Zeros and Ones

Johanna Beltran y Diego Triviño

2012

Índice

1. Introducción

'**Zeros and Ones**' es un problema de programación el cual encontramos en el juez virtual UVA con el número **10324**.
Este documento busca mostrar una de las tantas soluciones desde el enfoque matemático teniendo en cuenta que el objetivo es realizar la implementación de la solución del problema en cualquier lenguaje de programación con la ayuda de este documento.
Este problema puede ser resuelto utilizando la estrategia de 'divide y vencerás'.
Esta estrategia es una técnica de diseño de algoritmos la cual consiste en dividir de forma recurrente un problema en subproblemas más sencillos hasta que se encuentre un caso base.

Esta técnica consta fundamentalmente de los siguientes pasos:

1. Descomponer el problema a resolver en un cierto número de subproblemas más pequeños.
2. Resolver independientemente cada subproblema.
3. Combinar los resultados obtenidos para construir la solución del problema original.

2. Definición del problema

Dada una cadena de **0's** y **1's** y dos índices i y j se debe decir si todos los caracteres entre la posición i y la posición j son los mismos.

Se deben tener en cuenta las siguientes restricciones para la solución del problema:

1. Para cada caso de prueba se debe ingresar una cadena binaria b, un número entero n y n secuencias compuestas por dos números enteros i y j.
2. Para b se debe tener en cuenta que $1 \leq b \leq 1000000$ por lo tanto para los índices i y j se debe tener en cuenta $0 \leq i < 1000000$, $0 \leq j < 1000000$.

2.1. Entrada

Hay varios casos en la entrada. La primera línea de cada caso es una cadena de 0's y de 1's. La siguiente línea contiene un entero positivo n que indica el número de consultas para este caso. Las siguientes n líneas contienen los índices a consultar, una por línea. Cada consulta está dada por dos números enteros no negativos, i y j . Para cada consulta, usted debe imprimir **Yes** si todos los caracteres en la cadena entre la posición i y la posición j son los mismos, y **No** lo contrario.

EJEMPLO

0000011111

3

0 5

4 2

5 9

01010101010101010101010101111111111111111111111111111111111110000000000000000

5

4 4

25 60

1 3

62 76

24 62

1

1

0 0

2.2. Salida

Cada caso de la producción debería comenzar con un encabezado como en el ejemplo siguiente. La entrada termina con una cadena vacía que es una línea que contiene sólo el carácter de nueva línea, esta cadena no debe ser procesada.

EJEMPLO ANTERIOR

$Caso$ 1:

No

Yes

Yes

$Caso$ 2:

Yes

Yes

No

Yes

No

$Caso$ 3:

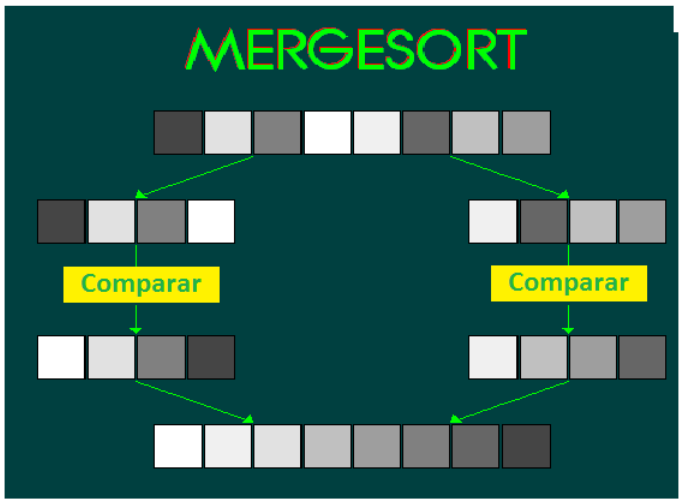
Yes

3. Modelamiento matemático

Dada una cadena binaria b y dos índices de consulta i y j, se toma la subcadena comparando así cada carácter que la conforma.

4. Planteamiento de la solución

Para este problema nos podemos basar en algunas características del algoritmo de ordenamiento **Mergesort** que permite mejorar el tiempo de ejecución puesto que se necesitan menos pasos para cumplir el objetivo principal; el algoritmo trabaja de la siguiente forma:



[1]

1. Si la longitud de la cadena es 0 ó 1, entonces la respuesta al problema es 'Yes'.
2. Dividir la cadena en dos subcadenas de aproximadamente la mitad del tamaño.
3. Comparar cada carácter de cada subcadena formada para asi tener la respuesta a cada caso de prueba.

Con este algoritmo se podrá dividir la subcadena hasta obtener un caso base y comparar fácilmente los caracteres que la conforman.

[1] http://throwingcodes.blogspot.com/

5. Conclusiones

1. Este algoritmo de ordenación es un ejemplo claro de que el método divide y vencerás es efectivo cuando tienes cantidades grandes de datos por trabajar y necesitas ahorrar tiempo y recursos.

2. Este algoritmo de ordenamiento tiene una complejidad de $(n \log n)$ que permite mejorar el tiempo de ejecución de un problema con listas grandes.