倍增 LCA 及其应用

BlueInRed

中国珂学院

April 19th, 2020

前置知识

搜索。

前置知识

搜索。
初一数学。

前置知识

搜索。
初一数学。
简单递推。

前置知识

搜索。
初一数学。
简单递推。
没了。

前置知识

搜索。
初一数学。
简单递推。
没了。
~~很简单吧，真的很简单 qwq~~

可能会涉及到的名词解释

祖先：对于一颗树的一个结点 u，她的爸爸的爸爸是她的祖先，她的爸爸的祖先也是她的祖先..... 以此类推，直到根结点，根结点也是她的祖先。

可能会涉及到的名词解释

祖先：对于一颗树的一个结点 u，她的爸爸的爸爸是她的祖先，她的爸爸的祖先也是她的祖先..... 以此类推，直到根结点，根结点也是她的祖先。

公共祖先：对于一棵树的两个结点 u, v，u, v 的共同的祖先称作她们的公共祖先。

可能会涉及到的名词解释

祖先：对于一颗树的一个结点 u，她的爸爸的爸爸是她的祖先，她的爸爸的祖先也是她的祖先..... 以此类推，直到根结点，根结点也是她的祖先。

公共祖先：对于一棵树的两个结点 u, v，u, v 的共同的祖先称作她们的公共祖先。

LCA：指 u, v 最近公共祖先，记为 $\mathrm{LCA}(u, v)$。最近公共祖先也就是 u, v 的公共祖先中最深的。

暴力

暴力做法：首先把 u, v 中深度大的结点提到和另一节点的同一深度，然后暴力一起向上提，直到 $u = v$ 为止。

暴力

暴力做法：首先把 u, v 中深度大的结点提到和另一节点的同一深度，然后暴力一起向上提，直到 $u = v$ 为止。

这么做的正确性是显然的：我们是**从下向上**提，故第一次 $u = v$ 时的结点一定是她们的 LCA ，后面无论咋提得到的结点深度都比第一次碰上的小。

Q：暴力究竟慢在哪里？

注意到我们珂爱的暴力算法是**一个一个**向上跳的。

Q：暴力究竟慢在哪里？

注意到我们珂爱的暴力算法是**一个一个**向上跳的。
诚然，这样得到的答案正确无疑，但是：真的有必要把这个 LCA 上路径的每一个点都走一遍吗？

Q：暴力究竟慢在哪里？

注意到我们珂爱的暴力算法是**一个一个**向上跳的。
诚然，这样得到的答案正确无疑，但是：真的有必要把这个 LCA 上路径的每一个点都走一遍吗？

现在想一下：您是珂以 AK IOI 的大神犇，那么您必然不愿意在教室里死磕文化课。

Q：暴力究竟慢在哪里？

注意到我们珂爱的暴力算法是**一个一个**向上跳的。
诚然，这样得到的答案正确无疑，但是：真的有必要把这个 LCA 上路径的每一个点都走一遍吗？

现在想一下：您是珂以 AK IOI 的大神犇，那么您必然不愿意在教室里死磕文化课。
所以您会干啥？**跳级！**

跳级

类比跳级，对于 LCA 的优化，您有什么猜想？

跳级

类比跳级，对于 LCA 的优化，您有什么猜想？
一个一个提 $\rightarrow$ **一次提很多个！**

跳级

类比跳级，对于 LCA 的优化，您有什么猜想？
一个一个提 $\rightarrow$ **一次提很多个！**
Q：一次提多少个？
A：一次跳 2^j 层。即，一次跳到当前的 u 和 v 的第 2^j 个祖先。

一个小 QUIZ

我们如何表示“跳”这个状态？

一个小 QUIZ

我们如何表示“跳”这个状态？

考虑设 $f_{x,j}$ 表示 x 向上跳 2^j 后所处的结点。
那么显然有：$f_{x,0} = \mathrm{fa}_x$
Q：如何快速递推出整个 f 数组呢？
对于 $f_{x,j}$ 而言，考虑其定义：设 $f_{x,j}$ 表示 x 向上跳 2^j 后所处的结点。
那么这个定义显然可以转化为：设 $f_{x,j}$ 表示 x 先向上跳 2^{j-1}，再向上跳 2^{j-1} 后所处的结点。
这是因为：$2^{j-1} + 2^{j-1} = 2 \times 2^{j-1} = 2^j$

一个小 QUIZ

我们如何表示“跳”这个状态？

考虑设 $f_{x,j}$ 表示 x 向上跳 2^j 后所处的结点。
那么显然有：$f_{x,0} = \mathrm{fa}_x$
Q：如何快速递推出整个 f 数组呢？
对于 $f_{x,j}$ 而言，考虑其定义：设 $f_{x,j}$ 表示 x 向上跳 2^j 后所处的结点。
那么这个定义显然可以转化为：设 $f_{x,j}$ 表示 x 先向上跳 2^{j-1}，再向上跳 2^{j-1} 后所处的结点。
这是因为：$2^{j-1} + 2^{j-1} = 2 \times 2^{j-1} = 2^j$
故我们得到递推式：

$$\text{令 } y = f_{x,j-1}\text{，那么 } f_{x,j} = f_{y,j-1}$$

倍增

刚刚我们干的那套事情，称为**倍增**。

倍增

刚刚我们干的那套事情，称为**倍增**。
假设我们要求 $\mathrm{LCA}(u, v)$
那么预处理出了 f 数组之后，我们就可以首先把 u, v 提到同一深度，然后按照**从大向小**的顺序枚举 2 的次幂 j，然后看若当前的 $f_{u,j}! = f_{v,j}$，则令 $u = f_{u,j}, v = f_{v,j}$；否则不动。这一整套过程下来后，$f_{u,j}$ 或 $f_{v,j}$ 就是答案。

代码实现

https://paste.ubuntu.com/p/MFk9FY2vVW/

END.

My email: nicest1919@163.com

Thanks for listening!