David Suárez

Análisis De Algoritmos **Invariantes** 2016/02/02

1. Búsqueda Lineal

Este algoritmo busca un índice i que indica la posición donde se encuentra un valor dado v en un arreglo $S = \langle a_0, a_1, \cdots, a_{N-1} \rangle$, donde N es el tamaño del arreglo, comparando los elementos uno por uno y de izquierda a derecha hasta que la posición donde se encuentra v es encontrada o no hay más elementos que comparar en el arreglo.
Si el arreglo posee múltiples ocurrencias de v el algoritmo retornará la posición i de la primera ocurrencia de v.
Si v no esta presente en el arreglo el algoritmo retornará N el tamaño del arreglo.

1.1. Formalmente:

Entradas: Arreglo S con N números y el valor v a buscar:

$$S = \langle a_0, a_1, a_2, \cdots, a_{N-1} \rangle \; ; N > 0 \tag{1}$$

Salidas: Índice i tal que:

$$v = a_i \Leftrightarrow v \in S \lor i = N \Leftrightarrow v \notin S \tag{2}$$

1.2. Pseudocódigo:

Algorithm 1 Búsqueda Lineal.

```
function BusquedaLineal( S, v )
var encontro ← false
for i ← 0 to |S| − 1 and encontro == false do
    if S[i] == v then
        encontro ← true
    end if
    i ← i + 1
end for
return i
end function
```

1.3. Invariantes:

Se pueden encontrar dos invariantes en este algoritmo.

1.3.1. El valor de i nunca será mayor que $N = |S|$:

1. Formalmente: $0 \leq i \leq N$
2. Prueba por inducción:
   a) Inicialización ($i \leftarrow 0$):
   En la primera iteración del ciclo se cumple que: $0 \leq i \leq N$ debido a la inicialización de $i \leftarrow 0$.
   b) Mantenimiento/Iteración/Actualización/Procesamiento ($1 \leq i \leq N$):
   Debido a la condición del ciclo **for** en cada iteración del ciclo se cumple que $0 \leq i \leq N$. De no ser así se terminaría el ciclo y el mayor valor de i sería N pero nunca $N + 1$.
   c) Terminación ($i \leftarrow N$):
   Si $v \in S \rightarrow i < N$.
   Si $v \notin S \rightarrow i = N$.

1.3.2. Debido a que es una búsqueda secuencial, de izquierda a derecha, el valor v no se encuentra en el subarreglo $S' = \langle a_0, \cdots, a_{i-1} \rangle$:

"EL valor v no estará en los elementos que el algoritmo ya ha procesado."

1. Formalmente: $v \notin S'$
2. Prueba por inducción:
   a) Inicialización ($i \leftarrow 0$):
   Si $v = a_0$ el subarreglo S' no se "crearía" ya que debido a la condición de la sentencia **if** este ciclo terminaría sin ni siquiera hacer su primera iteración.
   Si $v \neq a_0$ i tomaría el valor de 1, es decir, $i \leftarrow 1$ y el subarreglo S' sería $S' = \langle a_0 \rangle$ ya que $a_{i-1} = a_0 \neq v$ y se cumple que $v \notin S' = \langle a_0, \cdots, a_{i-1} \rangle$.
   b) Mantenimiento/Iteración/Actualización/Procesamiento ($1 \leq i \leq N$):
   Debido a la condición de la sentencia **if** y sabiendo que esta búsqueda es secuencial en algún instante cuando i tome un valor p quiere decir que lo elementos anteriores a $S[p] \neq v$.
   c) Terminación ($i \leftarrow N$):
   Si $i < N \rightarrow v \in S$ y está en la posición $S[i]$ por tanto $v \notin S' = \langle a_0, \cdots, a_{i-1} \rangle$.
   Si $i = N \rightarrow v \notin S = \langle a_0, \cdots, a_{N-1} \rangle$ y como $i = N \rightarrow S = S' \rightarrow v \notin S' = \langle a_0, \cdots, a_{i-1} \rangle$.

2. Conteo de Números Repetidos

Este algoritmo genera un arreglo S' que contiene duplas (x, n) donde n es la cantidad de veces que se repite un número $x \in S = \langle a_0, a_1, \cdots, a_{N-1} \rangle$, donde N es el tamaño del arreglo.
Hace uso de dos ciclos; el primero para recorrer cada elemento de S y evitar procesar un número ya "contado", con el segundo ciclo se cuenta la cantidad de veces (n) que aparece un número x.

2.1. Formalmente:

Entradas: Arreglo S con N números:

$$S = \langle a_0, a_1, a_2, \cdots, a_{N-1} \rangle \; ; N > 0 \tag{3}$$

Salidas: Arreglo $S' = \langle (x, n)_0, (x, n)_1, \cdots, (x, n)_{N-1} \rangle$ con N duplas donde cada dupla:

$$(x, n) | n \equiv x_i \;\; (Frecuencia\ Absoluta\ de\ x). \tag{4}$$

2.2. Pseudocódigo:

2.3. Invariantes:

Se pueden encontrar cuatro invariantes en este algoritmo.

2.3.1. El valor de j nunca será menor que i:

1. Formalmente: $j \geq i$.
2. Prueba por inducción:
   a) Inicialización ($j \leftarrow i$):
   En la primera iteración del ciclo más interno se cumple que: $j \geq i$ debido a la inicialización de $j \leftarrow i$.
   b) Mantenimiento/Iteración/Actualización/Procesamiento ($i \leq j \leq N$):
   Como j inicia en i y durante cada iteración del ciclo más interno j solo aumenta su valor, es decir, $j \leftarrow j + 1$ esto asegura que j nuca disminuirá su valor para llegar a ser $< i$.

Algorithm 2 Contar Números Repetidos.

```
function ContarRepetidos( S )
{Arreglo con N duplas.}
var S' ⟨x, n⟩ [N]
var X, rep
for i ← 0 to |S| − 1 do
    if S[i] ∉ S' then
        X ← S[i]
        rep ← 0
        for j ← i to |S| − 1 do
            if S[j] == X then
                rep ← rep + 1
            end if
            j ← j + 1
        end for
        S'[X] ← rep
    end if
    i ← i + 1
end for
return S'
end function
```

c) Terminación ($j = N$):
Cuando termina el ciclo más interno el mayor valor que j podrá tomar es N (debido a condición de este ciclo **for**) y como $N \geq i \rightarrow j \geq i$.

2.3.2. A un número X no se le contarán dos veces sus repetidos, es decir, cada numero distinto de S se procesará una sola vez:

1. Formalmente: Si $X \in S' \rightarrow$ no se procesa de nuevo.
2. Prueba por inducción:
   a) Inicialización ($i \leftarrow 0$):
   En la primera iteración del ciclo más externo se procesará el elemento $S[0]$ y como no se ha procesado ningún otro entonces $S[0] \notin S'$.
   b) Mantenimiento/Iteración/Actualización/Procesamiento ($1 \leq i \leq N$):
   Debido a la condición de la primera sentencia **if** no se entrará al ciclo más interno si $S[i] \in S'$ asegurando que si el elemento $S[i]$ ya se proceso no se procese de nuevo.
   c) Terminación ($i = N$):
   Cuando termina el ciclo más externo el arreglo S' solo contendrá duplas $|x_0 \neq x_1 \neq x_2 \neq \cdots \neq x_{N-1}$.

2.3.3. El valor de i nunca será mayor que $N = |S|$:

1. Formalmente: $0 \leq i \leq N$
2. Prueba por inducción:
   a) Inicialización ($i \leftarrow 0$):
   En la primera iteración del ciclo más externo se cumple que: $0 \leq i \leq N$ debido a la inicialización de $i \leftarrow 0$.
   b) Mantenimiento/Iteración/Actualización/Procesamiento ($1 \leq i \leq N$):
   Debido a la condición del ciclo **for** más externo en cada iteración del ciclo se cumple que $0 \leq i \leq N$. De no ser así se terminaría el ciclo y el mayor valor de i sería N pero nunca $N + 1$.
   c) Terminación ($i \leftarrow N$):
   Cuando $i = N$ el ciclo más externo termina debido a su condición.
   También puede terminar cuando haya contado las repeticiones de todos los números en S en cuyo caso i seguirá siendo $\leq N$.

2.3.4. El valor de j nunca será mayor que $N = |S|$:

1. Formalmente: $0 \leq j \leq N$
2. Prueba por inducción:
   a) Inicialización ($j \leftarrow 0$):
   En la primera iteración del ciclo más interno se cumple que: $0 \leq j \leq N$ debido a la inicialización de $j \leftarrow 0$.
   b) Mantenimiento/Iteración/Actualización/Procesamiento ($1 \leq j \leq N$):
   Debido a la condición del ciclo **for** más interno en cada iteración del ciclo se cumple que $0 \leq j \leq N$. De no ser así se terminaría el ciclo y el mayor valor de j sería N pero nunca $N + 1$.
   c) Terminación ($j = N$):
   Cuando $j = N$ el ciclo más interno termina debido a su condición.
   También puede terminar cuando haya contado las repeticiones de todos los números en S en cuyo caso j seguirá siendo $\leq N$.