ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC MÁY TÍNH

MÔ HÌNH HÓA TOÁN HỌC

Minimal-sum section problem

GVHD: Nguyễn An Khương
Sinh viên thực hiện: Nguyễn Thanh Hiên – 1810913

HCMC, Ngày 12 tháng 1 năm 2020

Mục lục

1 Problem: minimal-sum section

1.1 The problem and the solution

Let a[0], . . . , a[n− 1] be the integer values of an array a. A section of a is a continuous piece a[i], . . . ,a[j], where $0 \leq i \leq j < n$.We write $S_{i,j}$ for the sum of that section: a[i] + a[i + 1] + · · · + a[j]. A minimal-sum section is a section a[i], . . . , a[j] of a such that the sum $S_{i,j}$ is less than or equal to the sum $S_{i',j'}$ of any other section a[i'], . . . , a[j'] of a.

1.2 Pseudo code

Listing 1: Pseudo code of minimal section algorithm

```
k = 1;
t = a[0];
s = a[0];
while (k != n) {
t = min(t + a[k], a[k]);
s = min(s,t);
k = k + 1;
}
```

1.3 C++ code for minimal-sum section algorithm

Listing 2: Minimal-sum using C++ implementation.

```
#include<iostream>
#include<string>
#include<fstream>
#include<cmath>
#include<cstdio>
#include <algorithm>

using namespace std;

bool read_file(ifstream &infile,int *&a,int&N)
{
    if(!infile) return false;
    infile>>N;
    a=new int[N];
    for (int i=0;i<N;i++)
                infile>>a[i];
    return true;
}

int main(int argc,char* argv[])
{
    ifstream infile;

    int N;
    int i=0;
    int *a;
    do
        {
            infile.open(argv[++i]);
            if(read_file(infile,a,N))
            {
                int k = 1;
                int t = a[0];
                int s = a[0];
                while (k != N) {
                    t = min(t + a[k], a[k]);
                    s = min(s,t);
                    k++;
                }
                cout<<s<<endl;
            }
            infile.close();
        }while (infile);


    return 0;
}
```

2 The following problems with the algorithm

2.1 If we swap the first and second assignment in the while-statement of Min Sum, so that it first assigns to s and then to t, is the program still correct? Justify your answer.(Question 4.3.22)

Note: This definition is used for the following problems.

Inv1(s, k) = $\forall$i, j (0 $\leq$ i $\leq$ j < k $\rightarrow$ s $\leq S_{i,j}$)

If we swap the placements of those two variables, the program will be incorrect. The outcome value (s) will not calculate the element a[n] in the minimal-sum section since we consider the loop:

while (k!=n){

$(\!|$Inv1(s,k) $\wedge$ Inv2(t,k)$\wedge$ k$\neq$n$|\!)$

s = min(s,t);

$(\!|$Inv1(min(s,t),k) $\wedge$ Inv2(min(t + a[k],a[k]),k+1)$|\!)$

t = min(t + a[k], a[k]);

$(\!|$Inv1(min(s,t),k) $\wedge$ Inv2(t,k +1)$|\!)$

k = k + 1;

$(\!|$Inv1(s, k-1) $\wedge$ Inv2(t, k)$|\!)$ } $(\!|$Inv1(s, k) $\wedge$ Inv2(t, k) $\wedge$ $\neg\neg$(k = n)$|\!)$

Conclusion: The outcome value (s) is only correct with n-1 elements in the array. If the algorithm is changed this way, it is incorrect.

2.2 Prove the partial correctness of S2 for Min_ Sum.

Note: Both the following definitions are used for the following problems.

S1.⦇T⦈ Min_Sum ⦇ $\forall$ i, j (0 $\leq$ i $\leq$ j < n $\rightarrow$ s = $S_{i,j}$ ⦈

S2.⦇T⦈ Min_Sum ⦇ $\exists$ i, j (0 $\leq$ i $\leq$ j < n $\wedge$ s = $S_{i,j}$ ⦈

⦇T⦈	
⦇a[1]=S(1,1)⦈	Implied
⦇ $\exists$ i, j(i $\leq$ j $\leq$ n $\wedge$ a[1] = S(i,j))⦈	Implied
k = 2;	
t = a[1]	
s = a[1]	
⦇ $\exists$ i, j(i $\leq$ j $\leq$ n $\wedge$ s = S(i,j))⦈	Assignment
while (k !=n){	
⦇ $\exists$ i, j(i $\leq$ j $\leq$ n $\wedge$ s = S(i,j))⦈	Invariant Hyp. $\wedge$ guard
⦇ $\exists$ i, j(i $\leq$ j $\leq$ n $\wedge$ min(s,min(t+a[k],a[k]))=S(i,j)⦈	Implied
t=min(t+a[k],a[k]);	
⦇ $\exists$ i, j(i $\leq$ j $\leq$ n $\wedge$ min(s,t)=S(i,j)⦈	Assignment
s = min(s,t);	
⦇ $\exists$ i, j(i $\leq$ j $\leq$ n $\wedge$ s=S(i,j)⦈	Assignment
k=k+1;	
⦇ $\exists$ i, j(i $\leq$ j $\leq$ n $\wedge$ s=S(i,j)⦈	Assignment
}	
⦇ $\exists$ i, j(i $\leq$ j $\leq$ n $\wedge$ s=S(i,j)⦈	Partial while

2.3 The program Min Sum does not reveal where a minimal-sum section may be found in an input array. Adapt Min_ Sum to achieve that. Can you do this with a single pass through the array?

Finding the section with a single pass through the array is not complicated. We add two lines to store the addresses of the beginning and the ending point of the section into 2 variables "start" and "end":

if (a[k]==t)start=k;

if (s==t)end=k;

The loop will be like this:

```
while (k != N) {
        t = min(t + a[k], a[k]);
        if (a[k]==t)start=k;
        s = min(s,t);
        if (s==t)end=k;
        k++;
}
```

Simply, the "start" will be assigned as k whenever the new element is the better suitable beginning for the minimal-sum section. And when the program found the better minimal-sum section than the old "s" has, "end" variable will be assigned as the current k.

2.4 Consider the proof rule

$$\frac{(\!|\phi|\!)C(\!|\varphi_1|\!) \qquad (\!|\phi|\!)C(\!|\varphi_2|\!)}{(\!|\phi|\!)C(\!|\varphi_1 \wedge \varphi_2|\!)}\text{Conj}$$

Explain how this rule, or its derived version, is used to establish the overall correctness of Min_Sum.

We consider ϕ (Inv(a[0],1)). After the implementation of Min_Sum, ϕ is Inv1(s, k) and Inv2(t, k). They are considered φ1and φ2. Hence, the proof rule is used as:

(| Inv(a[0],1) |) Min_Sum (| Inv2(s, k) |)   (| Inv(a[0],1) |) Min_Sum (| Inv2(t, k) |)

The conjunction is used from the beginning of the algorithm in (|Inv1(a[0],1) $\wedge$ Inv2(a[0],1)|) and the program keeps processing until it achieves the result.

2.5 Prove total correctness of S1 and S2 for Min Sum

S1:

$(\!|T|\!)$

$(\!|a[1]=S(1,1)|\!)$ Implied

$(\!|\forall i, j(i \leq j \leq n \wedge a[1] = S(i,j))|\!)$ Implied

k = 2;

t = a[1];

s = a[1];

$(\!|\forall i, j(i \leq j \leq n \wedge s = S(i,j))|\!)$ Assignment

while (k !=n){

$(\!|\forall i, j(i \leq j \leq n \wedge s = S(i,j))\wedge 0\leq n\text{-}k<n\text{-}1|\!)$ Invariant Hyp.$\wedge$guard

$(\!|\forall i, j(i \leq j \leq n \wedge \min(s,\min(t+a[k],a[k]))=S(i,j)\wedge 0\leq n+1\text{-}k<n\text{-}1|\!)$ Implied

t=min(t+a[k],a[k]);

$(\!|\forall i, j(i \leq j \leq n \wedge \min(s,t)=S(i,j)\wedge 0\leq n\text{-}k<n\text{-}2|\!)$ Assignment

s = min(s,t);

$(\!|\forall i, j(i \leq j \leq n \wedge s=S(i,j)\wedge 0\leq n\text{-}k<n\text{-}2|\!)$ Assignment

k=k+1;

$(\!|\forall i, j(i \leq j \leq n \wedge s=S(i,j)\wedge 0\leq n\text{-}k\text{-}1<n\text{-}2|\!)$ Assignment

}

$(\!|\forall i, j(i \leq j \leq n \wedge s=S(i,j)\wedge k=n|\!)$ Total while

S2:

$\llparenthesis T \rrparenthesis$

$\llparenthesis a[1]=S(1,1) \rrparenthesis$ Implied

$\llparenthesis \exists i, j(i \leq j \leq n \wedge a[1] = S(i,j)) \rrparenthesis$ Implied

k = 2;

t = a[1];

s = a[1];

$\llparenthesis \exists i, j(i \leq j \leq n \wedge s = S(i,j)) \rrparenthesis$ Assignment

while (k !=n){

$\llparenthesis \exists i, j(i \leq j \leq n \wedge s = S(i,j)) \wedge 0 \leq n-k<n-1 \rrparenthesis$ Invariant Hyp.$\wedge$guard

$\llparenthesis \exists i, j(i \leq j \leq n \wedge \min(s,\min(t+a[k],a[k]))=S(i,j) \wedge 0 \leq n-k<n-1 \rrparenthesis$ Implied

t=min(t+a[k],a[k]);

$\llparenthesis \exists i, j(i \leq j \leq n \wedge \min(s,t)=S(i,j) \wedge 0 \leq n-k<n-2 \rrparenthesis$ Assignment

s = min(s,t);

$\llparenthesis \exists i, j(i \leq j \leq n \wedge s=S(i,j) \wedge 0 \leq n-k<n-2 \rrparenthesis$ Assignment

k=k+1;

$\llparenthesis \exists i, j(i \leq j \leq n \wedge s=S(i,j) \wedge 0 \leq n-k-1<n-2 \rrparenthesis$ Assignment

}

$\llparenthesis \exists i, j(i \leq j \leq n \wedge s=S(i,j) \wedge k=n \rrparenthesis$ Total while

Tài liệu

[1] Huth and Ryan. *Logic in computer science*. [*Modelling and Reasoning about Systems*]. Second Edition Chapter 4 Section 3.