Metodología de la Programación

Grado en Ingeniería Informática de Gestión y Sistemas de Información
Escuela de Ingeniería de Bilbao (UPV/EHU)
Departamento de Lenguajes y Sistemas Informáticos
Curso: 1^o
Curso académico: 2019-2020
Grupo 01
Tema 3: Verificación formal de programas
2 puntos

12-03-2020

Enunciado

Índice

Índice de figuras

Índice de tablas

****

1. Verificación formal de un programa iterativo (*2 puntos*)

Verificar, utilizando el Cálculo de Hoare, la corrección total del programa que se muestra en la figura 1 (página 3), con respecto a la precondición φ, la postcondición ψ, el invariante INV y la expresión cota E. Según la especificación pre-post formalizada mediante las fórmulas φ y ψ, dados un número entero positivo x que es una potencia de 2 y dos vectores no vacíos de enteros positivos $A(1..n)$ y $B(1..n)$ que solo contienen potencias de 2, el programa debería decidir en la variable booleana w si algún elemento de $A(1..n)$ multiplicado por x es igual al correspondiente elemento de $B(1..n)$.

En el programa de la figura 1, div representa la división entera. Ejemplos: $19\ div\ 2 = 9$; $19\ div\ 3 = 6$; $19\ div\ 4 = 4$; $17\ div\ 3 = 5$; $8\ div\ 12 = 0$.

Las potencias enteras de 2 son $2^0, 2^1, 2^2, 2^3, 2^4, 2^5, \ldots$. Es decir, $1, 2, 4, 8, 16, 32, \ldots$. El conjunto formado por todas las potencias enteras de 2 se puede definir formalmente de la siguiente manera: $\{y \mid y \in I\!N\ \wedge$

$\exists \ell(\ell \geq 0 \ \wedge \ y = 2^{\ell})\}$, donde $I\!N$ es el conjunto de los números naturales, es decir, $I\!N = \{0, 1, 2, 3, 4, \ldots\}$. Cualquier potencia entera de 2 es un número positivo (≥ 1).

Durante el proceso de verificación, al analizar el punto (IV) de la regla del While, es necesario utilizar la propiedad matemática $(Prop)$ de la figura 2 (página 3), que dice que, si se tienen tres valores enteros positivos m, p y q que son potencias de 2, entonces el que q sea igual al cociente de dividir m por p, quiere decir que el resultado de multiplicar q por p es igual a m. Por ejemplo, si $m = 32$, $p = 4$ y $q = 8$, puesto que se cumple $8 = 32 \ div \ 4$, también se cumple $8 * 4 = 32$. Nótese que no todos los enteros positivos cumplen lo expresado en la segunda parte de la implicación que aparece en la definición de $(Prop)$. Por ejemplo, $3 = 19 \ div \ 5$, pero $3 * 5 \neq 19$.

En la tabla 1 (página 4) se recogen las abreviaciones que se recomienda utilizar durante el proceso de verificación. En la tabla 2 (página 4) se recopilan las denominaciones de las letras griegas utilizadas en este documento. Finalmente, en la tabla 3 (página 4) se muestra la puntuación de los distintos pasos o apartados que han de ser considerados en el proceso de verificación.

Todos los elementos numéricos de las figuras 1 y 2 y de la tabla 1 representan números enteros. Por tanto, los valores representados por esos elementos pertenecen a $\mathbb{Z}$, donde el conjunto $\mathbb{Z}$ es el siguiente:

$$\{\ldots, -4, -3, -2, -1, 0, 1, 2, 3, 4, \ldots\}$$

Formalmente, $\mathbb{Z} = I\!N \ \cup \ \{-y \mid y \in I\!N \ \wedge \ y \geq 1\}$, donde $\cup$ es la unión de conjuntos.

Teniendo en cuenta las abreviaciones de la tabla 1, la propiedad $(Prop)$ de la figura 2 expresa que cada $\gamma(\ell)$ es igual al correspondiente $\sigma(\ell)$, siempre que se garantice que $\gamma(\ell)$, $\sigma(\ell)$ y x son potencias enteras de 2, lo cual es garantizado por λ.

Ejemplo. (Para el programa de la figura 1) *Sean $x = 4$ y los siguientes vectores $A(1..8)$ y $B(1..8)$:*

A(1..8)	*1*	*2*	*4*	*32*	*8*	*4*	*4*	*16*
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*
B(1..8)	*16*	*1*	*16*	*8*	*32*	*4*	*64*	*32*
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*

Para esos valores de x, $A(1..8)$ y $B(1..8)$, el programa de la figura 1 devolvería el valor booleano $True$ en w, porque al menos para un elemento de $A(1..8)$ se cumple que el resultado que se obtiene al multiplicarlo por x es igual al elemento que ocupa la misma posición de $B(1..8)$. En concreto, se cumple para $A(3)$ y $A(5)$. Es decir, $A(3) * x = B(3)$ y $A(5) * x = B(5)$.

Programa:

$\{\varphi\}$
i := 2;
while $\{INV\}$ $\{E\}$ i $\neq$ n + 2 **and not** w **loop**
    w := (A(i − 1) = B(i − 1) div x);
    i := i + 1;
end loop;
$\{\psi\}$

Definición de φ, INV, E y ψ:

$$\varphi \equiv \quad n \geq 1 \ \wedge \ pot\_dos\_num(x) \ \wedge \ pot\_dos\_vector(A(1..n)) \ \wedge \\ pot\_dos\_vector(B(1..n)) \ \wedge \ \neg w$$

$$INV \equiv \quad n \geq 1 \ \wedge \ pot\_dos\_num(x) \ \wedge \ pot\_dos\_vector(A(1..n)) \ \wedge \\ pot\_dos\_vector(B(1..n)) \ \wedge \ (2 \leq i \leq n+2) \ \wedge \\ (w \leftrightarrow algun\_cociente(x, A(1..n), B(1..n), i-2))$$

$$E = \quad n + 2 - i$$

$$\psi \equiv \quad w \leftrightarrow \exists k(1 \leq k \leq n \ \wedge \ A(k) * x = B(k))$$

Definición de los predicados utilizados:

$$pot\_dos\_num(z) \equiv \exists \ell(\ell \geq 0 \ \wedge \ z = 2^{\ell})$$

$$pot\_dos\_vector(H(1..r)) \equiv \forall k(1 \leq k \leq r \ \rightarrow \ pot\_dos\_num(H(k)))$$

$$algun\_cociente(y, F(1..r), G(1..r), pos) \equiv \exists k(1 \leq k \leq pos \ \wedge \ F(k) = G(k) \ div \ y)$$

Figura 1: Programa a verificar, definiciones de φ, INV, E y ψ y definiciones de los predicados utilizados.

Propiedad ($Prop$):

$$\forall m, p, q((pot\_dos\_num(m) \ \wedge \ pot\_dos\_num(p) \ \wedge \ pot\_dos\_num(q)) \rightarrow (q = m \ div \ p \leftrightarrow q * p = m))$$

Figura 2: Propiedad que cumplen tres potencias enteras de 2 cualesquiera.

Abreviaciones recomendadas:
$\lambda \equiv n \geq 1 \;\wedge\; pot\_dos\_num(x) \;\wedge\; pot\_dos\_vector(A(1..n)) \;\wedge\; pot\_dos\_vector(B(1..n))$
$B \equiv i \neq n+2 \; and \; not \; w$
$\gamma(\ell) \equiv A(\ell) = B(\ell) \; div \; x$
$\sigma(\ell) \equiv A(\ell) * x = B(\ell)$
$\mu(\ell) \equiv algun\_cociente(x, A(1..n), B(1..n), \ell)$

Tabla 1: Abreviaciones que se recomienda utilizar.

Letras griegas utilizadas:
φ : fi $\quad\psi$: psi $\quad\gamma$: gamma $\quad\sigma$: sigma $\quad\mu$: mu $\quad\lambda$: lambda

Tabla 2: Denominaciones de las letras griegas utilizadas.

Puntuación:

(a) Partición inicial y esquema: 0,100

(b) Verificación de la asignación inicial: 0,150
(Cálculo de fórmulas: 0,050. Comprobación de la implicación: 0,100)

(c) Punto (I) de la regla del while: 0,005

(d) Punto (II) de la regla del while: 0,020

(e) Punto (III) de la regla del while: 0,700
(Cálculo de fórmulas: 0,200. Comprobación de la implicación: 0,500)

(f) Punto (IV) de la regla del while: 0,500
(Casos $i = n+2$: 0,200. Caso $i \neq n+2$: 0,300)

(g) Punto (V) de la regla del while: 0,075

(h) Punto (VI) de la regla del while: 0,200
(Cálculo de fórmulas: 0,050. Comprobación de la implicación: 0,150)

(i) Demostración formal de la corrección: 0,250

- Cuando no se explique por qué se cumple una implicación, se contará cero. Es decir, indicar que una implicación sí se cumple sin razonar por qué se cumple cuenta 0.
- Para aprobar el ejercicio es obligatorio obtener al menos la mitad de la puntuación tanto del apartado *(e)* como del apartado *(f)* (puntos (III) y (IV) de la regla del while).

Tabla 3: Puntuación por apartados.