追击圣诞老人 解题报告

杭州学军中学 王逸松

1 试题来源

UOJ Round #4 C. 追击圣诞老人

2 试题大意

给一棵树，每个点有权值w_i ($w_i > 0$)。

对于第i个点给定三个参数a_i, b_i, c_i，第i个点只能走向a_i, b_i, c_i这三个点在原树中两两之间的路径上的点。

一个长度为l的序列$s_1, s_2, ..., s_l$的权值定义为$\sum_{i=1}^{l} s_i$，这个序列中对于任意$1 \le i < l$都要满足s_i能走向s_{i+1}。

求权值前k小的序列的权值。

3 数据范围

测试点编号	n	k
1,2	$n = 5$	$k = 10$
3,4	$n = 100$	$k = 100$
5,6	$n = 1000$	$k = 50000$
7,8	$n = 1000$	$k = 10^5$
9,10	$n = 15000$	$k = 10^5$
11,12	$n = 20000$	$k = 10^5$
13,14	$n = 10^5$	$k = 10^5$
15,16	$n = 10^5$	$k = 5 \times 10^5$
17,18	$n = 5 \times 10^5$	$k = 10^5$
19,20	$n = 5 \times 10^5$	$k = 5 \times 10^5$

4 算法介绍

4.1 算法一

暴力枚举所有长度不超过k的序列，复杂度$O(n^k)$，能通过1, 2号测试点。

4.2 算法二

定义$W(A)$表示序列A的权值。

我们把所有序列建成一张图，每一个序列向它最后加一个点形成的序列连边。那么这张图满足堆性质：如果u连向v，则$W(u) < W(v)$。

用一个优先队列维护已经扩展到的点（序列）。

一开始将所有入度为0的点（长度为1的序列）放入优先队列里，然后进行k次操作，每次操作是弹出优先队列中权值最小的序列（记为A），然后将A连向的所有序列放入优先队列里。第i次操作弹出的序列就是第i小的答案。

注意这张图不需要完全建出，只要在访问到x这个点时将x的出边建出即可。

由于一个点会连向$O(n)$个点，该算法的复杂度为$O(nk \log nk)$，能通过1 ~ 4号测试点。

4.3 算法三

对于上一个算法中图上的每个点，将它的所有出边按权值从小到大排序，那么就可以用左儿子右兄弟的方法将度数减少为2，这样优先队列中只会被放入$O(n + k)$个点。

由于一个序列最后能加的点只跟这个序列的最后一个点有关，所以只要将每个点能走到的所有点按权值排序即可。

时间复杂度$O(n^2 + k \log(n + k))$，空间复杂度$O(n^2 + k)$。能通过1 ~ 8号测试点。

4.4 算法四

上一个算法的瓶颈是将每个点能走到的点按权值排序。

我们可以将每个点能走到的点按权值建一个小根堆，优先队列中记录这些信息：(权值, 最后一个点, 指向一个堆的指针)。

弹出一个序列(W, x, H)时，只要将$(W + w_{H.l.id} - w_x, H.l.id, H.l)$、$(W + w_{H.r.id} - w_x, H.r.id, H.r)$ 和$(W + w_{Heap_x.id}, Heap_x.id, Heap_x)$放入优先队列即可。

其中$H.id$表示堆H中权值最小的点的编号，$Heap_i$表示一个包含点i能走到的所有点的堆。

现在问题转化成，对每个i求出$Heap_i$。

我们用可并堆（如左偏树）预处理第i个点到根的路径上前2^j个点的堆，然后每个点能走到的点可以拆成三条链，于是可以用倍增得到。

时间复杂度$O(n\log^2 n + k\log(n + k))$，空间复杂度$O(n\log^2 n + k)$。能通过第1 ~ 12号测试点。

4.5 算法五

考虑算法三，如果我们能在$O(f(n))$的时间内求得第i个点能走到的点中，权值第j大的点，就能在$O(g(n) + k\log(n + k) + k \cdot f(n))$的时间内解决原问题，其中$O(g(n))$的时间用来预处理。

算法三中$f(n) = O(1), g(n) = O(n^2)$。

如果我们用主席树（可持久化线段树）来维护1 ~ i路径上点的权值，就可以做到$O(\log n)$查询路径第k大，这样$f(n) = O(\log n), g(n) = O(n\log n)$。

于是总的时间复杂度为$O(n\log n + k\log(n + k))$，空间复杂度为$O(n\log n + k)$。

能通过1 ~ 16号测试点。对于$n = 5 \times 10^5$的数据会MLE。

4.6 算法六

考虑算法四，对于一条链(a, b)，将这条链上的点建成一个堆H，那么$H.id$就是这条链上权值最小的点。我们将这条链从这个点处断开，那么剩下的两段分别对应于$H.l$和$H.r$。

我们用一个二元组(a, b)表示(a, b)这条链的堆，就不用预处理出算法四中的$Heap_i$了。

复杂度为$O(n\log n + k\log(n + k) + k \cdot f(n))$，其中$f(n)$为求出一条链上权值最小的点的时间复杂度。

用倍增或树链剖分都可以做到$f(n) = O(\log n)$。

其中倍增的空间复杂度是O($n \log n$)的，会MLE。

树链剖分时要预处理每条重链的前缀最小值，就可以$O(\log n)$询问路径最小值。

5 参考程序

算法二：`Algorithm_2.cpp`

算法三：`Algorithm_3.cpp`

算法四：`Algorithm_4.cpp`

算法五：`Algorithm_5.cpp`

算法六：`Algorithm_6.cpp`