6036 -- 【10.24T3NOIP2018】明日之星

Solution

首先考虑如何快速查找一条链上的点对某个S的贡献，对这些点对应的字符串建AC自动机，在对应的位置加上a的值，之后把S丢到AC自动机上跑，每到自动机上的一个状态答案就加上它$fail$链上所有状态的权值和。

接下来考虑括号序，任何一条树上的链都可以拆成括号序上至多两段区间来表示，接着用对括号序建一棵线段树，对于线段树的某个区间$[l,r]$，将$[l,r]$上的所有点对应的字符串建AC自动机，接着再建出$fail$反链树，不难发现这样做时间复杂度和空间复杂度都是$O(\sum |s|\ log\ n)$的。接着再考虑a，当某个点加上某个权值时，会对其$fail$反链树上的子树中所有点都造成贡献，所以只需对每个线段树区间再用一个树状数组来维护即可，这样的时间复杂度是$O(\sum |s|\ log^2\ n)$。

询问的话，就找出u到v路径对应括号序的$log\ n$个线段树区间，依次把S丢到这$log\ n$个区间的AC自动机上跑，用树状数组统计贡献。
修改也差不多，可以发现也是一样的把需要修改的$log\ n$个区间抽出来，找到它在该区间对应的AC自动机上的位置(这个需要提前预处理)，在树状数组上修改即可。

总的复杂度$O(\sum |s| + \sum |S| + q\ log^2\ n)$

```
#include<iostream>
#include<cstdio>
#include<cstring>
#include<algorithm>
#define fo(i,j,l) for(int i=j;i<=l;++i)
#define fd(i,j,l) for(int i=j;i>=l;--i)
using namespace std;
typedef long long ll;
const ll N=22e4,M=N<<1,U=M<<2,K=M*20;
struct note{
    int c[5],fail,dfn,tail,val;
}k[K];
int n,tp,zy[200],uy,x,y,op,ans,fs,oo,l,keep,lr,q,rp,u,r,j;
char t[M];
int s[M],d[U];
int zb[M],yb[M];
int v[M],fa[M],dep[M];
int la[M],ne[U],lb[U];
int dq,be[U],en[U],cd[U],tr[K];
int bh[M],pre[M],bac[M],we[M];
int xs[M];
int f[N][20],po[M][30];
inline void llb(int a,int b){ne[++oo]=la[a]; la[a]=oo; lb[oo]=b;}
inline void dg(int o)
{
    pre[o]=++op;
```

```
        bh[op]=o; xs[op]=v[o];
        for(int y=la[o];y;y=ne[y])
        if(!fa[lb[y]])
        {
                fa[lb[y]]=f[lb[y]][0]=o;
                dep[lb[y]]=dep[o]+1;
                for(l=0;f[f[lb[y]][l]][l];++l)
                f[lb[y]][l+1]=f[f[lb[y]][l]][l];
                dg(lb[y]);
        }
        bac[o]=++op;
        bh[op]=o; xs[op]=-v[o];
}
inline void dfs(int o)
{
        k[o].dfn=k[o].tail=++uy;
        for(int y=la[o-keep];y;y=ne[y]){
                dfs(lb[y]);
                k[o].tail=k[lb[y]].tail;
        }
}
inline void cor(int o,int b,int del)
{
        for(;b<=cd[o];b=b+(b&(-b)))
        tr[b+be[o]-1]+=del;
}
inline void make(int o,int l,int r)
{       be[o]=++dq; uy=oo=0;
        k[dq].fail=dq;
        fo(i,l,r)
        {
                int mq=be[o]; int y=bh[i];
                fo(l,zb[y],yb[y])
                {
                        if(!k[mq].c[s[l]])k[mq].c[s[l]]=++dq;
                        mq=k[mq].c[s[l]];
                }
                k[mq].val+=xs[i];
                po[i][++po[i][0]]=mq;
        }
        d[1]=be[o];
        en[o]=dq;
        cd[o]=en[o]-be[o]+1;
        int le=0,ri=1;
```

```
	fo(i,1,cd[o])la[i]=0;
	keep=be[o]-1;
	while(le<ri)
	{
		r=d[++le];
		fo(i,0,4)if(k[r].c[i])
		{
			u=k[r].c[i];
			d[++ri]=u;
			j=k[r].fail;
			for(;j!=be[o]&&!k[j].c[i];)j=k[j].fail;
			if(k[j].c[i]!=u&&k[j].c[i])k[u].fail=k[j].c[i];
				else k[u].fail=be[o];
			llb(k[u].fail-keep,u);
		}
	}
	dfs(be[o]);
	fo(i,be[o],en[o])
	if(k[i].val!=0)cor(o,k[i].dfn,k[i].val),cor(o,k[i].tail+1,-k[i].val);
}
inline void build(int o,int l,int r)
{
	make(o,l,r);
	if(l==r)return;
	int mid=l+r>>1;
	build(o<<1,l,mid); build((o<<1)^1,mid+1,r);
}
inline void modify(int o,int l,int r,int posi,int c,int g)
{
	int wz=po[posi][g];
	cor(o,k[wz].dfn,c); cor(o,k[wz].tail+1,-c);
	if(l==r)return;
	int mid=l+r>>1;
	if(posi<=mid)modify(o<<1,l,mid,posi,c,g+1);
	else modify((o<<1)^1,mid+1,r,posi,c,g+1);
}

inline int ggg(int o,int b)
{
	int yy=0;
	for(;b;b=b-(b&(-b)))yy=yy+tr[b+be[o]-1];
	return yy;
}
inline int get(int a,int b)
```

```
{
	for(int l=19;l>=0;--l)if(dep[f[b][l]]>=dep[a])b=f[b][l];
	for(int l=19;l>=0;--l)if(f[b][l]!=f[a][l])a=f[a][l],b=f[b][l];
	fs=b;
	if(a!=b)a=fa[a];
	return a;
}
inline int ask(int o,int l,int r,int le,int ri)
{
	if(l==le&&r==ri)
	{
		int lj=0;
		int dq=be[o];
		fo(i,1,lr)
		{
			for(;!k[dq].c[we[i]]&&dq!=be[o];)dq=k[dq].fail;
			if(k[dq].c[we[i]])dq=k[dq].c[we[i]];
			if(dq!=be[o])lj=lj+ggg(o,k[dq].dfn);
		}
		return lj;
	}
	int mid=l+r>>1;
	if(ri<=mid)return ask(o<<1,l,mid,le,ri);
	else if(le>mid)return ask((o<<1)^1,mid+1,r,le,ri);
	else return ask(o<<1,l,mid,le,mid)+ask((o<<1)^1,mid+1,r,mid+1,ri);
}
int main()
{
	scanf("%d%d",&n,&tp);
	zy['A']=0; zy['C']=1; zy['G']=2; zy['T']=3; zy['U']=4;
	fo(i,1,n)
	{
		scanf("%s",t+1);
		zb[i]=yb[i-1]+1;
		yb[i]=yb[i-1]+strlen(t+1);
		fo(l,zb[i],yb[i])s[l]=zy[t[l-yb[i-1]]];
	}
	fo(i,1,n)scanf("%d",&v[i]);
	fo(i,1,n-1)
	{
		scanf("%d%d",&x,&y);
		llb(x,y); llb(y,x);
	}
	fa[1]=-1; dep[1]=1;
```

```
    dg(1);
    build(1,1,op);
    scanf("%d",&q);
    ans=0;
    fo(i,1,q)
    {
        scanf("%d%d%d",&rp,&x,&y);
        x^=(ans*tp);
        y^=(ans*tp);
        if(rp==2)
        {
            int zl=y-v[x];
            modify(1,1,op,pre[x],zl,1);
            modify(1,1,op,bac[x],-zl,1);
            v[x]=y;
        }
        else
        {
            scanf("%s",t+1);
            lr=strlen(t+1);
            fo(l,1,lr)we[l]=zy[t[l]];
            if(dep[x]>dep[y])swap(x,y);
            int lca=get(x,y);
            if(lca==x)ans=ask(1,1,op,pre[x],pre[y]);
            else ans=ask(1,1,op,pre[lca],pre[x])+ask(1,1,op,pre[fs],pre[y]);
            printf("%d\n",ans);
        }
    }
}
```