poj2823 单调队列（含单调队列的学习）

转自：http://www.sunhongfeng.com/2011/07/%E5%8D%95%E8%B0%83%E9%98%9F%E5%88%97-poj2823/

他的分析非常到位，顺便把单调队列给学了。很好，所以转了他的这篇文章。程序是我后来理解之后自己写的。

看这个问题：An array of size n ≤ 10^6 is given to you. There is a sliding window of size k which is moving from the very left of the array to the very right. You can only see the k numbers in the window. Each time the sliding window moves rightwards by one position.Your task is to determine the maximum and minimum values in the sliding window at each position.

也就是有一个数列a，要求你求数列b和c，b[i]是a[i]...a[i+w-1]中的最小值，c[i]是最大值。如果a是1,3,-1,-3,5,3,6,7，则b为-1,-3,-3,-3,3,3，c为3,3,5,5,6,7。

这个问题相当于一个数据流（数列a）在不断地到来，而数据是不断过期的，相当于我们只能保存有限的数据（sliding window中的数据，此题中就是窗口的宽度w），对于到来的查询（此题中查询是每时刻都有的），我们要返回当前滑动窗口中的最大值\最小值。注意，元素是不断过期的。

解决这个问题可以使用一种叫做单调队列的数据结构，它维护这样一种队列：

a)从队头到队尾，元素在我们所关注的指标下是递减的（严格递减，而不是非递增），比如查询如果每次问的是窗口内的最小值，那么队列中元素从左至右就应该递增，如果每次问的是窗口内的最大值，则应该递减，依此类推。这是为了保证每次查询只需要取队头元素。

b)从队头到队尾，元素对应的时刻（此题中是该元素在数列a中的下标）是递增的，但不要求连续，这是为了保证最左面的元素总是最先过期，且每当有新元素来临的时候一定是插入队尾。

满足以上两点的队列就是单调队列，首先，只有第一个元素的序列一定是单调队列。

那么怎么维护这个单调队列呢？无非是处理插入和查询两个操作。

对于插入，由于性质b，因此来的新元素插入到队列的最后就能维持b)继续成立。但是为了维护a)的成立，即元素在我们关注的指标下递减，从队尾插入新元素的时候可能要删除队尾的一些元素，具体说来就是，找到第一个大于（在所关注指标下）新元素的元素，删除其后所有元素，并将新元素插于其后。因为所有被删除的元素都比新元素要小，而且比新元素要旧，因此在以后的任何查询中都不可能成为答案，所以可以放心删除。

对于查询，由于性质b，因此所有该时刻过期的元素一定都集中在队头，因此利用查询的时机删除队头所有过期的元素，在不含过期元素后，队头得元素就是查询的答案（性质a），将其返回即可。

由于每个元素都进队出队一次，因此摊销复杂度为O(n)。

POJ2823就是上面描述的那道题。

我的程序：

10095685	gxmshxj	2823	Accepted

```
#include <iostream>
#include <fstream>

using namespace std;

#define MAX 1000001
int A[MAX]; //存储数据
int Q[MAX]; //队列
```

公告

昵称：Jason Damon
园龄：3年7个月
粉丝：67
关注：130
+加关注

< 2012年4月 >

日	一	二	三	四	五	六
25	26	27	28	29	30	31
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	1	2	3	4	5

搜索

常用链接

我的随笔
我的评论
我的参与
最新评论
我的标签
更多链接

我的标签

算法 (49) 数据结构 (30)
ACM 图算法 (16)
javascript (15)
ACM 搜索 (14) 心灵日记 (13)
ACM 字符串 (12) ACM (9)
ACM 线段树 (7) vim (7)
更多

随笔档案(256)

2014年10月 (2)
2014年6月 (6)
2014年5月 (1)
2014年4月 (3)
2013年11月 (2)
2013年10月 (6)

```
int P[MAX]; //存储A[i]中的下标i
int Min[MAX];  //输出最小
int Max[MAX];  //输出最大
int n,k;

void get_min()
{
    int i;
    int head=1,tail=0;
    for(i=0; i<k-1; i++)  //先把前两个入队
    {
        while(head<=tail && Q[tail]>=A[i])  //队尾元素大于要插入的数
            --tail;
        Q[++tail]=A[i];
        P[tail]=i;
    }

    for(; i<n; i++)
    {
        while(head<=tail && Q[tail]>=A[i])
            --tail;
        Q[++tail]=A[i];
        P[tail]=i;
        while(P[head]<i-k+1)  //判断数是否过时，即窗口是否已经划过这个数，我这是从0开始计数的。
        {
            head++;
        }
        Min[i-k+1]=Q[head];
    }
}

void get_max()
{
    int i;
    int head=1,tail=0;
    for(i=0; i<k-1; i++)
    {
        while(head<=tail && Q[tail]<=A[i])   //队尾元素小于要插入的值
            --tail;
        Q[++tail]=A[i];
        P[tail]=i;
    }

    for(; i<n; i++)
    {

        while(head<=tail && Q[tail]<=A[i])   //队尾元素小于要插入的值
            --tail;
        Q[++tail]=A[i];
        P[tail]=i;
        while(P[head]<i-k+1)
            {
            head++;
        }
        Max[i-k+1]=Q[head];
    }
}

void output()
{
       int i;
      //输出最下值
       for(i=0; i<n-k+1; i++)
       {
           if(i==0)
               printf("%d",Min[i]);
           else
               printf(" %d",Min[i]);
       }
       printf("\n");
       //输出最大值
       for(i=0; i<n-k+1; i++)
       {
           if(i==0)
               printf("%d",Max[i]);
           else
```


最新评论

阅读排行榜

评论排行榜

推荐排行榜

```
                printf(" %d",Max[i]);
        }
        printf("\n");
}


int main()
{
    int i;
    freopen("acm.txt","r",stdin);
    scanf("%d%d",&n,&k);
    for(i=0; i<n; i++)
    {
        scanf("%d",&A[i]);
    }
    get_min();
    get_max();
    output();
    return 0;
}
```

我是一名在校大学生，热爱编程，虽然起步晚了些，但我会努力的。呵呵！ 数据结构 算法

标签: 数据结构

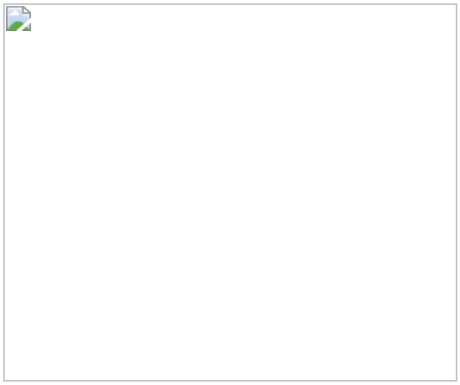
Jason Damon
关注 - 130
粉丝 - 67
+加关注

0 0

(请您对文章做出评价)

« 上一篇： 数据结构之“堆”
» 下一篇： poj2092 简单排序

posted @ 2012-04-19 21:33 Jason Damon 阅读(1386) 评论(0) 编辑 收藏



最新IT新闻:
· Windows平台主设计师Albert Shum展示部分Win 10设计定稿
· 微软发布Project Oxford，供Azure户免费集多项功能
· 我看全栈工程师
· 当微软说它允许移植别家应用的时候，它在说什么？
· 这里申请：Android和iOS应用至Win 10通用应用移植工具内测
» 更多新闻...

最新知识库文章:
· 携程App的网络性能优化实践
· 技术领导力：作为技术团队领导经常为人所忽略的技能和职责
· 在LinkedIn做面试官的故事
· 架构之重构的12条军规（上）
· 序列化和反序列化
» 更多知识库文章...