The lexicographic relationship between two arrays

Introduction

This note is a demonstration of an application of the Bounded Linear Search .

Our task is to design a program which determines the lexicographic relationship between two integer arrays $f[0..N)$ and $g[0..N)$. In particular, using $f \prec g$ to denote that f lexicographically precedes g , our program should assign to boolean variables a , b , and c such values that:

$$(a \equiv f \prec g) \quad \wedge \quad (b \equiv f = g) \quad \wedge \quad (c \equiv g \prec f) \qquad .$$

We are free to choose a refinement for $\prec$ as we see fit.

A formal specification

We may formally specify the problem as follows:

$|[$ **con** N : **int** ; f, g : **array** $[0..N)$ of **int** ;
var a, b, c : **bool** ;
$\{ Q \}$
Lexicographic order
$\{ R \}$
$]|$.

The precondition is:

Q : **true** .

The postcondition is:

R : $(a \equiv f \prec g) \quad \wedge \quad (b \equiv f = g) \quad \wedge \quad (c \equiv g \prec f)$.

Analyzing the postcondition

Our task is to design a program fragment with postcondition R . Since we are required to establish a relationship between arbitrary arrays, we might try to design a repetitive construct having R as its postcondition. We would then look to use R to design

an invariant and guard for this repetitive construct. However, the usual techniques for designing an invariant and guard from a given postcondition do not work very well with R .

Thus we are led to try and design another type of program construct which establishes R . Since there is little to guide us in the design of such a construct, we will try to formulate some simple properties of $\prec$ in the hope that they will serve as the required guide.

A simple property of $\prec$

One simple observation related to the lexicographic order $\prec$ is: for any f and g of the same size, exactly one of $f \prec g$, $f = g$, and $g \prec f$ will be **true** . Consequently, exactly one of a , b , and c will be **true** upon termination.

Selecting a suitable construct

The observation above leads us to consider a selection statement having the following shape:

```
|[
  { ¬a ∧ ¬b ∧ ¬c }
  if  ?   →   a := true   { R }
  []  ?   →   b := true   { R }
  []  ?   →   c := true   { R }
  fi
  { R }
]|
```

We must now design the guards of this selection statement. This involves identifying the precise conditions under which each of the assignments establish R . We shall derive these conditions in the next section.

Deriving the conditions under which each of the assignments establish R

We calculate the conditions in the context of the precondition of the selection statement:

$[\![$ Context: $\quad \neg\, a \quad \wedge \quad \neg\, b \quad \wedge \quad \neg\, c$

$wp.(a := \mathbf{true}).R$

$\equiv$ { Axiom of Assignment }

$\mathbf{true} \;\equiv\; f \prec g \qquad \wedge \qquad b \;\equiv\; f = g \qquad \wedge \qquad c \;\equiv\; g \prec f$

$\equiv$ { Property of $\prec$; Predicate Calculus }

$f \prec g \quad \wedge \quad \neg\, b \quad \wedge \quad \neg\, c$

$\equiv$ { Context }

$f \prec g$

$\Leftarrow$ { Using the following —standard— specification of $\prec$ }

$0 \le x < N \quad \wedge \quad f.x < g.x \quad \wedge \quad \langle \forall i : 0 \le i < x : \; f.i = g.i \rangle \qquad .$

$]\!]$

In the above, we have used the following specification for $\prec$:

$$f \prec g \quad \equiv \quad \langle \exists n : \; 0 \le n < N : \; f.n < g.n \; \wedge \; \langle \forall i : \; 0 \le i < n : \; f.i = g.i \; \rangle \; \rangle \quad .$$

Additionally, we introduced a fresh integer program variable x . In the interests of homogeneity, we shall use this same variable in the following calculations.

In a similar vein, we calculate:

$[\![$ Context: $\quad \neg\, a \quad \wedge \quad \neg\, b \quad \wedge \quad \neg\, c$

$wp.(c := \mathbf{true}).R$

$\Leftarrow$ { As above, details left to the reader }

$0 \le x < N \quad \wedge \quad g.x < f.x \quad \wedge \quad \langle \forall i : 0 \le i < x : \; f.i = g.i \rangle \qquad .$

$]\!]$

Finally,

$[\![$ Context: $\quad \neg\, a \quad \wedge \quad \neg\, b \quad \wedge \quad \neg\, c$

$wp.(b := \mathbf{true}).R$

$\equiv$ { Axiom of Assignment; Predicate Calculus; Context. (As in the first calculation) }

$f = g$

$\equiv$ { Refining $f = g$ }

$\langle \forall i : 0 \leq i < N : f.i = g.i \rangle$

$\Leftarrow$ { Predicate Calculus }

$0 \leq x = N \quad \wedge \quad \langle \forall i : 0 \leq i < x : f.i = g.i \rangle$.

$\rrbracket$

Thus we have identified the conditions under which assigning the value **true** to each of a , b , and c will establish the postcondition R .

Choosing the guards

he above conditions may be satisfied by placing some of the conjuncts in the precondition of the selection statement, while choosing others as guards.

In general, it is advisable to choose the guards to be as weak as possible. Thus, we shall choose to place all similar conjuncts in the precondition of the selection statement, since this will generally lead to weaker guards. Furthermore, we will try to choose our guards in such a way that the proof of non-abortion of the selection statement is straightforward.

We now proceed to place the expressions we have derived above in either the precondition of the selection or in the guards.

The conjuncts $0 \leq x$ and $\langle \forall i : 0 \leq i < x : f.i = g.i \rangle$ appear in all the three cases, and so we choose to place them in the precondition of the selection statement.

The conjunct $x = N$ appears only once and hence is chosen as a guard for $b := \mathbf{true}$.

The expression $x < N$ is a required precondition for both $a := \mathbf{true}$ and $c := \mathbf{true}$. In the interests of choosing the guards to be as weak as possible, we shall satisfy $x < N$ by choosing the weaker $x \neq N$ as a conjunct of the guards, and by simultaneously strengthening the precondition with $x \leq N$.

Finally, $f.x < g.x$ is added as a conjunct of the guard of $a := \mathbf{true}$, and $g.x < f.x$ is added as a conjunct of the guard of $c := \mathbf{true}$.

Thus we have derived the following guarded commands:

- $x \neq N \ \wedge \ f.x < g.x \quad \rightarrow \quad a := \mathbf{true}$
- $x = N \quad \rightarrow \quad b := \mathbf{true}$
- $x \neq N \ \wedge \ g.x < f.x \quad \rightarrow \quad c := \mathbf{true}$

In order to prove that the selection statement with these guarded commands does not abort, we have to prove the disjunction of the guards. In order to make this proof, we need $x = N \ \vee \ f.x \neq g.x$ as part of the context, as the reader may verify. Thus we shall strengthen the precondition of the selection with $x = N \ \vee \ f.x \neq g.x$. Hence we arrive at the first version of our program:

Program version 0

$$
\begin{array}{l}
|[\ \textbf{var}\ x : \ \textbf{int} \\
\quad \{\ 0 \leq x \leq N \quad \wedge \quad (x = N \ \vee \ f.x \neq g.x) \quad \wedge \quad \langle \forall i : 0 \leq i < x : f.i = g.i \rangle\ \} \\
\quad \{\ \neg a \ \wedge \ \neg b \ \wedge \ \neg c\ \} \\
\quad \textbf{if} \ \ x \neq N \ \wedge \ f.x < g.x \quad \rightarrow \quad a := \textbf{true} \\
\quad [] \ \ x = N \quad \rightarrow \quad b := \textbf{true} \\
\quad [] \ \ x \neq N \ \wedge \ g.x < f.x \quad \rightarrow \quad c := \textbf{true} \\
\quad \textbf{fi} \\
\quad \{\ R\ \} \\
]|
\end{array}
$$

Satisfying the precondition

Our remaining task is to design program fragments meeting the precondition of the selection statement.

The conjunct $\neg a \ \wedge \ \neg b \ \wedge \ \neg c$ of the precondition is easily satisfied by the statement $a,\ b,\ c \ := \ \textbf{false},\ \textbf{false},\ \textbf{false}$.

Thus we turn our attention to:

$$0 \leq x \leq N \quad \wedge \quad (x = N \ \vee \ f.x \neq g.x) \quad \wedge \quad \langle \forall i : 0 \leq i < x : f.i = g.i \rangle \qquad .$$

A moments reflection reveals that this is the precise postcondition of the Bounded Linear Search ! The Bounded Linear Search has the following specification:

$|[$ **con** $b :$ **array** $[0..N)$ of **bool** ;

var $x :$ **int** ;

$\{$ **true** $\}$

Bounded Linear Search

$\{\ 0 \leq x \leq N \quad \wedge \quad (x = N \ \vee \ b.x) \quad \wedge \quad \langle \forall i : 0 \leq i < x : \ \neg\, b.i \rangle\ \}$

$]|$.

for a boolean function b defined on the domain $[0..N)$. For the problem at hand, we define b as

$$b.i \quad \equiv \quad f.i \neq g.i \qquad .$$

Instantiating the standard refinement of the Bounded Linear Search with b as defined above, we arrive at the final version of our program.

The final program

The final version of our program is:

$|[$ **var** $x, y :$ **int** ;

$x, y \ := \ 0, N$

; **do** $x \neq y \quad \rightarrow$

if $f.x = g.x \quad \rightarrow \quad x := x + 1$

$[]$ $f.x \neq g.x \quad \rightarrow \quad y := x$

fi

od

; $a,\ b,\ c \ :=$ **false**, **false**, **false**

if $x \neq N \ \wedge \ f.x < g.x \quad \rightarrow \quad a :=$ **true**

$[]$ $x = N \quad \rightarrow \quad b :=$ **true**

$[]$ $x \neq N \ \wedge \ g.x < f.x \quad \rightarrow \quad c :=$ **true**

fi

$]|$

ARM5–6

Original: Mumbai, 13-16 September 2006

Revision: Mumbai, 4-9 April 2007

Apurva Mehta
apurva@mathmeth.com