Ejercicios en clase: División y conquista 2

Análisis y Diseño de Algoritmos

2 de octubre de 2021

Ejercicio 1. Corra el algoritmo para maximo subarreglo en el siguiente arreglo: ⟨2,-2,3,5,-3,0,3,-8,9⟩

Ejercicio 2. Corra el algoritmo de Karatsuba en los siguientes números: 16541533 y 40040534

Ejercicio 3. Corra el algoritmo de Strassen en las siguientes matrices:

$$A = \begin{pmatrix} 1 & 3 \\ 7 & 5 \end{pmatrix},$$

$$B = \begin{pmatrix} 2 & 4 \\ 8 & 6 \end{pmatrix}.$$

Ejercicio 4. Considere el siguiente problema.

Entrada: un arreglo $A[1..n]$ de números enteros.

Salida: El número de inversiones significativas, donde una *inversión significativa* es un par ordenado (i, j) tal que $i < j$ y $A[i] > 2A[j]$.

Diseñe un algoritmo $\Theta(n \lg n)$ para el problema anterior. Escriba la recurrencia para el tiempo de ejecución del algoritmo en el peor caso y resuélvala usando el teorema maestro.

Ejercicio 5. Considere el siguiente problema.

Entrada: Dos vectores $A[1..n]$, $B[1..n]$ con elementos distintos dos a dos, ordenados de manera creciente.

Salida: La mediana del conjunto de elementos que están en A o en B.

Es decir, el elemento v tal que existen exactamente $n - 1$ elementos menores en A o B.

Por ejemplo, si $A = [10, 30, 50, 70], B = [20, 40, 60, 80]$, la mediana correspondiente es 40, ya que $n = 4$ y existen 3 elementos menores que 40. Diseñe un algoritmo de división y conquista con complejidad $\Theta(\lg n)$ para el problema de la mediana. En este ejercicio puede considerar que n es potencia de 2. Escriba la recurrencia para el tiempo de ejecución del algoritmo en el peor caso y resuélvala usando el teorema maestro.

Ejercicio 6. Decimos que un vector $A[1..n]$ es *unimodal* si existe un índice p, llamado *pico* tal que $A[1..p]$ es una secuencia creciente, y $A[p + 1..n]$ es una secuencia decreciente. Diseñe un algoritmo de división y conquista que recibe un vector unimodal y encuentra el pico de A. Su algoritmo debe tener complejidad $\Theta(\lg n)$ en el peor caso. Escriba el pseudocódigo del algoritmo. Escriba la recurrencia para el tiempo de ejecución del algoritmo en el peor caso y resuélvala usando el teorema maestro.

Ejercicio 7. Considere el siguiente problema de búsqueda.

Entrada: un arreglo $A[1..n]$ de números enteros.

Salida: El valor $A[i]$ tal que existen más de $n/2$ números iguales que $A[i]$. Si no existe dicho valor, devolver -1. Por ejemplo, si $A = [2, 4, 2, 4, 2, 2, 1, 4, 2]$, el algoritmo debe devolver el valor 2.

Diseñe un algoritmo de división y conquista con complejidad $\Theta(n \lg n)$ para el problema anterior. Explique la idea de su algoritmo con un ejemplo. Escriba el pseudocódigo del algoritmo. Escriba la recurrencia para el tiempo de ejecución del algoritmo en el peor caso y resuélvala usando el teorema maestro. Obs: no puede usar ninguna rutina previa, por ejemplo de ordenación.

Ejercicio 8. Dado un arreglo $A[1..n]$, una k-rotación de A es un arreglo $B[1..n]$ tal que

$$B(k) = \begin{cases} A[i+k] & i+k \leq n \\ A[(i+k) \mod n] & \text{caso contrário} \end{cases}$$

Por ejemplo, si $A = [3, 6, 9, 10]$, una 2-rotación de A es $B = [9, 10, 3, 6]$.

Considere el siguiente problema. Entrada: Una k-rotación B de un arreglo ordenado de elementos diferentes. Salida: El número k. Por ejemplo, si $B = [9, 10, 3, 6]$, el algoritmo debe devolver el valor 2.

Diseñe un algoritmo $\Theta(\lg n)$ para el peor caso del problema. Escriba el pseudocódigo del algoritmo. Escriba una recurrencia para el peor caso de este algoritmo. Verifique con teorema maestro.

Ejercicio 9. Considere el siguiente problema.

Entrada: Un conjunto de k arreglos ordenados $A_1, A_2, \ldots A_k$, cada uno de ellos ordenado de manera creciente, que en conjunto tienen tamaño n.

Salida: Un arreglo $B[1..n]$ con todos los elementos de la entrada ordenados Por ejemplo, si $A_1 = [1, 3]$, $A_2 = [2, 7, 8]$, $A_3 = [3, 7]$, el algoritmo debe devolver $B = [1, 2, 3, 3, 7, 7, 8]$.

Diseñe un algoritmo de división y conquista que consuma tiempo $\Theta(n \lg k)$ en el peor caso. Escriba el pseudocódigo del algoritmo anterior. Escriba una recurrencia para el peor caso de este algoritmo. Verifique con teorema maestro.