Solution

高闻远

福建省长乐第一中学

July 26, 2017

Outline

命令序列

命令序列

- 考虑用两个变量 x, y 记录执行完一段命令后机器人所在的坐标，若 $x = y = 0$ 则说明走回了原点

命令序列

- 考虑用两个变量 x, y 记录执行完一段命令后机器人所在的坐标，若 $x = y = 0$ 则说明走回了原点
- $n \leq 400$：$O(n^2)$ 枚举每个连续子序列，再 $O(n)$ 模拟执行命令，最后根据 x, y 的值判断是否为解

命令序列

- 考虑用两个变量 x, y 记录执行完一段命令后机器人所在的坐标，若 $x = y = 0$ 则说明走回了原点
- $n \le 400$：$O(n^2)$ 枚举每个连续子序列，再 $O(n)$ 模拟执行命令，最后根据 x, y 的值判断是否为解
- $n \le 4000$：类似上面的做法，固定子序列左端点，右端点按顺序枚举时能顺便计算当前的 x, y 值，不需要再用 $O(n)$ 的时间从头模拟起。时间复杂度 $O(n^2)$

命令序列

- 考虑用两个变量 x, y 记录执行完一段命令后机器人所在的坐标，若 $x = y = 0$ 则说明走回了原点
- $n \leq 400$：$O(n^2)$ 枚举每个连续子序列，再 $O(n)$ 模拟执行命令，最后根据 x, y 的值判断是否为解
- $n \leq 4000$：类似上面的做法，固定子序列左端点，右端点按顺序枚举时能顺便计算当前的 x, y 值，不需要再用 $O(n)$ 的时间从头模拟起。时间复杂度 $O(n^2)$
- 命令序列只有 L 与 R，$n \leq 4 \times 10^5$：相当于在一个数轴上左右走

命令序列

- 考虑用两个变量 x, y 记录执行完一段命令后机器人所在的坐标，若 $x = y = 0$ 则说明走回了原点
- $n \leq 400$：$O(n^2)$ 枚举每个连续子序列，再 $O(n)$ 模拟执行命令，最后根据 x, y 的值判断是否为解
- $n \leq 4000$：类似上面的做法，固定子序列左端点，右端点按顺序枚举时能顺便计算当前的 x, y 值，不需要再用 $O(n)$ 的时间从头模拟起。时间复杂度 $O(n^2)$
- 命令序列只有 L 与 R，$n \leq 4 \times 10^5$：相当于在一个数轴上左右走
- 考虑将 L 视为-1，R 视为 $+1$，做这个序列的前缀和 sum_i

命令序列

- 考虑用两个变量 x, y 记录执行完一段命令后机器人所在的坐标，若 $x = y = 0$ 则说明走回了原点
- $n \leq 400$：$O(n^2)$ 枚举每个连续子序列，再 $O(n)$ 模拟执行命令，最后根据 x, y 的值判断是否为解
- $n \leq 4000$：类似上面的做法，固定子序列左端点，右端点按顺序枚举时能顺便计算当前的 x, y 值，不需要再用 $O(n)$ 的时间从头模拟起。时间复杂度 $O(n^2)$
- 命令序列只有 L 与 R，$n \leq 4 \times 10^5$：相当于在一个数轴上左右走
- 考虑将 L 视为-1，R 视为 +1，做这个序列的前缀和 sum_i
- $sum_r = sum_l$ 则 $[l + 1, r]$ 这段连续子序列即为一个合法答案

命令序列

- 考虑用两个变量 x, y 记录执行完一段命令后机器人所在的坐标，若 $x = y = 0$ 则说明走回了原点
- $n \leq 400$：$O(n^2)$ 枚举每个连续子序列，再 $O(n)$ 模拟执行命令，最后根据 x, y 的值判断是否为解
- $n \leq 4000$：类似上面的做法，固定子序列左端点，右端点按顺序枚举时能顺便计算当前的 x, y 值，不需要再用 $O(n)$ 的时间从头模拟起。时间复杂度 $O(n^2)$
- 命令序列只有 L 与 R，$n \leq 4 \times 10^5$：相当于在一个数轴上左右走
- 考虑将 L 视为-1，R 视为 $+1$，做这个序列的前缀和 sum_i
- $sum_r = sum_l$ 则 $[l+1, r]$ 这段连续子序列即为一个合法答案
- 用数组存下每个数的出现次数，$O(n)$ 扫一遍即可得出答案

Outline

找位置

找位置

- 将题目抽象成图的模型：将桌子看做点，若 $d_i = u_j$ 则 i 向 j 连一条有向边

找位置

- 将题目抽象成图的模型：将桌子看做点，若 $d_i = u_j$ 则 i 向 j 连一条有向边
- 每个点都恰好有一条出边，恰好有一条入边

找位置

- 将题目抽象成图的模型：将桌子看做点，若 $d_i = u_j$ 则 i 向 j 连一条有向边
- 每个点都恰好有一条出边，恰好有一条入边
- 一次询问即给定起点 k_i，问从起点走 s_i 步后在哪个点停下

找位置

- 将题目抽象成图的模型：将桌子看做点，若 $d_i = u_j$ 则 i 向 j 连一条有向边
- 每个点都恰好有一条出边，恰好有一条入边
- 一次询问即给定起点 k_i，问从起点走 s_i 步后在哪个点停下
- $n, Q, s_i \leq 20$：暴力模拟即可。$O(\sum s_i)$

找位置

- 将题目抽象成图的模型：将桌子看做点，若 $d_i = u_j$ 则 i 向 j 连一条有向边
- 每个点都恰好有一条出边，恰好有一条入边
- 一次询问即给定起点 k_i，问从起点走 s_i 步后在哪个点停下
- $n, Q, s_i \leq 20$：暴力模拟即可。$O(\sum s_i)$
- 注意到这张图一定由若干个环组成，所以每次询问时可以取出对应的环，并能直接算出最后停下时在环中的位置

找位置

- 将题目抽象成图的模型：将桌子看做点，若 $d_i = u_j$ 则 i 向 j 连一条有向边
- 每个点都恰好有一条出边，恰好有一条入边
- 一次询问即给定起点 k_i，问从起点走 s_i 步后在哪个点停下
- $n, Q, s_i \leq 20$：暴力模拟即可。$O(\sum s_i)$
- 注意到这张图一定由若干个环组成，所以每次询问时可以取出对应的环，并能直接算出最后停下时在环中的位置
- $n \leq 1000$：预处理出所有的环以及每个点所在的环，用二维数组按顺序存下环内元素。时空复杂度 $O(n^2)$

找位置

找位置

- $Q = 1$：求出询问点所在的环即可。时空复杂度 $O(n)$

找位置

- $Q = 1$：求出询问点所在的环即可。时空复杂度 $O(n)$
- 满分做法：二维数组存环空间无法承受，考虑将所有环按顺序放一起用一个一维数组存下，并记录每个环左右端点，询问时仍然可以 $O(1)$ 直接算出终点在这个数组中的位置。时空复杂度 $O(n)$

找位置

- $Q=1$：求出询问点所在的环即可。时空复杂度 $O(n)$
- 满分做法：二维数组存环空间无法承受，考虑将所有环按顺序放一起用一个一维数组存下，并记录每个环左右端点，询问时仍然可以 $O(1)$ 直接算出终点在这个数组中的位置。时空复杂度 $O(n)$
- C++ 选手可以使用 vector 来简便处理，但若比赛时没有开启 O2 优化则要慎用

Outline

摧毁道路

摧毁道路

- $n, m \le 15$：枚举每条道路是否摧毁，再以 s_1, s_2 为起点做 BFS 判定是否合法。$O(2^m \cdot n)$

摧毁道路

- $n, m \le 15$：枚举每条道路是否摧毁，再以 s_1, s_2 为起点做 BFS 判定是否合法。$O(2^m \cdot n)$
- 若原图 (s, t) 之间最短路大于 l 则无解；否则一定有解 (留下最短路)

摧毁道路

- $n, m \le 15$：枚举每条道路是否摧毁，再以 s_1, s_2 为起点做 BFS 判定是否合法。$O(2^m \cdot n)$
- 若原图 (s, t) 之间最短路大于 l 则无解；否则一定有解 (留下最短路)
- 摧毁最多相当于留下最少，而留下的图中，(s, t) 间的路径应该是唯一的

摧毁道路

- $n, m \leq 15$：枚举每条道路是否摧毁，再以 s_1, s_2 为起点做 BFS 判定是否合法。$O(2^m \cdot n)$
- 若原图 (s, t) 之间最短路大于 l 则无解；否则一定有解 (留下最短路)
- 摧毁最多相当于留下最少，而留下的图中，(s, t) 间的路径应该是唯一的
- $m = n - 1$：图是一棵树，原图路径唯一。若有解，则答案为都不在 $(s1, t1)$ 与 $(s2, t2)$ 路径上的边的边数，$O(n)$ 遍历求解即可

摧毁道路

- $n, m \le 15$：枚举每条道路是否摧毁，再以 s_1, s_2 为起点做 BFS 判定是否合法。$O(2^m \cdot n)$
- 若原图 (s, t) 之间最短路大于 l 则无解；否则一定有解 (留下最短路)
- 摧毁最多相当于留下最少，而留下的图中，(s, t) 间的路径应该是唯一的
- $m = n - 1$：图是一棵树，原图路径唯一。若有解，则答案为都不在 $(s1, t1)$ 与 $(s2, t2)$ 路径上的边的边数，$O(n)$ 遍历求解即可
- $s_1 = s_2, t_1 = t_2$：实际上只有一组限制，若有解则答案为总边数 – s,t 之间的最短路

摧毁道路

- $n, m \le 15$：枚举每条道路是否摧毁，再以 s_1, s_2 为起点做 BFS 判定是否合法。$O(2^m \cdot n)$
- 若原图 (s, t) 之间最短路大于 l 则无解；否则一定有解 (留下最短路)
- 摧毁最多相当于留下最少，而留下的图中，(s, t) 间的路径应该是唯一的
- $m = n - 1$：图是一棵树，原图路径唯一。若有解，则答案为都不在 $(s1, t1)$ 与 $(s2, t2)$ 路径上的边的边数，$O(n)$ 遍历求解即可
- $s_1 = s_2, t_1 = t_2$：实际上只有一组限制，若有解则答案为总边数 $-$ s,t 之间的最短路
- 两组限制时留下两组点之间最短路并集是错的

摧毁道路

- $n, m \leq 15$：枚举每条道路是否摧毁，再以 s_1, s_2 为起点做 BFS 判定是否合法。$O(2^m \cdot n)$
- 若原图 (s, t) 之间最短路大于 l 则无解；否则一定有解 (留下最短路)
- 摧毁最多相当于留下最少，而留下的图中，(s, t) 间的路径应该是唯一的
- $m = n - 1$：图是一棵树，原图路径唯一。若有解，则答案为都不在 $(s1, t1)$ 与 $(s2, t2)$ 路径上的边的边数，$O(n)$ 遍历求解即可
- $s_1 = s_2, t_1 = t_2$：实际上只有一组限制，若有解则答案为总边数 − s,t 之间的最短路
- 两组限制时留下两组点之间最短路并集是错的
- 最后留下的边不一定在原来 s, t 的最短路上

摧毁道路

摧毁道路

- $s_1 = s_2$：一定存在某个点 u，使得最后留下的边是 $(s_1, u), (u, t_1), (u, t_2)$ 三组点之间的最短路

摧毁道路

- $s_1 = s_2$：一定存在某个点 u，使得最后留下的边是 $(s_1, u), (u, t_1), (u, t_2)$ 三组点之间的最短路
- 从起点 s_1 出发，路径分叉后再相遇显然不优，因此分叉后一定按最短路直接到两个 t

摧毁道路

- $s_1 = s_2$：一定存在某个点 u，使得最后留下的边是 $(s_1, u), (u, t_1), (u, t_2)$ 三组点之间的最短路
- 从起点 s_1 出发，路径分叉后再相遇显然不优，因此分叉后一定按最短路直接到两个 t
- 枚举每个点当起点做 BFS，$O(n^2)$ 预处理出两点间最短路

摧毁道路

- $s_1 = s_2$：一定存在某个点 u，使得最后留下的边是 $(s_1, u), (u, t_1), (u, t_2)$ 三组点之间的最短路
- 从起点 s_1 出发，路径分叉后再相遇显然不优，因此分叉后一定按最短路直接到两个 t
- 枚举每个点当起点做 BFS，$O(n^2)$ 预处理出两点间最短路
- $O(n)$ 枚举点 u，判断是否合法并更新答案

摧毁道路

- $s_1 = s_2$：一定存在某个点 u，使得最后留下的边是 $(s_1, u), (u, t_1), (u, t_2)$ 三组点之间的最短路
- 从起点 s_1 出发，路径分叉后再相遇显然不优，因此分叉后一定按最短路直接到两个 t
- 枚举每个点当起点做 BFS，$O(n^2)$ 预处理出两点间最短路
- $O(n)$ 枚举点 u，判断是否合法并更新答案
- 满分做法：类似 $s_1 = s_2$ 时的做法。一定存在两个点 u, v，最后留下的边为 $(s_1, u), (s_2, u), (u, v), (v, t_1), (v, t_2)$ 或是 $(s_1, u), (t_2, u), (u, v), (v, t_1), (v, s_2)$ 五组点之间最短路

摧毁道路

- $s_1 = s_2$：一定存在某个点 u，使得最后留下的边是 $(s_1, u), (u, t_1), (u, t_2)$ 三组点之间的最短路
- 从起点 s_1 出发，路径分叉后再相遇显然不优，因此分叉后一定按最短路直接到两个 t
- 枚举每个点当起点做 BFS，$O(n^2)$ 预处理出两点间最短路
- $O(n)$ 枚举点 u，判断是否合法并更新答案
- 满分做法：类似 $s_1 = s_2$ 时的做法。一定存在两个点 u, v，最后留下的边为 $(s_1, u), (s_2, u), (u, v), (v, t_1), (v, t_2)$ 或是 $(s_1, u), (t_2, u), (u, v), (v, t_1), (v, s_2)$ 五组点之间最短路
- $O(n^2)$ 预处理最短路，$O(n^2)$ 枚举点 u, v 计算答案

Outline

L 君找工作

L 君找工作

- $n, m \leq 50$：按题意模拟即可

L 君找工作

- $n, m \leq 50$：按题意模拟即可
- $d_i, r_i \leq 9$：本质不同的工作只有 100 个，对这 100 个模拟即可

L 君找工作

- $n, m \le 50$：按题意模拟即可
- $d_i, r_i \le 9$：本质不同的工作只有 100 个，对这 100 个模拟即可
- $n \le 5000$ 或 $m \le 5000$：满分做法中部分工作可进行暴力处理

L 君找工作

L 君找工作

- 考虑将天按 t_i 排序，工作按 d_i 排序

L 君找工作

- 考虑将天按 t_i 排序，工作按 d_i 排序
- 按 d_i 的顺序处理每个工作，那么可进行工作的天数是单调不升的，且天数变化量只有 $O(n)$ 级别

L 君找工作

- 考虑将天按 t_i 排序，工作按 d_i 排序
- 按 d_i 的顺序处理每个工作，那么可进行工作的天数是单调不升的，且天数变化量只有 $O(n)$ 级别
- 按原顺序维护天数 t_i 的树状数组，每个工作查询时二分答案，那么我们可以直接在树状数组上查询出工作总量：$\sum t_i - d_j * days$

L 君找工作

- 考虑将天按 t_i 排序，工作按 d_i 排序
- 按 d_i 的顺序处理每个工作，那么可进行工作的天数是单调不升的，且天数变化量只有 $O(n)$ 级别
- 按原顺序维护天数 t_i 的树状数组，每个工作查询时二分答案，那么我们可以直接在树状数组上查询出工作总量：$\sum t_i - d_j * days$
- 天数变化时在树状数组上修改即可

L 君找工作

- 考虑将天按 t_i 排序，工作按 d_i 排序
- 按 d_i 的顺序处理每个工作，那么可进行工作的天数是单调不升的，且天数变化量只有 $O(n)$ 级别
- 按原顺序维护天数 t_i 的树状数组，每个工作查询时二分答案，那么我们可以直接在树状数组上查询出工作总量：$\sum t_i - d_j * days$
- 天数变化时在树状数组上修改即可
- 复杂度 $O((n+m)\log^2 m)$