计算机算法设计与分析

朱双贺　2010E8009070012

第七次作业:股票增值问题的贪心算法

题目：某公司过去 n 天的股票价格波动保存在数组 A 中，求在哪段时间（连续哪几天）其股价的累计增长最大。例如：过去七天内其股价波动序列为 +3，-6，+5,+2,-3,+4,-4,累计增长最大值出现在第 3 天到第 6 天，累计增长值为 5+2-3+4=8,。实际上就是求 i 和 j(0<=i<=j<=n-1)，使$\sum_{k=i}^{j} A[k]$最大。是设计贪心算法解决上述问题，证明算法的正确性并分析其时间复杂度。

解:

贪心算法设计：

1. 原问题可看作找出一个(i,j)，使得$\sum_{k=i}^{j} A[k]$最大。可在数组 A 中找出所有可能的(i,j),即局部最优解，存入数组 I[]、J[]中（p 为下标）。
2. 每趟求(I[p],J[p])时，找出遇到的第一个正数的下标 i，另 I[p]=J[p]=I;A[I[p]]记为 left,找到下一个正数 A[i]，记为 right，sum=$\sum_{k=I[p]}^{i} A[k]$，若 sum 大于 left 和 right，则另 J[p]=i，继续找出下一个正数并重复上述过程直到 i=n-1。若 sum 小于 left 或 right，则本趟结束，找到了一个(I[p],J[p])，将该连续序列和$\sum_{k=I[p]}^{J[p]} A[k]$存入 Sum[p]。继续下一趟求(I[p+1],J[p+1])的过程,直到 i=n-1。
3. 找出 Sum[]中的最大值，其下标 y 对应的 I[y],J[y]记为所求的 i 和 j，Max 为最大和的值。

算法如下：

Algorithm Find_MaxSum(A,n)

```
i=p=y=Max=0;
while i<n-1
    if A[i]>0
        left=right=sum=0;
        I[p]=J[p]=i;
        left=A[i];
        i=Find_next(i);
        right=A[i];
        sum=Sum(I[p],i);
        while sum>left && sum>right && i<n-1
            J[p]=i;
            left=sum;
            i=Find_next(i);
            right=A[i];
            sum=sum+Sum(J[p]+1,i);
        end
        Sum[p]=Sum(I[p],J[p]);
        if Sum[p]>Max then do Max=Sum[p]; y=p;
        p++;
    end
end
i=I[y], j=J[y];
```

Algorithm Find_next(i)

```
i++;
while A[i]<=0 && i<n-1 then do i++;
return   i;
```

Algorithm Sum(i,j)

```
a=0;
for k=i;k<=j;k++
    a=a+A[k];
return a;
```

证明算法的正确性：

只要证明：

1. 上述算法所求的 I[p],J[p]都是局部最优解；
2. 原问题的最优解一定在局部最优解中，且是其中最大的一个；

对于 1，上述算法求解 I[p],J[p]得过程已经证明 I[p],J[p]就是局部最优解；

对于 2，采用反证法：假设(m,n)为全局最优解，即 Sum(m,n)大于任何 Sum(i,j)，但(m,n)不是局部最优解，则存在局部最优解(m’,n’)使得 Sum(m’,n’)>Sum(m,n)，这与假设相矛盾。所以全局最优解也是局部最优解。又因为 Sum(m,n)大于任何 Sum(i,j)，所以 Sum(m,n)为局部最优解中序列和最大的一个。所以全局最优解也一定是局部最优解，且是最大的一个。

综上所述，该算法所得结果正确。

分析时间复杂度：

函数 **Find_next(i)**的比较次数记为 x，显然 x 最多为 n-i，而整个算法中在 **Find_next(i)**中执行的比较操作显然不会大于 n，否则 i 将大于 n-1，算法结束；

函数 **Sum(i,j)**中的循环次数显然与 **Find_next(i)**的比较次数相等;

除去在 **Find_next(i)**中执行的比较操作和在 **Sum(i,j)**中执行的加法操作，函数 **Find_MaxSum(A,n)**中的比较次数（加法次数）不会大于 4p；p 为局部最优解的个数，最坏情况下可能有 n/2 个局部最优解，如 4，-5，6，-7，8，-9，10，-11……，所以 4p<=2n;

所以算法的整体比较次数（还有加法次数）<=2(n-1)+2n=4n-2;

所以算法的时间复杂度为：**A(n)∈O(n)**
