A 班 7.14 动态规划 题解

题目名称	关键词
见面	图论、拓扑排序
道法自然	贪心、构造
选拔赛	概率期望 DP，组合数学
铁路规划	生成树、最短路、状压 DP

T1 见面

Source：ABC139E League

题目给了一张完全图，每天我们可以选择若干条不共点的边，但是有一些限制条件，某条边被选择之前需要先选择若干条边。

对于题目所给的这张完全图，把每条边看作一个新图中的点（那么，新图有 $\frac{n(n-1)}{2}$ 个点）。形如 (1,2) 见面要在 (1,3) 见面之前 这样的条件，把新图中的边 (1,2) 对应的点向边 (1,3) 对应的点连一条有向边。

这样我们得到了一张新图，有解当且仅当它是个 DAG。记 f_u 表示点 u 对应边被选中的最小天数。跑一遍拓扑排序，$f_v=\max_{(u,v)\in E}\{f_u+1\}$。

时间复杂度 $O(n^2)$，期望得分：100 。

T2 道法自然

Source：ARC123C 1, 2, 3 - Decomposition

算法 1

暴搜，期望得分：20 。

算法 2

先考虑判定性问题，钦定有 i 个优美的数，判断是否能组成 x。

每一位上，能组成的数字是 $[i,3i]\pmod{10}$ 。进位也需要考虑。

从低到高考虑。倘若某一位出现不能组成的数字，记为 p（$0\le p\le 9$）。若 $p<i$，那就扔掉一个数，组成的区间变成 $[i-1,3(i-1)]\pmod{10}$，以此类推直到 p 在区间内。若 $p>3i$，则无论如何都不能成功，说明不存在 i 个优美的数可以组成 x。

如果某个时候，扔掉一个数或使得这一位的和 $>=10$ 而向高一位进位都可以满足这一位的要求，可以简单证明第二种选择一定不劣。那么决策唯一，贪心判断即可。

注意到 5 个优美的数，在每一位上能组成的区间是 $[5,15]\pmod{10}$，$0\sim 9$ 均能取到。由此易证任何数都可以由至多 5 个优美的数组成。

枚举答案，贪心判断。时间复杂度 $\mathcal{O}(T\log_{10}n)$，期望得分：100 。

T3 选拔赛

Source：AGC060C Large Heap

题意：在所有满足 $\forall 1 \le i \le n$，$p_i > p_{2i} \land p_i > p_{2i+1}$ 的排列中任选一个，求 P_{2^A}>$P_{2^{B+1}-1}$ 的概率。

算法 1

枚举所有可能的排列，期望得分 10 。

算法 2

情况仅有 9 种，运用组合数学的知识手算结果，期望得分 20，结合前面部分共得分 30 。

算法 3

考虑这个排列是怎么形成的。回到题面那棵满二叉树，把数字从大到小依次填入树中，一个点可以填数当且仅当它的父节点已经填上数字了。

实际上就是（设根深度为 0）某个深度为 A 的点 u 和深度为 B 的点 v，u 拓扑序小于 v 的概率。

记 $f_{i,j}$ 表示仅考虑根节点到 u,v 的两条链，第一条链填到深度为 i，第二条链填到深度为 j，方案数与总方案数的比值。

设深度 $i+1$ 的点子树大小为 x，深度 $j+1$ 的点子树大小为 y，则：

$f_{i+1,j} \leftarrow f_{i,j} \times \frac{\binom{x-1+y}{x-1}}{\binom{x+y}{x}}$，组合数之比可化简为 $\frac{x}{x+y}$

最后 $ans \leftarrow \displaystyle\sum_{j<B} f_{A-1,j} \times \frac{size_A}{size_A + size_{j+1}}$，$size_A$ 表示深度为 A 的点子树大小。

时间复杂度 $\mathcal{O}(n^2)$（预处理逆元），期望得分：100 。

T4 铁路规划

Source：CF1149D Abandoning Roads

算法 1

搜出所有的最小生成树一一判断，期望得分 15 。

算法 2

先考虑仅有 a 边构成的图，形成若干连通块，那么最小生成树中一定是：块内的点用 a 边连接，块与块之间的用 b 边连接。

并查集处理连通块，按以上规则做单源最短路。注意通过 b 边到达的连通块不能重复，否则生成树就成环了。所以状压记录通过 b 边到达过的连通块。

若用 Dijkstra，时间复杂度 $O(2^n m \log(2^n m))$，期望得分：35 。

算法 3

注意到，连通块大小 ≤ 3 时，内部的点至多通过两条 a 边到达，从块外走至少要两条 b 边，一定更劣，最短路不会更新到。

所以只需记录大小 ≥ 4 的连通块。使用 Dijkstra 的时间复杂度为 $O(2^{\frac{n}{4}} m \log(2^{\frac{n}{4}} m))$，期望得分：$100$ 。

附：原定的 T4 题解

小 J 的饮料

Source：ARC066F Contest with Drinks Hard

算法 1

如果没有修改，容易想到用截距优化 DP。

记 $f\left(i\right)$ 表示从前往后进行到第 i 关时可以获得的最高积分。可以写出转移

$$f\left(i\right)=\max\left\{\max_{j<i}\left\{f\left(j\right)+\frac{\left(i-j\right)\left(i-j+1\right)}{2}-\sum_{k=j+1}^{i}T_k\right\},f\left(i-1\right)\right\}$$

直接做对于每组询问是 $O\left(n^2\right)$ 的。

记 S_i 表示 T_i 的前缀和，即 $S_i=\sum_{j=1}^{i}T_j$，我们可以将上式中里面的 $\max$ 改写成：

$$\begin{aligned}&\max_{j<i}\left\{f\left(j\right)+\frac{\left(i-j\right)\left(i-j+1\right)}{2}-T_i+T_j\right\}\\&=\max_{j<i}\left\{f\left(j\right)+\frac{i\left(i+1\right)}{2}-ij+\frac{j\left(j-1\right)}{2}-T_i+T_j\right\}\\&=\frac{i\left(i+1\right)}{2}-T_i+\max_{j<i}\left\{f\left(j\right)-ij+\frac{j\left(j-1\right)}{2}+T_j\right\}\end{aligned}$$

将其对偶成斜率为 i 的直线分别过每个满足 $j<i$ 的点 $\left(j,f\left(j\right)+\frac{j(j-1)}{2}+T_j\right)$ 时与 y 轴截距的最大值。

由于横坐标及直线斜率均具有单调性，每组询问的复杂度降低到了 $O\left(n\right)$。

每次修改做一遍 DP，时间复杂度 $O(n^2)$，期望得分：64 。

算法 2

为下文叙述方便，先推一下反向 DP 的做法。

记 $g\left(i\right)$ 表示从后往前进行到第 i 关时的最高积分。同样可以写出转移

$g\left(i\right)=\max\left\{\max_{j>i}\left\{g\left(j\right)+\frac{(j-i)(j-i+1)}{2}-\sum_{k=i}^{j-1}T_k\right\},g\left(i+1\right)\right\}$

内层的 $\max$ 经变形可以改写成

$\frac{i(i-1)}{2}-S'_i+\max_{j>i}\left\{g\left(j\right)-ij+\frac{j(j+1)}{2}+S'_j\right\}$

其中 $S'_i=\sum_{j=i}^{n}T_j$。

上述 $f\left(i\right)$ 和 $g\left(i\right)$ 的边界条件分别为 $f\left(0\right)=0$ 和 $g\left(n+1\right)=0$。

接下来考虑如果有修改应该如何计算。

显然，最终的答案可能有两种情况：

1. 放弃挑战第 P_i 关，此时答案可以由 $f\left(P_i-1\right)+g\left(P_i+1\right)$ 得到；
2. 挑战第 P_i 关，我们需要计算出 $\max\limits_{[l,r]\ni P_i}\left\{f\left(l-1\right)+\frac{\left(r-l+1\right)\left(r-l+2\right)}{2}-\sum\limits_{j=l}^{r}T_j+g\left(r+1\right)\right\}$，答案可以由其加上 $T_{P_i}-X_i$ 得到。

第 1 种经预处理可以对每组询问 $O\left(1\right)$ 查询。

我们现在考虑如何计算第 2 种的答案。（假设记其为 $h\left(i\right)$）

需要注意的是，对于某一关 i，必须挑战这一关的最优挑战方法中，包含 i 的区间很有可能不以 i 作为其左右端点。

最直接的处理手段是枚举左右端点 $[l,r]$，然后更新所有 $x\in[l,r]$ 的 $h\left(x\right)$。但是这样做的复杂度太大。

我们先考虑固定左端点 l，然后不断右移右端点，得到 $h'(x)$ 表示当右端点为 x 时包含 $[l,x]$ 的最优解。

这样我们可以用 $h'(i)$ 的后缀最大值来更新 $h(i)$。复杂度为 $O\left(n^2\right)$。

显然，如果需要枚举一个左端点并固定，复杂度难以做到比 $O\left(n^2\right)$ 更优秀。

我们考虑如果不固定左端点会发生什么。

假设我们一开始设当前的左右端点 $l=r=x_0$，然后每次右移右端点时让 l 移动到当前的最优决策点。显然，单调性在这里仍然满足。

这样，我们就可以对于 $x \geq x_0$ 的所有端点更新包含 x 的区间左端点 $\leq x_0$ 时的最优解。

类似地，每次左移左端点时让右端点移动到当前的最优决策点，可以对所有 $x \leq x_0$ 的区间更新包含 x 的区间右端点 $\geq x_0$ 时的最优解。

接下来还需要处理的就是左右端点均位于 $[1,x_0)$ 内或均位于 $(x_0,n]$ 内的端点，即不包含 x_0 的所有区间。

我们可以递归分治求解。

显然，每次取 $x_0=\frac{x_{\min}+x_{\max}}{2}$ 能使复杂度最优。这里 $x_{\min}$ 和 $x_{\max}$ 指的是当前处理的区间左右端点。

时间复杂度 $O\left(N\log N+M\right)$，期望得分：100 。