火车司机出秦川 命题报告

吴作凡

安徽师范大学附属中学

2016年5月3日

题目大意

有一个n个点m条边的带边权的仙人掌，有q组操作，每组操作由两个部分构成：

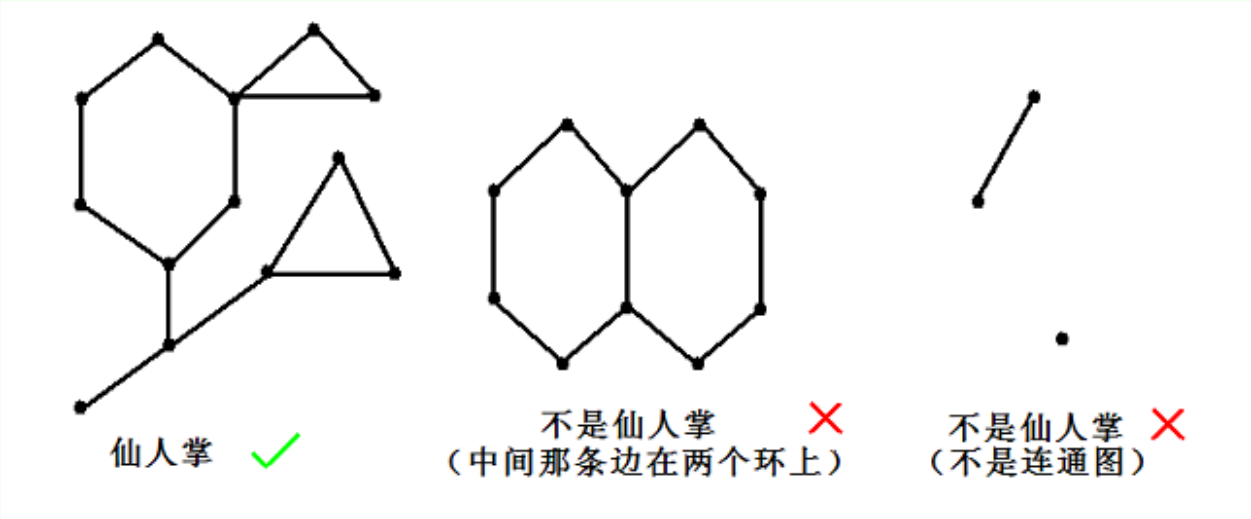
1. 给定k_i条路径，第j条是从s_j到t_j的路径，有一些是经过点最多的简单路径，有一些是经过点最少的简单路径，求出至少被一条路径经过的边的边权和。

2. 在询问结束以后，将第w_i条路径权值修改为x_i。

题目大意

如果一个无向连通图的任意一条边最多属于一个简单环，我们就称之为仙人掌。所谓简单环即不经过重复的结点的环，下文中的环均指简单环。

为了让经过点最多或最少的简单路径唯一，本题数据中的仙人掌的每一个环的长度都是奇数。没有自环。



数据范围和限制

保证任意时刻边长和都属于int范围内。边长均为正整数。
令$S = \sum_{i=1}^{q} k_i$，对100%的数据满足$n, q, S \leq 300000, m \leq 2n - 2$。
时间限制：5sec
空间限制：512MB

一些定义

为了便于描述算法我们对仙人掌作出一些定义。

一些定义

为了便于描述算法我们对仙人掌作出一些定义。
定义两个点u和v的最长链为他们间经过点最多的简单路径。

一些定义

为了便于描述算法我们对仙人掌作出一些定义。
定义两个点u和v的最长链为他们间经过点最多的简单路径。
定义两个点u和v的最短链为他们间经过点最少的简单路径。

一些定义

为了便于描述算法我们对仙人掌作出一些定义。
定义两个点u和v的最长链为他们间经过点最多的简单路径。
定义两个点u和v的最短链为他们间经过点最少的简单路径。
定义一个环的父亲是这个环距离根最近的结点。

一些定义

为了便于描述算法我们对仙人掌作出一些定义。

定义两个点u和v的最长链为他们间经过点最多的简单路径。

定义两个点u和v的最短链为他们间经过点最少的简单路径。

定义一个环的父亲是这个环距离根最近的结点。

定义一个点的子仙人掌是删去它到根的最长链和最短链经过的边以后它所在的连通块。

缩环

直接处理仙人掌过于困难，我们考虑将仙人掌中的环全部缩掉（因为一条边只会出现在一个环上，仙人掌会转化为树），然后再利用处理树的算法来解决仙人掌的问题。

缩环

直接处理仙人掌过于困难，我们考虑将仙人掌中的环全部缩掉（因为一条边只会出现在一个环上，仙人掌会转化为树），然后再利用处理树的算法来解决仙人掌的问题。

我们发现环的父亲到根的路径中并不会经过这个环的其它结点，而环中其它结点到根的最短链和最长链的并则会完全覆盖这个环，那么我们将环的父亲不缩到这个环中。

缩环

直接处理仙人掌过于困难，我们考虑将仙人掌中的环全部缩掉（因为一条边只会出现在一个环上，仙人掌会转化为树），然后再利用处理树的算法来解决仙人掌的问题。

我们发现环的父亲到根的路径中并不会经过这个环的其它结点，而环中其它结点到根的最短链和最长链的并则会完全覆盖这个环，那么我们将环的父亲不缩到这个环中。

那么每个点只会被缩到dfs树上它到父亲的边所在的环，这样每个点都会唯一对应一个新点（当然一个新点不只对应原来的一个点），如果这条边不在任何一个环中，这个点就不缩。

缩环

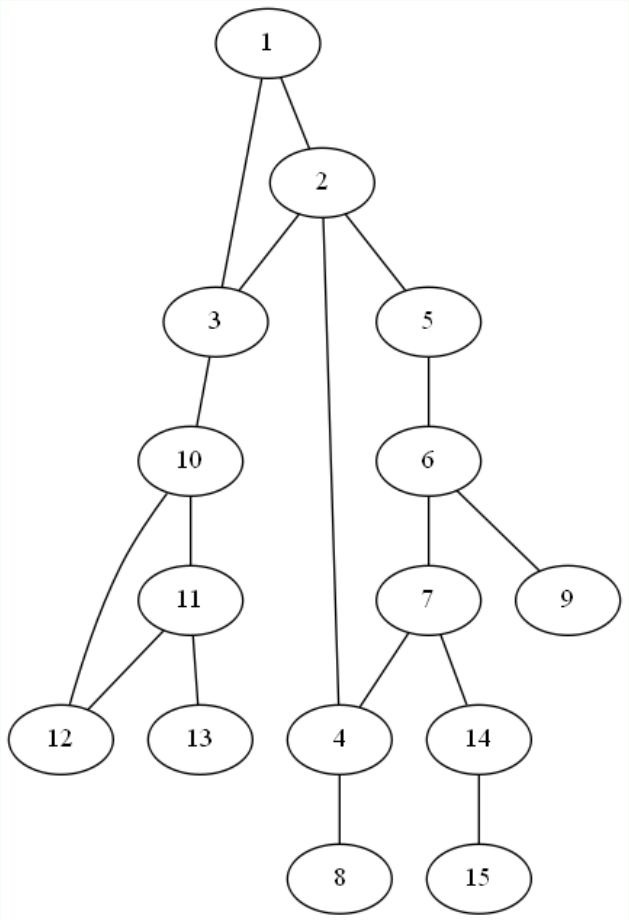
直接处理仙人掌过于困难，我们考虑将仙人掌中的环全部缩掉（因为一条边只会出现在一个环上，仙人掌会转化为树），然后再利用处理树的算法来解决仙人掌的问题。

我们发现环的父亲到根的路径中并不会经过这个环的其它结点，而环中其它结点到根的最短链和最长链的并则会完全覆盖这个环，那么我们将环的父亲不缩到这个环中。

那么每个点只会被缩到dfs树上它到父亲的边所在的环，这样每个点都会唯一对应一个新点（当然一个新点不只对应原来的一个点），如果这条边不在任何一个环中，这个点就不缩。

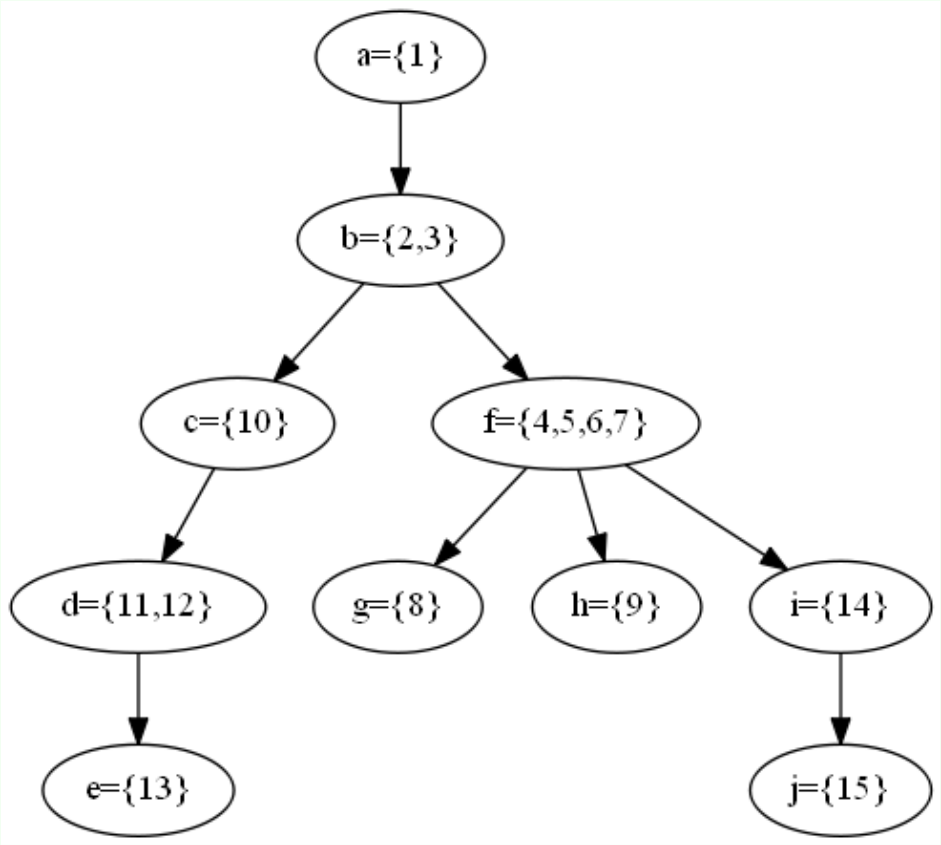
我们将这样缩环后形成的树称作缩环树。

例子



这是一个以1号点为根的仙人掌，下面的例子均指该仙人掌。

缩环树



虚树

如果仙人掌是树，我们可以用虚树算法，对于路径$< u, v >$，只需要在u,v和他们的Lca上打标记，然后遍历虚树判断每条边是否被经过。

虚树

如果仙人掌是树，我们可以用虚树算法，对于路径$< u, v >$，只需要在u,v和他们的Lca上打标记，然后遍历虚树判断每条边是否被经过。

虚树的一条边对应原树的一条链，我们对dfs序建立线段树维护每个点到根的路径长度就好了。

虚树

如果仙人掌是树，我们可以用虚树算法，对于路径$< u, v >$，只需要在u,v和他们的Lca上打标记，然后遍历虚树判断每条边是否被经过。

虚树的一条边对应原树的一条链，我们对dfs序建立线段树维护每个点到根的路径长度就好了。

修改是$O(\log n)$，对于k条路径的一次询问复杂度是$O(k \log n)$，总复杂度$O(S \log n)$。

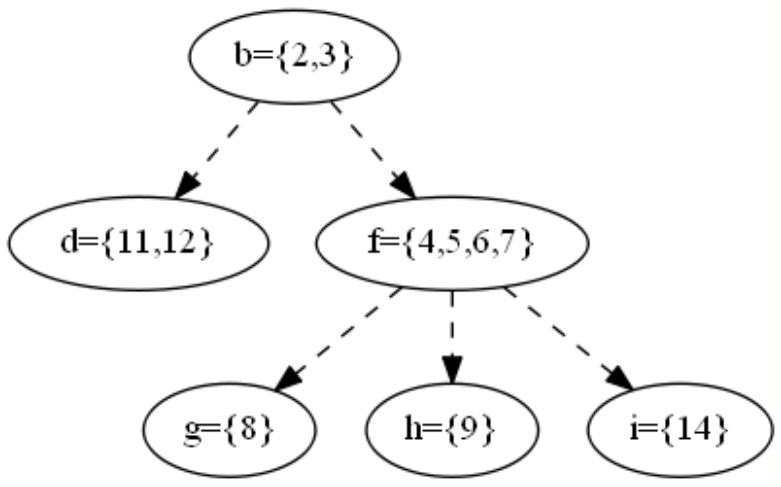
缩环树的虚树

我们已经将仙人掌转化为了缩环树，那么就可以对缩环树建虚树，然后遍历虚树统计答案。

缩环树的虚树

我们已经将仙人掌转化为了缩环树，那么就可以对缩环树建虚树，然后遍历虚树统计答案。

例如有两条路径，11到14的最长链和8到9的最短链，对应的缩环树上的点是d, i, h, g，虚树如下：



环内

和树不同，缩环树的环内依然存在边，在环内产生的贡献有两种方式。

环内

和树不同，缩环树的环内依然存在边，在环内产生的贡献有两种方式。

1. 作为一条路径在缩环树上的最近公共祖先产生的贡献。

比如8和9所属的点g和h的最近公共祖先f，8是4的子仙人掌中的结点，9是6的子仙人掌中的结点，所以产生了4-7-6的贡献。

环内

和树不同，缩环树的环内依然存在边，在环内产生的贡献有两种方式。

1. 作为一条路径在缩环树上的最近公共祖先产生的贡献。

比如8和9所属的点g和h的最近公共祖先f，8是4的子仙人掌中的结点，9是6的子仙人掌中的结点，所以产生了4-7-6的贡献。

2. 一条路径要越过这个环向上走的时候产生的贡献。

比如11到14的最长链要越过环f，14是7的子仙人掌中的结点，那么就产生了7到环f的父亲2的最长链7-6-5-2的贡献。

环内

分析一下这样的贡献数量，显然一条路径只会产生至多一个第一种贡献，那么有$O(k)$个，而每个第二种贡献都会对应虚树的一条边，那么也只有$O(k)$个。

环内

分析一下这样的贡献数量，显然一条路径只会产生至多一个第一种贡献，那么有$O(k)$个，而每个第二种贡献都会对应虚树的一条边，那么也只有$O(k)$个。

这两种贡献都是环上的一些区间，那么可以对这些区间求并去重，对每个环用树状数组维护，询问的时间复杂度是$O(k \log n)$，修改是$O(\log n)$。

虚树的边

和树相同的部分就是虚树的边，但是缩环树上的一条链对应仙人掌中的链有两种——最短链和最长链，产生的贡献就有三种：仅仅是最短链，仅仅是最长链，最长链和最短链的并。

虚树的边

和树相同的部分就是虚树的边，但是缩环树上的一条链对应仙人掌中的链有两种——最短链和最长链，产生的贡献就有三种：仅仅是最短链，仅仅是最长链，最长链和最短链的并。

我们可以维护仙人掌中的每个点到根的最长链、最短链、最长链和最短链的并的长度。同树上类似，我们可以按照dfs序进行维护。

虚树的边

和树相同的部分就是虚树的边，但是缩环树上的一条链对应仙人掌中的链有两种——最短链和最长链，产生的贡献就有三种：仅仅是最短链，仅仅是最长链，最长链和最短链的并。

我们可以维护仙人掌中的每个点到根的最长链、最短链、最长链和最短链的并的长度。同树上类似，我们可以按照dfs序进行维护。

如果修改的是一条不在环中的边，那么只有一个点的子仙人掌会受到影响，我们修改这个区间的三种权值就好。

虚树的边

和树相同的部分就是虚树的边，但是缩环树上的一条链对应仙人掌中的链有两种——最短链和最长链，产生的贡献就有三种：仅仅是最短链，仅仅是最长链，最长链和最短链的并。

我们可以维护仙人掌中的每个点到根的最长链、最短链、最长链和最短链的并的长度。同树上类似，我们可以按照dfs序进行维护。

如果修改的是一条不在环中的边，那么只有一个点的子仙人掌会受到影响，我们修改这个区间的三种权值就好。

如果修改的是某个环上的边，我们可以将这个环分成至多三段，每一段都会被修改最长链或最短链的一种，那么我们将同一个环的dfs序放在一起就可以很好地维护了。

虚树的边

和树相同的部分就是虚树的边，但是缩环树上的一条链对应仙人掌中的链有两种——最短链和最长链，产生的贡献就有三种：仅仅是最短链，仅仅是最长链，最长链和最短链的并。

我们可以维护仙人掌中的每个点到根的最长链、最短链、最长链和最短链的并的长度。同树上类似，我们可以按照dfs序进行维护。

如果修改的是一条不在环中的边，那么只有一个点的子仙人掌会受到影响，我们修改这个区间的三种权值就好。

如果修改的是某个环上的边，我们可以将这个环分成至多三段，每一段都会被修改最长链或最短链的一种，那么我们将同一个环的dfs序放在一起就可以很好地维护了。

我们可以用线段树来维护，那么需要查询的边数是$O(k)$的，每次询问的复杂度是$O(\log n)$，所以总时间复杂度是$O(k \log n)$，修改是$O(\log n)$。

至此本题的算法就描述完毕了，总时间复杂度是$O(S \log n)$，实现起来有很多细节。

本题还存在一个基于轻重链剖分的算法，时间复杂度$O(S \log^2 n)$，详细可以看我的论文。

命题思路和总结

近年来信息学竞赛出现了不少仙人掌相关的题目，2015年清华集训的《静态仙人掌》给了我很多启发，激起了我研究仙人掌的兴趣。

我通过研究被我称为“缩环树”的结构，发现在这个结构上虚树和轻重链剖分算法依然成立，时间复杂度也没有变化。本题在树上也算是一个经典问题，而且两种算法都很适用，所以就我将其扩展并命了这道题。

我希望这道题能起到抛砖引玉的作用，希望大家对于仙人掌产生一些兴趣。目前仙人掌相关的算法虽然时间复杂度都比较优秀，但是代码都相当难以实现，希望大家能提出一些更为有效、方便的算法。

感谢

感谢CCF提供的交流机会。
感谢叶国平老师在学习和生活上的帮助和关心。
感谢倪星宇同学、罗哲正同学对我命题的帮助。
感谢吕凯风学长、彭雨翔学长、王逸松学长、于纪平学长、毛啸同学等研究过仙人掌的同学给我的启发。
感谢其他给予过我帮助的老师和同学。
感谢大家的仔细聆听，欢迎大家提问。