栈、队列

黎伟诺

7.18.2024

栈

- 只有压入和弹出两种操作，一般栈顶的元素比较关键
- $stk[++len] = x; stk[len--] = 0;$
- 不建议使用 stack<int>，跑的慢而且是 vector<int> 的子集
- $v.push\_back(x); v.pop\_back();$

栈

- 在图论和单调栈题中应用的比较多
- 如 tarjan 算法求强连通分量、Fleury 算法求欧拉回路

树上的链

P1612

给定一棵有 n 个节点的树。每个节点有一个点权和一个参数。节点 i 的权值为 w_i，参数为 c_i。1 是这棵树的根。

现在，对每个节点 u，找到最长的一条链 $v_1, v_2, \ldots v_m$，满足如下条件：

1. $v_1 = u$。
2. 对 $2 \leq i \leq m$，有 v_i 是 v_{i-1} 的父节点。
3. 链上节点的点权和不超过 c_u，即 $\sum_{j=1}^{m} w_{v_j} \leq c_u$。

树上的链

P1612

- dfs，开栈记录 1 到 u 的路径点以及对应的权值
- 后缀和难以维护，改为维护前缀和
- 需要二分一个位置使得 $sum_u - sum_p \leq c_u$

受欢迎的牛（Tarjan 模板）

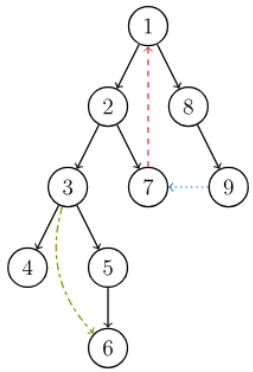
P2341

- 每头奶牛都梦想成为牛棚里的明星。被所有奶牛喜欢的奶牛就是一头明星奶牛。
- 所有奶牛都是自恋狂，每头奶牛总是喜欢自己的。奶牛之间的“喜欢”是可以传递的——如果 A 喜欢 B，B 喜欢 C，那么 A 也喜欢 C。
- 牛栏里共有 N 头奶牛，给定一些奶牛之间的爱慕关系，请你算出有多少头奶牛可以当明星。

Tarjan 算法

强连通分量：一个有向子图内的节点两两互相可达
DFS 生成树的四种边：树边、反祖边、横叉边、前向边。

- 树边：搜索时找到一个还没有访问过的结点
- 反祖边：指向祖先结点的边（例如 $7 \to 1$）
- 横叉边：搜索时遇到了一个访问过的结点，但这个结点不是当前结点的祖先。（例如 $9 \to 7$）
- 前向边：搜索时遇到子树中的结点（例如 $3 \to 6$）



Tarjan 算法

关键观察：如果结点 u 是某个强连通分量在搜索树中遇到的第一个结点，那么这个强连通分量的其余结点肯定是在搜索树中以 u 为根的子树中。
Why？
假设有个结点 v 在该强连通分量中但是不在以 u 为根的子树中，那么 u 到 v 的路径中肯定有一条离开子树的边。
这条边只能为横叉边或者反祖边，但指向的节点 dfs 序肯定比 u 小，与结点 u 是某个强连通分量在搜索树中遇到的第一个结点矛盾

Tarjan 算法

Tarjan 算法中需要为每个节点计算 dfn_u 、low_u。

- dfn_u：dfs 时结点 u 被搜索的次序（dfs 序）
- low_u：以下结点的 dfn 的最小值：$Subtree_u$ 中的结点；从 $Subtree_u$ 通过一条不在搜索树上的边能到达的结点。

Tarjan 算法

dfs 过程维护每个结点的 dfn 与 low 变量，且让搜索到的结点入栈。
dfs 过程中会遇到三种情况：
1、v 未被访问：dfs(v)。在回溯过程中，用 low_v 更新 low_u。
2、v 被访问过，已经在栈中：根据 low 值的定义，用 dfn_v 更新 low_u。
3、v 被访问过，已不在栈中：说明 v 已搜索完毕，其所在连通分量已被处理，所以不用对其做操作。

Tarjan 算法

连通图中有且仅有一个 u 使得 $dfn_u = low_u$。该结点一定是在深度遍历的过程中，该连通分量中第一个被访问过的结点。
在回溯的过程中，判定 $dfn_u = low_u$ 是否成立，如果成立，则栈中 u 及其上方的结点构成一个强连通分量。

王室联邦

P2325

- 把一个有 n 个点的树划分成若干个省，要求每个省至少要有 B 个城市，最多可以有 $3B$ 个城市。
- 每个省必须有一个省会，这个省会可以位于省内，也可以在该省外。
- 但是该省的任意一个城市到达省会所经过的道路上的城市（除了该省省会）都必须属于该省。
- 另外，一个城市可以作为多个省的省会。

王室联邦

P2325

- DFS 整棵树，处理每个节点时，将其一部分子节点分块，将未被分块的子节点返回到上一层。
- 枚举 u 的每个子节点 v，递归处理子树后，将每个子节点返回的未被分块的节点 S_v 添加到集合 S_u 中，一旦 $|S_u| \geq B$ 就把 S_u 作为一个新的块并将 u 作为省会，然后清空 S_u 并继续枚举 v。
- 处理完所有子树后，将 u 也加入到集合 S_u 中，此时在 S_u 中的节点为未被分块的节点，将被返回到上一层，显然 S_u 的大小最大为 $B-1$ 个子树节点加上 u 节点本身，即 $|S_u| \leq B$。

王室联邦

P2325

- 由于返回的集合的大小最大为 B，对于一个子树会对集合最多增加 $B-1$ 个节点，那么每个块的大小最大为 $2B-1$，满足条件。
- 在全局的 DFS 结束后，集合 S(也就是 S_1) 中可能还有节点（最多有 B 个)，那么我们把这 B 个节点并入最后一个块 (以根节点 1 为省会的最后一个块) 中，那么这个块的大小最大为 $3B-1$，符合条件。
- 可以用一个栈来实现这些集合，在 dfs 到 u 的时候，先记录一个栈的大小 sz_{pre}，这个代表 u 的兄弟子树未被分块的节点集合的大小和，每当目前栈的大小 $sz_{now} >= sz_{pre} + B$ 就意味着 S_u 需要出来一个新的块，把栈清空至 sz_{pre}。

单调栈

栈的结构
但是其中的元素满足某种单调性

单调栈

插入的元素如果会使栈不满足单调性怎么办？
一直弹栈，直到插入的元素能满足单调性。

最大全 1 子矩阵

POJ3494

一个 $n \times m$ 的 01 矩阵，找出面积最大的全 1 子矩阵

最大全 1 子矩阵

POJ3494

- 每个位置算出 $up_{i,j}$ 代表 (i,j) 这个位置往上走能连续走多少个 1
- 枚举子矩阵的最后一行 i，假如限定子矩阵的列为 $l \sim r$，那么极大的子矩阵大小为 $\min\{up_{i,l}, up_{i,l+1}, \ldots, up_{i,r}\} \times (r-l+1)$
- 假设 $up_{i,j}$ 是这一段的最小值，我们需要知道 $up_{i,j}$ 作为最小值时往左以及往右能延伸到的最远的地方 $l_{i,j}, r_{i,j}$

最大全 1 子矩阵

POJ3494

- $r_{i,j}$ 可以通过对称的方式来求，看看怎么求 $l_{i,j}$
- 维护一个值递增的单调栈
- $up_{i,j}$ 入栈前弹走比它小的所有值，未被弹走的栈顶位置 $+1$ 就是 $l_{i,j}$
- 最终的最大面积就是 $\max_{i,j}\{up_{i,j} \times (r_{i,j} - l_{i,j} + 1)\}$

最大数

P1198

维护一个初始为空数列，要求提供以下两种操作：

- 查询操作：查询当前数列中末尾 L 个数中的最大的数。
- 插入操作：$n = (n + t) \bmod D$，其中 t 是最近一次查询操作的答案，将所得答案插入到数列的末尾。

数据范围 $1 \leq M \leq 2 \times 10^5$，$1 \leq D \leq 2 \times 10^9$。

最大数

P1198

- 如果新进来的数比前面的大，那前面这个数肯定不能成为最大值。
- 如果比前面的小，那这个新进来的数将在长度小的末尾成为最大值，前面的数在长度大的末尾成为最大值
- 维护一个值递减的单调栈，单调栈里存需要存进来的数的值和位置。

最大数

P1198

- 假如新进来的数 x 比栈顶大，把栈里比 x 小的数全部弹出
- 我们可以利用单调栈的性质做一些事情
- 在单调栈中按位置二分出第一个大于等于 $size - L + 1$ 的位置，对应的值就是末尾 L 个数中的最大的数

音乐会的等待

P1823

- n 个人正在排队进入一个音乐会。
- 队列中任意两个人 a 和 b，如果他们是相邻或他们之间没有人比 a 或 b 高，那么他们是可以互相看得见的。
- 求出有多少对人可以互相看见。

音乐会的等待

P1823

- 维护一个单调栈存储每个人的身高，身高需要满足递减
- 当一个身高比栈顶高的人要进栈时，先将栈中所有身高比他矮（不包括相同的）的人弹掉，弹掉几个答案就加几个
- 当要进栈的人和栈中的人身高相等时，因为相等也看得见，也要加进答案，有几个加几个，但是不弹出
- 假如栈中还有东西，答案就要多加 1，因为要进栈的数和当前比它大的数也是可以互相看见的

队列

- 压入队尾，弹出队头
- $q[++tail] = x; \quad x = q[h++];$
- queue<int> 还是比较方便的
- $q.push(x); x = q.front(); q.pop();$

队列

- bfs，求图的无权最短路
- 优先队列（贪心）、单调队列（优化 DP 等）

Fortune Wheel

2nd Universal Cup. Stage 18

一个转盘有 n 个位置 $0, 1, \ldots, n-1$，你初始在 x，
有 K 种固定走法，$k_1, k_2, \ldots, k_K$，代表顺时针走对应的格子数
还有一种随机走法：等概率随机生成在转盘的一个位置
问到 0 期望最少多少步。
$n \leq 10^5, K \leq 500$

假设我没有随机走法，那应该就是从 0 号点反向 bfs 得出每个点到它的步数
将距离排序，会选择随机 roll 的点肯定处于距离最大的一段后缀
枚举一个后缀，钦定它选择随机 roll，计算出对应的期望步数，看是否小于原来的步数

优先队列

- 压入队列，弹出在某个比较符（$<$）下的最小值（最大值）
- 一般拿堆实现
- 贪心、dijkstra 求最短路、优化 DP

Heap Operation

CF681C

堆支持 3 种操作

- insert x：将 x 插入到最小堆中。
- getMin x：获取最小堆的堆顶，要求最小值为 x
- removeMin: 将最小堆的堆顶弹出。

题目给出了一些操作，不一定合法，往里再添加一些操作使得所有的操作都合法，求添加操作最少的情况并按序输出全部操作（新添加的和已存在的）

Heap Operation

CF681C

- 优先队列模拟最小堆
- insert x 向队列里加入 x
- removeMin 时将堆顶 top 给 pop(如果堆为空，则任意入队一个数然后再 pop)
- getMin x:
  1. 如果堆为空，则加入 x
  2. 不为空且 top()=x，直接 getMin x
  3. 不为空但 top()>x，加入 x
  4. 不为空但 top()<x，一直 removeMin，直到 top()>=x 或堆为空

  (之后按 1,2,3 方法处理)

合并果子

P1090

- n 堆果子
- 每一次合并，多多可以把两堆果子合并到一起，消耗的体力等于两堆果子的重量之和。
- $n-1$ 次合并最后合成 1 堆，求最小的体力耗费值。

Playlist

CF1140C

- n 首歌，每首歌有长度和美丽度
- 选择一些曲子产生的价值是长度和乘上这些曲子的美丽度最小值
- 在选择不超过 K 首曲子的前提下，美丽度的最大值

Playlist

CF1140C

- 按美丽度降序排序
- 插入并维护一个优先队列（大小不超过 K 的小根堆），里面放的是长度的前 K 大

Messenger in MAC

CF1935C

- $n \leq 2000$ 条信息，每条信息有 a_i 和 b_i
- 读一串信息的时间是 a_i 的和加上相邻 b_i 差的绝对值
- 总时间限定为 L 的情况下应该怎么安排读信息的顺序使得能读的数目最大

Messenger in MAC

CF1935C

- 读信息的顺序一定是按 b_i 升序读，相当于枚举 b_l 和 b_r 在里面贪心的按 a_i 选
- 改为枚举 l，再枚举 r 往右移动，移动的过程中往优先队列加 a_i
- 每当 $L-(b_r-b_l)\geq \sum a_i$ 的时候把最大的 a_i pop 掉

种树

P1484

- cyrcyr 今天在种树，他在一条直线上挖了 n 个坑。
- 这 n 个坑都可以种树，但为了保证每一棵树都有充足的养料，cyrcyr 不会在相邻的两个坑中种树。
- 而且由于 cyrcyr 的树种不够，他至多会种 k 棵树。
- 假设每个坑种树的获利会是多少已知（可能为负）
- 求最大获利

种树

P1484

- 带反悔贪心
- 对于一棵树 i，我们先假设它一定被种。但是，如果它不被种，那么它的左右两边一定会被选。所以，如果”反悔”，这等价于又种了一棵收益为 $a_{left(i)} + a_{right(i)} - a_i$ 的树。
- 所以，用一个堆来选择当前收益最大的树，再用一个链表维护每棵树左右两侧的树，最后选择收益最大的 k 棵树。

单调队列

- 与单调栈类似，数据内部满足单调性
- 需要支持 $pop_back()$ 的操作，也就是双端队列 deque（但是 STL 的 deque 很慢）
- 可以处理一些滑动窗口题目，也可以利用这个单调性结构去优化 DP

求 m 区间内的最小值

P1440

- 一个含有 n 项的数列，求出每一项前的 m 个数到它这个区间内的最小值。
- 若前面的数不足 m 项则从第 1 个数开始，若前面没有数则输出 0。

宝物筛选

P776

- 单调队列优化多重背包
- $f_{k_1 w+d} = \max\{f_{k_1 w+d-kw} + kv\} = \max\{f_{k_1 w+d-kw} - (k_1 - k)v\} + k_1 v$

Flowerpot S

P2698

老板需要你帮忙浇花。给出 N 滴水的坐标，y 表示水滴的高度，x 表示它下落到 x 轴的位置。
每滴水以每秒 1 个单位长度的速度下落。你需要把花盆放在 x 轴上的某个位置，使得从被花盆接着的第 1 滴水开始，到被花盆接着的最后 1 滴水结束，之间的时间差至少为 D。
我们认为，只要水滴落到 x 轴上，与花盆的边沿对齐，就认为被接住。
给出 N 滴水的坐标和 D 的大小，请算出最小的花盆的宽度 W。

Flowerpot S

P2698

做法很多，充分发挥想象力

- 单调栈维护最大值最小值，枚举 r 然后二分 l
- 二分长度，单调队列滑动窗口维护最大值最小值
- 双指针，单调队列滑动窗口维护最大值最小值，r 向右走到最大值减最小值大于等于 D 之后，l 再往右走

谢谢