mx的仙人掌 解题报告

杭州学军中学 王逸松

1 试题来源

UOJ #87. mx的仙人掌

2 试题大意

现给定一棵仙人掌，每条边有一个正整数权值，每次给k 个点（可以存在相同点），问从它们中选出两个点（可以相同），它们之间最短路的最大值是多少。

3 数据范围

边权不超过$2^{31}-1$。

Q 表示询问的个数，tot 表示询问的总点数。

对于5% 的数据，$n, tot \leq 7$。

对于10% 的数据，$n, tot \leq 5000$。

对于另外10% 的数据，$Q \leq 10$，其中存在$Q \leq 2$ 的数据。

对于另外30% 的数据，$m = n - 1$。

对于100% 的数据，$n, tot \leq 300000$。

时间限制5s

空间限制512MB

4 算法介绍

4.1 算法一

对每个询问暴力。

对于树上的情况，每个询问只要做一次树形DP即可。

对于仙人掌上的情况，同样可以做仙人掌上的DP。

以1号结点为根，对每个结点x求出x这棵子仙人掌中询问点到x的最大距离，记为$f[x]$。（子仙人掌x的定义是，删掉根节点到x之间的所有简单路径上的边之后，x所在的连通块）

然后对于每个结点，用不同儿子的最大距离+次大距离来更新答案。对于每个环，还要对环上的每对x, y，用($f[x] + f[y]$ + x到y在环上的最短路长度)来更新答案。可以在环上按顺时针枚举x，然后满足x到y的最短路是顺时针走的y在环上是一个区间，而且这个区间的另一个端点也在环上顺时针移动，所以可以用单调队列维护。

时间复杂度$O(nQ)$。

4.2 算法二

对于树上的情况，可以对每个询问的点集建一棵虚树[1]，然后直接在虚树上DP即可。

虚树的建法是，把询问的点集按照dfs序排序，然后将排序后相邻两个结点的lca也加进虚树中，虚树中两个结点的距离为它们在原树上的距离。

时间复杂度为$O(n \log n)$，可以通过另外30%的数据。（我们认为$Q = O(n)$, $tot = O(n)$）

4.3 算法三

对于仙人掌上的情况，可以将虚树推广到仙人掌上[1] 。

首先我们把仙人掌的每个环看成一个结点。这个环节点的父亲是环上离根最近的结点。这个环节点的儿子是环上除父亲外的所有结点。

[1]一个优秀的做法是，建立一棵“虚仙人掌”来解决这道题，由于细节较多不做详细介绍，详见mx的仙人掌 题解

对于每个环节点，把它和它的父亲用一条权值为0的边连起来，再把这个环的每个儿子都跟这个环节点连起来，边权为这个儿子到环的父亲的最短路长度。

最后把所有环上原来的边删掉，于是这棵仙人掌就变成了一棵树。我们在这棵树上对询问的点集建一棵虚树，然后进行树形DP。

树形DP的过程是，枚举虚树上每个结点x，如果是仙人掌上的结点，就枚举这个结点的儿子，然后用不同儿子的最大距离+次大距离来更新答案；如果是仙人掌的环结点，就类似仙人掌上的DP，用单调队列扫一遍来更新答案。

这个算法的时间复杂度是$O(n \log n)$，能通过所有数据，但是有点难写。

4.4 算法四

考虑算法一在树上的情况，如果一个询问给了cnt个点，那么有效的更新答案的次数为$cnt - 1$次，所以对于所有询问，有效的更新答案的次数之和是$O(n)$的。

我们可以用$f[x][i]$表示在子树x中第i个询问的点集中的点到结点x的最大距离，然后树形DP的过程相当于合并这些数组，并更新相应询问的答案。

现在问题转化成，对于每个结点，合并它的每个儿子的DP数组，并快速找到需要被更新答案的询问编号。

这相当于，维护很多数的集合，支持把其中两个合并，并且在合并的时候返回它们的交集。

可以直接用数组来存集合中的数，然后启发式合并，即每次合并时，将小的集合暴力插入到大的集合中。由于要求交集，可以对每个集合开一个哈希表（或直接在全局开哈希表），然后合并的时候判一下即可。

时间复杂度$O(n \log n)$，空间复杂度$O(n)$，可以通过另外30%的数据。

4.5 算法五

考虑把算法四推广到仙人掌上。

对于除了环以外的部分，跟树上的情况一样处理。

对于每个环，先将环的每个儿子的DP数组处理出来，然后用每对DP数组的交集来更新答案。

由于没法从一个集合中删除另一个集合的元素，所以不能使用带删除操作的单调队列。

可以按照从环的父亲往下走哪边近，将环分成左右两部分。每一部分内部的点对之间的最短路一定是在内部走。对于左边的每一个结点，在右边的结点中逆时针走到它较近的结点一定是右边这条链的一个前缀。对于右边的每一个结点也是这样。

所以只要对环的每一个部分，把DP数组按顺序合并起来，并与另一部分的DP数组求交更新答案即可。

在环上更新完答案后还要撤销所有合并操作，这可以在合并的过程中记录下合并时进行的操作来实现。

最后要把环上所有儿子的DP数组全部合并。

复杂度分析：

不考虑环上更新答案时DP数组的合并，总时间复杂度是$O(n\log n)$的，因为每个询问点在合并时作为“较小的集合”最多$O(\log n)$次。

考虑环上更新答案的过程，本质上跟把环上所有DP数组合并是一样的，所以不会影响复杂度，所以总时间复杂度是$O(n\log n)$。

假设环上有若干个集合，总的大小为N，那么可以证明，把这些集合按顺序一个一个合并的复杂度不会超过$O(N)$，所以合并时记录下的操作个数也是$O(N)$，所以总的空间复杂度也是$O(n)$。

这个算法可以通过全部测试数据，而且比较好写。

5 参考代码

见`mx-cactus.cpp`（算法五）

参考文献

[1] 徐寅展, 线段树在一类分治问题上的应用, 2014年国家集训队论文