A 班 7.13 动态规划 题解

题目名称	关键词
积木大赛	动态规划、栈
表达式求值	动态规划、贪心
文本编辑器	区间 DP、KMP
移球游戏	树型 DP、换根 DP、矩阵快速幂

T1 积木大赛

Source：JOI 2013 Final T2「IOI 列車で行こう」

算法 1

暴搜（？）暴力枚举扔掉的积木以及剩下的出栈顺序。

期望得分： $0 \sim 20$。

算法 ???

各种奇怪的贪心。没有子任务很难卡掉。

期望得分： $0 \sim 100$。

算法 2

考虑 DP。注意到题目的要求是选择两段后缀，栈序构造最长的序列。

枚举后缀的时间消耗太大。记 $g_{i,j}$ 表示第 1 个栈 $[i,n]$，第 2 个栈 $[j,m]$ 的按题意取积木形成的最长 `OIOIOI...` 型序列。$f_{i,j}$ 表示第 1 个栈 $[i,n]$，第 2 个栈 $[j,m]$ 的按题意取积木形成的最长 `IOIOIOI...` 型序列。

记第 1 个栈为 $a[\,]$，第 2 个栈为 $b[\,]$ 那么有转移：

$g_{i,j} \leftarrow g_{i+2,j} + 2$，满足 $a_i =$ `O`，$a_{i+1} =$ `I`

$g_{i,j} \leftarrow g_{i,j+2} + 2$，满足 $b_j =$ `O`，$b_{j+1} =$ `I`

$g_{i,j} \leftarrow g_{i+1,j+1} + 2$，满足 $a_i =$ `O`，$b_j =$ `I`，交换也成立

$f_{i,j} \leftarrow g_{i+1,j} + 1$，满足 $a_i =$ `I`

$f_{i,j} \leftarrow g_{i,j+1} + 1$，满足 $b_j =$ `I`

最终答案 $\max\{f_{i,j}\}$

时间复杂度 $\mathcal{O}(nm)$，期望得分： 100 。

T2 表达式求值

Source：ARC066E Addition and Subtraction Hard

算法 1

暴搜，期望得分：10 。

算法 2

注意到只需要在减号后加左括号。记 $f_{i,j}$ 表示当前做到第 i 个数，还有 j 个左括号未被右括号匹配的表达式最大值。
若第 $i+1$ 个数字前面是减号，则：

1、什么都不做，$f_{i+1,j} \leftarrow f_{i,j} + (-1)^{j+1}a_i$

2、减号前添上右括号，$f_{i+1,j-1} \leftarrow f_{i,j} + (-1)^{j}a_i$

3、减号后添上左括号，$f_{i+1,j+1} \leftarrow f_{i,j} + (-1)^{j+1}a_i$

数字前是加号的情况类似：

1、什么都不做，$f_{i+1,j} \leftarrow f_{i,j} + (-1)^{j}a_i$

2、加号前添上右括号，$f_{i+1,j-1} \leftarrow f_{i,j} + (-1)^{j-1}a_i$

最终答案 $\max\limits_{0\le i<n}\{f_{n,i}\}$，因为可以在最后添上任意多的右括号（中间也可以，但没必要）。

注意边界情况。

时间复杂度 $\mathcal{O}(n^2)$ ，期望得分：60 。

算法 3

注意到我们并不关心有多少个左括号未匹配，只关心左括号个数的奇偶性，同时还需记录有无未匹配的左括号。

DP 式改为 $f_{i,0/1/2}$，分别表示没有未匹配的左括号、奇数个未匹配的左括号、偶数（>0）个未匹配的左括号。转移类似。

时间复杂度 $\mathcal{O}(n)$ ，期望得分：100 。

事实上，任意方案均存在一种等价方案，满足任意位置未匹配的左括号数目不超过 2。在特定位置添右括号来改变奇偶性即可。

算法 4

事实上不需要 DP。

假设在某个减号后的位置添上左括号，那么自此之后的第一个减号开始，每个数都可以满足对答案贡献为正。把一个减号连着后面若干连续加号的数括起来就好了，例如：

$$a_1+a_2-(a_3+a_4-(a_5+a_6+a_7)-(a_8+a_9+a_{10})-a_{11})$$

枚举第一个左括号的位置，通过预处理前后缀和等方法 $\mathcal{O}(1)$ 计算贡献。注意别漏了不加任何括号的情况。

时间复杂度 $\mathcal{O}(n)$ ，期望得分：100 。

T3 文本编辑器

Source：JOISC2022D2T1 复制粘贴 3

算法 1

暴搜，期望得分： 12 。

算法 2

注意到若最优方案进行了一次剪切 + 粘贴操作，那么粘贴次数一定是尽可能多的，得到一个形如 `..Y....Y.YY..Y...` 的字符串，中间其他字符用操作 1 补上。

对于所有字符均为 `a` 的情况，记 f_i 表示生成 i 个 `a` 的最小代价，则：

$$f_i \leftarrow f_{i-1} + A$$

$$f_i \leftarrow \min_{j<i}\left\{f_j + B + C \times \left\lfloor \frac{i}{j} \right\rfloor + A \times (i \bmod j)\right\}$$

时间复杂度 $\mathcal{O}(n^2)$ ，期望得分： 12，拼上算法 1 共得分 24 。

算法 3

由算法 2 的启发，考虑区间 DP。

记 $f_{l,r}$ 表示生成区间 $[l,r]$ 字符的最小代价。

有转移 1：$f_{l,r} \leftarrow \min(f_{l+1,r}, f_{l,r-1}) + A$ 。

从大到小枚举区间的左端点 l，然后从小到大枚举右端点 r，枚举串 $[l,r]$ 的 border 长度贡献答案。假设一个 border 长度为 k，在 $[l,r]$ 不重叠的最大出现次数为 cnt ，则：

转移 2：$f_{l,r} \leftarrow f_{l,l+k-1} + B + cnt \times C + (l + r - 1 - cnt \times k) \times A$

可以在枚举 r 的同时对后缀串 $[l,n]$ 做 KMP ，并实时记录 las_i 表示长度为 i 的 border 上一次出现的位置（便于求上文的 cnt），以实现上述操作。

时间复杂度 $\mathcal{O}(n^3)$ ，但是常数很小（只有全部字符相同能卡满）。期望得分： $60 \sim 100$ ，拼上算法 2 后，期望得分： $72 \sim 100$ 。

算法 4

注意到，因为有转移 1，我们考虑一个 border 在区间内不重叠出现的所有末尾位置，转移 2 只需要在这些位置上处理即可。

用 KMP $\mathcal{O}(n^2)$ 预处理 $nex_{i,len}$ 表示子串 $[i, i + len - 1]$ 在串 S 中下一次出现的起点位置。

采用主动转移，这样 $f_{l,r}$ 至多贡献 $\frac{n-l+1}{r-l+1}$ 次，用调和级数证明时间复杂度是 $\mathcal{O}(n^2 \ln n)$ 级别。期望得分： 100 。

T4 移球游戏

Source：CEOI2020D1T3 Star Trek

算法 1

对于 $N=2$：容易发现不管怎么连边最终结果都是小 A 获胜，而任意两个树型图连边方式有 4 种，所以答案是 4^D 。

时间复杂度 $\mathcal{O}(\log\ D)$ ，期望得分： 8 。

算法 2

对于 $D=1$，$N\le 1000$:

先考虑 $D=0$ 的情况。

如果在 i 的子树内，从 i 出发先手必胜，则称 i 为必胜点，反之为必败点。那么 $D=0$ 就是询问结点 1 是否是必胜点。

那么记 p_i 表示 i 是否为必胜点，树型 DP。

显然 $p_i=0$ 当且仅当结点 i 的所有儿子都是必胜点，否则 i 一定可以走到某个必败点儿子，让对方必败。

所以 $p_i=\bigvee_{x\in son_i}\ [p_x=\ 0]$

那么 $D=1$ 呢？考虑新复制的树上的点连到原始树，如果从某个点出发先手必胜，这个点不管连到原始树的哪里都相当于空点（肯定不会走），不会对结果产生影响。

再考虑出发先手必败的点，记 $size_i$ 表示 i 的子树大小，g_i 表示一个必败点连到 i 的子树内，使得 i 为必胜点的方案数，然后树型 DP。

若 i 有两个及以上的儿子为必败点，那么不管怎么连边 i 都一定是必胜点， $g_i=size_i$；

若 i 只有一个儿子 x 是必败点，那么答案为连边仍使这个儿子是必败点的方案数，即 $g_i=size_i-g_x$；

若 i 所有儿子都是必胜点，那么答案为连边使它们中任意一个变成必败点的方案数，即 $g_i=size_i-\sum_{x\in son_i}g_x$ 。

再处理出 " $1\le i\le n$ ，以 i 为根时，p_i为必胜点 " 的数量，记为 s_1；同理 s_2 表示 " $1\le i\le n$ ，以 i 为根时，p_i为必败点 " 的数量（n 次树型 DP）。

那么答案为必胜点、必败点连到初始树，$p_i=1$ 的方案数和，即 $p_1\times n\times s_1+g_1\times s_2$ 。

时间复杂度 $\mathcal{O}(n^2)$ ，期望得分： 28 ，结合前面算法共 36 分。

算法 3

对于 $N\le 1000$，$D\le 10^5$:

处理出 $1\le i\le n$ ，以 i 为根时， g_i 的值（n 次树型 DP），令 $s_3=\sum g_i$, $s_4=n^2-s_3$ 。

考虑 D 个新树型图从后往前 DP。记 $f_{i,1}$ 表示“做到第 i 个树型图，向第 $i-1$ 个树型图连出一个必胜点的方案数”； $f_{i,0}$ 表示“做到第 i 个树型图，向第 $i-1$ 个树型图连出一个必败点的方案数”。

如果第 $i+1$ 个树型图连出必胜点（相当于空点），则不会影响结果，第 i 个树型图所有点接这条边结果都不变，则向第 $i-1$ 棵树连一个必胜点的方案有 $n\times s_1$ ，必败点 $n\times s_2$ 种。

如果第 $i+1$ 个树型图连出必败点，第 i 个树型图的点 x 必胜方案为 g_x ，必败方案为 $n-g_x$ ，则向第 $i-1$ 棵树连一个必胜点的方案有 s_3 种，必败点 s_4 种。

所以， $f_{i,0}=f_{i+1,1}\times s_2\times n+f_{i+1,0}\times s_4$, $f_{i,1}=f_{i+1,1}\times s_1\times n+f_{i+1,0}\times s_3$ 。边界 $f_{d,1}=s_1$, $f_{d,0}=s_2$ 。

答案即为 $f_{1,1}\times p_1\times n+g_1\times f_{1,0}$ 。

时间复杂度 $\mathcal{O}(n^2+d)$ ，期望得分： 40 ，结合前面算法共 48 分。

算法 4

对于 $N\le 10^5$，$D\le 10^5$:

有没有办法在 $\mathcal{O}(n)$ 时间内处理出 $1\le i\le n$,以 i 为根时 p_i , g_i 的值呢？考虑换根 DP。

记 up_i 表示结点 i 向上走一步到 i 的父亲 fa_i后，fa_i 是否为必胜点（相当于以 fa_i 为根的树，剔除子树 i 后，fa_i 是否为必胜点）；

h_i 表示结点 i 向上走一步到 i 的父亲 fa_i后 ,一个必败点连到 fa_i 的子树内，使得 fa_i 为必胜点的方案数（相当于以 fa_i 为根的树，剔除子树 i 后的 g_{fa_i} ）。

则再一遍 dfs ，每次相当于把一颗根节点为 fa_i ，大小为 $n - size_i$ 的向点 i上方生长的子树塞给结点 i ，用 up_i , h_i 更新 p_i , g_i 的值。

up_i , h_i 的转移类似 p_i , g_i :

up_i 从 up_{fa_i} 与 fa_i 的其他儿子节点的 p 值转移，即 $up_i = [up_{fa_i} = 0] \ \vee \left(\bigvee_{x \neq i \wedge x \in son_{fa_i}} [p_x = 0] \right)$ 。

h_i 从 h_{fa_i} 与 fa_i 的其他儿子节点的 g 值转移，枚举 fa_i 除 i 以外 p 值为 0 的儿子个数，记为 cnt ;

如果 $up_{fa_i} = 0$，那么 cnt 需要再增加 1（因为以 fa_i 为根往上长的树被看作子树了）；

然后像算法 2 中那样分 " $cnt = 0$ " , " $cnt = 1$ " , "$cnt \geq 2$" 三种情况讨论 h_i 的值。

做一次换根 DP，$\mathcal{O}(n)$ 搞定。总时间复杂度 $O(n + d)$ ，期望得分：64 ，结合前面算法共 72 分。

算法 5

对于 $N \leq 10^5$，$D \leq 10^{18}$

我们发现 $f_{i,0}$, $f_{i,1}$ 的转移只和 $f_{i+1,0}$ $f_{i+1,1}$ 以及 s_1, s_2, s_3, s_4 有关。

于是用矩阵快速幂优化，构造矩阵 $\begin{bmatrix} s_4 & s_2 \times n \\ s_3 & s_1 \times n \end{bmatrix}^{d-1} \times \begin{bmatrix} f_{d,0} \\ f_{d,1} \end{bmatrix}$ 即可求出答案。

时间复杂度 $\mathcal{O}(n + \log\ d)$, 期望得分：100 。