# 一种基于LCT的树上同色连通块维护算法

绍兴市第一中学 任之洲

2016年1月26日

BZOJ3914

summary O O O

**■** 990

一种基于LCT的树上同色连通块维护算法

写在前面

简单介绍

前置技能

▶ 一种由Memphis同学原创的算法。

BZOJ3914

summary

前置技能

写在前面

简单介绍

- ▶ 一种由Memphis同学原创的算法。
- ▶ 用于解决带树链修改的同色连通块维护问题。
- ► 以QTREE7的算法为基础,利用LCT做出了一些扩展,也可以认为是一种新的动态树数据结构。

- ▶ 一种由Memphis同学原创的算法。
- ▶ 用于解决带树链修改的同色连通块维护问题。
- ▶ 以QTREE7的算法为基础,利用LCT做出了一些扩展,也可以认为是一种新的动态树数据结构。
- ▶ 暂时没有正式命名。

- ▶ 一种由Memphis同学原创的算法。
- ▶ 用于解决带树链修改的同色连通块维护问题。
- ▶ 以QTREE7的算法为基础,利用LCT做出了一些扩展,也可以认为是一种新的动态树数据结构。
- ▶ 暂时没有正式命名。
- ► Link-Cut Memphis (LCM)

► Link-Cut Tree

QTREE7

0000000

BZOJ3914

000000 000000 0000000000 summary

**■** 990

000

写在前面

前置技能

前置技能

•00

BZOJ3914

summary

**■** 990

000

一种基于LCT的树上同色连通块维护算法

写在前面

前置技能

前置技能

▶ 经典的动态树数据结构。

•00

▶ Link-Cut Tree

BZOJ3914

summary

QTREE7

► Link-Cut Tree

写在前面

▶ 经典的动态树数据结构。

前置技能

•00

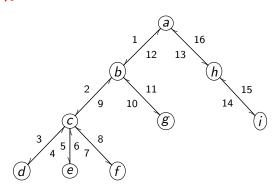
▶ 虚实边的更改次数有O(N log N)的均摊保证。

- ▶ Link-Cut Tree
  - ▶ 经典的动态树数据结构。
  - ▶ 虚实边的更改次数有O(N log N)的均摊保证。
- ► Euler Tour Tree

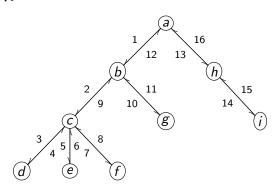
- ▶ Link-Cut Tree
  - ▶ 经典的动态树数据结构。
  - ▶ 虚实边的更改次数有O(N log N)的均摊保证。
- Euler Tour Tree
  - ▶ 用数据结构维护树的欧拉遍历序。

写在前面

► 树的欧拉遍历序: 从根开始的欧拉回路, 将经过的点依次加入序列。



▶ 树的欧拉遍历序:从根开始的欧拉回路,将经过的点依次加入序列。



ightharpoonup a 
ightharpoonup b 
ightharpoonup c 
i

BZOJ3914

summary

**■** 990

000

写在前面

前置技能

前置技能

▶ 欧拉遍历序的一些性质:

000

BZOJ3914

summary

**■** 990

000

写在前面

前置技能

前置技能

▶ 欧拉遍历序的一些性质: ▶ 序列长度为2N-1。

000

- 前置技能
- ▶ 欧拉遍历序的一些性质:
  - ▶ 序列长度为2N-1。
  - ▶ 两点路径上经过的深度最浅的点就是两点的LCA。

- ▶ 欧拉遍历序的一些性质:
  - ▶ 序列长度为2N-1。
  - ▶ 两点路径上经过的深度最浅的点就是两点的LCA。
  - ▶ 一棵完整的子树的欧拉遍历序是连续的一段。

- ▶ 欧拉遍历序的一些性质:
  - ▶ 序列长度为2N-1。
  - ▶ 两点路径上经过的深度最浅的点就是两点的LCA。
  - ▶ 一棵完整的子树的欧拉遍历序是连续的一段。
- ▶ 严格欧拉序的维护相对麻烦,大多数情况可以简化为括号序(DFS序)。

- ▶ 欧拉遍历序的一些性质:
  - ▶ 序列长度为2N-1。
  - ▶ 两点路径上经过的深度最浅的点就是两点的LCA。
  - ▶ 一棵完整的子树的欧拉遍历序是连续的一段。
- 严格欧拉序的维护相对麻烦、大多数情况可以简化为括号 序(DFS序)。
- ► [a [b [c [d] [e] [f] ] [g] ] [h [i] ] ]

- ▶ 欧拉遍历序的一些性质:
  - ▶ 序列长度为2N-1。
  - ▶ 两点路径上经过的深度最浅的点就是两点的LCA。
  - ▶ 一棵完整的子树的欧拉遍历序是连续的一段。
- ▶ 严格欧拉序的维护相对麻烦,大多数情况可以简化为括号 序(DFS序)。
- ► [a [b [c [d] [e] [f] ] [g] ] [h [i] ] ]
- ▶ 很多时候右括号也可以省略。

- ▶ 欧拉遍历序的一些性质:
  - ▶ 序列长度为2N-1。
  - ▶ 两点路径上经过的深度最浅的点就是两点的LCA。
  - ▶ 一棵完整的子树的欧拉遍历序是连续的一段。
- ▶ 严格欧拉序的维护相对麻烦,大多数情况可以简化为括号序(DFS序)。
- ► [a [b [c [d] [e] [f] ] [g] ] [h [i] ] ]
- ▶ 很多时候右括号也可以省略。
- ▶ DFS序可以将子树维护问题转为序列维护问题。

- ▶ 欧拉遍历序的一些性质:
  - ▶ 序列长度为2N-1。
  - ▶ 两点路径上经过的深度最浅的点就是两点的LCA。
  - ▶ 一棵完整的子树的欧拉遍历序是连续的一段。
- 严格欧拉序的维护相对麻烦、大多数情况可以简化为括号 序(DFS序)。
- ► [a [b [c [d] [e] [f] ] [g] ] [h [i] ] ]
- ▶ 很多时候右括号也可以省略。
- ▶ DFS序可以将子树维护问题转为序列维护问题。
- ▶ 根据具体情况选择相应的数据结构进行维护。

- ▶ 给出一棵n个点的无根树, 每个点有一个初始点权。
- ▶ 点的颜色有黑白两种, 最初所有点颜色为黑色。
- 需要维护m次操作:
  - ▶ 0 u, 求点u所在的同色连通块中的最大点权。
  - ▶ 1 u, 将点u反色。
  - ▶ 2 u w, 将点u的点权修改为w。
- ▶  $n, m < 10^5$
- source: SPO I

00

•000000

BZOJ3914

summary

**■** 990

000

写在前面

模型转化

前置技能

▶ 定义1号点为整棵树的根。

BZOJ3914

summary

QTREE7

•000000

▶ 定义1号点为整棵树的根。

前置技能

▶ 并且根据颜色维护两棵树:黑树、白树。

- ▶ 定义1号点为整棵树的根。
- ▶ 并且根据颜色维护两棵树: 黑树、白树。
  - ▶ 所有黑色节点在黑树上与它的父亲节点相连。

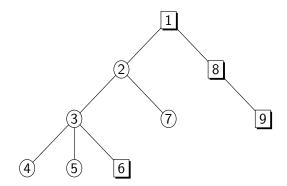
- ▶ 定义1号点为整棵树的根。
- ▶ 并且根据颜色维护两棵树: 黑树、白树。
  - ▶ 所有黑色节点在黑树上与它的父亲节点相连。
  - ▶ 所有白色节点在白树上与它的父亲节点相连。

- ▶ 定义1号点为整棵树的根。
- ▶ 并且根据颜色维护两棵树: 黑树、白树。
  - ▶ 所有黑色节点在黑树上与它的父亲节点相连。
  - ▶ 所有白色节点在白树上与它的父亲节点相连。

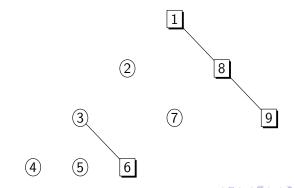
ZOJ3914 summary 00000 0 000000 0 000000000 0

模型转化

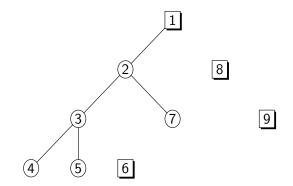
- ▶ 定义1号点为整棵树的根。
- ▶ 并且根据颜色维护两棵树:黑树、白树。
  - ▶ 所有黑色节点在黑树上与它的父亲节点相连。
  - ▶ 所有白色节点在白树上与它的父亲节点相连。



- ▶ 定义1号点为整棵树的根。
- ▶ 并且根据颜色维护两棵树: 黑树、白树。
  - ▶ 所有黑色节点在黑树上与它的父亲节点相连。
  - ▶ 所有白色节点在白树上与它的父亲节点相连。



- ▶ 定义1号点为整棵树的根。
- ▶ 并且根据颜色维护两棵树:**黑树、白树**。
  - ▶ 所有黑色节点在黑树上与它的父亲节点相连。
  - ▶ 所有白色节点在白树上与它的父亲节点相连。



0000000

BZOJ3914

summary

**■** 990

000

一种基于LCT的树上同色连通块维护算法

写在前面

模型转化

前置技能

▶ 考虑修改2号点的颜色。

BZOJ3914

summary

QTREE7

0000000

前置技能

▶ 黑树和白树上均只有一条边受到影响。

写在前面

BZOJ3914

summary

QTREE7

0000000

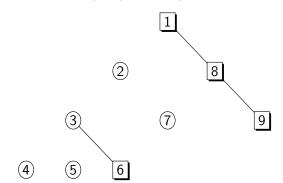
▶ 考虑修改2号点的颜色。

前置技能

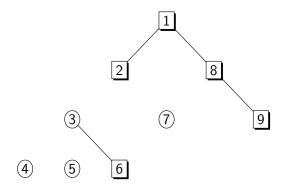
写在前面

▶ 黑树和白树上均只有一条边受到影响。

- ▶ 考虑修改2号点的颜色。
- ▶ 黑树和白树上均只有一条边受到影响。

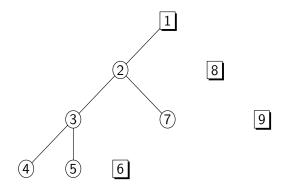


- ▶ 考虑修改2号点的颜色。
- ▶ 黑树和白树上均只有一条边受到影响。



模型转化

- ▶ 考虑修改2号点的颜色。
- ▶ 黑树和白树上均只有一条边受到影响。



模型转化

- ▶ 考虑修改2号点的颜色。
- ▶ 黑树和白树上均只有一条边受到影响。

2 8 7 9 4 5 6

BZOJ3914

summary O O O

▶ 在黑树中,每个连通块都是除根以外均为黑色节点的树。

QTREE7

•0

写在前面

结构维护

前置技能

▶ 在白树中,每个连通块都是除根以外均为白色节点的树。

结构维护

写在前面

summary

- ▶ 在黑树中,每个连通块都是除根以外均为黑色节点的树。
- ▶ 在白树中,每个连通块都是除根以外均为白色节点的树。
- ▶ 修改一个点的颜色,在黑树和白树上都只会影响一条边。

- ▶ 在黑树中、每个连通块都是除根以外均为黑色节点的树。
- ▶ 在白树中,每个连通块都是除根以外均为白色节点的树。
- ▶ 修改一个点的颜色,在黑树和白树上都只会影响一条边。
- ▶ 修改一个点的父亲,即添加子树或移除子树。

结构维护

- ▶ 在黑树中、每个连通块都是除根以外均为黑色节点的树。
- ▶ 在白树中,每个连通块都是除根以外均为白色节点的树。
- ▶ 修改一个点的颜色,在黑树和白树上都只会影响一条边。
- ▶ 修改一个点的父亲,即添加子树或移除子树。
- ▶ 查找一个点离根最近的祖先。

▶ 在黑树中,每个连通块都是除根以外均为黑色节点的树。

QTREE7

- ▶ 在白树中,每个连通块都是除根以外均为白色节点的树。
- ▶ 修改一个点的颜色,在黑树和白树上都只会影响一条边。
- ▶ 修改一个点的父亲,即添加子树或移除子树。
- ▶ 查找一个点离根最近的祖先。
- ▶ 维护子树信息。

QTREE7

▶ 可以用平衡树维护DFS括号序(Euler Tour Tree)。

0000000

BZOJ3914

summary

**■** 990

000

一种基于LCT的树上同色连通块维护算法

写在前面

结构维护

前置技能

- ▶ 可以用平衡树维护DFS括号序(Euler Tour Tree)。
- ▶ 为了方便提出子树,不省略右括号。

- ▶ 可以用平衡树维护DFS括号序(Euler Tour Tree)。
- ▶ 为了方便提出子树,不省略右括号。
- ▶ 添加和移除子树都转化为DFS序中一段区间的移动。

- ▶ 可以用平衡树维护DFS括号序(Euler Tour Tree)。
- ▶ 为了方便提出子树,不省略右括号。
- ▶ 添加和移除子树都转化为DFS序中一段区间的移动。
- ▶ 既然已经使用平衡树维护了, 那么子树信息的维护就自然不 成问题了。

- ▶ 可以用平衡树维护DFS括号序(Euler Tour Tree)。
- ▶ 为了方便提出子树,不省略右括号。
- ▶ 添加和移除子树都转化为DFS序中一段区间的移动。
- ▶ 既然已经使用平衡树维护了,那么子树信息的维护就自然不 成问题了。
- ▶ 查找离根最近的祖先可以用LCT。

- ▶ 可以用平衡树维护DFS括号序(Euler Tour Tree)。
- ▶ 为了方便提出子树,不省略右括号。
- ▶ 添加和移除子树都转化为DFS序中一段区间的移动。
- ▶ 既然已经使用平衡树维护了,那么子树信息的维护就自然不 成问题了。
- ▶ 查找离根最近的祖先可以用LCT。
  - ▶ 也许有更简便的做法。

- ▶ 可以用平衡树维护DFS括号序(Euler Tour Tree)。
- ▶ 为了方便提出子树,不省略右括号。
- ▶ 添加和移除子树都转化为DFS序中一段区间的移动。
- ▶ 既然已经使用平衡树维护了,那么子树信息的维护就自然不 成问题了。
- ▶ 查找离根最近的祖先可以用LCT。
  - ▶ 也许有更简便的做法。
- ▶ 时间复杂度O(n log n)

- ▶ 给出一棵n个点的无根树, 每条树边有一个正权。
- ▶ 点的颜色有黑白两种, 最初所有点颜色为黑色。
- ▶ 需要维护m次操作:
  - ▶ 1 u, 求点u所在的同色连通块中最远点对的距离。
  - ▶ 2 u v c, 将树链(u, v)上的点覆盖为颜色c。
- ▶  $n, m \le 10^5, c \in \{1, 2\}$
- source: by Memphis

QTREE7

▶ 同色连通块是一棵树,需要求这棵树的直径。

BZOJ3914

000000

•00000

summary

**■** 990

000

写在前面

树的直径

前置技能

- ▶ 同色连通块是一棵树,需要求这棵树的直径。
- ▶ 一个比较经典的暴力做法:

- ▶ 同色连通块是一棵树,需要求这棵树的直径。
- ▶ 一个比较经典的暴力做法:
  - ▶ 随便找一个起点S。

- ▶ 同色连通块是一棵树,需要求这棵树的直径。
- ▶ 一个比较经典的暴力做法:
  - ▶ 随便找一个起点S。
  - ▶ BFS找到其中一个离S最远的点u。

- ▶ 同色连通块是一棵树,需要求这棵树的直径。
- ▶ 一个比较经典的暴力做法:
  - ▶ 随便找一个起点S。
  - ▶ BFS找到其中一个离S最远的点u。
  - ▶ 用同样的方法找到离u最远的点v。

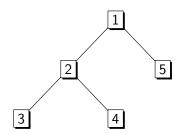
- ▶ 同色连通块是一棵树,需要求这棵树的直径。
- ▶ 一个比较经典的暴力做法:
  - ▶ 随便找一个起点S。
  - ▶ BFS找到其中一个离S最远的点u。
  - ▶ 用同样的方法找到离u最远的点v。
  - ▶ (u, v)间的距离即为这棵树的直径长度。

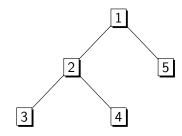
- ▶ 同色连通块是一棵树,需要求这棵树的直径。
- ▶ 一个比较经典的暴力做法:
  - ▶ 随便找一个起点S。
  - ▶ BFS找到其中一个离S最远的点u。
  - ▶ 用同样的方法找到离u最远的点v。
  - ▶ (u, v)间的距离即为这棵树的直径长度。
  - ▶ 可以用反证法证明。

- ▶ 同色连通块是一棵树、需要求这棵树的直径。
- ▶ 一个比较经典的暴力做法:
  - ▶ 随便找一个起点S。
  - ▶ BFS找到其中一个离S最远的点u。
  - ▶ 用同样的方法找到离µ最远的点v。
  - ▶ (u, v)间的距离即为这棵树的直径长度。
  - 可以用反证法证明。
- ▶ 很遗憾这一种做法很难高效地维护。

- ▶ 同色连通块是一棵树、需要求这棵树的直径。
- ▶ 一个比较经典的暴力做法:
  - ▶ 随便找一个起点S。
  - ▶ BFS找到其中一个离S最远的点u。
  - ▶ 用同样的方法找到离µ最远的点v。
  - ▶ (u, v)间的距离即为这棵树的直径长度。
  - 可以用反证法证明。
- ▶ 很遗憾这一种做法很难高效地维护。
- ▶ 有另外一道经典题: ZJOI2007 捉迷藏

## ▶ 观察下面这棵树的DFS括号序:





► [1 [2 [3] [4] ] [5] ]

QTREE7

▶ 括号序为[1 [2 [3] [4] ] [5] ], 假设所有边权为1。

BZOJ3914

000000

summary

写在前面

树的直径

前置技能

- ▶ 括号序为[1 [2 [3] [4] ] [5]], 假设所有边权为1。
- ▶ 提出(2,4)间的括号:

- ▶ 括号序为[1 [2 [3] [4] ] [5] ], 假设所有边权为1。
- ▶ 提出(2,4)间的括号:
  - ▶ 2 [3] [4 → [] [ → [

- ▶ 括号序为[1 [2 [3] [4] ] [5] ], 假设所有边权为1。
- ▶ 提出(2,4)间的括号:
  - ▶ 2 [3] [4 → [] [ → [
  - dist(2,4) = 1

- ▶ 括号序为[1 [2 [3] [4] ] [5] ], 假设所有边权为1。
- ▶ 提出(2,4)间的括号:
  - ▶ 2 [3] [4 → [] [ → [
    - $\rightarrow$  dist(2, 4) = 1
- ▶ 提出(3,5)间的括号:

- ▶ 括号序为[1 [2 [3] [4] ] [5] ], 假设所有边权为1。
- ▶ 提出(2,4)间的括号:
  - ▶ 2 [3] [4 → [] [ → [
    - $\rightarrow$  dist(2, 4) = 1
- ▶ 提出(3,5)间的括号:
  - ▶ 3] [4] ] [5 → ] [ ] ] [ → ] ] [

- ▶ 括号序为[1 [2 [3] [4] ] [5] ], 假设所有边权为1。
- ▶ 提出(2,4)间的括号:
  - ▶ 2 [3] [4 → [] [ → [
    - $\rightarrow$  dist(2, 4) = 1
- ▶ 提出(3,5)间的括号:
  - ▶ 3] [4] ] [5 → ] [ ] ] [ → ] ] [
  - dist(3,5) = 3

- ▶ 括号序为[1 [2 [3] [4] ] [5]], 假设所有边权为1。
- ▶ 提出(2,4)间的括号:
  - ▶ 2 [3] [4 → [] [ → [
    - $\rightarrow$  dist(2, 4) = 1
- ▶ 提出(3,5)间的括号:
  - ▶ 3] [4] ] [5 → ] [ ] ] [ → ] ] [
  - dist(3,5) = 3
- ▶ 化简后的括号序列长度即为两点在树上的距离。

QTREE7

BZOJ3914

000000

000000

summary

**■** 990

000

写在前面

树的直径

前置技能

▶ 设S为整棵树的DFS括号序。

- ▶ 设S为整棵树的DFS括号序。
- ▶ 一个括号序的化简结果用二元组(A, B)表示,即



- ▶ 设S为整棵树的DFS括号序。
- ▶ 一个括号序的化简结果用二元组(A, B)表示,即

$$\overbrace{\left[\right]\left[\right]\left[\right]}^{A}\cdots \left[\left[\right]\right]$$
 
$$\left[\left[\left[\left[\right]\cdots \left[\right]\right]\right]$$

▶ 对于S的所有子串s求出 $(A_s, B_s)$ ,  $A_s + B_s$ 的最大值即为该树的直径。

- ▶ 设S为整棵树的DFS括号序。
- ▶ 一个括号序的化简结果用二元组(A, B)表示,即

$$\overbrace{\left[\right]\left[\right]\left[\right]}^{A}\cdots \left[\left[\right]\right]$$

- ▶ 对于S的所有子串s求出 $(A_s, B_s)$ ,  $A_s + B_s$ 的最大值即为该树 的直径。
- ▶ 考虑使用数据结构进行维护。

▶ 合并两个二元组(A<sub>1</sub>, B<sub>1</sub>)(A<sub>2</sub>, B<sub>2</sub>), 即将这两段括号序相接。

QTREE7

BZOJ3914

000000

summary

₹ 990

写在前面

树的直径

前置技能

- ▶ 合并两个二元组(A<sub>1</sub>, B<sub>1</sub>)(A<sub>2</sub>, B<sub>2</sub>), 即将这两段括号序相接。
- ▶ 合并结果有两种:

- ▶ 合并两个二元组(A<sub>1</sub>, B<sub>1</sub>)(A<sub>2</sub>, B<sub>2</sub>), 即将这两段括号序相接。
- ▶ 合并结果有两种:
  - ▶ 对于 $B_1 < A_2$ 的情况,合并结果为 $(A_1 + (A_2 B_1), B_2)$ 。

- ▶ 合并两个二元组(A<sub>1</sub>, B<sub>1</sub>)(A<sub>2</sub>, B<sub>2</sub>), 即将这两段括号序相接。
- ▶ 合并结果有两种:
  - ▶ 对于 $B_1 < A_2$ 的情况,合并结果为 $(A_1 + (A_2 B_1), B_2)$ 。
  - ▶ 对于 $B_1 \ge A_2$ 的情况,合并结果为 $(A_1, (B_1 A_2) + B_2)$ 。

- ▶ 合并两个二元组(A<sub>1</sub>, B<sub>1</sub>)(A<sub>2</sub>, B<sub>2</sub>), 即将这两段括号序相接。
- ▶ 合并结果有两种:
  - ▶ 对于 $B_1 < A_2$ 的情况,合并结果为 $(A_1 + (A_2 B_1), B_2)$ 。
  - ▶ 对于 $B_1 \ge A_2$ 的情况,合并结果为 $(A_1, (B_1 A_2) + B_2)$ 。
- ▶ 设合并出的二元组为(A<sub>3</sub>, B<sub>3</sub>), 那么A<sub>3</sub> + B<sub>3</sub>的值为

$$Max(A_1 + (A_2 - B_1) + B_2, A_1 + (B_1 - A_2) + B_2)$$

- ▶ 合并两个二元组(A<sub>1</sub>, B<sub>1</sub>)(A<sub>2</sub>, B<sub>2</sub>), 即将这两段括号序相接。
- ▶ 合并结果有两种:
  - ▶ 对于 $B_1 < A_2$ 的情况,合并结果为 $(A_1 + (A_2 B_1), B_2)$ 。
  - ▶ 对于 $B_1 \ge A_2$ 的情况,合并结果为 $(A_1, (B_1 A_2) + B_2)$ 。
- ▶ 设合并出的二元组为(A<sub>3</sub>, B<sub>3</sub>), 那么A<sub>3</sub> + B<sub>3</sub>的值为

$$Max(A_1 + (A_2 - B_1) + B_2, A_1 + (B_1 - A_2) + B_2)$$

QTREE7

BZOJ3914

000000

00000

summary

**■** 990

000

写在前面

树的直径

前置技能

- ▶ 可以用平衡树维护对应的信息,完成直径的计算。
- ► 假如每次只修改一个点的颜色,那么套用QTREE7的算法就可以完成维护。

QTREE7

▶ 同样维护**黑树和白树**,相对地定义**主树和辅树**。

BZOJ3914

•00000

summary O O O

**₹** 990

写在前面

模型转化

前置技能

- ▶ 同样维护黑树和白树,相对地定义主树和辅树。
  - ► 对于**黑色节点**,黑树为主树,白树为辅树。

- ▶ 对于黑色节点,黑树为主树,白树为辅树。
  - ▶ 对于白色节点,白树为主树,黑树为辅树。

- ▶ 同样维护**黑树和白树**,相对地定义**主树和辅树**。
  - ▶ 对于黑色节点,黑树为主树,白树为辅树。
    - ▶ 对于白色节点,白树为主树,黑树为辅树。
- ▶ 怎样在黑树和白树上高效完成链颜色覆盖?

- ▶ 同样维护黑树和白树,相对地定义主树和辅树。
  - ▶ 对于黑色节点,黑树为主树,白树为辅树。
  - ▶ 对于白色节点、白树为主树、黑树为辅树。
- ▶ 怎样在黑树和白树上高效完成链颜色覆盖?
- ▶ 不难发现覆盖后整条链变成了同一个连通块。

- ▶ 同样维护黑树和白树,相对地定义主树和辅树。
  - ▶ 对于黑色节点,黑树为主树,白树为辅树。
  - ▶ 对于白色节点,白树为主树,黑树为辅树。
- ▶ 怎样在黑树和白树上高效完成链颜色覆盖?
- ▶ 不难发现覆盖后整条链变成了同一个连通块。
  - ► LCT!

- ▶ 同样维护黑树和白树、相对地定义主树和辅树。
  - ▶ 对于黑色节点,黑树为主树,白树为辅树。
  - ▶ 对于白色节点、白树为主树、黑树为辅树。
- ▶ 怎样在黑树和白树上高效完成链颜色覆盖?
- ▶ 不难发现覆盖后整条链变成了同一个连通块。
  - ► I CTI
- ▶ 不妨将链颜色覆盖看作LCT操作,可以将问题转化 为O(nlog n)次同色链反色。

QTREE7

一种基于LCT的树上同色连通块维护算法

写在前面

模型转化

前置技能

▶ 对黑树和白树各自维护一棵LCT。

BZOJ3914

000000

summary O O O

**₹** 990

- ▶ 对黑树和白树各自维护一棵LCT。
  - ▶ LCT中需要保证只有颜色相同的点才会以实边相连。

- ▶ 对黑树和白树各自维护一棵LCT。
  - ▶ LCT中需要保证只有颜色相同的点才会以实边相连。
  - ▶ 链覆盖操作时,将修改完的链用实边连结起来。

- ▶ 对黑树和白树各自维护一棵LCT。
  - ▶ LCT中需要保证只有颜色相同的点才会以实边相连。
  - ▶ 链覆盖操作时,将修改完的链用实边连结起来。
- ▶ 为了配合LCT. 黑树和白树的结构定义也作一些修改。

- ▶ 对黑树和白树各自维护一棵LCT。
  - ▶ LCT中需要保证只有颜色相同的点才会以实边相连。
  - ▶ 链覆盖操作时,将修改完的链用实边连结起来。
- ▶ 为了配合LCT. 黑树和白树的结构定义也作一些修改。
- ▶ 对于一个节点u,设它的父亲节点为v

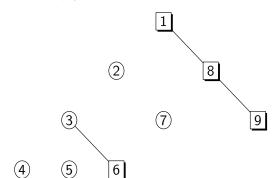
- ▶ 对黑树和白树各自维护一棵LCT。
  - ▶ LCT中需要保证只有颜色相同的点才会以实边相连。
  - ▶ 链覆盖操作时,将修改完的链用实边连结起来。
- ▶ 为了配合LCT. 黑树和白树的结构定义也作一些修改。
- ▶ 对于一个节点u,设它的父亲节点为v
  - ▶ u在它所对应的主树上与v相连。

- ▶ 对黑树和白树各自维护一棵LCT。
  - ▶ LCT中需要保证只有颜色相同的点才会以实边相连。
  - ▶ 链覆盖操作时,将修改完的链用实边连结起来。
- ▶ 为了配合LCT,黑树和白树的结构定义也作一些修改。
- ▶ 对于一个节点u, 设它的父亲节点为v
  - ▶ u在它所对应的主树上与v相连。
  - ► 假如u在它所对应的辅树的LCT中与v以实边相连,那么它们在辅树上相连。

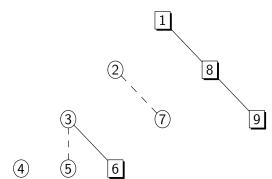
- ▶ 对黑树和白树各自维护一棵LCT。
  - ▶ LCT中需要保证只有颜色相同的点才会以实边相连。
  - ▶ 链覆盖操作时,将修改完的链用实边连结起来。
- ▶ 为了配合LCT, 黑树和白树的结构定义也作一些修改。
- ▶ 对于一个节点u,设它的父亲节点为v
  - ▶ u在它所对应的主树上与v相连。
  - ► 假如u在它所对应的辅树的LCT中与v以实边相连,那么它们 在辅树上相连。
- ▶ 根据以上的定义可以得到黑树和白树的实际形态。

伏王代

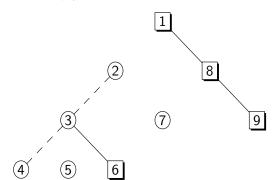
▶ 几种可能的黑树形态:



## ▶ 几种可能的黑树形态:



▶ 几种可能的黑树形态:



► 在新的定义中, **黑树**可视为由以下两部分构成:

QTREE7

BZOJ3914

00000

summary O O O

**₹** 990

写在前面

模型转化

前置技能

- ► 在新的定义中, **黑树**可视为由以下两部分构成:
  - ▶ 由完整的黑色连通块构成的子树。

- ▶ 在新的定义中, 黑树可视为由以下两部分构成:
  - ▶ 由完整的黑色连通块构成的子树。
  - ▶ 由LCT中以实边相连的白色节点构成的链。

- ▶ 在新的定义中, 黑树可视为由以下两部分构成:
  - ▶ 由完整的黑色连通块构成的子树。
  - ▶ 由LCT中以实边相连的白色节点构成的链。
  - ▶ 白色链下接有黑色子树, 但黑色节点下一定没有白色节点。

- ▶ 在新的定义中, 黑树可视为由以下两部分构成:
  - ▶ 由完整的黑色连通块构成的子树。
  - ▶ 由LCT中以实边相连的白色节点构成的链。
  - ▶ 白色链下接有黑色子树,但黑色节点下一定没有白色节点。
- ▶ 同色连通块仍为一段连续的DFS括号序。

- ▶ 在新的定义中, 黑树可视为由以下两部分构成:
  - ▶ 由完整的黑色连通块构成的子树。
  - ▶ 由LCT中以实边相连的白色节点构成的链。
  - ▶ 白色链下接有黑色子树,但黑色节点下一定没有白色节点。
- ▶ 同色连通块仍为一段连续的DFS括号序。
- ▶ 剩下的问题是依照定义严格地维护黑树和白树的形态。

▶ 定义expose操作,为access操作的变种。

QTREE7

BZOJ3914

000000

summary O O O

一种基于LCT的树上同色连通块维护算法

写在前面

expose操作

前置技能

- ▶ 定义expose操作,为access操作的变种。
- ▶ 设v为u所处的同色连通块中深度最浅的点。

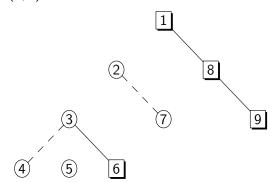
- ▶ 定义expose操作,为access操作的变种。
- ▶ 设v为u所处的同色连通块中深度最浅的点。
- ▶ expose(*u*)的具体操作为:

- ▶ 定义expose操作,为access操作的变种。
- ▶ 设v为u所处的同色连通块中深度最浅的点。
- ▶ expose(*u*)的具体操作为:
  - ▶ 与access操作类似,将路径(u,v)以实边相连,同时也断开u下方的实边。

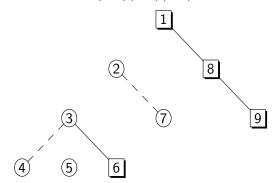
- ▶ 定义expose操作,为access操作的变种。
- ▶ 设v为u所处的同色连通块中深度最浅的点。
- ▶ expose(*u*)的具体操作为:
  - ▶ 与access操作类似,将路径(u,v)以实边相连,同时也断 开u下方的实边。
  - ▶ 在更改实边时,对ETT做相应的修改。

- ▶ 定义expose操作,为access操作的变种。
- ▶ 设v为u所处的同色连通块中深度最浅的点。
- ▶ expose(u)的具体操作为:
  - ▶ 与access操作类似,将路径(u,v)以实边相连,同时也断 开川下方的实边。
  - ▶ 在更改实边时,对ETT做相应的修改。
- ▶ expose操作是对同色链进行的, 所以只在辅树上进行。

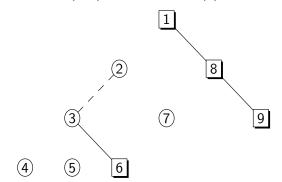
▶ 将路径(6,1)覆盖为黑色。



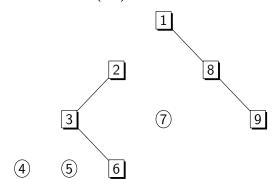
▶ 根据颜色更替,分为(6,6)(3,2)(1,1)进行操作。



▶ 需要修改的只有(3,2)一段, 由expose(3)对黑树进行修改。



▶ 需要添加的边只剩下(1,2)一条。



BZOJ3914

summary

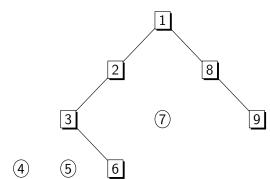
QTREE7

expose採

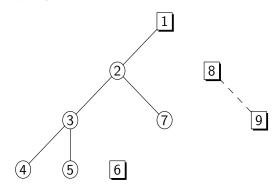
写在前面

## ▶ 黑树维护完成。

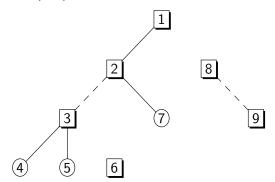
前置技能



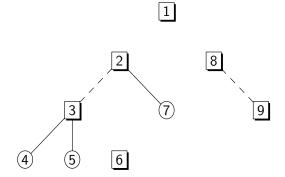
▶ 考虑白树的形态。



▶ 只需要去除(1,2)一条边。



## ▶ 白树维护完成。



QTREE7

BZOJ3914

000000 000000 0000000000 summary

**■** 990

000

写在前面

expose操作

前置技能

▶ 流程整理, 以黑色覆盖为例:

BZOJ3914

summary

QTREE7

expose操作

写在前面

▶ 流程整理, 以黑色覆盖为例:

前置技能

▶ 最初,所有需要修改的节点均为白色,此时黑树为辅树。

- ▶ 流程整理, 以黑色覆盖为例:
  - ▶ 最初, 所有需要修改的节点均为白色, 此时黑树为辅树。
  - ► 在**黑树**上用若干次expose操作将所有需要修改的白色链相 连、并加上链顶的边。

- ▶ 流程整理, 以黑色覆盖为例:
  - ▶ 最初,所有需要修改的节点均为白色,此时黑树为辅树。
  - ▶ 在黑树上用若干次expose操作将所有需要修改的白色链相 连, 并加上链顶的边。
  - ▶ 黑树修改完成后,将所有被修改点的颜色信息更改为黑色。

- ▶ 流程整理, 以黑色覆盖为例:
  - ▶ 最初,所有需要修改的节点均为白色,此时黑树为辅树。
  - ▶ 在黑树上用若干次expose操作将所有需要修改的白色链相 连, 并加上链顶的边。
  - ▶ 黑树修改完成后,将所有被修改点的颜色信息更改为黑色。
  - ▶ 此时白树作为辅树,用expose操作将被修改树链作为黑色链 连通。

- ▶ 流程整理, 以黑色覆盖为例:
  - ▶ 最初,所有需要修改的节点均为白色,此时黑树为辅树。
  - ▶ 在黑树上用若干次expose操作将所有需要修改的白色链相 连, 并加上链顶的边。
  - ▶ 黑树修改完成后,将所有被修改点的颜色信息更改为黑色。
  - ▶ 此时白树作为辅树,用expose操作将被修改树链作为黑色链 连通。
- ▶ 复杂度上界为O(n log²n)。

QTREE7

BZOJ3914

000000 000000 0000000000 summary

• 0 0

一种基于LCT的树上同色连通块维护算法

写在前面

just for fun

前置技能

• • • •

just for fun

- ▶ 这个算法是否可能普及?
  - ▶ 请参考toptree的发展趋势

- ▶ 这个算法是否可能普及?
  - ▶ 请参考toptree的发展趋势
  - ▶ 将来说不定会被搬到仙人掌上

- ▶ 这个算法是否可能普及?
  - ▶ 请参考toptree的发展趋势
  - ▶ 将来说不定会被搬到仙人掌上
    - ▶ 某变色的超仙人掌?

- ▶ 这个算法是否可能普及?
  - ▶ 请参考toptree的发展趋势
  - ▶ 将来说不定会被搬到仙人掌上
    - ▶ 某变色的超仙人掌?
- ▶ 这个算法有什么应用方向?

- ▶ 这个算法是否可能普及?
  - ▶ 请参考toptree的发展趋势
  - ▶ 将来说不定会被搬到仙人掌上
    - ▶ 某变色的超仙人掌?
- ▶ 这个算法有什么应用方向?
  - ▶ DFS序可以维护的题都可以在外面套上链改色

- ▶ 这个算法是否可能普及?
  - ▶ 请参考toptree的发展趋势
  - ▶ 将来说不定会被搬到仙人掌上
    - ▶ 某变色的超仙人掌?
- ▶ 这个算法有什么应用方向?
  - ▶ DFS序可以维护的题都可以在外面套上链改色
    - ▶ 大概比扔在树链剖分上要"有趣"一些

- ▶ 这个算法是否可能普及?
  - ▶ 请参考toptree的发展趋势
  - ▶ 将来说不定会被搬到仙人掌上
    - ▶ 某变色的超仙人掌?
- ▶ 这个算法有什么应用方向?
  - ▶ DFS序可以维护的题都可以在外面套上链改色
    - ▶ 大概比扔在树链剖分上要"有趣"一些
    - ▶ 请不要把这类题出给CodeChef以外的比赛

感谢

## 感谢

- ▶ 感谢ccf提供这次营员交流的机会。
- ▶ 感谢陶渊政同学在这个算法上对我的指导。
- ▶ 感谢国家集训队教练余林韵的验稿。
- ▶ 感谢清华大学的俞鼎力、董宏华、何奇正、张恒捷、王鉴浩 学长对我的帮助。
- ▶ 感谢大家的细心聆听。

写在前面

## 参考资料

- ▶ 陶渊政, 动态树拓展相关, Memphis's Blog
- ▶ 黄志翱, 浅谈动态树的相关问题及简单扩展, 2014集训队论文