Warm-Up Problem

No Warm-Up Today!

Program Verification
Arrays

Carmen Bruni

Lecture 21

Based on slides by Jonathan Buss, Lila Kari, Anna Lubiw and Steve Wolfman with thanks to B. Bonakdarpour, A. Gao, D. Maftuleac, C. Roberts, R. Trefler, and P. Van Beek

Last Time

- Partial correctness for while loops
- Determine whether a given formula is an invariant for a while loop.
- Find an invariant for a given while loop.
- Prove that a Hoare triple is satisfied under partial correctness for a program containing while loops.

Learning Goals

- Introducing the array assignment rule
- Annotate code using this rule
- Prove that a Hoare triple is satisfied under partial correctness for a program containing array assignment statments.

Assignment of Values of an Array

Let `A` be an array of `n` integers: `A[1]`, `A[2]`, ..., `A[n]`.

Assignment may work as before:

$$\{\!|\, P[A[x]/v] \,|\!\}$$
```
v = A[x] ;
```
$$\{\!|\, P \,|\!\}$$

assignment

But a complication can occur:

$$\{\!|\, A[y] = 0 \,|\!\}$$
```
A[x] = 1;
```
$$\{\!|\, A[y] = 0 \,|\!\}$$

???

The conclusion is not valid if $x = y$.

A correct rule must account for possible changes to `A[y]`, `A[z+3]`, etc., when `A[x]` changes.

Assignment to a Whole Array

Our solution: Treat an assignment to an array value

$$\texttt{A[e}_1\texttt{]} \texttt{ = e}_2$$

as an assignment of the whole array:

$$\texttt{A = A}\{\texttt{e}_1 \leftarrow \texttt{e}_2\} \texttt{ ;}$$

where the term "$\texttt{A}\{\texttt{e}_1 \leftarrow \texttt{e}_2\}$" denotes an array identical to A except the $\texttt{e}_1^{th}$ element is changed to have the value $\texttt{e}_2$.

Array Assignment: Definition and Examples

Definition: $A\{i \leftarrow e\}$ denotes the array with entries given by

$$A\{i \leftarrow e\}[j] = \begin{cases} e, & \text{if } j = i \\ A[j], & \text{if } j \neq i \end{cases} .$$

Examples:

1. $A\{1 \leftarrow 3\}[1] = 3$
2. $A\{1 \leftarrow 3\}\{1 \leftarrow 4\}[1] = 4$

The Array-Assignment Rule

Array assignment:

$$\frac{}{(\!| \, Q[A\{e_1 \leftarrow e_2\}/A] \, |\!) \;\; \texttt{A[e}_1\texttt{]} \;=\; \texttt{e}_2 \;\; (\!| \, Q \, |\!)} \text{(Array assignment)}$$

where

$$A\{i \leftarrow e\}[j] = \begin{cases} e, & \text{if } j = i \\ A[j], & \text{if } j \neq i \end{cases} .$$

Example

Prove the following is satisfied under partial correctness.

$$\{\!|\ ((A[x] = x_0) \wedge (A[y] = y_0))\ |\!\}$$

```
t = A[x] ;
```

```
A[x] = A[y] ;
```

```
A[y] = t ;
```

$$\{\!|\ ((A[x] = y_0) \wedge (A[y] = x_0))\ |\!\}$$

We do assignments bottom-up, as always....

Example: push up assertions for assignments

$\{\!|\ ((A[x] = x_0) \wedge (A[y] = y_0))\ |\!\}$

```
t = A[x] ;
```

```
A[x] = A[y] ;
```

$\{\!|\ ((A\{y \leftarrow t\}[x] = y_0) \wedge (A\{y \leftarrow t\}[y] = x_0))\ |\!\}$

```
A[y] = t ;
```

$\{\!|\ ((A[x] = y_0) \wedge (A[y] = x_0))\ |\!\}$ array assignment

Example: push up assertions for assignments

$$\{\!|\ ((A[x] = x_0) \wedge (A[y] = y_0))\ |\!\}$$

```
t = A[x] ;
```

$$\{\!|\ ((A\{x \leftarrow A[y]\}\{y \leftarrow t\}[x] = y_0) \wedge (A\{x \leftarrow A[y]\}\{y \leftarrow t\}[y] = x_0))\ |\!\}$$

```
A[x] = A[y] ;
```

$$\{\!|\ ((A\{y \leftarrow t\}[x] = y_0) \wedge (A\{y \leftarrow t\}[y] = x_0))\ |\!\}$$ array assignment

```
A[y] = t ;
```

$$\{\!|\ ((A[x] = y_0) \wedge (A[y] = x_0))\ |\!\}$$ array assignment

Example: push up assertions for assignments

$$\{\!|\ ((A[x] = x_0) \wedge (A[y] = y_0))\ |\!\}$$

$$\{\!|\ ((A\{x \leftarrow A[y]\}\{y \leftarrow A[x]\}[x] = y_0) \wedge (A\{x \leftarrow A[y]\}\{y \leftarrow A[x]\}[y] = x_0))\ |\!\}$$

```
t = A[x] ;
```

$$\{\!|\ ((A\{x \leftarrow A[y]\}\{y \leftarrow t\}[x] = y_0) \wedge (A\{x \leftarrow A[y]\}\{y \leftarrow t\}[y] = x_0))\ |\!\}$$ assignment

```
A[x] = A[y] ;
```

$$\{\!|\ ((A\{y \leftarrow t\}[x] = y_0) \wedge (A\{y \leftarrow t\}[y] = x_0))\ |\!\}$$ array assignment

```
A[y] = t ;
```

$$\{\!|\ ((A[x] = y_0) \wedge (A[y] = x_0))\ |\!\}$$ array assignment

Example: push up assertions for assignments

$\{\!|\ ((A[x] = x_0) \wedge (A[y] = y_0))\ |\!\}$

$\{\!|\ ((A\{x \leftarrow A[y]\}\{y \leftarrow A[x]\}[x] = y_0)$
$\quad \wedge (A\{x \leftarrow A[y]\}\{y \leftarrow A[x]\}[y] = x_0))\ |\!\}$ — implied (a)

```
t = A[x] ;
```

$\{\!|\ ((A\{x \leftarrow A[y]\}\{y \leftarrow t\}[x] = y_0)$
$\quad \wedge (A\{x \leftarrow A[y]\}\{y \leftarrow t\}[y] = x_0))\ |\!\}$ — assignment

```
A[x] = A[y] ;
```

$\{\!|\ ((A\{y \leftarrow t\}[x] = y_0) \wedge (A\{y \leftarrow t\}[y] = x_0))\ |\!\}$ — array assignment

```
A[y] = t ;
```

$\{\!|\ ((A[x] = y_0) \wedge (A[y] = x_0))\ |\!\}$ — array assignment

Example: Proof of implied

As "implied (a)", we need to prove the following.

Lemma:

$$(A\{x \leftarrow A[y]\}\{y \leftarrow A[x]\}[x] = A[y])$$

and

$$(A\{x \leftarrow A[y]\}\{y \leftarrow A[x]\}[y] = A[x]) \ .$$

Proof.

In the second equation, the index element is the assigned element.

For the first equation, we consider two cases.

- If $y \neq x$, the "$\{y \leftarrow \ldots\}$" is irrelevant, and the claim holds.
- If $y = x$, the result on the left is $A[x]$, which is also $A[y]$.

Example: Alternative proof

For an alternative proof, use the definition of $M\{i \leftarrow e\}[j]$, with $A\{x \leftarrow A[y]\}$ as M, $i = y$ and $e = A[x]$:

$$A\{x \leftarrow A[y]\}\{y \leftarrow A[x]\}[j] = \begin{cases} A[x], & \text{if } y = j \\ A\{x \leftarrow A[y]\}[j], & \text{if } y \neq j \end{cases} .$$

At index $j = y$, this is just $A[x]$, as required.

In the case $j = x$, we get the required value $A[y]$. (Why?)

And, finally, if $j \neq x$ and $j \neq y$, then

$$A\{x \leftarrow A[y]\}\{y \leftarrow A[x]\}[j] = A[j] \ ,$$

as we should have required.

Example: reversing an array

Example: Given an array R with n elements $R[1], ..., R[n]$, reverse the elements.

Algorithm: exchange $R[j]$ with $R[n+1-j]$, for each $1 \leq j \leq \lfloor n/2 \rfloor$.

A possible program is

```
j = 1 ;
while ( 2*j <= n ) {
    t = R[j] ;
    R[j] = R[n+1-j] ;
    R[n+1-j] = t ;
    j = j + 1 ;
}
```

Needed: a postcondition, and a loop invariant.

Reversal code: conditions and an invariant

Precondition: $\Big(\forall x \Big((1 \leq x \leq n) \to (R[x] = r_{x})\Big)\Big)$.

Postcondition: $\Big(\forall x \Big((1 \leq x \leq n) \to (R[x] = r_{n+1-x})\Big)\Big)$.

Invariant? When has an exchange occurred at position x?

- If $x < j$ or $x > n + 1 - j$, then $R[x]$ and $R[n + 1 - x]$ have already been exchanged.
- If $j \leq x \leq n + 1 - j$, then no exchange has happened yet.

Thus let *Inv'*(j) be the formula

$$\Big(\forall x \Big(\Big((1 \leq x < j) \to (R[x] = r_{n+1-x} \wedge R[n + 1 - x] = r_{x})\Big) \wedge \Big((j \leq x \leq (n + 1)/2) \to (R[x] = r_{x} \wedge R[n + 1 - x] = r_{n+1-x})\Big)\Big)\Big) .$$

and *Inv*$(j) =$ *Inv'*$(j) \wedge (1 \leq j \leq n/2 + 1)$.

Reversal: Annotations around the loop

The annotations surrounding the `while`-loop:

$\{\!|\ \Big((n \geq 0) \wedge \Big(\forall x \Big((1 \leq x \leq n) \rightarrow (R[x] = r_x)\Big)\Big)\Big)\ |\!\}$

$\{\!|\ \mathit{Inv}(1)\ |\!\}$ — implied (a)

```
j = 1 ;
```

$\{\!|\ \mathit{Inv}(j)\ |\!\}$ — assignment

```
while ( 2*j <= n ) {
```

$\{\!|\ (\mathit{Inv}(j) \wedge (2j \leq n))\ |\!\}$ — partial-`while`

$\vdots$

$\{\!|\ \mathit{Inv}(j)\ |\!\}$ — *(TBA)*

```
}
```

$\{\!|\ (\mathit{Inv}(j) \wedge (2j > n))\ |\!\}$ — partial-`while`

$\{\!|\ \Big(\forall x \Big((1 \leq x \leq n) \rightarrow (R[x] = r_{n+1-x})\Big)\Big)\ |\!\}$ — implied (b)

Reversal code: annotations inside the loop

We must now handle the code inside the loop.

$\{\!|\ (\mathit{Inv}(j) \wedge 2j \leq n)\ |\!\}$ partial-`while`

$\{\!|\ \mathit{Inv}(j+1)[R'/R]$, *where* R' *is*

$\quad R\{j \leftarrow R[n+1-j]\}\{(n+1-j) \leftarrow R[j]\}\ |\!\}$ implied (c)

```
t = R[j]; R[j] = R[n+1-j]; R[n+1-j] = t;
```

$\{\!|\ \mathit{Inv}(j+1)\ |\!\}$ Lemma

```
j = j + 1 ;
```

$\{\!|\ \mathit{Inv}(j)\ |\!\}$ assignment

Proof of Implied Condition (c)

Recall *Inv'*(j):

$$\Big(\forall x \Big(\big((1 \le x < j) \to (R[x] = r_{n+1-x} \wedge R[n+1-x] = r_x)\big) \\ \wedge \big((j \le x \le (n+1)/2) \to (R[x] = r_x \wedge R[n+1-x] = r_{n+1-x})\big)\Big)\Big) \ .$$

We need this to imply *Inv'*$(j+1)[R'/R]$, which is

$$\Big(\forall x \Big(\big((1 \le x < j+1) \to (R'[x] = r_{n+1-x} \wedge R'[n+1-x] = r_x)\big) \\ \wedge \big((j+1 \le x \le n/2) \to (R'[x] = r_x \wedge R'[n+1-x] = r_{n+1-x})\big)\Big)\Big) \ ,$$

which by the construction of R' is equivalent to

$$\Big(\forall x \Big(\big((1 \le x < j) \to (R[x] = r_{n+1-x} \wedge R[n+1-x] = r_x)\big) \\ \wedge (R'[j] = r_{n+1-j}) \wedge (R'[n+1-j] = r_j) \\ \wedge \big((j+1 \le x \le n/2) \to (R[x] = r_x \wedge R[n+1-x] = r_{n+1-x})\big)\Big)\Big) \ .$$