树

xcy

写在前面

梳理一下关于树的知识点，配一些你们都能秒切的题目.

树是什么

树是什么

无环的无向连通图.

树是什么

无环的无向连通图.

由于它没有环, 就会有很多优秀的性质, 比如说连接任意两点有且只有一个.

树是什么

无环的无向连通图.

由于它没有环, 就会有很多优秀的性质, 比如说连接任意两点有且只有一个.

既然我们知道树是什么了, 就可以做几道不需要更多关于树的知识的题目了.

CF1109F Sasha and Algorithm of Silence's Sounds

一个$n \times m$的网格, 每个格子有一个数, 为$1 \sim n \times m$的排列.

一个区间$[l, r](1 \le l \le r \le n \times m)$是好的, 当且仅当数值在该区间内的格子, 构成一棵树(有公共边的格子连边).

统计好区间数.

$n, m \le 2 \times 10^3, n, m \le 2 \times 10^5$.

Solution

图是树当且仅当无环，且$|E| = |V| - 1$.

Solution

图是树当且仅当无环, 且$|E| = |V| - 1$.

two pointers, LCT维护无环, 线段树判$|E| = |V| - 1$(每个点存$|V| - |E|$, 看它是否$= 1$, 森林一定≥ 1, 故只需要查min及其个数即可).

CF650E Clockwork Bomb

给出两棵n个结点的有标号树.

每次操作删去第一棵树的一条边, 再加上一条边, 需要保证此时还是一棵树.

构造一种操作序列, 将第一棵树变成第二棵树,使得操作数最小.

$n \leq 5 \cdot 10^5$.

Solution

对于两棵树中都出现的边，它们不需要改变．它们会形成森林．用并查集把它们缩起来．

Solution

对于两棵树中都出现的边，它们不需要改变．它们会形成森林．用并查集把它们缩起来．

在新生成的树上删边，连边即可．注意需要保证每次操作后都还是一棵树．一种可行的方法是从下往上修改每个点的父边，这样既不会成环，也不会使图不连通．$O(n \times \alpha(n))$.

Luogu P4824 概率充电器

给出一棵n个节点的树，每个点i有q_i的概率直接带电,对于每条边(a,b)，有$p_{a,b}$的概率通电，电流能从带电的点经由通电的边到达其他的点．求带电的节点期望个数.

$n \leq 5 \cdot 10^5$.

Solution

只要求每个点带电的概率就行.

Solution

只要求每个点带电的概率就行.

先dfs一次, 求出每个点子树内的点使它带电的概率, 再换根dp做一次. $O(n)$.

CF1039D You Are Given a Tree

有一棵n个点的树.

一个简单路径的集合被称为"k合法"当且仅当: 树的每个点至多属于其中一条路径(可以不属于), 且每条路径恰好包含k个点.

对于$k \in [1, n]$, 求出"k合法"路径集合的最多路径数.

$n \leq 10^5$.

Solution

$O(n^2)$就是对每个k都从下往上贪心选链.

Solution

$O(n^2)$就是对每个k都从下往上贪心选链.

然后发现$k \geq \sqrt{n}$时答案$\leq \sqrt{n}$, 于是$(\sqrt{n}, n]$那部分按答案分一下块, 每块二分其右端点. $O(n\sqrt{n}logn)$.

树的直径

树的直径

定义: 树上最长的链.

树的直径

定义: 树上最长的链.

求法:

树的直径

定义: 树上最长的链.

求法:

1. 贪心. 从树上任意一个点开始, dfs一次, 找出离它最远的点, 再从这个点dfs, 求出的最长链就是直径.

树的直径

定义: 树上最长的链.

求法:

1. 贪心. 从树上任意一个点开始, dfs一次, 找出离它最远的点, 再从这个点dfs, 求出的最长链就是直径. 证明:

树的直径

定义: 树上最长的链.

求法:

1. 贪心. 从树上任意一个点开始, dfs一次, 找出离它最远的点, 再从这个点dfs, 求出的最长链就是直径. 证明: 自己想一下.

树的直径

定义: 树上最长的链.

求法:

1. 贪心. 从树上任意一个点开始, dfs一次, 找出离它最远的点, 再从这个点dfs, 求出的最长链就是直径. 证明: 自己想一下.

2. DP. 对每个点维护其子树中的最长链和次长链长度, dfs, 回溯时把当前点的最长链传到父亲更新父亲的最长链或次长链.

树的直径

定义: 树上最长的链.

求法:

1. 贪心. 从树上任意一个点开始, dfs一次, 找出离它最远的点, 再从这个点dfs, 求出的最长链就是直径. 证明: 自己想一下.

2. DP. 对每个点维护其子树中的最长链和次长链长度, dfs, 回溯时把当前点的最长链传到父亲更新父亲的最长链或次长链.

时间复杂度都是$O(n)$的.

树的直径

维护，及查询子树直径:

把dfs序求出来(用于查询子树直径)，线段树维护，pushup时取4个点对中最远的那个.

用RMQ求LCA可做到$O(nlogn)$.

树的直径

维护，及查询子树直径:

把dfs序求出来(用于查询子树直径)，线段树维护，pushup时取4个点对中最远的那个.

用RMQ求LCA可做到$O(nlogn)$.

性质:

1. 从树的任意一点出发，最长链一定能到直径的一个端点.
2. 在树上再连上一个点，直径至少有一个端点是不变的.

树的重心

定义: 树上最大儿子子树最小的点.

树的重心

定义: 树上最大儿子子树最小的点.

求法: $O(n)$dfs.

树的重心

定义: 树上最大儿子子树最小的点.

求法: $O(n)$dfs.

性质:

1. 一个点是重心当且仅当它所有子树的大小$\leq \frac{n}{2}$.

2. 一个点是重心当且仅当它是树上到所有点距离之和最小的点.

3. 把两棵树用一条边连起来, 新的树的重心一定在连接原来两棵树重心的路径上.

4. 一棵树添加/删除一个叶子, 其重心最多只移动一条边的距离.

树相关的序列

树相关的序列

dfs序.

树相关的序列

dfs序. 性质：任意一棵子树中的点在dfs序中都是连续的.

树相关的序列

dfs序. 性质: 任意一棵子树中的点在dfs序中都是连续的.

树链的并: 树链的并集(其中的每个点, 每条边只记录一次).

树相关的序列

dfs序. 性质: 任意一棵子树中的点在dfs序中都是连续的.

树链的并: 树链的并集(其中的每个点, 每条边只记录一次).

性质: 若干个点到点x的树链的并, 等于它们到x的链之和, 减去把它们按dfn排序后相邻两点LCA到x的链之和.

Luogu P3320 [SDOI2015]寻宝游戏

给定一棵n个点的树. 每个点有一个状态$w_i \in \{0,1\}$, 一开始所有点状态都是0. 边有长度.

m次操作, 每次操作将一个点的状态异或1.

每次操作后问从任意一个点出发, 在树上随意走, 经过所有状态为1的点后回到出发点所走过路程的最小值.

$n, m \le 10^5$, 边的长度$\le 10^9$.

Solution

设状态为1的点为$x_1, x_2, ..., x_k$. 发现要求的是这些点到$LCA(x_1, x_2, ..., x_k)$的树链的并.

Solution

设状态为1的点为$x_1, x_2, ..., x_k$. 发现要求的是这些点到$LCA(x_1, x_2, ..., x_k)$的树链的并.

把$\{x_k\}$按dfn排序, 则相当于求到$LCA(x_1, x_k)$的树链的并.

Solution

设状态为1的点为$x_1, x_2, ..., x_k$. 发现要求的是这些点到$LCA(x_1, x_2, ..., x_k)$的树链的并.

把$\{x_k\}$按dfn排序, 则相当于求到$LCA(x_1, x_k)$的树链的并.

设dis_x表示x到根的距离. 则上述等于

$(dis_1 + dis_2 - dis_{LCA(1,2)} + dis_3 - dis_{LCA(2,3)} + ...) - dis_{LCA(1,n)}$, $\times 2$即是答案.

Solution

上式可化为

$\frac{1}{2}(\sum_{i=1}^{k-1}(dis_i + dis_{i+1} - 2dis_{LCA(i,i+1)}) + (dis_1 + dis_k - 2dis_{LCA(1,k)}))$.

Solution

上式可化为

$\frac{1}{2}(\sum_{i=1}^{k-1}(dis_i + dis_{i+1} - 2dis_{LCA(i,i+1)}) + (dis_1 + dis_k - 2dis_{LCA(1,k)}))$.

这个式子只和相邻两点的距离，1和k的距离有关.

Solution

上式可化为

$\frac{1}{2}(\sum_{i=1}^{k-1}(dis_i + dis_{i+1} - 2dis_{LCA(i,i+1)}) + (dis_1 + dis_k - 2dis_{LCA(1,k)}))$.

这个式子只和相邻两点的距离, 1和k的距离有关.

用set维护加点, 删点即可. $O(mlogn)$.

树相关的序列

欧拉序: dfs每次回溯时都把父亲加入序列.

树相关的序列

欧拉序: dfs每次回溯时都把父亲加入序列. 用于RMQ维护LCA.

树相关的序列

欧拉序: dfs每次回溯时都把父亲加入序列. 用于RMQ维护LCA.

括号序列: dfs时每个点进栈, 出栈都加入序列一次.

树相关的序列

欧拉序: dfs每次回溯时都把父亲加入序列. 用于RMQ维护LCA.

括号序列: dfs时每个点进栈, 出栈都加入序列一次.

用于树上莫队. 设点x在括号序列中第一次, 第二次出现的位置分别是st_x, ed_x. 令$st_x < st_y$, 若$LCA(x, y) = x$, 则相当于查询st_x, st_y, 若$LCA(x, y) \neq x$, 则相当于查询ed_x, st_y. 序列上如果一个点的左右括号都出现了, 就不计它的贡献.

树相关的序列

prufer序列：对于一棵无根树，不断找到编号最小的度数为1的点，把它的父亲加入序列，然后把它删掉，直到剩下2个点，这样得到的长度为$(n-2)$的序列就是prufer序列.

树相关的序列

prufer序列：对于一棵无根树，不断找到编号最小的度数为1的点，把它的父亲加入序列，然后把它删掉，直到剩下2个点，这样得到的长度为$(n-2)$的序列就是prufer序列.

无根树与prufer序列一一对应.

树相关的序列

prufer序列：对于一棵无根树，不断找到编号最小的度数为1的点，把它的父亲加入序列，然后把它删掉，直到剩下2个点，这样得到的长度为$(n-2)$的序列就是prufer序列.

无根树与prufer序列一一对应.

关于prufer序列的性质:

树相关的序列

prufer序列：对于一棵无根树，不断找到编号最小的度数为1的点，把它的父亲加入序列，然后把它删掉，直到剩下2个点，这样得到的长度为$(n-2)$的序列就是prufer序列.

无根树与prufer序列一一对应.

关于prufer序列的性质：

1. n个点无根树的棵数：n^{n-2}.

树相关的序列

prufer序列：对于一棵无根树，不断找到编号最小的度数为1的点，把它的父亲加入序列，然后把它删掉，直到剩下2个点，这样得到的长度为$(n-2)$的序列就是prufer序列.

无根树与prufer序列一一对应.

关于prufer序列的性质：

1. n个点无根树的棵数：n^{n-2}.

2. n个点有根树的棵数：n^{n-1}.

树相关的序列

prufer序列：对于一棵无根树，不断找到编号最小的度数为1的点，把它的父亲加入序列，然后把它删掉，直到剩下2个点，这样得到的长度为$(n-2)$的序列就是prufer序列.

无根树与prufer序列一一对应.

关于prufer序列的性质：

1. n个点无根树的棵数：n^{n-2}.

2. n个点有根树的棵数：n^{n-1}.

3. n个点，其中点i度数为d_i的无根树的棵数：$\frac{(n-2)!}{\prod_i (d_i-1)!}$.

Luogu P2624 [HNOI2008]明明的烦恼

给出标号为1到n的点,以及其中k个点最终的度数d_i,允许在任意两点间连线,可产生多少棵度数满足要求的树?

$k \le n \le 10^3$.

Solution

设$left = (n-2) - \sum_i (d_i - 1)$，则$Ans = \frac{(n-2)!}{\prod_i (d_i - 1)! \times left} \times left^{n-k}$.

剖分

剖分

重链剖分：用于树链剖分，让时间复杂度乘上$logn$，把一些树上问题转化为序列上的问题.

剖分

重链剖分: 用于树链剖分, 让时间复杂度乘上$logn$, 把一些树上问题转化为序列上的问题.

长链剖分: 一般用于优化和树的深度相关的树上DP. 有时可代替点分治.

剖分

重链剖分: 用于树链剖分, 让时间复杂度乘上$logn$, 把一些树上问题转化为序列上的问题.

长链剖分: 一般用于优化和树的深度相关的树上DP. 有时可代替点分治.

实链剖分: LCT用的.

BZOJ4855 [JSOI2016]轻重路径

一棵n个点的有根二叉树,对它跑一边重链剖分. 如果u两个儿子的大小一样, 认为左儿子为重儿子。

q次操作, 每次删掉一个当前二叉树的叶子, 每次操作后输出重儿子的编号和.

删除一个点之后, 如果以u两个儿子为根的子树大小一样, 重儿子保持不变.

$n, q \leq 2 \cdot 10^5$.

Solution

- 先对原树树剖，在一次删点操作后从根节点开始二分，如果一条边从重边变成轻边，必然有 $size_u \leq \frac{1}{2} size_{rt}$ (取等号是特判对应儿子消失)，二分后，将这个位置作为顶端递归寻找。容易发现这样操作的次数 $< logn$ 次。
- 判定一条边是否从重边变成轻边的依据是父亲的重儿子之前指向 u ，同时删除节点后有 $size_u + 1 = size_{another\_son}$，注意特判 u 是父亲子树最后一个节点的情况。
- 时间复杂度 $O(nlog^2n)$

Luogu P5290 [十二省联考2019]春节十二响

给定一棵n个点，以1为根的树。每个点有权值w_i。需要把所有点划分到若干个集合中，满足每个集合内不存在两点互为祖先/后代。每个集合的权值为其内点的权值的最大值。一种划分的权值为划分出的所有集合的权值之和。求划分的权值的最小值。

$n \le 2 \cdot 10^5$.

Solution

对于任意一条到根的链或其子链, 其上的所有点一定分属不同的集合. 每条链用堆存权值, 合并两条链时从大到小依次合并. 这样相当于做了长链剖分. $O(nlogn)$.

LCA的求法

LCA的求法

倍增.

LCA的求法

倍增.

RMQ.

LCA的求法

倍增.

RMQ. 在欧拉序上用RMQ维护深度最小的点，查询时找到两点在欧拉序上第一次出现的位置.

LCA的求法

倍增.

RMQ. 在欧拉序上用RMQ维护深度最小的点, 查询时找到两点在欧拉序上第一次出现的位置. $O(nlogn) - O(1)$.

LCA的求法

倍增.

RMQ. 在欧拉序上用RMQ维护深度最小的点, 查询时找到两点在欧拉序上第一次出现的位置. $O(nlogn) - O(1)$.

Tarjan.

LCA的求法

倍增.

RMQ. 在欧拉序上用RMQ维护深度最小的点, 查询时找到两点在欧拉序上第一次出现的位置. $O(nlogn) - O(1)$.

Tarjan. 离线. dfs, 用并查集维护, 搞完一个点后把它的并查集合并到它父亲上. 查LCA时看当前点要查的那个点是否被访问过, 如果被访问过, 它所在并查集的根就是这两个点的LCA.

LCA的求法

倍增.

RMQ. 在欧拉序上用RMQ维护深度最小的点, 查询时找到两点在欧拉序上第一次出现的位置. $O(nlogn) - O(1)$.

Tarjan. 离线. dfs, 用并查集维护, 搞完一个点后把它的并查集合并到它父亲上. 查LCA时看当前点要查的那个点是否被访问过, 如果被访问过, 它所在并查集的根就是这两个点的LCA. $O(n \times \alpha(n))$.

Luogu P4211 [LNOI2014]LCA

给定一棵n个点，以1为根的树. dep_i表示点i的深度，$dep_1 = 1$.

q次询问，每次询问给定$l\ r\ z$，求$\sum_{i=l}^{r} dep_{LCA(i,z)}$.

$n, q \leq 5 \cdot 10^4$.

Solution

扫描线. 考虑加一个点会对每个点产生什么贡献. 然后转化为至根路径加, 至根路径查询. 树剖即可.

如果强制在线, 就把线段树可持久化.

Luogu P5305 [GXOI/GZOI2019]旧词

给定一棵n个点，以1为根的树．dep_i表示点i的深度，$dep_1=1$．给定常数k．

q次询问，每次询问给定x,y，求$\sum_{i\le x} dep_{LCA(i,y)}^k$．

$n,q\le 5\cdot 10^4, 1\le k\le 10^9$．

Solution

把dep^k差分一下，然后按树剖的那个序列做一下前缀和.

dsu on tree

dsu on tree

有时可代替点分治.

dsu on tree

有时可代替点分治.

时间复杂度是$O(nlogn)$的.

dsu on tree

有时可代替点分治.

时间复杂度是$O(nlogn)$的.

证明:

dsu on tree

有时可代替点分治.

时间复杂度是$O(nlogn)$的.

证明: 每次合并只会访问小的集合, 且合并后的集合大小至少为小的集合的2倍. 即每个元素最多被访问$logn$次.

BZOJ4855 [JSOI2016]轻重路径(加强版)

题意不变，要求时间复杂度为$O(nlogn)$.

Solution

每个点分开算贡献.

Solution

每个点分开算贡献.

对于一个点x, 若它只有一个儿子, 该儿子在被删之前一直有贡献. 若它有两个儿子u, v, $siz_u \ge siz_v$, 则x重儿子的变化只可能发生在它子树内的最后$siz_v \times 2$个操作(不包括删除x本身的操作).

Solution

用Splay维护每个点被删除的时间, $O(siz_v)$暴力算最后$siz_v \times 2$个操作的影响, 因为前面那$(siz_u - siz_v)$个操作(这些操作中重儿子相同), 需要差分, 最后再求前缀和. Splay启发式合并即可.

Solution

用Splay维护每个点被删除的时间, $O(siz_v)$暴力算最后$siz_v \times 2$个操作的影响, 因为前面那$(siz_u - siz_v)$个操作(这些操作中重儿子相同), 需要差分, 最后再求前缀和. Splay启发式合并即可.

由于siz_v都是相对u较小的子树, 根据重链剖分相关理论, $\sum siz_v$在$O(n)$级别.

Solution

用Splay维护每个点被删除的时间, $O(siz_v)$暴力算最后$siz_v \times 2$个操作的影响, 因为前面那$(siz_u - siz_v)$个操作(这些操作中重儿子相同), 需要差分, 最后再求前缀和. Splay启发式合并即可.

由于siz_v都是相对u较小的子树, 根据重链剖分相关理论, $\sum siz_v$在$O(n)$级别.

$O(nlogn)$.

点分治

点分治

一般用于解决树上路径相关的问题.

点分治

一般用于解决树上路径相关的问题.

两种计算信息的方法: 先把子树信息全部算出来, 再对于每个子树, 剔除该子树内部的信息; 依次合并每个子树的信息.

Luogu P2664 树上游戏

给定一棵树，每个点有一个颜色c_i，设$s(i,j)$表示点i到点j的路径的颜色数量，设$sum_i = \sum_{j=1}^{n} s(i,j)$。对$i \in [1,n]$，求出所有的$sum_i$。

$n, c_i \le 10^5$.

Solution

发现一个颜色第一次出现时，它会产生它子树大小的贡献.

Solution

发现一个颜色第一次出现时，它会产生它子树大小的贡献.

点分治，用第一种计算信息的方法. 对每个分治中心x，先算出每个点到根的答案. 进入每棵子树k时，把子树内的点的贡献消去，然后对于子树内的一个点，首先把它到x(不包括x)的颜色所产生的贡献都消去(即这些颜色第一次出现时的子树大小之和)，然后统计当前点到x(不包括x)的颜色数量num，使答案加上$num \times (siz_x - siz_k)$.

Luogu P2664 树上游戏(加强版)

题意不变，要求时间复杂度为$O(n)$.

Solution

考虑每种颜色对答案的贡献.

Solution

考虑每种颜色对答案的贡献.

对每种颜色, 把该颜色的点删掉, 就会形成一个森林. 对于森林中的每个点, 若它所在的树的大小为Siz, 则这种颜色会对这个点产生$(n - Siz)$的贡献.

Solution

考虑每种颜色对答案的贡献.

对每种颜色, 把该颜色的点删掉, 就会形成一个森林. 对于森林中的每个点, 若它所在的树的大小为Siz, 则这种颜色会对这个点产生$(n-Siz)$的贡献.

故$sum_i = nk - \sum Siz$, 其中k为颜色数, $\sum Siz$为删掉每种颜色后该点所在的树的大小之和. 树上差分搞一下就行.

树分块

树分块

DFS 序分块

通过 DFS 序每 $\sqrt{N}$ 个点分成一块，好写，非常有利于与 DFS 序有关的题目，且严格保证了块的大小，不保证块直径

Height 分块

通过 Depth 每 $\sqrt{Hmax}$ 分成一块，好写，严格保证块直径，不保证块大小，保证连通

Size 分块

检验父亲所在的 size 是否小于 $\sqrt{N}$，小于就加入，否则新开一块，严格保证块直径，大小，连通性，不保证块个数

王室联邦分块

参加下文，保证块直径，大小，个数，不保证联通

Luogu P2325 [SCOI2005]王室联邦

一个国家有n个城市, 编号为1..n. 它们形成了一棵树. 现在你要把它们分成若干个省, 每个省的城市数在$[B, 3B]$内.

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(除了最后一个城市, 即该省省会)都必须属于该省.

一个城市可以作为多个省的省会.

n可以开到10^6.

Solution

对树进行dfs. 对于一个点x, 每棵子树返回的没有分成块的点数是$\leq B$的. 依次遍历每棵子树, 一旦访问到的没有分成块的点数$\geq B$, 就把这些点分成一块, 并设其省会为x. 最后把剩下的那些没有分成块的点, 连同根一起分入最后创建的那个块中. 这样那个块中的点数也不会超过$3B$. 时间复杂度$O(n)$.

Luogu P2137 Gty的妹子树

n个点, 以1为根的有根树, 每个点有点权w_i. m个操作, 操作分3种:

1. 0 u x, 询问以u为根的子树中, 严格大于x的数的个数.

2. 1 u x, 把u的权值改为x.

3. 2 u x, 添加一个点, 编号为当前编号最大点的编号+1, 父亲为u, 权值为x.

$n, m \leq 3 \cdot 10^4$. 强制在线.

Solution

像王室联邦那样分块, 块之间认一下祖先后代关系. 每块内维护一个set. 加点时, 如果其父亲所在块大小$< 3B$就加进它父亲所在的块内, 否则给它新开一个块(此时这个新块只有它一个元素).

$O(m\sqrt{n}logn)$.

虚树

虚树

构建方法：先把原树的根加入虚树，再把关键点按dfn排序，依次加入虚树，顺便加入LCA.

虚树

构建方法：先把原树的根加入虚树，再把关键点按dfn排序，依次加入虚树，顺便加入LCA.

由于建出的虚树只含关键点及其LCA(顶多还有一个原树的根)，点数是$O(k)$级别的，其中k是关键点数.

虚树

证明LCA的个数也是$O(k)$级别的:

虚树

证明LCA的个数也是$O(k)$级别的:

因为关键点按dfn排序了，对于排序后的任意3个点x_1, x_2, x_3，$LCA(x_1, x_2)$和$LCA(x_2, x_3)$都在x_2到根的路径上，即$LCA(x_1, x_2, x_3)$一定等于两者中的其中一个.

虚树

证明LCA的个数也是$O(k)$级别的:

因为关键点按dfn排序了，对于排序后的任意3个点x_1, x_2, x_3，$LCA(x_1, x_2)$和$LCA(x_2, x_3)$都在x_2到根的路径上，即$LCA(x_1, x_2, x_3)$一定等于两者中的其中一个.

因为已按dfn排序，x_2一定在以$LCA(x_1, x_3)$为根的子树内，即$LCA(x_1, x_3) = LCA(x_1, x_2, x_3)$.

虚树

证明LCA的个数也是$O(k)$级别的:

因为关键点按dfn排序了，对于排序后的任意3个点x_1, x_2, x_3，$LCA(x_1, x_2)$和$LCA(x_2, x_3)$都在x_2到根的路径上，即$LCA(x_1, x_2, x_3)$一定等于两者中的其中一个.

因为已按dfn排序，x_2一定在以$LCA(x_1, x_3)$为根的子树内，即$LCA(x_1, x_3) = LCA(x_1, x_2, x_3)$.

故只要算dfn相邻的两点的LCA，就是所有的LCA了.

CF925E May Holidays

给定一棵n个点的有根树，每个点有一个权值t_i和颜色(黑/白)。m次操作，操作有两种类型：

1. 翻转一个点的颜色.
2. 询问有多少个点x满足：x为黑色，x的白色后代数$> t_x$.

$n, m \leq 10^5, |w_i|, |z| \leq 10^9$，时限5s.

Solution

设$w_i = t_i -$子树内白点数, 1操作即翻转点的颜色, 并将点到根的路径上的$w_i + 1 - 1$, 2操作即查询$w_i > 0$的黑点的个数.

Solution

设$w_i = t_i -$子树内白点数, 1操作即翻转点的颜色, 并将点到根的路径上的$w_i + 1 - 1$, 2操作即查询$w_i > 0$的黑点的个数.

把每k个操作分成一块, 建虚树. 虚树上的一条边对应原树上的一条路径, 且该路径在这k个操作中点权变化是一样的. 对于虚树上的每条边, 把点权排序后存到一个数组中, 进行1操作时对它打全局标记即可. 时间复杂度$O(nlogn\frac{m}{k})$(建虚树)+$O(km)$(1操作)+$O(klognm)$(2操作)+$O(n\frac{m}{k})$(每个块操作完后把信息更新到原树上), $k = \sqrt{n}$时理论复杂度最优, 为$O(m\sqrt{n}logn)$.

Solution

设$w_i = t_i -$子树内白点数, 1操作即翻转点的颜色, 并将点到根的路径上的$w_i + 1 - 1$, 2操作即查询$w_i > 0$的黑点的个数.

把每k个操作分成一块, 建虚树. 虚树上的一条边对应原树上的一条路径, 且该路径在这k个操作中点权变化是一样的. 对于虚树上的每条边, 把点权排序后存到一个数组中, 进行1操作时对它打全局标记即可. 时间复杂度$O(nlogn\frac{m}{k})$(建虚树)+$O(km)$(1操作)+$O(klognm)$(2操作)+$O(n\frac{m}{k})$(每个块操作完后把信息更新到原树上), $k = \sqrt{n}$时理论复杂度最优, 为$O(m\sqrt{n}logn)$.

注意被修改的点的颜色变化.

Luogu P2137 Gty的妹子树(加强版)

题意不变, $n, m \le 2 \cdot 10^5$, 不强制在线.

Solution

把操作分块. 每块操作前先算出每个点的答案. 块内暴力算2操作的影响. RMQ求LCA就可以做到$O(m\sqrt{m})$.

Solution

把操作分块. 每块操作前先算出每个点的答案. 块内暴力算2操作的影响. RMQ求LCA就可以做到$O(m\sqrt{m})$.

如果强制在线就只能倍增求LCA了. 这是$O(m\sqrt{m}logn)$的.

Kruskal重构树

Kruskal重构树

用于解决一类问题: 边带权的图上, 一个点只经过边权$\geq x$或$\leq x$的边, 能到达哪些点.

Kruskal重构树

用于解决一类问题: 边带权的图上, 一个点只经过边权$\geq x$或$\leq x$的边, 能到达哪些点.

以$\leq x$为例. 求出图的最小生成树. 如果除去边权$> x$的边后两点在最小生成树上连通(A), 则它们一定在原图上连通(B). 反之也成立.

Kruskal重构树

用于解决一类问题: 边带权的图上, 一个点只经过边权$\geq x$或$\leq x$的边, 能到达哪些点.

以$\leq x$为例. 求出图的最小生成树. 如果除去边权$> x$的边后两点在最小生成树上连通(A), 则它们一定在原图上连通(B). 反之也成立.

证明:

Kruskal重构树

用于解决一类问题: 边带权的图上, 一个点只经过边权$\geq x$或$\leq x$的边, 能到达哪些点.

以$\leq x$为例. 求出图的最小生成树. 如果除去边权$> x$的边后两点在最小生成树上连通(A), 则它们一定在原图上连通(B). 反之也成立.

证明: $A \to B$显然. 下面证明$B \to A$. 如果B成立而A不成立, 那么两点之间的路径不完全在最小生成树上, 即经过了边权$\geq$最小生成树上的路径边权的边. 而如果这条路径能使两点连通, 最小生成树上的路径一定能使两点连通.

Kruskal重构树

用于解决一类问题: 边带权的图上, 一个点只经过边权$\geq x$或$\leq x$的边, 能到达哪些点.

以$\leq x$为例. 求出图的最小生成树. 如果除去边权$> x$的边后两点在最小生成树上连通(A), 则它们一定在原图上连通(B). 反之也成立.

证明: $A \to B$显然. 下面证明$B \to A$. 如果B成立而A不成立, 那么两点之间的路径不完全在最小生成树上, 即经过了边权$\geq$最小生成树上的路径边权的边. 而如果这条路径能使两点连通, 最小生成树上的路径一定能使两点连通.

Kruskal重构树

于是只需考虑生成树上的边.

Kruskal重构树

于是只需考虑生成树上的边.

发现进行Kruskal时，是从小到大加边的. 建一棵树，每加一条边(x, y)，在树上新加一个点t，点权为所加边的边权，然后连边$(t, x), (t, y)$，建成的树就是Kruskal重构树.

Kruskal重构树

于是只需考虑生成树上的边.

发现进行Kruskal时, 是从小到大加边的. 建一棵树, 每加一条边(x, y), 在树上新加一个点t, 点权为所加边的边权, 然后连边$(t, x), (t, y)$, 建成的树就是Kruskal重构树.

由此可发现Kruskal重构树是棵二叉堆.

Luogu P5109 归程

n点m边的无向图，边有两个权值l, a，分别表示长度和水位线.

q个询问，每个询问形如$v\ p$. 你会从v号点开车出发，只经过水位线不超过p的边. 如果走到死路，就停车步行. 步行所有边都可以走. 求你到1号点步行走的边的长度之和的最小值.

$n \leq 2 \cdot 10^5, m, q \leq 4 \cdot 10^5$，强制在线.

Solution

求出以1号点为起点到所有点的最短路, 设距离为d_i. 按a建重构树. 重构树上每个点的d_i为其子树d_i的最小值.

Solution

求出以1号点为起点到所有点的最短路, 设距离为d_i. 按a建重构树. 重构树上每个点的d_i为其子树d_i的最小值.

询问时在重构树上从v出发, 倍增到最高的$< p$的点x, 答案就是它的d_x.

Luogu P4899 [IOI2018]werewolf 狼人

n点m边的无向图. q个询问, 每个询问形如$s\ e\ l\ r$. 狼人需要从s走到e. 在s时必须是人形, 在e时必须是狼形. 人形时只能到编号$\in [1, r]$的点, 狼形时只能到编号$\in [l, n]$的点. 能且只能进行一次形态转换, 从人形转为狼形. 需要在$[l, r]$的点内进行转换. 问满足条件的路径是否存在.

$n, q \leq 2 \cdot 10^5, m \leq 4 \cdot 10^5$.

Solution

建最小，最大生成树的重构树，边权分别为两端点编号的最大，最小值．然后转化为两集合是否有交的问题了．主席树维护．

Thanks