问题概述

LCA的全称是Least Common Ancestors，顾名思义，LCA问题就是求树上两个点的最近公共祖先的问题。由定义，两个点的最近公共祖先是满足这两个点都在其子树中的点里面深度最大的那个，因此有可能这两个点的最近公共祖先就是其中一个。

倍增算法求LCA

求LCA最简单的方法就是先把两个点中深度更深的那个移动到和另外一个深度相同，然后两个一起向自己的父亲移动，直到移动到同一个点，但是这个方法需要花费的时间较长，我们考虑如何优化这一算法。我们知道，任何一个数都可以表示成二进制，所以我们如果能每次向上跳一个2的幂次深度，我们就可以把时间复杂度从$O(n)$优化到$O(\log n)$。

令$f[i][j]$表示点i的第2^j个祖先是谁，显然$f[i][0]$就是i的父亲，而这个信息我们可以递推，显然$f[i][j] = f[f[i][j-1]][j-1]$，即我的第$2^{j-1}$个祖先的第$2^{j-1}$个祖先是我的第$2^j$个祖先，所以我们只要知道了所有的$f[i][0]$就可以推出所有的信息。

在实际求两个点的LCA时，我们依照刚刚的想法，先把深度更深的点向上跳到和深度浅的点同深度，这一过程可以依据f数组在$O(\log n)$的时间内完成，然后两个点一起往上跳，时间也是$O(\log n)$。具体代码如下

```
inline void init()
{
    for(int j = 1;j <= deps + 1;++j)
        for(int i = 1;i <= n;++i)
            f[i][j] = f[f[i][j - 1]][j - 1];
}

inline int lca(int a, int b)
{
    int i, j;
    if(deep[a] < deep[b])
    {
        int y = a;a = b;b = y;
    }

    for(j = deps;j >= 0;--j)
        if(deep[a] - (1 << j) >= deep[b])
            a = f[a][j];
    if(a == b)
        return a;

    for(j = deps;j >= 0;--j)
        if(f[a][j] != -1 && f[a][j] != f[b][j])
            a = f[a][j], b = f[b][j];
    return f[a][0];
}
```

Tarjan算法求LCA

树链剖分求LCA

RMQ算法求LCA

例题——BZOJ1001狼抓兔子

题目大意

解题思路

代码实现