路径需要被改造。
- 且被改造的路径长度应大于它们超出长度的最大值。
- 这题做完了。

图论

- *NOIP2008*: 双栈排序
- 存在逆序关系的两个数不能放在同一个栈中。在它们之间连边。
- 二分图染色判断是否有解，然后构造方案十分显然。
- （比贪心简单多了2333）

图论

- *NOIP2009*: 最优贸易
- *tarjan*缩环。
- 然后拓扑排序，维护链上的最低价格，最高价格，做差更新答案。



图论

- *NOIP2000*: 方格取数
- k 取方格数? 费用流板子题。
- 每个点拆成两个, 连费用正, 流量1的边, 费用0, 流量inf 的边。
- 跑最大费用流即可。
- *NOIP2008*: 传纸条
- 和上一题思路相同点拆成两个, 只连费用为输入值, 流量1的边。
- 跑最大费用流。

数学

- CSP范围内的数学题知识有：
- 取模，同余，逆元， $exgcd$ ，中国剩余定理，合并同余方程
- 素数判定（根号， $millier - rabin$ ），线性筛法
- 基础组合数学（组合数，卡特兰数，第一、二类斯特林数，斯特林反演），置换群，染色，Polya定理，Burnside引理等。
- 数论函数（ μ, φ, id ）等，基础的莫比乌斯反演
- 期望，线性方程组，高斯消元
- 进制转化，高精度， FFT, NTT, FWT (及 FWT 的构造)等
- 划掉的部分可以直接选择暴力滚粗了2333

数学

- *NOIP2011*: 聪明的质监局
- 显然 Y 随 w 增大单调不增。
- 所以 $|Y - S|$ 随 w 增大先减后增。
- ~~大力三分 w 即可。~~
- 二分 w , 求出使得 $Y \leq S$ 的最大的 w , 然后带入 w 和 $w + 1$ 进行运算取min。

数学

- *NOIP2004*: 虫食算
- 从低位到高位暴搜每个字母代表的数字。
- 低位不符合表达式则剪枝。



数学

- *NOIP2018*: 填数游戏
- 考虑找规律。
- $n = 1$ 时, 显然答案为 2^m 。
- $n = 2$ 时, 答案为 $4 * 3^{m-1}$ 。
- $n = 3$ 时, 答案为 $112 * 3^{m-3}$ 。
- 65分到手。之后你可以继续找, 也可以弃疗了……
- (这题当年难倒集训队2333)



数据结构

- 数据结构是 $NOIP$ 中的经典考题。
- 主要锻炼选手的代码能力（并满足出题人毒瘤欲望）
- 需要熟练掌握的数据结构有：
- 二叉堆，左偏树；
- 线段树(动态开点/可持久化)、树状数组、轻量级平衡树(非旋转 $treap$ ，替罪羊树)，重量级平衡树($Splay$)， LCT ， ETT ；
- 根号数据结构： $KD-Tree$ ，可持久化数组。
- 单调队列，对顶堆， $pbds$ 等科技。
- 数据结构题和模拟题一样，如果别人都 AC 了就你挂分了，那你凉凉了。

数据结构

- *NOIP2016*: 蚯蚓
- 一个变量维护增加值。
- 初始的降序排序，开三个队列，维护初始的，第一段，第二段。
- 显然先切的分成两段也比后切的分成的两段长。
- 所以三个单调队列就搞定了。
- 想想当年多少写二叉堆暴力的铁憨憨，其实直接 *pbds* 配对堆就能……

数据结构

- *NOIP2017*: 列队
- 动态开点线段树或平衡树大力维护每一行和最后一列。
- 线段树/平衡树中每个节点代表一段区间。
- ~~(别说了, 我当年就是个Splay没调出来的菜狗。这有何难.jpg)~~

数据结构

- *NOIP2016*: 天天爱跑步
- 这题有非数据结构的做法，但是这里我想讲一个利用数据结构的暴力做法。
- 路径可以拆成上行/下行两段。
- 上行路径上每个人到路径上的点*i*的时间可看作 $t_0 - dep[i]$ 。
- 下行路径上每个人到路径上的点*i*的时间可看作 $t_1 + dep[i]$ 。
- 而观察员观察的时间，可看作 $a - dep[i]$ 或 $b + dep[i]$ 。
- 按照加减分类，按照常数排序，转换为树链加，单点查问题。
- （当然也可以不转换，对一条链进行操作时直接主席树区间加）
- （其实打标记，开一个桶，在遍历整棵树的时候维护一下就好了）

字符串

- 字符串题目在 $NOIP$ 中出现不多，相对难度不大。
- （不过今年不一定）
- 怕是难度全在 IO 。
- 需要掌握的算法有：
- KMP (深入理解)， AC 自动机。
- 后缀数据结构：后缀数组，后缀自动机，后缀平衡树，后缀树。
- 为什么不试试神奇的哈希呢？

字符串

- *NOIP2015*: 子串
- 注意到相邻的两个字符串不会被强行拼成一个。
- $f[i][j][k]$ 表示考虑利用 a 串前 i 位, b 串前 j 位已经匹配, 子串数量为 k , 方案数, $sum[j][k]$ 表示 $f[i][j][k]$ 在 i 维度上的前缀和。
- 如果 $b[i] == a[j]$ 的话, 向前枚举本次匹配串的左端点, 进行转移。
- 发现维度 i 没什么用, 滚动数组滚掉。
- 最终答案即为 $sum[len(b)][input_k]$ 。

字符串

- *NOIP*: 统计单词个数
- $f[i][j]$ 表示前 i 个字符, 分 j 段, 最优解。
- 枚举前一段结束位置, 中间串的匹配数量暴力计算。



搜索

- 搜索是指：从已知状态的临域出发，寻找未访问过的新状态或更优状态的方法。
- 常见的搜索方法有： dfs 和 bfs 。
- 常用于图论题和动态规划题。
- 以及各种暴力（万物皆可 $dfs.jpg$ ）
- 需要掌握的技巧有：
 - $a*$, $ida*$, 蒙特卡洛搜索。

搜索

- *NOIP*2015：斗地主
- 先暴搜顺子，之后贪心：
- 先四带两对，四带两单，四带一对，三带二，三带一。
- 之后每种分开出。
- （记得当年ZTY大佬高一上学期12月就水掉了这题）

搜索

- *NOIP2011*: Mayan游戏
- 暴搜方案，然后手动模拟消除、下落。
- 代码量不大。这有何难？
- 对每一层开一组数组或把状态存储为一个类。
- 因为可行解中消除的次数不会太多，所以复杂度正确。

搜索

- NOIP2009：靶形数独
- 对每行每列每个 3×3 幻方状态压缩。
- 暴力搜索每个空白位置。通过存储的状态计算出每个位置能填的数字有那些，并保存、更新状态。
- 不需要对权值进行剪枝。只对可填的数字进行剪枝就够了。

国际惯例的

谢谢大家

