

## Ejemplos Invariantes de Ciclo

### Problema de bisqueda en un arreglo arbitrario

Entrada: Un arreglo  $A[0..N]$  de números y un número  $v$ .

Solida:  $\exists p \in [0..N] . A[p] = v$

```

1. bool solve(vector<int> &A, int v) {
2.     bool ans=false;
3.     int i=0;
4.
5.     while(i < A.size()){
6.         if(A[i]==v)
7.             ans=true;
8.         ++i;
9.     }
10.
11.    return ans;
12. }
```

Invariantes:

$$I_0 : 0 \leq i \leq N$$

$$I_1 : \text{ans} = \exists p \in [0..i] . A[p] = v$$

Teorema: Los invariantes  $I_0$  y  $I_1$  se cumplen.

Demarcación: Se procede mostrando la validez de los invariantes para la inicialización y la estabilidad.

Inicialización: De acuerdo a los líneas 2-3 inicialemente  $i=0$  y  $ans=false$

Para el invariente  $I_0$  se tiene que:

$$0 \leq i \leq N$$

$$0 \leq 0 \leq N$$



Para el invariante  $I_1$ , se tiene que:

$$ans = \exists p \in [0..i]. A[p] = v$$

$$\text{false} = \exists p \in [0..0]. A[p] = v$$

$$\text{false} = \text{false}$$

dado que el rango de posiciones  $[0..0]$  es un rango vacío y en consecuencia no puede existir el valor  $p$  que haga verdadero el existencial.

Por lo tanto, los invariantes  $I_0$  y  $I_1$  se cumplen en la inicialización.

**Establecida:** Se considera una iteración arbitraria  $i=j$  y se asume que antes de ejecutar esa iteración se cumplen los invariantes. Esto quiere decir que se asume que las expresiones

$$1 \leq j \leq N$$

$$ans = \exists p \in [0..j]. A[p] = v$$

son verdaderas. El objetivo es mostrar que después de esta iteración o sea antes de la iteración  $i=j+1$  los invariantes son ciertos y por lo tanto las expresiones:

$$1 \leq j+1 \leq N$$

$$ans = \exists p \in [0..j+1]. A[p] = v$$

se cumplen. Al ejecutar las líneas 6-8:

```
if (A[j] == v)
    ans = true;
    ++i;
```

Hay dos posibilidades:

- $A[j] = v$ : En este caso  $ans$  pasará a valer  $true$ :

$$ans = \exists p \in [0..j+1). A[p] = v$$

y esto es correcto porque la posición  $j$  del arreglo  $A$  es igual a  $v$ .

$\neg A[i] = v$ ; En este caso  $ans$  no se modificará y conservará el valor que tenía antes de la iteración.

$$ans = \exists p \in [0..j+1). A[p] = v = \underbrace{\exists p \in [0..j). A[p] = v}_{\downarrow \\ \text{j valor de ans se conserva}}$$

Esto es correcto porque dado que el valor no cambia, si antes de la iteración se cumplía el invariante entonces se va a seguir cumpliendo.

Similarmemente, después de ejecutar la línea 8 el valor de  $i$  pasará a ser  $j+1$ . Luego, el invariante  $I_0$  es verdadero ya que:

$$i \leq j+1 \leq N$$

Tiene que ser cierto ya que  $j < N$  puesto que de lo contrario no se habrían ejecutado las líneas del ciclo.

Finalmente, el ciclo terminará ya que el valor de  $i$  se incrementará de 1 en 1 hasta eventualmente llegar a  $N$ . Luego por la estabilidad del invariante  $I_1$  se satisface:

$$ans = \exists p \in [0..i). A[p] = v$$

$$ans = \exists p \in [0..N). A[p] = v$$

Lo que coincide con la poscondición.

**Teorema:** La invocación  $solve(A, v)$  para algún arreglo  $A$  y número  $v$  produce  $true$  si  $v$  está en  $A$  y  $false$  en caso contrario.

**Demonstración:** Es trivial a partir de la correctitud de los invariantes  $I_0$  y  $I_1$ .

## Problema de mezclar arreglos

Entrada: Arreglos  $A[0..N]$  y  $B[0..M]$  de números ordenados ascendentemente.

Salida: Arreglo  $C[0..N+M]$  con los valores en  $A[0..N]$  y  $B[0..M]$  ordenados ascendentemente.

$$c = [1, 1] \quad A = [1, 5, 8, 9] \quad B = [1, 1, 6] \quad i=0 \quad j=2$$

```

1. def mezclar(A,B):
2.     i,j,C = 0,0, []
3.     while i < len(A) or j < len(B):
4.         if i < len(A) and j < len(B):
5.             if A[i] <= B[j]:
6.                 C.append(A[i])
7.                 i += 1
8.             else:
9.                 C.append(B[j])
10.                j += 1
11.            elif i < len(A):
12.                C.append(A[i])
13.                i += 1
14.            elif j < len(B):
15.                C.append(B[j])
16.                j += 1
17.    return C

```

Invariantes:

$$I_0 : 0 \leq i \leq N$$

$$I_1 : 0 \leq j \leq M$$

$I_2 : C = [c_0, c_1, \dots, c_{i+j-1}]$  tal que  $C$  contiene los  $i+j$  elementos más pequeños de  $A$  y  $B$  con los primeros  $i$  elementos de  $A$  y los primeros  $j$  elementos de  $B$  ordenados ascendentemente

Teorema: Los invariantes  $I_0$ ,  $I_1$  y  $I_2$  se cumplen.

Demarcación: Se procede mostrando la validez de los invariantes para la inicialización y la estabilidad.

Inicialización: De acuerdo a los linea 2 inicialmente  $i=0, j=0$  y  $C = []$ .

Para los invariantes  $I_0$  y  $I_1$  se tiene que

$$0 \leq i \leq N$$

$$0 \leq 0 \leq N$$

$$0 \leq j \leq M$$

$$0 \leq 0 \leq M$$

Para el invariante  $I_2$  se tiene que:

$$C = [c_0, c_1, \dots, c_{i+j-1}] = [c_0, c_1, \dots, c_{i+j-1}]$$

$$C = [c_0, \dots, c_{-1}]$$

$$C = []$$

dado que el rango de posiciones  $[0..-1]$  es un rango vacío la lista C debe estar vacía. Además, trivialmente se satisface que C contiene los primeros i elementos de A (i elementos) y los primeros j elementos de B ordenados ascendenteamente.

Por lo tanto, los invariantes  $I_0$ ,  $I_1$  y  $I_2$  se cumplen en la inicialización.

**Establecidos:** Se considera una iteración arbitraria en la que  $i \leq k$  y  $j \leq h$  y se asume que antes de ejecutar esa iteración se cumplen los invariantes. Esto quiere decir que se asume que las expresiones

$$\begin{array}{c} 0 \leq k \leq N \\ 0 \leq h \leq M \\ C = [c_0, c_1, \dots, c_{k+h-1}] \end{array}$$

son verdaderas. De esta forma se asume que antes de esta iteración C contiene los primeros k elementos de A y los primeros h elementos de B ordenados ascendenteamente. El objetivo es mostrar que después de esta iteración o sea antes de la siguiente iteración donde  $i = i'$  y  $j = j'$  los invariantes son ciertos y por lo tanto las expresiones:

$$0 \leq i' \leq N ? \quad 0 \leq j' \leq M ?$$

$$C = [c_0, c_1, \dots, c_{i'+j'-1}] ?$$

se cumplen. Al ejecutar los líneas 4-16 se tienen las siguientes posibilidades:

- La condición en la linea 4 es verdadera, osea que se tiene que  $k < N$  y  $h < M$ . Luego, para los líneas

5-10 se tienen dos posibilidades adicionales:

→ La condición en la linea 5 es verdadera, osea que se tiene que  $A[k] \leq B[h]$ . En este caso se ejecutarán las líneas 6-7:

c.append(A[k])  
i += 1

y en consecuencia:

$$c = [c_0, c_1, \dots, c_{k+h-1}, A[k]] \quad i' = i + 1 = k+1 \quad j' = j = h$$

De esta forma, los invariantes  $I_0$  y  $I_1$ , continúaran siendo verdaderos:

$$\begin{array}{ll} 0 \leq i' \leq N & 0 \leq j' \leq M \\ 0 \leq k+1 \leq N & 0 \leq h \leq M \end{array}$$

puesto que el valor de  $j$  no cambia y  $k < N$ . Además, la lista  $c$  contiene  $k+1+h$  elementos y contiene los  $k+1$  primeros elementos de  $A$  y los primeros  $h$  elementos de  $B$  ordenados ascendente mente. El elemento  $A[k]$  que se agregó al final de  $C$  será mayor que los elementos que ya estaban allí ya que los elementos de  $A$  y  $B$  están ordenados ascendente mente y por ende  $A[k]$  es mayor o igual a todos los elementos en  $A[0..k]$ . Por lo tanto, el invariante  $I_2$  continúará siendo verdadero.

→ La condición en la linea 5 es falsa, osea que se tiene que  $A[k] > B[h]$ . En este caso se ejecutarán las líneas 9-10:

c.append(B[h])  
j += 1

y en consecuencia:

$$c = [c_0, c_1, \dots, c_{k+h-1}, B[h]] \quad i' = i = k \quad j' = j + 1 = h + 1$$

De esta forma, los invariantes  $I_0$  y  $I_1$  continuarán siendo verdaderos:

$$\begin{array}{l} 0 \leq i' \leq N \\ 0 \leq k \leq N \end{array}$$

$$\begin{array}{l} 0 \leq j' \leq M \\ 0 \leq h+1 \leq M \end{array}$$

sabiendo que el valor de  $i$  no cambia y  $h < M$ . Además, la lista  $C$  contiene  $k+h+1$  elementos y contiene los  $k$  primeros elementos de  $A$  y los primeros  $h+1$  elementos de  $B$  ordenados ascendenteamente. El elemento  $B[h]$  que se agregó al final de  $C$  es mayor que los elementos que ya estaban allí ya que los elementos de  $A$  y  $B$  están ordenados ascendenteamente y por ende  $A[h]$  es mayor o igual a todos los elementos en  $B[0..h]$ . Por lo tanto, el invariante  $I_2$  continuará siendo verdadero.

- \* La condición en la línea 11 es verdadera, osea que  $k < N$ . En este caso se ejecutarán las líneas 12-13:

$c.append(A[k])$   
 $i := i + 1$

y en consecuencia:

$$c = [c_0, c_1, \dots, c_{k+h-1}, A[k]] \quad i' = i + 1 = k + 1 \quad j' = j = h$$

El análisis del cumplimiento de los invariantes es igual al realizado previamente para cuando la condición en la línea 5 es verdadera y se omite por espacio.

- \* La condición en la línea 14 es verdadera, osea que  $h < M$ . En este caso se ejecutarán las líneas 15-16:

$c.append(B[h])$   
 $j := j + 1$

y en consecuencia:

$$c = [c_0, c_1, \dots, c_{k+h-1}, B[h]] \quad i' = i = k \quad j' = j + 1 = h + 1$$

El análisis del cumplimiento de los invariantes es igual al realizado previamente para cuando la

condición en la línea 5 es falsa y se omite por espacio.

A partir de lo anterior se evidencia que en cualquier caso al finalizar la iteración los invariantes  $I_0$ ,  $I_1$  y  $I_2$  continuarán siendo verdaderos.

Finalmente, el ciclo terminará ya que el valor de  $i$  y  $j$  se incrementarán de 1 en 1 hasta eventualmente llegar a  $N$  y  $M$ . Luego por la estabilidad del invariante  $I_2$  se satisface:

$$C = [c_0, c_1, \dots, c_{i+j-1}]$$

$$C = [c_0, c_1, \dots, c_{N+M-1}]$$

y  $C$  contiene los primeros  $N$  elementos de  $A$  y los primeros  $M$  elementos de  $B$  ordenados ascendente mente o sea todos los elementos en ambos arreglos como es requerido en la postcondición. //

**Teorema:** La invocación `mezclar(A,B)` para cualquier arreglo  $A$  y cualquier arreglo  $B$  ordenados ascendente mente produce un arreglo con los valores en  $A$  y  $B$  ordenados ascendente mente.

**Demostación:** Es trivial a partir de la correctitud de los invariantes  $I_0$ ,  $I_1$  y  $I_2$ . //

---