

【样例 3】

见选手目录下的 `transmit/transmit3.in` 与 `transmit/transmit3.ans`。该样例满足测试点 3 的限制。

【样例 4】

见选手目录下的 `transmit/transmit4.in` 与 `transmit/transmit4.ans`。该样例满足测试点 20 的限制。

【数据范围】

对于所有的测试数据, 满足 $1 \leq n \leq 2 \times 10^5$, $1 \leq Q \leq 2 \times 10^5$, $1 \leq k \leq 3$, $1 \leq a_i$, $b_i \leq n$, $1 \leq s_i$, $t_i \leq n$, $s_i \neq t_i$ 。

测试点	n	Q	k	特殊性质
1	≤ 10	≤ 10	$= 2$	是
2			$= 3$	
3	≤ 200	≤ 200	$= 2$	
4, 5			$= 3$	
6, 7	≤ 2000	≤ 2000	$= 1$	否
8, 9			$= 2$	
10, 11			$= 3$	
12, 13	$\leq 2 \times 10^5$	$\leq 2 \times 10^5$	$= 1$	
14	$\leq 5 \times 10^4$	$\leq 5 \times 10^4$	$= 2$	是
15, 16	$\leq 10^5$	$\leq 10^5$		否
17, 18, 19	$\leq 2 \times 10^5$	$\leq 2 \times 10^5$		
20	$\leq 5 \times 10^4$	$\leq 5 \times 10^4$	$= 3$	是
21, 22	$\leq 10^5$	$\leq 10^5$		否
23, 24, 25	$\leq 2 \times 10^5$	$\leq 2 \times 10^5$		

特殊性质: 保证 $a_i = i+1$, 而 b_i 则从 $1, 2, \dots, i$ 中等概率选取。

CCF CSP-J/S 2022 第二轮提高级解题报告

“假期计划”解题报告

Google 苏黎世 钟诚

【概述】

图的邻接矩阵、邻接表的存储方法, 和图的深度、宽度优先遍历算法是信息学竞赛中较

为基础的知识点,不超过 NOI 大纲中入门级的范围。本题正是基于这些基础的数据结构和算法,考查选手对此掌握的扎实程度。同时,在满分解法中,还需要进行两处预处理,并在此结果的基础上,合理地对枚举循环的层数进行剪枝,有一定的思维深度。

【题目大意】

输入一张无向图,其中 1 号点是家, $2 \sim n$ 号点都是景点,每条边都代表双向直达的线路。现从家出发去 4 个不同的景点游玩,完成 5 段行程后回家:家 \rightarrow 景点 A \rightarrow 景点 B \rightarrow 景点 C \rightarrow 景点 D \rightarrow 家,要求每段行程最多转车 k 次。每个景点都有一个分数,求景点的分数之和最大值。

【解法一】

对于 $k=0$ 的数据,也就是不能转车,我们可以直接使用 4 个 for 循环,来枚举所有可能的景点访问顺序。为了便于判断任意两点之间是否有边,此时无向图宜用邻接矩阵存储。时间复杂度 $O(n^4)$ 。

【解法二】

在解法一的基础上,当 $k>0$ 时,仍然使用 4 个 for 循环,枚举所有可能的景点访问顺序。在此之外,另写一个函数,判断两个点之间的距离是否小于等于 $k+1$ 。对于每种可能的顺序,调用 5 次该函数,判断每段行程是否符合规定。时间复杂度 $O(n^5)$ 。

【解法三】

在解法二的基础上,做一定的预处理。我们用宽度优先遍历算法,预先计算出与每个点距离小于等于 $k+1$ 的点,从而形成一张新的无向图。然后,再用 4 个 for 循环,枚举所有可能的景点访问顺序。此预处理相当于代替了解法二中的那个函数,节约了时间。预处理时间复杂度 $O(nm)$, for 循环时间复杂度 $O(n^4)$,考虑到 $m \leq O(n^2)$,总的时间复杂度 $O(n^4)$ 。

【解法四】

在解法三的基础上,只枚举景点 B、景点 C,然后再用一重循环枚举剩下所有的景点,看每个景点是否可以作为行程中的景点 A、景点 D,而不单独分别枚举景点 A、景点 D。如果能作为景点 A 里分数最高的,和能作为景点 D 里分数最高的是不同的点,那直接选取它们,否则再考虑次高即可。时间复杂度 $O(n^3)$ 。

【解法五】

在解法四的基础上,再做一个预处理。在新的无向图中,预先找出每个点和家的共同通达的景点中,分数最高的 3 个景点。这样,在枚举景点 B、景点 C 的时候,就不再用一重循环找景点 A、景点 D 了,而可以直接在分数最高的 3 个景点中寻找。之所以要预处理保存 3 个分数最高的景点,是为了避免后续枚举时与已有的景点重复。此预处理的复杂度是 $O(n^2)$,枚举景点 B、景点 C 的复杂度是 $O(n^2)$,加上解法三中预处理的 $O(nm)$,所以总的时间复杂度是 $O(n^2+nm)$ 。

【总结】

本题源于基础，只要对图的最基本的存储数据结构和最基本的遍历算法有扎实地掌握，就能得到高分；但要得到满分，还需要有一定的优化和技巧。因此作为提高组的第一题，难度安排合理，分数梯度适当。

“策略游戏” 解题报告

清华大学 张艺缤

【概述】

本题是一道比较有趣的“博弈论”题目，题目背景是一个两人博弈游戏。但作为 CSP-S 的第二题，本题并不涉及较难的博弈论知识，主要考查贪心算法和 ST 表的应用，以及一些分类讨论技巧，总体难度在 CSP-S 系列中属于中等。

【题目大意】

给定长为 n 的数组 A 和长为 m 的数组 B ，有 q 次询问，每次给定 l_1, r_1, l_2, r_2 。先手需要在 $l_1 \sim r_1$ 之间选择一个 i ，后手再在 $l_2 \sim r_2$ 之间选择一个 j ，则本轮游戏的得分为 $A_i \times B_j$ 。先手希望最大化得分，后手希望最小化得分，求在两人最优决策下的得分。

数据范围： $n, m, q \leq 10^5$ 。

【解法一】

按照题意进行模拟。

每次询问先在 $l_1 \sim r_1$ 之间枚举 i ，再在 $l_2 \sim r_2$ 之间枚举 j ，计算 $A_i \times B_j$ ，内层循环取最小值，外层循环取最大值。

复杂度 $O(nmq)$ ，期望得分 25 分。

【解法二】

进行第一层思考：当外层循环枚举 i 后，选择哪个 B_j 能使得 $A_i \times B_j$ 尽可能小？

容易发现需要根据 A_i 的正负进行分类讨论：当 $A_i \geq 0$ 时，应选择区间内最小的 B_j ；反之，当 $A_i < 0$ 时，应选择区间内最大的 B_j 。

于是问题转化为了：外层循环枚举 i ，再求出 $l_2 \sim r_2$ 区间内 B_j 的最大值或最小值。可以使用 ST 表实现 $O(1)$ 的区间求最值。

复杂度 $O(nq + m \log m)$ ，期望得分 60 分。

【解法三】

在解法二的基础上进行第二层思考：外层的 A_i 又该如何选择？

根据解法二, 容易发现本质不同的 A_i 选择只有非负和负数两种。

如果要选择 A_i 非负, 那么后手一定会选择区间内最小的 B_j , 记为 B_{\min} 。这时又可以分类讨论: 如果 $B_{\min} \geq 0$, 那么 A_i 可以选择非负数的最大值, 以使得乘积 $A_i \times B_{\min}$ 尽量大; 而当 $B_{\min} < 0$ 时, 就应当选择非负数的最小值(注意仍然需要为非负数, 因为这是这一段讨论的前提)。

同理, 如果要选择 A_i 为负, 那么后手会选择区间内最大的 B_j , 记为 B_{\max} 。为了最大化 $A_i \times B_{\max}$, 需要当 $B_{\max} \geq 0$ 时选择最大的负数 A_i , 否则选择最小的负数 A_i 。

最后的答案, 便是在选择非负数 A_i 和负数 A_i 中选择一种答案最大的。

在解法二的基础上, 我们还需要用 ST 表分别维护: A 数组的区间非负数最大值、非负数最小值、负数最大值、负数最小值。为方便实现, 可以分别把原来 A 数组中的负数(或非负数)全部替换成 inf (或 $-\text{inf}$)。

总复杂度 $O(n \log n + m \log m + q)$, 可以获得 100 分。

【总结】

本题作为一道带有博弈色彩的题目, 推理出合理的贪心策略是解题的关键。这需要选手准确理解题目所表述的内涵, 并具有至少两层的思考和推理能力, 最终将问题转化为求区间最值问题, 并使用数据结构加以维护。作为 CSP-S 的第二题而言, 具有一定的思维难度, 在代码实现上也有一定的细节和分类讨论需要处理, 对具有一等奖水平的选手有较好的区分作用。

“星战” 解题报告

北京大学 黄蔚尧

【概述】

本题以星际战争为题目背景, 主要考查内容为图论建模、基环树、随机哈希。本题需要选手理解基环树的图论性质和判定条件, 并根据题目所要求的动态变化选择合适的数据结构来回答询问。

【题目大意】

你需要维护一张 n 个点 m 条边有向图, 有 q 次操作:

1. 将一条边染白;
2. 将到达某个点 u 的所有边染白;
3. 将一条边染黑;
4. 将到达某个点 u 的所有边染黑。

初始时所有边均为黑色。每次操作后你均需要回答所有黑色的边是否构成原图的一个基环内向生成森林。

【解法一】

按照题意暴力模拟,判断是否是基环内向树等价于要求一个(弱)连通块中每个点都有出边,且这个连通块内点数和边数相同。这样单次操作 1 和 3 的时间复杂度为 $O(1)$, 单次操作 2 和 4 的时间复杂度为 $O(n)$, 回答询问的时间复杂度为 $O(n+m)$ 。总时间复杂度 $O(q(n+m))$, 可以获得 40 分。

【解法二】

可以发现:黑边构成原图的一个基环内向生成森林等价于每个点有且仅有一条出边是黑边。对于没有操作 2 和 4 的情况,我们可以直接维护每个点的黑色出边个数,进而维护有多少个点有且仅有一条出边是黑边。这样回答询问的时间复杂度降为 $O(1)$ 。总时间复杂度 $O(q+n+m)$ 。可以获得 50 分。

【解法三】

对于没有操作 4 的情况,容易发现一条边从黑色变成白色有两种情况:

1. 从初始时的黑色变成白色;
2. 这条边之前经过了操作 3 从白变黑,再经过一次操作 1 或者 2 从黑变白。

第一种情况至多进行 m 次。第二种情况发生次数至多为操作 3 的进行次数。

一条边从白色变成黑色发生的次数也至多为操作 3 的进行次数。

而操作 3 至多进行 q 次。也就是说我们仍然可以暴力维护修改操作,但是对于操作 2 我们需要避免无效操作(也就是黑色变成黑色、白色变成白色这种情况)。这个可以通过使用容器维护每个结点黑色入边的个数来完成。结合解法二,可以获得 60 分。

【解法四】

不妨设一条有向边 (u, v) 的权值为 u 。那么黑边构成原图的一个基环内向生成森林等价于所有黑边权值集合为 $\{1, 2, \dots, n\}$ 。但是如果直接使用容器维护当前所有黑边的权值集合,时间复杂度过高。

考虑哈希。我们只需要判断当前所有黑边权值集合的哈希值是否等于 $\{1, \dots, n\}$ 的哈希值即可。

设结点 u 的所有前驱结点构成集合 $S(u)$, 设当前黑边权值集合为 T , 那么题目中的操作可以转化为:

1. 在 T 中删除一个元素;
2. 在 T 中删除 $S(u)$ 中所有在 T 中的元素;
3. 在 T 中插入一个元素;
4. 在 T 中插入 $S(u)$ 中所有不在 T 中的元素。

因此我们需要满足结合律且有逆元的哈希模型，比如模质数正整数乘法群、非负整数异或群、取模加法群等等。采用多重哈希或者随机化的方式可以进一步降低出错的概率。在每个结点 u 上维护 u 的在 T 中的前驱集合的哈希值即可。时间复杂度 $O(n+m+q)$ ，可以通过本题。

“数据传输” 解题报告

北京大学 周雨扬

【概述】

本题以常见的计算机网络中的路径规划为题目背景，主要考查内容为最短路、动态规划、树上数据结构。本题需要选手掌握较高树上数据结构的设计与编写能力，并通过观察题目的性质设置出合理的状态，从而辅助最终算法的设计。考虑到整个分析过程的难度较高，部分分设计有意引导选手从较简单的最短路情形内情形出发，通过 k 的值引导选手思考更加一般的解法。总体而言题题目难度中等偏上。

【题目大意】

给定一颗大小为 n 的树，每个点有非负点权 v_i ，每条边长度都为 1。

Q 次询问，每次询问给定 s, t, k ，你需要找到序列 a_1, a_2, \dots, a_m ，使得：

- $a_1 = s, a_m = t$
- 对于任意 $1 \leq i < m$ ， a_i, a_{i+1} 在树上的距离不超过 k 。

这样的序列可能有很多，输出所有上述序列中 $\sum_{i=1}^m v_{a_i}$ 的最小值即可。

$$n, Q \leq 200\,000, k \leq 3$$

【解法一】

$n \leq 200$ 。

我们可以找到所有在树上距离不超过 k 的点对。点对之间的距离即为到达点的点权。此时原问题便可以是为在新图上的一个多源最短路问题。

可以简单地使用 floyd 算法进行解决，时间复杂度 $O(n^3)$ ，期望得分 20 分。

【解法二】

$n \leq 2000$ 。

观察原题目，我们不难发现可以对题目做如下转化：

- 从起点出发，每次可以移动到一个邻居。
- 可以给当前所在的点做标记，做标记有代价。起点终点必须做标记。
- 做标记后最多只能连续移动 k 次，并不在当前点做标记。

据此我们可以对原图做拆点最短路。假设 $f[i][j]$ 表示目前到达了结点 i ，从当前结点出发至多还能连续移动 j 次就必须做标记，所需要的最小花费。转移要么是移动到相邻点，要么是选择在当前点做标记。

对于每一个起点都可以做一次做短路，此时便可以预处理出所有的答案。如果使用堆优化 dijkstra 算法时间复杂度为 $O(n^2 \log n)$ ，期望得分 44 分。

【解法三】

$k=1$ 。

原问题此时简化为只有点权的树上最短路问题。利用树的特殊性质，只有点权的树上最短路问题等价于树上两点之间的路径的点权和。

此时原问题可以转化为更加简单的树链求和问题，时间复杂度 $O(n \log n)$ ，期望得分 16 分，结合解法一可以获得 36 分。

【解法四】

$k=2$ 。

由于树的性质，如果存在最优解跳到了树上路径以外的点 x 并进行标记，则必然 x 到树上最短路径距离不超过 1。假定其在 x 之前标记的最后一个点为 y ，之后标记的第一个点为 z ，则 y, z 树上最短路径距离必然不超过 2。因此我们可以让其直接从 y 行走到 z 再做标记，这样子得到的方案严格优于原方案。

因此 $k=2$ 时候的最短路必然只使用树上最短路径的点。此时原问题就变成了从路径上选点，使得相邻两个点至少选择一个，且点权和最小。

这是一个经典的树上动态规划问题，可以使用树链剖分+线段树维护动态规划转移矩阵进行解决。根据实现方法不同时间复杂度为 $O(n \log^2 n)$ 或 $O(n \log n)$ ，结合解法二、解法三可以获得 68 分。

【解法五】

基于解法四的基础上，观察 $k=3$ 的不同之处。使用类似的证明思路，我们可以证明最优解至多只会标记距离树上最短路径不超过 1 的结点。同时，题目样例也给出了使用这类结点的例子。

因此我们可以通过增加状态的方式，在解法四的基础上额外维护距离树上最短路径不超过 1 的结点所表示的状态与转移。具体维护方法较为复杂，这里略去。

加强后仍然是一个经典的树上动态规划问题，因此可以使用相同的方法解决。根据实现方法不同时间复杂度为 $O(n \log^2 n)$ 或 $O(n \log n)$ ，期望得 100 分。

【解法六】

另一种满分做法我们考虑基于解法二进行修改。

首先我们求出原树的点分树。考虑起点 s 和终点 t 在点分树上的最近公共祖先 L 。不难发

现在使用解法 2 的时候求解最短路的时候必然会经过 $f[L][0/1/2/3]$ 中的某一个状态。

因此我们不妨考虑倒着做, 求解从状态 $f[L][0/1/2/3]$ 出发, 到达点分树子树中每一个结点所需要的最小花费。最后我们只在 L 处合并点 s 和终点 t 分别到达状态 $f[L][0/1/2/3]$ 的最短路即可。

利用点分树的性质, 我们只需要求解 $O(n \log n)$ 对状态之间的最短路。同时利用树的性质, 我们甚至可以删除堆优化 Dijkstra 算法转而采用类似于树上 dp 的思路解决该问题。根据实现方法不同时间复杂度为 $O(n \log^2 n)$ 或 $O(n \log n)$, 期望得分 100 分。

【总结】

本题的思维难度中等, 但是实现一个简明易懂的算法, 需要跳出传统的树链剖分+线段树维护动态规划转移矩阵的思路。这对于不少选手来说是个挑战。赛后了解到绝大多数选手选择了以解法五为基础的算法。但相较来说, 我个人认为解法六在 k 较大的时候设计会明显比解法五简单。

本题目比赛时绝大多数选手受时间所限, 仅能实现解法一或者解法三。但是有部分实力较强的选手仍能解出该题。最终本题得分分布与预期相符。

总体来说, 本题难度在 CSP-S 的所有题目中最高, 但仍然低于省选等竞赛的试题, 作为 CSP-S 的压轴题较为适宜。