Algoritmos de División y conquista (Divide and Conquer)

En un algoritmo de división y conquista hay dos etapas:

(1) El input se divide en subproblemas que se resuelven independientemente. (típicamente mediante una llamada recursiva) (División)

(2) Las soluciones de estos problemas individuales se recombinan (Conquista)

Ejemplo: El ejemplo más sencillo es merge Sort. que tiene complejidad $O(n\log(n))$

Empezaremos con un ejemplo más sustancial, el problema de contar el Número de inversiones

Def: Sea A una lista de enteros distintos. Una INVERSIÓN de A es una pareja de posiciones (i,j) con

$$i<j \quad y \quad A[i] > A[j]$$

Ejemplo:

$$\begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \\ 5 & 4 & 3 & 1 & 2 \end{array} \leftarrow \text{índice}$$

Inversiones: 1 2, 1 3, 1 4, 1 5,
2 3, 2 4, 2 5
3 4, 3 5
4 5

9 inversiones

NO inversión.

$$0 \leq \#\text{inversiones} \leq \binom{n}{2}$$

Obs: Hay un algoritmo directo, de exploración por fuerza bruta que toma $O(n^2)$ pasos.

Idea:

1 2 3 4 5 6 ← indices

6 5 4 3 1 2

Partimos la sucesión en dos subproblemas:

654 3 1 2

3 — 65, 64, 54 31, 32 — 2

Pero los problemas no son "independientes", falta considerar inversiones que tienen un lado en la primera mitad y el otro en la segunda

63, 61, 62

53, 51, 52

43, 41, 42

puede haber muchas de estas, del orden de $O\left(\left(\frac{n}{2}\right)^2\right) = O(n^2)$

escogiendo $\left(n, n-1, \ldots, \frac{n}{2}, 1\ 2, \ldots, \frac{n}{2}\right)$ por ejemplo.

Necesitamos una idea para poder contarlas correctamente...

Sabemos que ordenar un arreglo puede hacerse muy rápido así que podemos ordenar los subproblemas para ayudarnos en ese conteo. (no importa el orden interno sino solo como se comparan entre sí)

456 123

merge

1 2 3 4 5 6

3 3 3

Otro ejemplo

421 563

124 356

1 2 3 4 5 6

0 0 1

\# inversiones que cruzan = \# rojos a la izquierda

Si mantenemos un contador que nos diga cuántos rojos hemos visto en cada instante podemos contar inversiones cruzadas muy rápido, aunque hayan muchas

$8\ 7\ 6\ 5$ — 6 inv $\qquad$ $4\ 3\ 2\ 1$ — 6 inv

$\downarrow$ ordena $\qquad$ $\downarrow$ ordena

$5\ 6\ 7\ 8$ $\qquad$ $1\ 2\ 3\ 4$

$1\ 2\ 3\ 4\ 5\ 6\ 7\ 8$

$4\ 4\ 4\ 4$ — 16 inversiones cruzadas

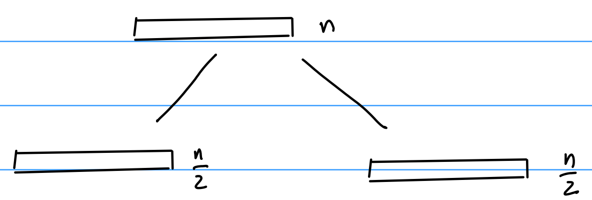
$$\text{total} = 6 + 6 + 16 = 28 = \binom{8}{2}$$

Lema: Suponga que A y R son listas ordenadas. El número de inversiones cruzadas $a > r$ iniciando en a para $a \in A$, $r \in R$ es igual al número de elementos de R a la izquierda de a en la lista $A \cup R$ ordenada de manera creciente.

Dem: Si $A \cup R$ está ordenada de manera creciente y $a \in A$ entonces todo elemento r^* de R a la izquierda (menor) que a determina una inversión cruzada porque $a > r^*$.

Luego un contador es suficiente para calcular las inversiones cruzadas en tiempo $O(n)$. pues basta recorrer toda la lista una vez.

Cuánto tiempo se requiere en total?



(1) Resuelvo probl recursivamente

(2) Ordeno los outputs $2O\left(\frac{n}{2}\log(n)\right) = O\left(n \log(n)\right)$

(3) Recorro lista contando las inversiones $O(n)$

luego $T(n) \leq 2T(\frac{n}{2}) + O(n) + O(n\log(n))$

Qué podemos decir en gral ?

← más tarde ... (MASTER)

$T(n) = O(n\log(n))$

Ejemplo: Multiplicación de matrices

$$\underbrace{\begin{pmatrix} A & B \\ C & D \end{pmatrix}}_{X} \cdot \underbrace{\begin{pmatrix} E & F \\ G & H \end{pmatrix}}_{Y} = \begin{pmatrix} A\cdot E + B\cdot G & A\cdot F + B\cdot H \\ C\cdot E + D\cdot G & C\cdot F + D\cdot H \end{pmatrix}$$

$$T(n) = \underbrace{8T\left(\frac{n}{2}\right)}_{\text{calcula los productos recursivamente}} + \underbrace{4\,O\left(\left(\frac{n}{2}\right)^2\right)}_{\text{realizar las sumas}}$$

⋮ más tarde (MASTER)

$O(n^3)$ (igual que el método ingenuo original :( )

No obstante Strassen inventó una manera de expresar el producto mediante SÓLO 7 llamadas recursivas. Es difícil de creer, pero cierto:

$$X \cdot Y = \begin{pmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{pmatrix}$$

con
$$\left.\begin{aligned}
P_1 &= A\cdot(F - H) \\
P_2 &= (A + B)\cdot H \\
P_3 &= (C + D)\cdot E \\
P_4 &= D\cdot(G - E) \\
P_5 &= (A + D)\cdot(E + H) \\
P_6 &= (B - D)\cdot(G + H) \\
P_7 &= (A - C)\cdot(E + F)
\end{aligned}\right\}$$

Hay solo 7 y no 8 multiplicaciones

Concluimos que la complejidad en el tiempo de Strassen es así:

(1) Calculamos las sumas (y restas) de bloques necesarias

$\leq 10\ O(n^2)$

(2) Calculamos los productos

$7\,T\left(\frac{n}{2}\right)$

(3) Calculamos las sumas de los resultados obtenidos

$8\ O(n^2)$

Concluimos

$$T(n) \leq 7\,T\left(\frac{n}{2}\right) + O(n^2)$$

$\vdots$ (MASTER METHOD)

$$T(n) \leq ?$$