Talle 2.1 - Notación Binaria Inversa

Edwin Fabian Vesga Escobar[1]

[1]Departamento de Ingeniería de Sistemas, Pontificia Universidad Javeriana
Bogotá, Colombia
vesga.e@javeriana.edu.co

16 de agosto de 2022

Índice

1. Introducción

Los algoritmos "divide y vencerás"se basan en la idea de reducir un problema a subproblemas, siendo estos últimos instancias del problema universal planteado, pero con valores o muestras de evaluación menores. En este taller, plantearemos un problema (sección 2) el cual puede ser resuelto por medio de este método de resolución de problemas, y además veremos la definición formal de algoritmos que podría resolver el mismo (sección 3).

2. Definición del problema

El problema que se nos plantea es el siguiente:

"A partir de un número natural, calcular su representación binaria inversa"

Esto a grandes rasgos puede definir el problema así:

1. Un numero $l \in \mathbb{N}$ el cual será el que se evalúa para convertir en binario inverso.
2. Una secuencia S cuyo cardinal $|S| = n$ y con elementos $s_1, s_2, ..., s_n$ cuyos elementos $s_i \in \mathbb{N}$ y que además, es una representación binaria del número l.
3. Cualquier número $s_i = \begin{cases} 0 \\ 1 \end{cases}$

Como resultado, se espera obtener una secuencia inversa S^{-1}, donde sus elementos se ordenan así:
$s_n, s_{n-1}, ..., s_2, s_1$, siendo la representación binaria inversa.

3. Algoritmos de para solucionar el problema

A continuación, se definen dos algoritmos que permiten realizar la respectiva resolución del problema, uno de forma iterativa y otro, de usando la técnica de ”divide y vencerás”. Es por eso, que asumiremos que existe una función *toBinary*(x) que recibe como parámetro un valor $x \in \mathbb{N}$ y retorna una secuencia S con las características planteadas más arriba en la sección 2.

3.1. Algoritmo Iterativo

La idea de este algoritmo es realizar iterativamente la solución del problema.

Algoritmo 1 Algoritmo Iterativo - Binario Inverso.

Require: $S = s_1, s_2, ..., s_n$
Ensure: S será cambiado por $S^{-1} = s_n, s_{n-1}, ..., s_2, s_1$

procedure InvertBinaryIterative(l)
    $S \leftarrow toBinary(l)$
    $S^{-1} \leftarrow \{\emptyset\}$
    for $i \leftarrow |S|$ **to** 1 **do**
        Append s_i **to** S^{-1}
    end for
end procedure

3.1.1. Análisis de complejidad

Por inspección de código: hay un ciclo *para-todo* que, en el peor de los casos, recorren todo la secuencia de datos; entonces, este algoritmo es $O(|S|)$.

3.1.2. Invariante

Después de cada iteración controlada por el contador i, los elementos de S quedan en S^{-1} ordenados así: $S^{-1}_{n-i} = S_i$.

3.2. Algoritmo Divide y vencerás

La idea de este algoritmo es realizar con el método de divide y vencerás, la solución del problema.

Algoritmo 2 Algoritmo Divide y vencerás - Binario Inverso.

Require: $S = s_1, s_2, ..., s_n$
Ensure: S será cambiado por $S^{-1} = s_n, s_{n-1}, ..., s_2, s_1$

procedure InvertBinaryDivide(l)
    $S \leftarrow toBinary(l)$
    $S^{-1} \leftarrow \{\emptyset\}$
    if $|S| = 1$ **then**
        Append S **to** S^{-1}
    end if
    if $|S| > 1$ **then**
        Append $InvertSubBinary(S, 1, |S|)$ **to** S^{-1}
    end if
    Return S^{-1}
end procedure

Algoritmo 3 Algoritmo Invertir Subconjunto Binario - Binario Inverso.

Require: $S = s_1, s_2, ..., s_n$ b el cual es la posición inicial y un e que es la posición final.

Ensure: *invert* que es un conjunto de números con el valor inverso de una secuencia binaria S

procedure InvertSubBinary(S, b, e)
  invert $\leftarrow \{\emptyset\}$
  if b=e **then**
    Append S_b **to** *invert*
    Return *invert*
  end if
  $q \leftarrow (b + e)/2$
  Append $InvertSubBinary(S, q, e)$ **to** *invert*
  Append $InvertSubBinary(S, b, q - 1)$ **to** *invert*
  Return *invert*
end procedure