Escritura del un algoritmo de dividir-y-vencer

Juan Jose Bolaños Melo[1]

[1]Departamento de Ingeniería de Sistemas, Pontificia Universidad Javeriana
Bogotá, Colombia
{bolanos.jj}@javeriana.edu.co

18 de agosto de 2022

Resumen

En este documento se presenta la formalización del problema de conversión de un numero decimal a un numero binario en su forma invertida, junto con la descripción de dos algoritmos que lo solucionan. Además, se presenta un análisis experimental de la complejidad de esos dos algoritmos. **Palabras clave:** conversión, algoritmo, formalización, experimentación, complejidad, binario.

Índice

1. Introducción

Los algoritmos de conversión de números decimales a binario y viceversa han resultado demasiado útiles a los largo de los años, sin embargo en la mayoría de los casos solo conocemos la solución estándar que se realiza por el método iterativo, es por esto que en el presente documento se realizo la comparación de esta solución estándar frente a la solución realiza por el algoritmo de dividir y vencer, con el objetivo de mostrar: la formalización del problema (sección 2), la escritura formal de los dos algoritmos (sección 3) y un análisis experimental de la complejidad de cada uno de ellos (sección **??**).

2. Formalización del problema

Cuando se piensa en el algoritmo de*Dividir y vencer* se hace referencia a que la solución del problema es dividir este en numero de subproblemas que son instancias mas pequeñas del mismo problema. Sin embargo, para que el algoritmo siga con el paradigma de dividir y vencer debe cumplir con los siguientes pasos:

1. Dividir: Se debe descomponer el problema en sub-problemas, donde cada sub-problema representa una parte del problema original.

2. Vencer: En este paso procedemos a resolver cada uno de los sub-problemas de forma recursiva.

3. Combinar: Por ultimo se procede a combinar cada una de las respuestas apropiadamente.

Recordemos que este método no es la verdad absoluta, ni siempre va ser la mejor solución, sin embargo en algunos caso esta demostrado que este paradigma nos permite reducciones bastantes significativa en la complejidad tiempo del algoritmo .

2.1. Definición del problema del "ordenamiento de datos"

Así, el problema de conversión de un numero decimal a binario invertido se define a partir de:

1. Sea X perteneciente al subconjunto de numeros $a \in \mathbb{N}$

producir un nuevo numero $Xfinal'$ cuya representación decimal sea la de $xInicial'$ binaria invertida.

- Entradas:
  - $XInicial$ perteneciente al subconjunto de números $a \in \mathbb{N}$
- Salidas:
  - $X'Final$ perteneciente al subconjunto de números $a \in \mathbb{N}$.

3. Algoritmos de solución

3.1. Método "Iterativo"

La idea de este algoritmo es: hacer uso iterativamente del bucle While, a la vez que vamos disminuyendo el valor del entero a la mitad en cada interacción, obteniendo así los residuos que serán evaluados por una condición con el fin de obtener el valor binario final <.

Algoritmo 1 Algoritmo "Iterativo".

Require: $XInicial$ perteneciente al subconjunto de números $a \in \mathbb{N}$
Ensure: $XInicial$ será cambiado por $X'Final$ perteneciente al subconjunto de números $a \in \mathbb{N}$.

```
1: procedure pasaraBinario(num)
2:     String : numeroBinario
3:     while num != 0 do
4:         numeroBinario+ = num(modulo)2 == 0?("0" : "1")
5:         num/ = 2
6:     end while
7:     return numeroBinario
8: end procedure
```

3.1.1. Análisis de complejidad

Por inspección de código: el algoritmo cuenta con solamente un ciclo que, en todos los casos, va ir reduciendo iterativamente el entero recibido a la mitad hasta que este sea 0; entonces, este algoritmo es $O(|n|)$.

3.2. Algoritmo por "Dividir-y-Vencer"

La idea de este algoritmo se divide en tres pasos fundamentales:

1. Dividir el numero binario en dos o mas (subproblemas) donde cada división queda mas pequeña el numero binario original, hasta que llegue al caso base que es numeroInicial = numeroFinal (Dividir)

2. Resolver cada sub-división de forma recursiva (vencer)

3. Por ultimo se procede a combinar cada unos de los resultados mediante un vector el cual ira guardando los bits hasta llegar a la forma inversa del numero binario (Combinar) .

El pseudocódigo mostrado a continuación cumple estrictamente los criterios del paradigma de dividir y vencer, invirtiendo el numero binario y mostrando la representación decimal de este.

Algoritmo 2 Algoritmo por "Dividir-y-Vencer".

Require: $XInicial$ perteneciente al subconjunto de números $a \in \mathbb{N}$
Ensure: $XInicial$ será cambiado por $X'Final$ perteneciente al subconjunto de números $a \in \mathbb{N}$.

```
1: procedure pasaraBinarioDividirVencer(inicio,final)
2:     if inicio == fin then
3:         binarioInvertido[poss] == binario[inicio]
4:         poss + 1
5:         return  mitad = (inicio + fin)/2;
6:     end if
7:     mitad ← (inicio + fin)/2;
8:     pasaraBinarioDividirVencer(mitad + 1, fin)
9:     pasaraBinarioDividirVencer(inicio, mitad)
10: end procedure
```

3.2.1. Análisis de complejidad

Por el Teorema Maestro : Nos podemos encontrar que la recurrencia usada en la funcion principal cumple con el caso 2 del teorema por lo tanto tenemos que:

1. $T(n) = O(n.logna2.logn)$
2. $T(n) = O(nlogn)$