Precondición más débil de ciclos

Algoritmos y Estructuras de Datos I

Repaso: Triplas de Hoare

- Consideremos la siguiente tripla de Hoare:

$$\{P\}\ \mathbf{S}\ \{Q\}.$$

- Esta tripla es válida si se cumple que:

  1. Si el programa S comienza en un estado que cumple P ...
  2. ... entonces termina luego de un número finito de pasos ...
  3. ... Y además en un estado que cumple Q.

Repaso: Lenguaje SmallLang

- Definimos un lenguaje imperativo basado en variables y las siguientes instrucciones:
  1. Nada: Instrucción **skip** que no hace nada.
  2. Asignación: Instrucción **x := E**.
- Además, tenemos las siguientes estructuras de control:
  1. Secuencia: **S1; S2** es un programa, si **S1** y **S2** son dos programas.
  2. Condicional: **if B then S1 else S2 endif** es un programa, si **B** es una expresión lógica y **S1** y **S2** son dos programas.
  3. Ciclo: **while B do S endwhile** es un programa, si **B** es una expresión lógica y **S** es un programa.

Repaso: Precondición más débil

- **Definición.** La precondición más débil de un programa **S** respecto de una postcondición Q es el predicado P más débil posible tal que $\{P\}\mathbf{S}\{Q\}$.
- **Notación.** $wp(\mathtt{S}, Q)$.
- **Teorema:** Decimos que $\{P\}$ **S** $\{Q\}$ es válida sii $P \Rightarrow_L wp(\mathtt{S}, Q)$

Repaso: Axiomas wp

- **Axioma 1.** $wp(\texttt{x := E}, Q) \equiv \operatorname{def}(E) \land_L Q^x_E$.
- **Axioma 2.** $wp(\texttt{skip}, Q) \equiv Q$.
- **Axioma 3.** $wp(\texttt{S1; S2}, Q) \equiv wp(\texttt{S1}, wp(\texttt{S2}, Q))$.
- **Axioma 4.** $wp(\texttt{if B then S1 else S2 endif}, Q) \equiv$

$$def(B) \land_L \Big((B \land wp(\texttt{S1}, Q)) \lor (\neg B \land wp(\texttt{S2}, Q)) \Big)$$

- **Observación:** $wp(\texttt{b[i]:=E}, Q) \equiv wp(\texttt{b:=setAt(b,i,E)}, Q)$

¿Cuál es la precondición más débil?

$\{???\}$

```
while (x>0) do
  x := x -1
endwhile
```

$\{x = 0\}$

$$wp(\texttt{while ...}, x = 0) \equiv x \geq 0$$

¿Cuál es la precondición más débil?

$\{???\}$

```
i := 0;
while (x<5) do
  x := x + 1;
  i := i + 1
endwhile
```

$\{x = 5 \wedge i = 5\}$

$$wp(\texttt{i:=0; while ...}, x = 5 \wedge i = 5) \equiv x = 0$$

¿Cuál es la precondición más débil?

$\{???\}$

```
while (x==5) do
  x := 5
endwhile
```

$\{x \neq 5\}$

$$wp(\texttt{while ...}, x \neq 5) \equiv x \neq 5$$

¿Es válida la siguiente tripla de Hoare?

$\{n \geq 0 \wedge i = 1 \wedge s = 0\}$

```
while (i <= n) do
  s := s + i;
  i := i + 1
endwhile
```

$\{s = \sum_{k=1}^{n} k\}$

Precondición más débil de un ciclo

- Supongamos que tenemos el ciclo **while B do S endwhile**.
- **Definición.** Definimos $H_k(Q)$ como el predicado que define el conjunto de estados a partir de los cuales la ejecución del ciclo termina en exactamente k iteraciones:

$$\begin{aligned} H_0(Q) &\equiv \operatorname{def}(B) \land \neg B \land Q, \\ H_{k+1}(Q) &\equiv \operatorname{def}(B) \land B \land wp(\mathtt{S}, H_k(Q)) \qquad \text{para } k \geq 0. \end{aligned}$$

- **Propiedad:** Si el ciclo realiza a lo sumo k iteraciones, entonces

$$wp(\mathtt{while\ B\ do\ S\ endwhile}, Q) \equiv \vee_{i=0}^{k} H_i(Q)$$

Ejemplo

$\{???\}$

```
while (0<i && i<3) do
  i := i +1
endwhile
```

$\{i = 3\}$

- A lo sumo, se va a ejecutar 2 veces el cuerpo del ciclo
- ¿Cuál es la precondición más débil?

$$
\begin{aligned}
& wp(\texttt{while } 0 < i < 3 \texttt{ do i:=i+1 endwhile}, i = 3) \\
\equiv \quad & \vee_{i=0}^{2} H_i(i = 3) \\
\equiv \quad & H_0(i = 3) \vee H_1(i = 3) \vee H_2(i = 3) \\
\equiv \quad & i = 1 \vee i = 2 \vee i = 3
\end{aligned}
$$

Otro ejemplo

$\{???\}$

```
while (0<i && i<n) do
  i := i +1
endwhile
```

$\{i \geq 0\}$

- ¿Cuántas veces se va a ejecutar el cuerpo del ciclo?
- ¿Podemos usar la propiedad anterior para conocer la precondición más débil?
- ¡No! Porque no podemos fijar a priori una cota superior a la cantidad de iteraciones que va a realizar el ciclo.

Precondición más débil de un ciclo

- **Intituivamente:** $wp(\texttt{while B do S endwhile}, Q)$ tiene que ser una fórmula lógica capaz de capturar todos los estados tales que, luego de ejecutar el ciclo una cantidad arbitraria de veces, vale Q.
- **Axioma 5:**

$$wp(\texttt{while B do S endwhile}, Q) \equiv (\exists_{i \geq 0})\big(H_i(Q)\big)$$

Precondición más débil de un ciclo

- Ahora tratemos de usar el **Axioma 5**:

$$
\begin{aligned}
&wp(\texttt{while B do S endwhile}, Q) \\
&\qquad \equiv \quad (\exists_{i \geq 0}) H_i(Q) \\
&\qquad \equiv \quad H_0(Q) \vee H_1(Q) \vee H_2(Q) \vee \ldots \\
&\qquad \equiv \quad \vee_{i=0}^{\infty}(H_i(Q))
\end{aligned}
$$

¡Es una fórmula infinitaria!

- Por lo tanto, no podemos usar mecánicamente el **Axioma 5** para demostrar la corrección de un ciclo con una cantidad no acotada a priori de iteraciones :(

Recap: Teorema del Invariante

- **Teorema.** Si def(B) y existe un predicado I tal que

  1. $P_C \Rightarrow I$,
  2. $\{I \wedge B\}\ S\ \{I\}$,
  3. $I \wedge \neg B \Rightarrow Q_C$,

  ... y **el ciclo termina**, entonces la siguiente tripla de Hoare es válida:

  $$\{P_C\}\ \textbf{while B do S endwhile}\ \{Q_C\}$$

- Esta observación es un teorema que se deduce de la definición anterior.

- Las condiciones 1-3 garantizan la corrección parcial del ciclo (la hipótesis de terminación es necesaria para garantizar corrección).

Ejemplo: suma de índices

- Sea la siguiente tripla de Hoare:

  $\{n \geq 0 \wedge i = 1 \wedge s = 0\}$

  ```
  while (i <= n) do
    s = s + i;
    i = i + 1;
  endwhile
  ```

  $\{s = \sum_{k=1}^{n} k\}$

- Habíamos identificamos los predicados necesarios para aplicar el Teorema del Invariante:
  - $P_C \equiv n \geq 0 \wedge i = 1 \wedge s = 0$
  - $Q_C \equiv s = \sum_{k=1}^{n} k$
  - $B \equiv i \leq n$
  - $I \equiv 1 \leq i \leq n + 1 \wedge \mathrm{s} = \sum_{k=1}^{i-1} k$

Repaso: $P_C \Rightarrow I$

$$
\begin{aligned}
P_C &\equiv n \geq 0 \wedge i = 1 \wedge s = 0 \\
&\Rightarrow 1 \leq i \leq n+1 \wedge s = 0 \\
&\Rightarrow 1 \leq i \leq n+1 \wedge s = \sum_{k=1}^{0} k \\
&\Rightarrow 1 \leq i \leq n+1 \wedge s = \sum_{k=1}^{i-1} k \\
&\equiv I \checkmark
\end{aligned}
$$

Repaso: $I \land \neg B \Rightarrow Q_C$

$$
\begin{aligned}
I \land \neg B &\equiv 1 \leq i \leq n+1 \land s = \sum_{k=1}^{i-1} k \land \neg(i \leq n) \\
&\equiv 1 \leq i \leq n+1 \land s = \sum_{k=1}^{i-1} k \land i > n \\
&\Rightarrow 1 \leq i \leq n+1 \land s = \sum_{k=1}^{i-1} k \land i = n+1 \\
&\Rightarrow 1 \leq i \leq n+1 \land s = \sum_{k=1}^{n+1-1} k \land i = n+1 \\
&\Rightarrow s = \sum_{k=1}^{n} k \equiv Q_C \checkmark
\end{aligned}
$$

$\{I \wedge B\}\ S\ \{I\}$

Para demostrar $\{I \wedge B\}\ S\ \{I\}$ tenemos que probar que:

$$I \wedge B \Rightarrow wp(\mathtt{S}, I)$$

$$\begin{aligned}
& wp(\mathtt{s:=s+i;i:=i+1}, 1 \leq i \leq n+1 \wedge s = \sum_{k=1}^{i-1} k) \\
\equiv\ & wp(\mathtt{s:=s+i}, wp(\mathtt{i:=i+1}, 1 \leq i \leq n+1 \wedge s = \sum_{k=1}^{i-1} k)) \\
\equiv\ & wp(\mathtt{s:=s+i}, \mathit{def}(i+1) \wedge_L \left(1 \leq i+1 \leq n+1 \wedge s = \sum_{k=1}^{i+1-1} k\right)) \\
\equiv\ & wp(\mathtt{s:=s+i}, 1 \leq i+1 \leq n+1 \wedge s = \sum_{k=1}^{i+1-1} k) \\
\equiv\ & \mathit{def}(s+i) \wedge_L \left(1 \leq i+1 \leq n+1 \wedge s+i = \sum_{k=1}^{i+1-1} k\right)
\end{aligned}$$

$\{I \wedge B\}\ S\ \{I\}$

$$\begin{aligned}
&\equiv \quad 0 \leq i \leq n \wedge s + i = \sum_{k=1}^{i} k \\
&\equiv \quad 0 \leq i \leq n \wedge s = (\sum_{k=1}^{i} k) - i \\
&\equiv \quad 0 \leq i \leq n \wedge s = \sum_{k=1}^{i-1} k
\end{aligned}$$

- Luego de simplificar, nos falta probar que:

$$\underbrace{\Big(1 \leq i \leq n+1 \wedge s = \sum_{k=1}^{i-1} k}_{I} \wedge \underbrace{i \leq n}_{B}\Big) \Rightarrow \underbrace{\Big(0 \leq i \leq n \wedge s = \sum_{k=1}^{i-1} k\Big)}_{wp(\mathtt{S}, I)}$$

- Lo cual es trivialmente cierto.
- Por lo tanto podemos concluir que $\{I \wedge B\}\ S\ \{I\}$ es una tripla de Hoare válida (i.e., verdadera)

Ejemplo: suma de índices

- Habiendo probado las hipótesis del Teorema del Invariante podemos decir que si el ciclo siempre termina, entonces la siguiente tripla de Hoare es válida:

  $\{n \geq 0 \wedge i = 1 \wedge s = 0\}$

  ```
  while (i <= n) do
    s = s + i;
    i = i + 1;
  endwhile
  ```

  $\{s = \sum_{k=1}^{n} k\}$

- **Pero** ..., ¡todavía no probamos que el ciclo siempre termina!
- ¿Cómo podemos probar si dada una precondición, un ciclo siempre termina?
  - Para eso tenemos el Teorema de terminación.

Teorema de terminación de un ciclo

- **Teorema.** Sea $\mathbb{V}$ el producto cartesiano de los dominios de las variables del programa y sea I un invariante del ciclo **while B do S endwhile**. Si existe una función $fv : \mathbb{V} \to \mathbb{Z}$ tal que

  1. $\{I \wedge B \wedge v_0 = fv\}$ **S** $\{fv < v_0\}$,
  2. $I \wedge fv \leq 0 \Rightarrow \neg B$,

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

- La función fv se llama función variante del ciclo.

- El Teorema de terminación nos permite demostrar que si un ciclo termina (i.e. no se cuelga).

Ejemplo: Suma de índices

- Sea la siguiente tripla de Hoare:

  $\{n \geq 0 \wedge i = 1 \wedge s = 0\}$

  ```
  while (i <= n) do
    s = s + i;
    i = i + 1;
  endwhile
  ```

  $\{s = \sum_{k=1}^{n} k\}$

- Ya probamos que el siguiente predicado es un invariante de este ciclo.

$$I \equiv 1 \leq i \leq n + 1 \wedge \mathrm{s} = \sum_{k=1}^{i-1} k$$

- ¿Cúal sería una buena función variante para este ciclo?

- Ejecutemos el ciclo con $n = 6$.

Iteración	i	s	n	n+1-i
0	1	0	6	6
1	2	1	6	5
2	3	3	6	4
3	4	6	6	3
4	5	10	6	2
5	6	15	6	1
6	7	21	6	0

- Una función variante representa una cantidad que se va reduciendo a lo largo de las iteraciones. En este caso es la cantidad de índices que falta sumar.

- Proponemos entonces $fv \; = \; n + 1 - i$

Ejemplo: Suma de índices

- Veamos que se cumplen las dos condiciones del teorema.

1. Para verificar que $\{I \wedge B \wedge fv = v_0\}$ S $\{fv < v_0\}$ para todo v_0, calculamos $wp(\texttt{S}, fv < v_0)$.

$$
\begin{aligned}
& wp(\texttt{s:=s+1;i:=i+1}, fv < v_0) \\
& \equiv wp(\texttt{s:=s+1;i:=i+1}, (n+1-i) < v_0) \\
& \equiv wp(\texttt{s:=s+1}, wp(\texttt{i:=i+1}, (n+1-i) < v_0)) \\
& \equiv wp(\texttt{s:=s+1}, \mathit{def}(i+1) \wedge_L (n+1-(i+1)) < v_0)) \\
& \equiv wp(\texttt{s:=s+1}, (n+1-(i+1)) < v_0)) \\
& \equiv \mathit{def}(s+1) \wedge_L n-i < v_0 \\
& \equiv n-i < v_0 \\
& \equiv n-i < n+1-i \\
& \equiv n-i < n-i+1 \checkmark
\end{aligned}
$$

Ejemplo: Suma de índices

- Veamos que se cumplen las dos condiciones del teorema.

2. Verifiquemos que $I \wedge fv \leq 0 \Rightarrow \neg B$

$$
\begin{aligned}
I \wedge fv \leq 0 \quad &\equiv \quad \overbrace{1 \leq i \leq n+1 \wedge s = \sum_{k=1}^{i-1} k}^{I} \wedge \overbrace{n+1-i \leq 0}^{fv \leq 0} \\
&\Rightarrow \quad i \leq n+1 \wedge n+1-i \leq 0 \\
&\Rightarrow \quad i \leq n+1 \wedge n+1 \leq i \\
&\Rightarrow \quad i = n+1 \\
&\Rightarrow \quad \neg(i \leq n) \\
&\Rightarrow \quad \neg B \checkmark
\end{aligned}
$$

Ejemplo: Suma de índices

Recapitulando, sean

- $I \equiv 1 \leq i \leq n + 1 \wedge \mathtt{s} = \sum_{k=1}^{i-1} k$
- $fv \ = \ n + 1 - i$

En la teórica pasada ya habíamos probado que el ciclo es **parcialmente** correcto dado que:

1. $P_C \Rightarrow I$
2. $\{I \wedge B\}\ \mathtt{S}\ \{I\}$
3. $I \wedge \neg B \Rightarrow Q_C$

Ahora acabamos de probar que el ciclo siempre termina ya que:

4. $\{I \wedge B \wedge v_0 = fv\}\ \ \mathbf{S}\ \ \{fv < v_0\}$,
5. $I \wedge fv \leq 0 \Rightarrow \neg B$,

Por lo tanto, por (1)-(5) tenemos (finalmente) que ...

Ejemplo: Suma de índices

- Que la siguiente tripla de Hoare:

  $\{P_C : n \geq 0 \wedge i = 1 \wedge s = 0\}$

  ```
  while (i <= n) do
    s = s + i;
    i = i + 1;
  endwhile
  ```

  $\{Q_C : s = \sum_{k=1}^{n} k\}$

  ¡es una tripla de Hoare válida!

- Esto significa que:

  1. Si el ciclo comienza en un estado que cumple P_C
  2. ... entonces termina luego de un número finito de pasos
  3. y además en un estado que cumple Q_C

Otro ejemplo: Chequeo de paridad

Sea una secuencia de booleans s, contar la cantidad de posiciones de la secuencia iguales a `true`.

Algunas propiedades de $\#apariciones$:

- $\#apariciones(\langle\rangle, true) = 0$
- $\#apariciones(concat(s, \langle e\rangle), true) =$
  $\#apariciones(s, true) + (\text{if } e = true \text{ then } 1 \text{ else } 0 \text{ fi})$

Otro ejemplo: Chequeo de paridad

- ¿Es válida la siguiente tripla de Hoare?

- $\{i = 0 \wedge c = 0\}$

  ```
  while( i < |s| ) do
     if s[i]=true then
       c := c + 1
     else
       skip
     endif;
     i := i +1
  endwhile
  ```

  $\{c = \#apariciones(s, true)\}$

Otro ejemplo: Chequeo de Paridad

- Para probar que se cumplen las condiciones del Teorema del Invariante tenemos que demostrar formalmente que se cumple:

  1. $P_C \Rightarrow I$
  2. $\{I \wedge B\}$ S $\{I\}$
  3. $I \wedge \neg B \Rightarrow Q_C$

- ¿Cuál es el predicado P_C, Q_C, I y B?
  - $P_C \equiv i = 0 \wedge c = 0$
  - $Q_C \equiv c = \#apariciones(s, true)$
  - $B \equiv i < |s|$
  - $I \equiv 0 \leq i \leq |s| \wedge_L c = \#apariciones(subseq(s, 0, i), true)$

Otro ejemplo: Chequeo de Paridad

1. $P_C \Rightarrow I$?

$$
\begin{array}{rcl}
P_C & \equiv & i = 0 \wedge c = 0 \\
 & \Rightarrow & 0 \leq i \leq |s| \wedge c = 0 \\
 & \Rightarrow & 0 \leq i \leq |s| \wedge c = \#apariciones(\langle\rangle, true) \\
 & \Rightarrow & 0 \leq i \leq |s| \wedge c = \#apariciones(subseq(s, 0, 0), true) \\
 & \Rightarrow & 0 \leq i \leq |s| \wedge c = \#apariciones(subseq(s, 0, i), true) \\
 & \equiv & I \checkmark
\end{array}
$$

Otro ejemplo: Chequeo de Paridad

3. $I \wedge \neg B \Rightarrow Q_C$?

$$
\begin{array}{rcl}
I \wedge \neg B & \equiv & 0 \leq i \leq |s| \wedge c = \#apariciones(subseq(s, 0, i), true) \\
 & & \wedge\neg(i < |s|) \\
 & \Rightarrow & i = |s| \wedge c = \#apariciones(subseq(s, 0, i), true) \\
 & \Rightarrow & c = \#apariciones(subseq(s, 0, |s|), true) \\
 & \Rightarrow & c = \#apariciones(s, true) \\
 & \equiv & Q_C \checkmark
\end{array}
$$

Otro ejemplo: Chequeo de Paridad

2. Finalmente, tenemos que demostrar que $\{I \wedge B\}\ \mathtt{S}\ \{I\}$, para lo cual debemos probar que:

$$I \wedge B \ \Rightarrow_L wp(\mathtt{S}, I).$$

- Calculamos:

$$\begin{array}{cl}
 & wp(\mathtt{if...endif;i:=i+1}, I) \\
\equiv & wp(\mathtt{if...endif}, wp(\mathtt{i:=i+1}, I)) \\
\equiv & wp(\mathtt{if...endif}, I^i_{i+1}) \\
\equiv & (s[i] = \mathit{true} \wedge wp(\mathtt{c:=c+1}, I^i_{i+1})) \vee (s[i] = \mathit{false} \wedge wp(\mathtt{skip}, I^i_{i+1})) \\
\equiv & (s[i] = \mathit{true} \wedge (I^i_{i+1})^c_{c+1})) \vee (s[i] = \mathit{false} \wedge I^i_{i+1})
\end{array}$$

- Para probar que esto es verdadero, separemos en 2 casos: $s[i] = \mathit{true}$ y $s[i] = \mathit{false}$

Otro ejemplo: Chequeo de Paridad

- Si $s[i] = \mathit{true}$, entonces podemos simplificar el predicado:

$$
\begin{array}{ll}
 & (s[i] = \mathit{true} \wedge (I^i_{i+1})^c_{c+1}) \vee (s[i] = \mathit{false} \wedge I^i_{i+1}) \\
\equiv & (s[i] = \mathit{true} \wedge (I^i_{i+1})^c_{c+1}) \\
\equiv & (I^i_{i+1})^c_{c+1} \\
\equiv & 0 \leq i+1 \leq |s| \wedge_L c+1 = \#\mathit{apariciones}(\mathit{subseq}(s, 0, i+1), \mathit{true})
\end{array}
$$

- Ahora probemos que este predicado es verdadero:
  - Por hipótesis, $0 \leq i \leq |s|$ y $i < |s|$
  - Por lo tanto,

$$
\begin{array}{ll}
 & 0 \leq i < |s| \\
\Rightarrow & 0 \leq i+1 \leq |s|
\end{array} \tag{1}
$$

Otro ejemplo: Chequeo de Paridad

Por hipótesis:

$$c = \#apariciones(subseq(s, 0, i), true)$$

Por lo tanto,

$$\begin{aligned}
c + 1 &= \#apariciones(subseq(s, 0, i), true) + 1 \\
&= \#apariciones(subseq(s, 0, i), true) + (\text{if } true = true \text{ then } 1 \text{ else } 0 \text{ fi}) \\
&= \#apariciones(subseq(s, 0, i), true) + (\text{if } s[i] = true \text{ then } 1 \text{ else } 0 \text{ fi}) \\
&= \#apariciones(subseq(s, 0, i + 1), true) \qquad (2)
\end{aligned}$$

Finalmente, por (1) y (2) demostramos que $I \wedge B \Rightarrow wp(\mathtt{S}, I)$ para el caso que $s[i] = true$ (pero aún falta probarlo para $s[i] = false$)

Otro ejemplo: Chequeo de Paridad

- Si $s[i] = \mathit{false}$, entonces podemos nuevamente simplificar el predicado:

$$\begin{aligned}
& (s[i] = \mathit{true} \land (I^i_{i+1})^c_{c+1}) \lor (s[i] = \mathit{false} \land I^i_{i+1}) \\
\equiv\ & (s[i] = \mathit{false} \land I^i_{i+1}) \\
\equiv\ & I^i_{i+1} \\
\equiv\ & 0 \leq i+1 \leq |s| \land_L c = \#\mathit{apariciones}(\mathit{subseq}(s, 0, i+1), \mathit{true})
\end{aligned}$$

- Ahora probemos que este predicado es verdadero:
  - Por hipótesis, $0 \leq i \leq |s|$ y $i < |s|$
  - Análogo al caso $s[i] = \mathit{true}$, podemos probar que:

$$0 \leq i+1 \leq |s| \tag{3}$$

Otro ejemplo: Chequeo de Paridad

Por hipótesis:

$$c = \#apariciones(subseq(s, 0, i), true)$$

Por lo tanto,

$$\begin{aligned}
c &= \#apariciones(subseq(s, 0, i), true) + 0 \\
&= \#apariciones(subseq(s, 0, i), true) + (\text{ if } false = true \text{ then } 1 \text{ else } 0 \text{ fi}) \\
&= \#apariciones(subseq(s, 0, i), true) + (\text{ if } s[i] = true \text{ then } 1 \text{ else } 0 \text{ fi}) \\
&= \#apariciones(subseq(s, 0, i+1), true) \qquad (4)
\end{aligned}$$

Finalmente, por (3) y (4) demostramos que $I \wedge B \Rightarrow wp(\mathtt{S}, I)$ para el caso que $s[i] = false$. Y como ya probamos lo mismo para $s[i] = true$, podemos concluir que:

$$\{I \wedge B\}S\{I\} \checkmark$$

Otro ejemplo: Chequeo de Paridad

- Ya que probamos
  - $P_C \Rightarrow I$
  - $\{I \wedge B\}S\{I\}$
  - $I \wedge \neg B \Rightarrow Q_C$
- usando el teorema del invariante pudimos probar que (si el ciclo termina), se cumple Q_C.
- Ya probamos que $I \equiv 0 \leq i \leq |s| \wedge_L c = \#apariciones(s, true)$ es un invariante del ciclo.
- ¡Pero no probamos todavía que la ejecución del ciclo termina!

Otro ejemplo: Chequeo de Paridad

- La función variante representa una cantidad que se va reduciendo.
- Pero... ¿Cuál la condición para que se detenga el ciclo?
  - $B \equiv i < |s|$
  - Necesitamos que $fv \leq 0$ implique $i < |s|$
- Por lo que proponemos entonces:

$$fv \;=\; |s| - i$$

Otro ejemplo: Chequeo de Paridad

- Sea la siguiente función candidato a función variante:

$$fv \;=\; |s| - i$$

- Veamos como evoluciona con los valores para $|s| = 4$

Iteración	$\lvert s\rvert$	i	$fv \;=\; \lvert s\rvert - i$
0	4	0	4-0=4
1	4	1	4-1=3
2	4	2	4-2=2
3	4	3	4-3=1
4	4	4	4-4=0

Otro ejemplo: Chequeo de Paridad

- Con esta definición de fv, veamos si se cumplen las dos condiciones del Teorema de Terminación:
  1. $\{I \wedge B \wedge fv = v_0\}$ S $\{fv < v_0\}$
  2. $I \wedge fv \leq 0 \Rightarrow \neg B$

Otro ejemplo: Chequeo de Paridad

- $\{I \wedge B \wedge fv = v_0\}$ S $\{fv < v_0\}$?

  Para demostrarlo tenemos que probar que:

  $$I \wedge B \wedge fv = v_0 \Rightarrow wp(\mathtt{S}, fv < v_0)$$

Otro ejemplo: Chequeo de Paridad

- Comenzamos con la definición de la *wp*:

$$
\begin{aligned}
& wp(\texttt{if...endif;i:=i+1}, |s| - i < v_0) \\
\equiv\quad & wp(\texttt{if...endif}, wp(\texttt{i:=i+1}, |s| - i < v_0)) \\
\equiv\quad & wp(\texttt{if...endif}, (|s| - i < v_0)^{i}_{i+1}) \\
\equiv\quad & wp(\texttt{if...endif}, |s| - (i+1) < v_0) \\
\equiv\quad & (s[i] = \mathit{true} \wedge wp(\texttt{c:=c+1}, |s| - (i+1) < v_0)) \\
& \vee (s[i] = \mathit{false} \wedge wp(\texttt{skip}, |s| - (i+1) < v_0)) \\
\equiv\quad & (s[i] = \mathit{true} \wedge (|s| - (i+1) < v_0)^{c}_{c+1}) \\
& \vee (s[i] = \mathit{false} \wedge |s| - (i+1) < v_0) \\
\equiv\quad & (s[i] = \mathit{true} \wedge |s| - (i+1) < v_0) \\
& \vee (s[i] = \mathit{false} \wedge |s| - (i+1) < v_0) \\
\equiv\quad & (s[i] = \mathit{true} \vee s[i] = \mathit{false}) \wedge |s| - (i+1) < v_0 \\
\equiv\quad & |s| - (i+1) < v_0
\end{aligned}
$$

Otro ejemplo: Chequeo de Paridad

- ... como $fv = v_0$ equivale a $|s| - i$, reemplazamos v_0 con esa expresión

$$\begin{aligned} &\equiv \quad |s| - (i+1) < |s| - i \\ &\equiv \quad -(i+1) < -i \\ &\equiv \quad (i+1) > i \end{aligned}$$

- Lo cual es verdadero.
- Por lo tanto, demostramos que

$$I \wedge B \wedge fv = v_0 \Rightarrow wp(\mathtt{S}, fv < v_0) \checkmark$$

Otro ejemplo: Chequeo de Paridad

2. $I \wedge fv \leq 0 \Rightarrow \neg B$?

$$\begin{aligned} fv \leq 0 &\equiv |s| - i \leq 0 \\ &\equiv |s| \leq i \\ &\Rightarrow \neg(i < |s|) \\ &\equiv \neg B \checkmark \end{aligned}$$

- Por lo tanto, probamos que $I \wedge fv \leq 0 \Rightarrow \neg B$
- Ya que se cumplen sus hipótesis, por el teorema de terminación podemos concluir que el ciclo siempre termina.

Otro ejemplo: Chequeo de Paridad

- Finalmente, probamos que:

    1. $P_C \Rightarrow I$
    2. $\{I \wedge B\} S \{I\}$
    3. $I \wedge \neg B \Rightarrow Q_C$
    4. $\{I \wedge v_0 = fv\} S \{fv < v_0\}$
    5. $I \wedge fv \leq 0 \Rightarrow \neg B$

- Entonces, por (1)-(5) , se cumplen las hipótesis de ambos teoremas (teorema del invariante + teorema de terminación).

- Por lo tanto, la tripla de Hoare es válida (i.e., dada P_C, el ciclo siempre termina y vale Q_C)

Recap #1: Teorema del invariante

- **Teorema.** Si def(B) y existe un predicado I tal que

  1. $P_C \Rightarrow I$,
  2. $\{I \wedge B\}\ S\ \{I\}$,
  3. $I \wedge \neg B \Rightarrow Q_C$,

  ... y **el ciclo termina**, entonces la siguiente tripla de Hoare es válida:

$$\{P_C\}\ \textbf{while B do S endwhile}\ \{Q_C\}$$

Recap #2: Teorema de terminación de un ciclo

- **Teorema.** Sea $\mathbb{V}$ el producto cartesiano de los dominios de las variables del programa y sea I un invariante del ciclo **while B do S endwhile**. Si existe una función $fv : \mathbb{V} \rightarrow \mathbb{Z}$ tal que

  1. $\{I \wedge B \wedge v_0 = fv\}$ **S** $\{fv < v_0\}$,
  2. $I \wedge fv \leq 0 \Rightarrow \neg B$,

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

- La función fv se llama función variante del ciclo.

Teorema de corrección de un ciclo

- **Teorema.** Sean un predicado I y una función $fv : \mathbb{V} \to \mathbb{Z}$ (donde $\mathbb{V}$ es el producto cartesiano de los dominios de las variables del programa), y supongamos que $I \Rightarrow \text{def}(B)$. Si

  1. $P_C \Rightarrow I$,
  2. $\{I \wedge B\}\ S\ \{I\}$,
  3. $I \wedge \neg B \Rightarrow Q_C$,
  4. $\{I \wedge B \wedge v_0 = fv\}$ **S** $\{fv < v_0\}$,
  5. $I \wedge fv \leq 0 \Rightarrow \neg B$,

  ... entonces la siguiente tripla de Hoare es válida:

  $$\{P_C\}\ \textbf{while B do S endwhile}\ \{Q_C\}$$

Teorema de corrección de un ciclo

- El **teorema de corrección de un ciclo** nos permite demostrar la validez de una tripla de Hoare cuando el programa es un ciclo.

- Por definición, si probamos que:

$$\{P_C\} \text{ while B do S endwhile } \{Q_C\}$$

... entonces probamos que:

$$P_C \Rightarrow wp(\texttt{while B do S endwhile}, Q_C)$$

- **¡Cuidado!** Probar lo anterior no significa haber obtenido un predicado que caracteriza a la precondición más débil del ciclo:

$$wp(\text{while B do S endwhile}, Q_C)$$

Programas con ciclos

- En general, no se puede definir un mecanismo efectivo para obtener una fórmula cerrada que represente la precondición más débil de un ciclo.
- Entonces, ¿cómo hacemos para probar la corrección y terminación de un programa que incluye ciclos intercalados con otras instrucciones?

Guía para demostrar programas con ciclos

¿Qué tenemos que hacer para probar que $\{Pre\}\texttt{S1;while...;S3}\{Post\}$ es válida?

1. $Pre \Rightarrow_L wp(S1, P_C)$
2. $P_C \Rightarrow_L wp(while..., Q_C)$
3. $Q_C \Rightarrow_L wp(S3, Post)$

Por monotonía, esto nos permite demostrar que $Pre \Rightarrow_L wp(\texttt{S1;while...;S3}, Post)$ es verdadera.

Recap: SmallLang

- Para las demostraciones de corrección, introdujimos un lenguaje sencillo y con menos opciones (mucho más simple que C++). Llamemos **SmallLang** a este lenguaje.
- SmallLang tiene únicamente:
  - Nada: `skip`
  - Asignación: `x := E`
  - Secuencia: `S1;S2`
  - Condicional: `if B then S1 else S2 endif`
  - Ciclo: `while B do S endwhile`
- No posee memoria dinámica (punteros), aliasing, llamados a función, estructura `for`, etc.

C++ → SmallLang

Pero dado un programa en C++ podemos traducirlo a SmallLang preservando su semántica (comportamiento).
Por ejemplo:

Versión C++

```
for (int i =0; i < s.size(); i++) {
  if (s[i]==0) {
    s[i]++;
  }
}
```

Versión SmallLang

```
i:=0;
while (i < s.size()) do
  if (s[i]==0)
    s[i]:=s[i]+1
  else
    skip
  endif;
  i:=i+1
endwhile
```

Ambos programas tienen el mismo comportamiento.

Corrección de programas en C++

Para demostrar la corrección de un programa en C++ con respecto a una especificación, podemos:

1. Traducir el programa C++ a SmallLang preservando su comportamiento.
2. Demostrar la corrección del programa en SmallLang con respecto a la especificación.
3. Entonces, probamos la corrección del comportamiento del programa original.

Bibliografía

- David Gries - The Science of Programming
  - Part II - The Semantics of a Small Language
    - Chapter 11 - The Iterative Command