

alpc60 ACM/ICPC程序设计

成长的路.....源



```
2008年9月
<
                   五.
                   5
31
        2
            3
               4
           10 11 12 13
          17 18 <u>19</u> 20
14 <u>15</u> 16
21 22 23 24 25 26 27
28 29 30
              2
                   3
                   10 11
        7
```

常用链接

我的随笔

我的评论

我参与的随笔

留言簿(7)

给我留言

查看公开留言

查看私人留言

随笔分类

ACM/ICPC(20) XML

随笔档案

2008年9月 (3)

2008年8月 (4)

2008年7月 (1)

2008年6月 (2)

2008年5月(2)

2008年4月 (5)

2007年9月(3)

文章档案



树状数组学习心得

POJ 3321 Apple Tree

http://acm.pku.edu.cn/JudgeOnline/problem?id=3321

POJ 2481 Cows

http://acm.pku.edu.cn/JudgeOnline/problem?id=2481

POJ 2155 Matrix

http://acm.pku.edu.cn/JudgeOnline/problem?id=2155

今天在POJ淘了这几道题目,学习了一下树状数组的用法,跟大家分享一下心得。可以把树状数组看成一种数据结构,对于一个数组,如果有多次操作,每次的操作有两种: 1、修改数组中某一元素的值,2、求和,求数组元素a[1]+a[2]+...a[num]的和。这是树状数组最基本的应用了,用树状数组可以实现O(log(n))的修改以及O(log(n))的求和。当然用线段树完全可以胜任这些计算,但是线段树写起来代码比较长,并且线段树要占用2*n大小的空间。下面先给出树状数组的三个基本操作的函数:

```
int lowbit(int k)
{
    return k&(-k);
}
//lowbit 函数是计算k的二进制位最低位不为0的数字的权值。
void Modify(int num, int v)
```

```
2008年4月(1)
alpcs
 alpc02
 alpc12
 alpc16
 alpc40
 alpc47
 alpc55
 alpc62
 zzningxp
最新随笔
 1. 树状数组学习心得
 2. 数学中的爆搞题
 3. 最小点覆盖 POJ 2060 Taxi Cab Sc
 heme
 4. 从希望走向失望
 5. POJ 1112 Team Them Up! 求补图
 ,连通分量, DP
 6. 个人比赛时心理素质不好
 7. POJ 2047
 8. 7月16-7月21荒废的一周!!!
 9. POJ 2335 浮点数的gcd
 10. Written to alpcs in Normal
搜索
最新评论 XML
 1. re: A*搜索求最短路
 "使得由s1生成的每状态s2,都有h(s1)
 <=h(s2)"应该是"都有h(s1)<=h(s2)+c(s
 1,s2)"吧
                         --MJ
 2. re: 网络流
 看来主要是建模的问题啊
             --yyz_max@163.com
 3. re: 网络流
 很好很强大
             --yyz_max@163.com
 4. re: 树状数组学习心得
 评论内容较长,点击标题查看
                    --Mr.moon
```

5. re: A*搜索求最短路

评论内容较长,点击标题查看

```
while(num \leq n)
     c[num]+=v;
     num+=lowbit(num);
  }
}
//Modify函数是往数组c中修改值,更新整个数组的值,实现了操作1;
int Sum(int num)
  int ans=0;
  while(num > 0)
     ans+=c[num];
     num-=lowbit(num);
  return ans;
//Sum函数返回数组元素a[1]+a[2]+...a[num]的和,实现操作2;
```

树状数组的巧妙之处在于对于数组下标的二进制的操作,假设a[1...N]为原数组,定义c[1...N]为对应的树状数组:

$$c[i] = a[i - 2^k + 1] + a[i - 2^k + 2] + ... + a[i]$$

其中k为i的二进制表示末尾0的个数,所以 2^k 即为i的二进制表示的最后一个1的权值.

可以把树状数组看作是把数组分成了若干个 2^k 大小的空间。对于一个下标i, c[i]的值是由i/(lowbit(i))个数组元素的值所组成的,每次步进的单位是k=lowbit(i),这个有点像二分归并的思想!这样就可以实现O(log(n))的修改和查询。

下面是树状数组的具体应用:

3321 Apple Tree 一棵树上长了苹果,每一个树枝节点上有长苹果和不长苹果两种状态,两种操作,一种操作能够改变树枝上苹果的状态,另一种操作询问某一树枝节点一下的所有的苹果有多少。具体做法是做一次dfs,记下每个节点的开始时间low[i]和结束时间high[i],那么对于i节点的所有子孙的开始时间和结束时间都应位于low[i]和high[i]之间,另外用一个数组c[i]记录附加在节点i上的苹果的个数,然后用树状数组统计low[i]到high[i]之间的附加苹果总数。这里用树状数组统计区间可以

--pandy

用Sum(high[i])-Sum(low[i]-1)来计算。

```
6. re: POJ 2335 浮点数的gcd
 敬爱的......都不更新了
                      --yumi
 7. re: POJ 2335 浮点数的qcd
 大牛啊。。最近我都在学习你的blog呢
 写的不错啊!!!
                       --11
 8. re: 从希望走向失望
 呵呵
                       --11
 9. re: 从希望走向失望
 这个这个, 玩玩也是不错的, 权作娱乐
 而已,要学还是要自己学的
                    --alpc40
 10. re: POJ 1112 Team Them Up! 求
 补图,连通分量, DP
 不愧是国防科技大学的大牛!!
                    --ssadwlll
阅读排行榜
 1. 动态规划专题(2052)
 2. 树状数组学习心得(1884)
 3. 迭代加深搜索(1821)
 4. A*搜索求最短路(1789)
 5. POJ 1112 Team Them Up! 求补图
 ,连通分量, DP(1558)
评论排行榜
 1. 从希望走向失望(8)
 2. 不用IDE的结果(6)
 3. A*搜索求最短路(6)
 4. 动态规划专题(4)
 5. POJ 2335 浮点数的qcd(3)
```

```
1 #include <stdio.h>
2 #include <string.h>
3 #include <vector>
   using namespace std;
6 //vector<int> g[100005];
7 struct Node
8⊟{
9 | int v;
10 struct Node *next;
11^{\perp} g[100005];
12 int n,m,cnt,low[100005],high[100005],c[100005],flag[100005];
   bool mark[100005];
14
15 void dfs(int v)
16⊟{
17
     struct Node *p=g[v].next;
18
     mark[v]=true;
19 cnt++;
20 \mid low[v]=cnt;
21 | while(p)
22 申 {
23
        if(!mark[p->v])
24
          dfs(p->v);
25
        p=p->next;
26 | }
27
     high[v]=cnt;
28 <sup>L</sup>}
29 int lowbit(int k)
30⊟{
31 return k&(-k);
32 <sup>L</sup>}
33 void Modify(int num, int v)
34⊟{
35
    while(num <= n)</pre>
36草 {
37
        c[num]+=v;
38
        num+=lowbit(num);
39 - }
40 <sup>L</sup>}
41 int Sum(int num)
42⊟{
43
      int ans=0;
44
      while(num > 0)
```

```
45 申 {
46
       ans+=c[num];
47
       num-=lowbit(num);
48 - }
49
     return ans;
50 L}
51
52 int main()
53⊟{
54
     int i,j,a,b,ans;
     char temp[10];
55
56
     struct Node *p;
57
     //freopen("in.txt","r",stdin);
     scanf("%d",&n);
58
59
     memset(g, 0, sizeof(g));
60
      for(i=1; i<n; i++)
61中 {
62
       scanf("%d%d",&a,&b);
63
       p=new Node;
       p->next=g[a].next;
64
65
       p->v=b;
66
       g[a].next=p;
       p=new Node;
67
68
        p->next=g[b].next;
69
       p->v=a;
70
       g[b].next=p;
71 | }
72
     memset(mark,false,sizeof(mark));
     memset(c, 0, sizeof(c));
73
74
     for(i=1; i<=n; i++)
75
       flag[i]=1;
     cnt=0;
76
77
     dfs(1);
     scanf("%d",&m);
78
79
     while(m--)
80阜 {
81
       scanf("%s",temp);
82
       if(temp[0] == 'Q')
83草
84
         scanf("%d",&a);
85
         ans=high[a]-low[a]+1+Sum(high[a])-Sum(low[a]-1);
86
          printf("%d\n",ans);
87 |
       }
88
        else
89草
        {
90
         scanf("%d",&a);
91
         if(flag[a]) Modify(low[a],-1);
92
         else Modify(low[a],1);
```

2481 Cows 给n个区间[Si,Ei],区间[Sj,Ej]< [Si,Ei] 有 Si <= Sj and Ej <= Ei and Ei - Si > Ej - Sj。按y坐标从小到达, \mathbf{x} 坐标从大到小的顺序排序,然后从后往前扫描,记录i之前所有的j区间Sj<Si的个数,这个用树状数组实现。扫描一遍可得出结果。

```
1 #include <stdio.h>
2 #include < string.h >
3 #include <algorithm>
4 using namespace std;
5
6 struct P
7⊟{
8 \mid int x, y, id;
9^{\perp}}p[100005];
10 int n,a[100005],max_n,b[100005];
11
12 int lowbit(int k)
13⊟{
14 return k&(-k);
15 <sup>L</sup>}
16 void Modify(int num, int v)
17⊟{
18 while(num <= max_n)
19草 {
20
        a[num]+=v;
21
       num+=lowbit(num);
22 | }
23 <sup>L</sup>}
24 int Sum(int num)
25⊟{
26 | int ans=0;
27 | if (num <= 0) return 0;
28 while(num)
29 申 {
30
        ans+=a[num];
31
       num-=lowbit(num);
32 | }
```

```
33
      return ans;
34 <sup>L</sup>}
35 bool operator < (const P a, const P b)
36⊟{
37 if (a.y == b.y) return a.x > b.x;
38
      return a.y < b.y;
39 <sup>L</sup>}
40
41 int main()
42□{
43
      int i;
44
      //freopen("in.txt","r",stdin);
45
      while(scanf("%d",&n), n)
47
        max_n=0;
48
        for(i=0; i<n; i++)
49草
50
          scanf("%d%d",&p[i].x,&p[i].y);
51
          p[i].id=i;
52
          p[i].x++;
53
          p[i].y++;
          if(p[i].y > max_n) max_n = p[i].y;
54
55 -
        }
56
        sort(p,p+n);
        memset(a, 0, sizeof(a));
57
        for (i=n-1; i>=0; i--)
58
59草
60
           if(i != n-1 \&\& p[i].y == p[i+1].y \&\& p[i].x == p[i+1].x)
61
            b[p[i].id]=b[p[i+1].id];
62
          else
63
             b[p[i].id]=Sum(p[i].x);
64
          Modify(p[i].x,1);
65 -
        }
66
        for(i=0; i<n; i++)
67草
68
           if(i) printf(" ");
69
           printf("%d",b[i]);
70 -
        }
71
        printf("\n");
72 | }
73
      return 0;
74 <sup>L</sup>}
75
76
```

2155 Matrix 有**n*****n**的**0**, **1**矩阵, 两种操作, **1**、翻转矩形 (**x1**,**y1**) (**x2**,**y2**)的 值, **2**、输出位置为(**x**,**y**)矩阵处的值。先考虑一维的情况,设**A**<**B**,那么要翻转[**A**,**B**]之

间的值,可以分解为两步操作,先翻转[1,A-1],然后再翻转[1,B],其中翻转的次数就可以用树状数组来计算。然后再将次操作扩展到二维的情形,只需将x方向与y方向套成一个二重循环即可。从这里我们也可以看到树状数组处理类似问题时比线段树的优越性。从代码的长度,空间消耗上面,树状数组都有明显的优势。

```
1 #include <stdio.h>
   #include <string.h>
3
   int a[1005][1005],n,m;
5
6 int lowbit(int k)
7⊟{
8
     return k&(-k);
9 <sup>L</sup>}
10 void Modify(int x1, int y1, int x2, int y2)
11⊟{
12 | int i,j;
      for(i=x1-1; i>0; i-=lowbit(i))
13
14 中 {
15
        for(j=y1-1; j>0; j-=lowbit(j))
16草
        {
17
          a[i][j]^=1;
18 -
19
        for (j=y2; j>0; j=lowbit(j))
20草
        {
21
          a[i][j]^{=1};
22 |
        }
23 | }
24
      for(i=x2; i>0; i-=lowbit(i))
25 中 {
26
        for(j=y1-1; j>0; j-=lowbit(j))
27申
        {
28
          a[i][j]^=1;
29
30
        for (j=y2; j>0; j=lowbit(j))
31阜
        {
32
          a[i][j]^=1;
33
34 - }
35 <sup>L</sup>}
36 int Sum(int x, int y)
37⊟{
38
      int ans=0,i,j;
39
      for(i=x; i<=n; i+=lowbit(i))</pre>
40阜 {
41
        for (j=y; j<=n; j+=lowbit(j))
42
          ans^=a[i][j];
43 - }
```

```
44
      return ans;
45 <sup>L</sup>}
46
47 int main()
48⊟{
49 int i,j,x1,x2,y1,y2,cases,ic=0;
50 | char temp[10];
51 //freopen("in.txt","r",stdin);
52
      scanf("%d",&cases);
      while(cases--)
53
54 申 {
55
        if(ic++) printf("\n");
56
        scanf("%d%d",&n,&m);
57
        memset(a, 0, sizeof(a));
58
        while(m--)
59草
        {
60
          scanf("%s",temp);
61
          if(temp[0] == 'C')
62草
          {
63
            scanf("%d%d%d%d",&x1,&y1,&x2,&y2);
64
            Modify(x1,y1,x2,y2);
65 -
          }
66
          else
67草
68
            scanf("%d%d",&x1,&y1);
            printf("%d\n",Sum(x1,y1));
69 l
70 <del>|</del>
          }
71 |-
        }
72 | }
73 | return 0;
74 <sup>L</sup>}
75
76
```

posted on 2008-09-24 16:35 <u>飞飞</u> 阅读(1885) <u>评论(1)</u> <u>编辑 收藏 引用</u> 所属分类: <u>ACM/ICPC</u>

FeedBack:

刷新评论列表

博问-解决您的IT难题

博客园 博问 IT新闻 C++程序员招聘

标题

姓名

主页

验证码

* 6645

内容(提交失败后,可以通过"恢复上次提交"恢复刚刚提交的内容)

Remember Me?

登录 使用高级评论 新用户注册 返回页首 恢复上次提交

[使用Ctrl+Enter键可以直接提交]



IT新闻:

- · 美国研制黑猩猩机器人 用于人猿交流
- · 如何开启Windows 8新视觉主题Aero Lite
- · 郑志昊: 开放平台成助推互联网发展新动力
- · AT&T公司开始接受消费者Lumia900预订单
- · 微软不希望出现Metro版avast!

博客园首页随笔:

- · WPF使用--数据编辑GridView
- · Portal-Basic Web 应用开发框架 —— 前言

- · Contoso 大学 2 实现基本的增删改查
- · Android学习笔记——Activity的启动和创建
- · MOON.ORM 3.5 MYSQL的配置及使用方法(最新版免费下载使用.欢迎加盟)

知识库:

- · 论道WP (一): 你为什么选择Windows Phone?
- · 减少javascript垃圾回收[译]
- · 心如止水的程序员
- · 解决「问题」, 不要解决问题
- · 为什么我不再做.NET开发

相关文章:

数学中的爆搞题

最小点覆盖 POJ 2060 Taxi Cab Scheme

从希望走向失望

POJ 1112 Team Them Up! 求补图, 连通分量, DP

个人比赛时心理素质不好

POJ 2047

7月16-7月21荒废的一周!!!

POJ 2335 浮点数的gcd

Written to alpcs in Normal

网站导航: 博客园 IT新闻 BlogJava 知识库 程序员招聘 管理



Copyright ©2012 飞飞 Powered By 博客园 模板提供: 沪江博