Verificación de programas II: Teorema del Invariante

Román Gorojovsky

Algoritmos y Estructuras de Datos

4 de septiembre de 2024

Plan del día

Plan del día

- Precondición más débil de ciclos
- Teorema del Invariante
- Teorema de Terminación
- Un ejercicio de un parcial

Precondición más débil

Precondición más débil – Idea informal

Es la P que permite que el programa **S** funcione correctamente, pero restringiendo lo menos posible.

Principio de diseño

Ser cuidadoso con los resultados que se emiten y generoso con los parámetros que se reciben.

Axiomas

Definiciones (copiadas de la teórica)

- **Axioma 1:** $wp(\textbf{x := E}, Q) \equiv \text{def}(E) \wedge_L Q^x_E$
- **Axioma 2:** $wp(\textbf{skip}, Q) \equiv Q$
- **Axioma 3:** $wp(\textbf{S1; S2}, Q) \equiv wp(\textbf{S1}, wp(\textbf{S2}, Q))$
- **Axioma 4:** Si $\textbf{S} = \textbf{if B then S1 else S2 endif}$, entonces

$$wp(\textbf{S}, Q) \equiv def(B) \wedge_L \Big((B \wedge wp(\textbf{S1}, Q)) \vee (\neg B \wedge wp(\textbf{S2}, Q)) \Big)$$

¿Qué hacemos con los ciclos?

¿Cómo calculo la WP de este programa?

$\texttt{proc sumar}(\text{in } s : seq\langle\mathbb{Z}\rangle) : \mathbb{Z}$

```
while (i < s.size()) do
    res := res + s[i];
    i := i + 1
endwhile
```

$Q \equiv \{res = \sum_{i=0}^{|s|-1} s[i]\}$

- A ojo $\longrightarrow WP = \{res = 0 \land i = 0\}$
- Formalmente no existe un Axioma 5, termina siendo una fórmula infinita (detalles en la teórica)
- Sólo voy a poder probar que la tripla $\{P\}$ S $\{Q\}$ es válida

Invariante de un ciclo

Dado un ciclo de la forma

```
while (B) do
        S1;
        S2;
        // ...
endwhile
```

El **Invariante** del ciclo es

- Un predicado I que se cumple:
  - Antes de "entrar" en el ciclo, es decir, antes de cada iteración
  - Al terminar cada iteración (si se cumplía B)

Teorema del Invariante

Teorema del invariante

Si existe un predicado I tal que ...

1. $P_c \Rightarrow \mathsf{I}$
2. $\{I \wedge B\}\ \mathtt{S}\ \{I\}$
3. $I \wedge \neg B \Rightarrow Q_c$

entonces el ciclo **while(B)** $\{$**S**$\}$ es *parcialmente correcto* respecto de la especificación (P_c, Q_c).

Más tarde vemos qué falta para que sea totalmente correcto

Ejemplo

```
proc sumar(in s : seq⟨ℤ⟩) : ℤ

res := 0
i := 0
while (i < s.size()) do
   res := res + s[i];
   i := i + 1
endwhile
```

- $P_c \equiv \{res = 0 \wedge i = 0\}$
- $Q_c \equiv \{res = \sum_{i=0}^{|s|-1} s[i]\}$
- $B \equiv \{i < |s|\}$
- $I \equiv \{0 \leq i \leq |s| \wedge res = \sum_{j=0}^{i-1} s[j]\}$

Ejemplo

```
while (i < s.size()) do
   res := res + s[i];
   i := i + 1
endwhile
```

- $P_c \equiv \{res = 0 \wedge i = 0\}$
- $Q_c \equiv \{res = \sum_{i=0}^{|s|-1} s[i]\}$
- $B \equiv \{i < |s|\}$
- $I \equiv \{0 \leq i \leq |s| \wedge res = \sum_{j=0}^{i-1} s[j]\}$

- $P_c \Rightarrow$ I
  - $0 \leq i \leq |s| \equiv 0 \leq 0 \leq |s|$ ✓
  - $res = \sum_{j=0}^{i-1} s[j] \equiv 0 = \sum_{j=0}^{0-1} s[j] = 0$ ✓

Ejemplo

```
while (i < s.size()) do
  res := res + s[i];
  i := i + 1
endwhile
```

- $P_c \equiv \{res = 0 \land i = 0\}$
- $Q_c \equiv \{res = \sum_{i=0}^{|s|-1} s[i]\}$
- $B \equiv \{i < |s|\}$
- $I \equiv \{0 \leq i \leq |s| \land res = \sum_{j=0}^{i-1} s[j]\}$

- $\{I \land B\}\ \mathtt{S}\ \{I\} \leftrightarrow \{I \land B\} \rightarrow WP(\mathtt{S}, I)$
  - $WP(\mathtt{S}, I)$
    $\equiv WP(\texttt{res := res + s[i]; i := i + 1}, \{0 \leq i \leq |s| \land res = \sum_{j=0}^{i-1} s[j]\})$
    $\equiv WP\big(\texttt{res := res + s[i]}, WP(\texttt{i := i + 1}, \{0 \leq i \leq |s| \land res = \sum_{j=0}^{i-1} s[j]\})\big)$
    $\equiv WP(\texttt{res := res + s[i]}, 0 \leq i+1 \leq |s| \land res = \sum_{j=0}^{i} s[j])$
    $\equiv \mathsf{def}(s[i]) \land_L (-1 \leq i \leq |s| - 1 \land res + s[i] = \sum_{j=0}^{i} s[j])$
    $\equiv 0 \leq i < |s| \land_L (-1 \leq i \leq |s| - 1 \land res = \sum_{j=0}^{i} s[j] - s[i])$
    $\equiv 0 \leq i < |s| \land_L res = \sum_{j=0}^{i-1} s[j]$ *(Combino los rangos y resto de la sumatoria)*
  - $\{I \land B\} \equiv \{0 \leq i < |s| \land res = \sum_{j=0}^{i-1} s[j]\}$
  - ¿$\{I \land B\} \rightarrow WP(\mathtt{S}, I)$? ✓

Ejemplo

```
while (i < s.size()) do
   res := res + s[i];
   i := i + 1
endwhile
```

- $P_c \equiv \{res = 0 \wedge i = 0\}$
- $Q_c \equiv \{res = \sum_{i=0}^{|s|-1} s[i]\}$
- $B \equiv \{i < |s|\}$
- $I \equiv \{0 \leq i \leq |s| \wedge res = \sum_{j=0}^{i-1} s[j]\}$

- $I \wedge \neg B \Rightarrow Q_C$
  - $I \wedge \neg B \equiv |s| \leq i \leq |s| \wedge res = \sum_{j=0}^{i-1} s[j]$
    $\equiv i = |s| \wedge res = \sum_{j=0}^{i-1} s[j]$
    $\equiv res = \sum_{j=0}^{|s|-1} s[j] \equiv Q_c$ ✓

¿Qué podemos demostrar hasta ahora?

- Correctitud parcial: Probando las hipótesis que vimos hasta acá sabemos que si el ciclo termina la tripla de Hoare $\{P_c\}$ S $\{Q_c\}$ es válida
- Falta probar que el ciclo efectivamente termine
- Teorema de Terminación

Teorema de Terminación

Teorema de Terminación

Si existe una función $fv : \mathbb{V} \to \mathbb{Z}$ tal que

1. $\{I \wedge B \wedge f_v = v_0\}$ S $\{f_v < v_0\}$,
2. $I \wedge f_v \leq 0 \to \neg B$,

entonces la ejecución del ciclo **while B do S endwhile** siempre termina.

- La función f_v se llama función variante del ciclo.
- $\mathbb{V}$ son valores que toman las variables del programa

Ejemplo

proc sumar(in $s : seq\langle\mathbb{Z}\rangle) : \mathbb{Z}$

```
res := 0
i := 0
while (i < s.size()) do
   res := res + s[i];
   i := i + 1
endwhile
```

- $P_c \equiv \{res = 0 \land i = 0\}$
- $Q_c \equiv \{res = \sum_{i=0}^{|s|-1} s[i]\}$
- $B \equiv \{i < |s|\}$
- $I \equiv \{0 \leq i \leq |s| \land res = \sum_{j=0}^{i-1} s[j]\}$
- $f_v = |s| - i$

Ejemplo

```
while (i < s.size()) do
   res := res + s[i];
   i := i + 1
endwhile
```

- $P_c \equiv \{res = 0 \land i = 0\}$
- $Q_c \equiv \{res = \sum_{i=0}^{|s|-1} s[i]\}$
- $B \equiv \{i < |s|\}$
- $I \equiv \{0 \leq i \leq |s| \land res = \sum_{j=0}^{i-1} s[j]\}$
- $f_v = |s| - i$

- $\{I \land B \land f_v = v_0\}\ \mathtt{S}\ \{f_v < v_0\} \leftrightarrow$
  $\{I \land B \land f_v = v_0\} \rightarrow WP(\mathtt{S}, f_v < v_0)$
  - $WP(\mathtt{S}, f_v < v_0) \equiv WP(\texttt{res := res + s[i]; i := i + 1}, |s| - i < v_0)$
    $\equiv WP(\texttt{res := res + s[i]}, WP(\texttt{i := i + 1}, |s| - i < v_0))$
    $\equiv WP(\texttt{res := res + s[i]}, |s| - i + 1 < v_0) \equiv \mathsf{def}(s[i]) \land_L |s| - (i+1) < v_0$
    $\equiv 0 \leq i < |s| \land_L |s| - i - 1 < v_0$
  - $\{I \land B \land f_v = v_0\} \equiv \{0 \leq |s| - 1 \land |s| - 1 = v_0 \land res = \sum_{j=0}^{i-1} s[j]\}$
    - Puedo ignorar lo que corresponde al rango y a res
  - $\{I \land B \land f_v = v_0\} \rightarrow WP(\mathtt{S}, f_v < v_0) \leftrightarrow |s| - i - 1 < |s| - 1 \leftrightarrow -1 < 0$ ✓

Ejemplo

```
while (i < s.size()) do
   res := res + s[i];
   i := i + 1
endwhile
```

- $P_c \equiv \{res = 0 \land i = 0\}$
- $Q_c \equiv \{res = \sum_{i=0}^{|s|-1} s[i]\}$
- $B \equiv \{i < |s|\}$
- $I \equiv \{0 \leq i \leq |s| \land res = \sum_{j=0}^{i-1} s[j]\}$
- $f_v = |s| - i$

- $I \land f_v \leq 0 \rightarrow \neg B$
  - $I \land f_v \leq 0 \equiv 0 \leq i \leq |s| \land |s| - 1 \leq 0 \land res = \sum_{j=0}^{i-1} s[j]$
    - Una vez más ignoro lo que corresponde a res porque no lo necesito para esta demostración
  - $0 \leq i \leq |s| \land |s| - i \leq 0 \leftrightarrow i \leq |s| \land |s| \leq i \leftrightarrow i = |s|$
  - $i = |s| \rightarrow \neg(i < |s|) \rightarrow \neg B$ ✓

Ejercicio de parcial

E4. Correctitud del ciclo

Dado el siguiente programa con su especificación

$P_c \equiv \{n > 0 \land n \ mod \ 2 = 0 \land i = 1 \land res = 1\}$

```
While( i <= n/2){
    res := res * i * (n+1-i);
    i := i+1;
}
```

$Q_c \equiv \{res = n!\}$

Contamos con el siguiente invariante, que sabemos que es incorrecto:

$I \equiv \{1 \leq i \leq n/2 + 1 \land res = \prod_{j=1}^{2(i-1)} j\}$

a) Señale qué axiomas del teorema del invariante no se cumplen. Justifique con palabras en forma precisa.
b) Escriba un invariante que resulte correcto.
c) Proponga una función variante y demuestre formalmente que es correcta.