

Chapter 7: Formalizing Programming Principles

The Basic Technique: Slope Search

Programming Problem:

Given an array

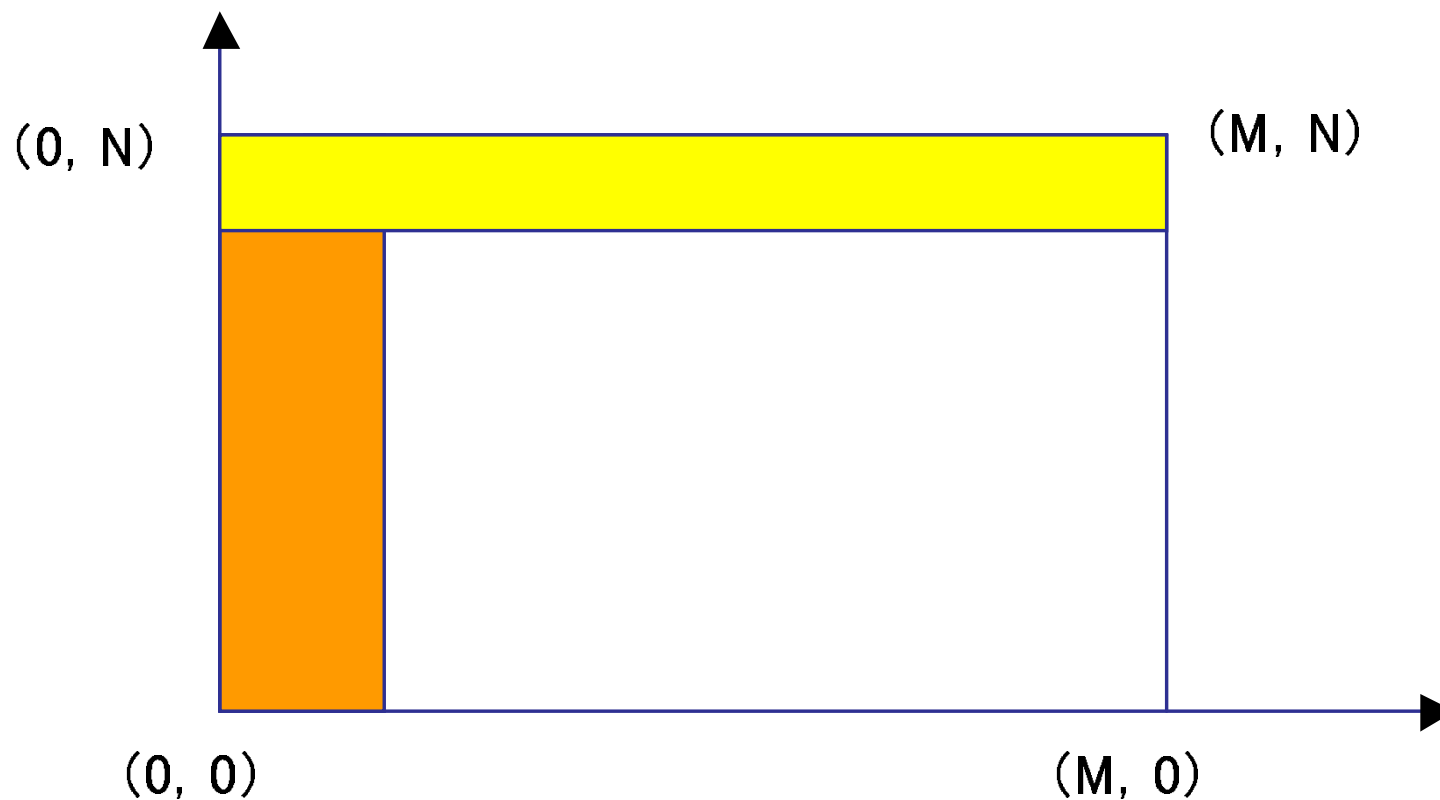
$$f : [0..M] \times [0..N] \rightarrow \mathbb{Z}$$

which is ascending in both arguments, and assuming X occurs in f , derive a program that establishes for integer a and b such that

$$f.a.b = X.$$

The post condition is:

$$R : 0 \leq a \leq M \wedge 0 \leq b \leq N \wedge f.a.b = X$$



Choosing $(0, N)$ for inspection:

$$f.0.N > X \Rightarrow (\forall i : 0 \leq i \leq M : f.i.N > X)$$

$$f.0.N < X \Rightarrow (\forall j : 0 \leq j \leq N : f.0.j < X)$$

To be formal, we define (I, J) be the point satisfying

$$R : 0 \leq I \leq M \wedge 0 \leq J \leq N \wedge f.I.J = X$$

and choose the following invariant:

$$P : 0 \leq a \leq I \wedge J \leq b \leq N$$

which is established by $a, b := 0, N$.

$$\begin{aligned} & f.a.b < X \\ \Rightarrow & \{ f \text{ is ascending in its second argument, } J \leq b \} \\ & f.a.J < X \\ \Rightarrow & \{ f.I.J = X \} \\ & a \neq I \\ \equiv & \{ a \leq I \} \\ & a + 1 \leq I \end{aligned}$$

So

$$P \wedge f.a.b < X \Rightarrow P(a := a + 1)$$

$$\begin{aligned} & f.a.b > X \\ \Rightarrow & \{ f \text{ is ascending in its second argument, } a \leq I \} \\ & f.I.b > X \\ \Rightarrow & \{ f.I.J = X \} \\ & b \neq J \\ \equiv & \{ a \leq I \} \\ & J \leq b - 1 \end{aligned}$$

So

$$P \wedge f.a.b > X \Rightarrow P(b := b - 1)$$

This yields the following slope searching solution:

```

$$\begin{array}{l} a, b := 0, N; \\ \mathbf{do} \ f.a.b < X \rightarrow a := a + 1 \\ \quad [] \ f.a.b > X \rightarrow b := b - 1 \\ \mathbf{od} \end{array}$$

```

It has time complexity of $\mathcal{O}(M + N)$. A similar program can be obtained by choosing $(M, 0)$ as starting point.

Notice the tail invariant:

$$\begin{aligned} & (\exists i, j : 0 \leq i < M \wedge 0 \leq j < N : f.i.j = X) \\ & \equiv (\exists i, j : a \leq i < M \wedge 0 \leq j < b : f.i.j = X) \end{aligned}$$

Application 1: Searching

Given an array

$$f : [0..M] \times [0..N] \rightarrow \mathcal{Z}$$

which is ascending in both arguments, determine whether value X occurs in f .

```
[[  
  con  $M, N, X : int$ ;  
   $f : \mathbf{array} [0..M) \times [0..N) \mathbf{of} int$ ;  
  { $f$  is ascending in both arguments}  
  var  $r : bool$ ;  
   $S$   
  { $r \equiv (\exists i, j : 0 \leq i < M \wedge 0 \leq j < N : f.i.j = X)$ }  
  ]]
```

Following the approach of “slope searching”, we define a “tail” invariant:

$$G.a.b \equiv (\exists i, j : a \leq i < M \wedge 0 \leq j < b : f.i.j = X)$$

So:

$$R : r \equiv G.0.N$$

$$P_0 : r \vee G.a.b \equiv G.0.N$$

$$P_1 : 0 \leq a \leq M \wedge 0 \leq b \leq N$$

It is not difficult to prove that

$$P \wedge (a = M \vee b = 0 \vee r) \Rightarrow R$$

What is the guard?

Investigating an increase of a by 1.

$$\begin{aligned}
 & G.a.b \\
 \equiv & \quad \{ \text{definition of } G \} \\
 & (\exists i, j : a \leq i < M \wedge 0 \leq j < b : f.i.j = X) \\
 \equiv & \quad \{ \text{split off } i = a \} \\
 & G.(a + 1).b \vee (\exists j : 0 \leq j < b : f.a.j = X) \\
 \equiv & \quad \{ \text{assuming } f.a.(b - 1) < X, f \text{ is ascending in its second argument} \} \\
 & G.(a + 1).b \vee \text{false} \\
 \equiv & \quad \{ \text{predicate calculus} \} \\
 & G.(a + 1).b
 \end{aligned}$$

Hence,

$$f.a.(b - 1) < X \Rightarrow (G.a.b = G.(a + 1).b)$$

Similarly(?), we have

$$f.a.(b - 1) > X \Rightarrow (G.a.b = G.a.(b - 1))$$

And for the case $f.a.(b - 1) = X$, we have

$$P_0 \wedge (f.a.(b - 1) = X) \Rightarrow P_0(r := \text{true})$$

Our final program:

```
[[  
  var  $a, b : int$ ;  
   $a, b, r := 0, N, false$ ;  
  do  $a \neq M \wedge b \neq 0 \wedge \neg r \rightarrow$   
    if  $f.a.(b - 1) < X \rightarrow a := a + 1$ ;  
    []  $f.a.(b - 1) > X \rightarrow b := b - 1$ ;  
    []  $f.a.(b - 1) = X \rightarrow r := true$   
    fi  
  od  
]]
```

The time complexity is $\mathcal{O}(M + N)$. How to show this algorithm is optimal?

Application 2: Decomposition in a sum of two squares

Derive a program for the computation of the number of ways in which a natural number N can be written as the sum of two squares.

```
[[  
  con  $N : int \{N \geq 0\};$   
  var  $r : int;$   
   $S$   
   $\{r = (\#x, y : 0 \leq x \leq y : x^2 + y^2 = N)\}$   
  ]]
```

Since $x^2 + y^2$ is increasing in both arguments, we define

$$G.a.b = (\#x, y : a \leq x \leq y \leq b : x^2 + y^2 = N)$$

Hence

$$R : \quad r = G.0.\sqrt{N}$$

$$P_0 : \quad r + G.a.b = G.0.\sqrt{N}$$

$$P_1 : \quad 0 \leq a \leq b \leq \sqrt{N}$$

The invariants can be established by:

$$a, b, r := 0, \sqrt{N}, 0$$

Investigate the effect of increase of a by 1.

$$\begin{aligned}
 & G.a.b \\
 = & \quad \{ \text{definition of } G \} \\
 & (\#x, y : a \leq x \leq y \leq b : x^2 + y^2 = N) \\
 = & \quad \{ \text{split } x = a \} \\
 & G.(a + 1).b + (\#y. a \leq y \leq b : a^2 + y^2 = N) \\
 = & \quad \{ a \leq b \} \\
 & G.(a + 1).b + 0, \quad \text{if } a^2 + b^2 < N \\
 & G.(a + 1).b + 1, \quad \text{if } a^2 + b^2 = N
 \end{aligned}$$

Investigate the effect of decrease of b by 1.

$$\begin{aligned}
 & G.a.b \\
 = & \quad \{ \text{definition of } G \} \\
 & (\#x, y : a \leq x \leq y \leq b : x^2 + y^2 = N) \\
 = & \quad \{ \text{split } y = b \} \\
 & G.a.(b - 1) + (\#x.a \leq x \leq b : x^2 + b^2 = N) \\
 = & \quad \{ a \leq b \} \\
 & G.a.(b - 1) + 0, \quad \text{if } a^2 + b^2 > N \\
 & G.a.(b - 1) + 1, \quad \text{if } a^2 + b^2 = N
 \end{aligned}$$

Our final program:

```
[[  
  var  $a, b : int$ ;  
   $r, a := 0, 0; b := 0$ ; do  $b * b < N \rightarrow b := b + 1$  od;  
  do  $a \leq b \rightarrow$   
    if  $a * a + b * b < N \rightarrow a := a + 1$   
    []  $a * a + b * b > N \rightarrow b := b - 1$   
    []  $a * a + b * b = N \rightarrow r, a := r + 1, a + 1$   
    []  $a * a + b * b = N \rightarrow r, b := r + 1, b - 1$   
    fi  
  od  
]]
```

The time complexity is $\mathcal{O}(\sqrt{N})$.

Application 3: Minimal distance

Derive a program for the computation of the minimal distance of two ascending sequences.

```

[[
  con  $M, N : int \{M \geq 0 \wedge N \geq 0\}$ ;
   $f : \text{array } [0..M) \text{ of } int \{f \text{ is ascending}\}$ ;
   $g : \text{array } [0..N) \text{ of } int \{g \text{ is ascending}\}$ ;
  var  $r : int$ ;
   $S$ 
   $\{r = (\text{min } x, y : 0 \leq x < M \wedge 0 \leq y < N : |f.x - g.y|)\}$ 
]]

```

Note that $f.x - g.y$ is increasing in x and decreasing in y , but $|f.x - g.y|$ does not have this property. Anyway, slope search is still possible.

Define $G.a.b$ by

$$G.a.b = (\mathbf{min} \ x, y : a \leq x < M \wedge b \leq y < N : |f.x - g.y|)$$

So

$$R : \quad r = G.0.0$$

$$P_0 : \quad r \ \mathbf{min} \ G.a.b = G.0.0$$

$$P_1 : \quad 0 \leq a \leq M \wedge 0 \leq b \leq N$$

The invariants can be established by

$$a, b, r = 0, 0, \infty$$

Furthermore

$$P_0 \wedge (a = M \vee b = N) \Rightarrow R$$

$$\begin{aligned}
& G.a.b \\
= & \quad \{ \text{definition of } G \} \\
& (\mathbf{min} \ x, y : a \leq x < M \wedge b \leq y < N : |f.x - g.y|) \\
= & \quad \{ \text{split off } x = a \} \\
& G.(a + 1).b \mathbf{min} (\mathbf{min} \ y : b \leq y < N : |f.a - g.y|) \\
= & \quad \{ \text{assuming } g.b \geq f.a, g \text{ is increasing} \} \\
& G.(a + 1).b \mathbf{min} (\mathbf{min} \ y : b \leq y < N : g.y - f.a) \\
= & \quad \{ g \text{ is increasing} \} \\
& G.(a + 1).b \mathbf{min} (g.b - f.a)
\end{aligned}$$

Hence,

$$g.b \geq f.a \Rightarrow G.a.b = G.(a + 1).b \mathbf{min} (g.b - f.a)$$

Symmetrically,

$$f.a \geq g.b \Rightarrow G.a.b = G.a.(b + 1) \mathbf{min} (f.a - g.b)$$

Exercises

Problem 8:

Derive an efficient program S satisfying the following specification:

```
[[  
  con  $N : int \{N \geq 0\};$  var  $r : bool;$   
   $S$   
   $\{r \equiv (\exists x, y : 0 \leq x \wedge 0 \leq y : N = 2^x + 3^y)\}$   
]]
```