Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Las marcas en rojo son correcciones a la publicación original

Demostración de corrección de ciclos en SmallLang

Teorema del invariante: corrección de ciclos

Ejercicio 1. Consideremos el problema de sumar los elementos de un arreglo y la siguiente implementación en SmallLang, con el invariante del ciclo.

Especificación

`proc sumar` (in s: $array < \mathbb{Z} >$) : $\mathbb{Z}$
`requiere` $\{True\}$
`asegura` $\{res = \sum_{j=0}^{|s|-1} s[j]\}$

Implementación en SmallLang

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

Invariante de Ciclo

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

a) Escribir la precondición y la postcondición del ciclo.

b) ¿Qué punto falla en la demostración de corrección si el primer término del invariante se reemplaza por $0 \leq i < |s|$?

c) ¿Qué punto falla en la demostración de corrección si el límite superior de la sumatoria $(i-1)$ se reemplaza por i?

d) ¿Qué punto falla en la demostración de corrección si se invierte el orden de las dos instrucciones del cuerpo del ciclo?

e) Mostrar la corrección parcial del ciclo, usando los primeros puntos del teorema del invariante.

f) Proponer una función variante y mostrar la terminación del ciclo, utilizando la función variante.

Ejercicio 2. Dadas la especificación y la implementación del problema `sumarParesHastaN`, escribir la precondición y la postcondición del ciclo, y mostrar su corrección a través del teorema del invariante.

Especificación

`proc sumarParesHastaN` (in n: $\mathbb{Z}$) : $\mathbb{Z}$
`requiere` $\{n \geq 0\}$
`asegura` $\{res = \sum_{j=0}^{n-1}(\text{if } j \ mod\ 2 = 0 \text{ then } j \text{ else } 0 \text{ fi})\}$

Implementación en SmallLang

```
res := 0;
i := 0;
while (i < n) do
  res := res + i;
  i := i + 2
endwhile
```

Invariante de ciclo

$$I \equiv 0 \leq i \leq n+1 \wedge i \ mod\ 2 \ = \ 0 \wedge res = \sum_{j=0}^{i-1}(\text{if } j \ mod\ 2 = 0 \text{ then } j \text{ else } 0 \text{ fi})$$

Ejercicio 3. Considere el problema `sumaDivisores`, dado por la siguiente especificación:

`proc sumaDivisores` (in n: $\mathbb{Z}$) : $\mathbb{Z}$
`requiere` $\{n \geq 1\}$
`asegura` $\{res = \sum_{j=1}^{n}(\text{if } n \ mod\ j = 0 \text{ then } j \text{ else } 0 \text{ fi})\}$

a) Escribir un programa en SmallLang que satisfaga la especificación del problema y que contenga exactamente un ciclo.

b) Escribir la pre y post condición del ciclo y su invariante.

c) Considere el siguiente invariante para este problema

$$I \equiv 1 \leq i \leq n/2 \wedge res = \sum_{j=1}^{i} (\textsf{if } n \bmod j = 0 \textsf{ then } j \textsf{ else } 0 \textsf{ fi})$$

Si no coincide con el propuesto en el inciso anterior, ¿qué cambios se le deben hacer al programa para que lo represente este invariante? ¿Deben cambiar la pre y post condición?

Ejercicio 4. Considere la siguiente especificación e implementación del problema `copiarSecuencia`, y la pre y post condiciones del ciclo

Especificación

`proc copiarSecuencia` (in s: $array < \mathbb{Z} >$, inout r: $array < \mathbb{Z} >$)
`requiere` $\{|s| = |r| \wedge r = r_0\}$
`asegura` $\{|s| = |r| \wedge_L (\forall j : \mathbb{Z})(0 \leq j < |s| \rightarrow_L s[j] = r[j])\}$

Implementación en SmallLang

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

$$P_c \equiv |s| = |r| \wedge i = 0$$

$$Q_c \equiv (\forall j : \mathbb{Z})(0 \leq j < |r| \rightarrow_L s[j] = r[j])$$

a) ¿Qué variables del programa deben aparecer en el invariante?

b) Proponer un invariante e indicar qué clausula del mismo es necesario para cada paso de la demostración.

c) Proponer una función variante que permita demostrar que el ciclo termina.

d) Comparar la solución propuesta con la que ofrecemos al final de la guía.

Ejercicio 5. Sea el siguiente ciclo con su correspondiente precondición y postcondición:

```
while (i >= s.size() / 2) do
    suma := suma + s[s.size()-1-i];
  i := i - 1
endwhile
```

$$P_c : \{|s| \bmod 2 = 0 \wedge i = |s| - 1 \wedge suma = 0\}$$

$$Q_c : \{|s| \bmod 2 = 0 \wedge i = |s|/2 - 1 \ \wedge_L \ suma = \sum_{j=0}^{|s|/2-1} s[j]\}$$

a) Proponer un invariante e indicar qué clausula del mismo es necesario para cada paso de la demostración.

b) Proponer una función variante que permita demostrar que el ciclo termina.

c) Comparar la solución propuesta con la que ofrecemos al final de la guía.

Ejercicio 6. Dado el siguiente problema

`proc sumarElementos` (in s: $array < \mathbb{Z} >$) : $\mathbb{Z}$
`requiere` $\{|s| \geq 1\}$
`asegura` $\{res = \sum_{j=0}^{|s|-1} s[j]\}$

a) Corregir las siguientes implementaciones

b) Dar un invariante y función variante para cada una de estas implementaciones

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

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

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

d)
```
res := 0
i := 0
while (i > s.size() / 2) do
        res := res + s[i] + s[s.size() - i];
        i := i + 1
endwhile
```

Ejercicio 7. Considerando el siguiente Invariante:

$$I \equiv \{0 \leq i \leq |s| \wedge (\forall j : \mathbb{Z})(0 \leq j < i \rightarrow_L ((j\ mod\ 2 = 0 \wedge s[j] = 2 \times j) \vee (j\ mod\ 2 \neq 0 \wedge s[j] = 2 \times j + 1)))\}$$

- Escribir un programa en SmallLang que se corresponda al invariante dado.
- Defina las P_c, B y Q_c que correspondan a su programa.
- Dar una función variante para que se pueda completar la demostración.

Ejercicio 8. Considerando el siguiente Invariante:

$$I \equiv \{0 \leq i \leq |s|/2 \wedge (\forall j : \mathbb{Z})(0 \leq j < i) \rightarrow_L (s[j] = 0 \wedge s[|s| - j - 1] = 0)\}$$

- Escribir un programa en SmallLang que se corresponda al invariante dado.
- Defina las P_c, B y Q_c que correspondan a su programa.
- Dar una función variante para que se pueda completar la demostración.

Ejercicio 9. Indique si el siguiente enunciado es verdadero o falso; fundamente:

Si dados B y I para un ciclo S existe una función f_v que cumple lo siguiente:

- $\{I \wedge B \wedge f_v = V_0\}S\{f_v > V_0\}$
- $\exists(k : \mathbb{Z})(I \wedge f_v \geq k \rightarrow \neg B)$

entonces el ciclo siempre termina.

Ejercicio 10. Considere la siguiente especificación y su implementación

Especificación

`proc existeElemento` (in s: $array < \mathbb{Z} >$, in e: $\mathbb{Z}$) : Bool
`requiere` $\{True\}$
`asegura` $\{res = \text{true} \leftrightarrow$
$((\exists k : \mathbb{Z})(0 \leq k < |s|) \wedge_L s[k] = e)\}$

Implementación en SmallLang

```
i := 0;
j := -1;
while (i < s.size()) do
  if (s[i] = e) then
     j := i
 else
    skip
 endif;
 i := i + 1
endwhile;
if (j != -1)
   res := true
else
   res := false
endif
```

Escribir los pasos necesarios para demostrar la correctitud de la implementación respecto a la especificación usando WP y el teorema del invariante

---

Soluciones

Ejercicio 4

- $I \equiv 0 \leq i \leq |s| \wedge (\forall j : \mathbb{Z})(0 \leq j < i \rightarrow_L s[j] = r[j])$
- $f_v = |s| - i$

Ejercicio 5

a) $I \equiv |s|/2 - 1 \leq i \leq |s| - 1 \wedge_L suma = \sum_{j=0}^{|s|-2-i} s[j]$

b) $f_v = i - (|s|/2 - 1)$