

Tarea 1: Implementación de algoritmos Divide y Vencerás

Delgado Romero, Gustavo ¹ Torres Reategui, Joaquin²

7 de septiembre de 2025

Problema 1: Moda de un vector

Cálculo de la moda con Divide y Vencerás (Parte I)

```
1: function MODA_DIVIDE_VENCERAS(arr)
2:   if longitud(arr) = 1 then
3:     return { arr[0] : 1 }
4:   end if
5:   mid  $\leftarrow$  longitud(arr) / 2
6:   freq_izq  $\leftarrow$  Moda_Divide_Vencerás(arr[0:mid])
7:   freq_der  $\leftarrow$  Moda_Divide_Vencerás(arr[mid:])
8:   return Merge_Freq(freq_izq, freq_der)
9: end function
```

Problema 1: Moda de un vector

Cálculo de la moda con Divide y Vencerás (Parte II)

```
1: function MERGE_FREQ(freq_izq, freq_der)
2:   combinado  $\leftarrow$  copia(freq_izq)
3:   for all (k, v) en freq_der do
4:     combinado[k]  $\leftarrow$  combinado.get(k,0) + v
5:   end for
6:   return combinado
7: end function
```

Consideramos la recurrencia:

$$T(n) = \begin{cases} 2, & n = 1, \\ 2T\left(\frac{n}{2}\right) + cn, & n > 1, \end{cases}$$

donde $c > 0$ es una constante.

Expansión de la recurrencia

Aplicamos el método iterativo (expansión):

$$T(n) = 2T\left(\frac{n}{2}\right) + cn$$

$$T(n) = 2\left(2T\left(\frac{n}{2^2}\right) + c\frac{n}{2}\right) + cn = 2^2T\left(\frac{n}{2^2}\right) + 2cn$$

$$T(n) = 2^iT\left(\frac{n}{2^i}\right) + i cn$$

Conclusión

Cuando $\frac{n}{2^k} = 1 \implies k = \log_2 n$:

$$T(n) = 2^{\log_2 n} T(1) + cn \log_2 n$$

$$T(n) = n \cdot 2 + cn \log_2 n$$

Conclusión:

$$T(n) = \Theta(n \log n).$$

Problema 2: Multiplicación de enteros grandes

Multiplicación de enteros grandes por divide y vencerás

```
1: function MULTIPLICACION( $x, y$ )
2:   if  $x < 10$  or  $y < 10$  then
3:     return  $x \times y$ 
4:   end if
5:    $n \leftarrow \text{máx}(\text{len}(x), \text{len}(y))$ 
6:    $m \leftarrow \lfloor n/2 \rfloor$ 
7:   Dividir  $x$  en  $x_1, x_0$  tal que  $x = x_1 \cdot 10^m + x_0$ 
8:   Dividir  $y$  en  $y_1, y_0$  tal que  $y = y_1 \cdot 10^m + y_0$ 
9:    $z_2 \leftarrow \text{Multiplicacion}(x_1, y_1)$ 
10:   $z_0 \leftarrow \text{Multiplicacion}(x_0, y_0)$ 
11:   $z_1 \leftarrow \text{Multiplicacion}(x_1 + x_0, y_1 + y_0) - z_2 - z_0$ 
12:  return  $z_2 \cdot 10^{2m} + z_1 \cdot 10^m + z_0$ 
13: end function
```

▷ Caso base: 1 dígito

La recurrencia asociada al algoritmo es:

$$T(n) = \begin{cases} 4, & n = 1, \\ 3 T\left(\frac{n}{2}\right) + c n, & n > 1, \end{cases}$$

donde $c > 0$ es una constante que representa el coste lineal de las operaciones de suma, resta y combinación.

Expansión de la recurrencia

Desarrollamos por expansión iterativa:

$$T(n) = 3T\left(\frac{n}{2}\right) + cn$$

$$T(n) = 3\left(3T\left(\frac{n}{2^2}\right) + c\frac{n}{2}\right) + cn = 3^2T\left(\frac{n}{2^2}\right) + cn\left(1 + \frac{3}{2}\right).$$

Después de i expansiones:

$$T(n) = 3^i T\left(\frac{n}{2^i}\right) + cn \sum_{j=0}^{i-1} \left(\frac{3}{2}\right)^j.$$

Evaluación de la suma

La suma geométrica es:

$$\sum_{j=0}^{i-1} \left(\frac{3}{2}\right)^j = \frac{\left(\frac{3}{2}\right)^i - 1}{\frac{3}{2} - 1} = 2\left(\left(\frac{3}{2}\right)^i - 1\right).$$

Por tanto:

$$T(n) = 3^i T\left(\frac{n}{2^i}\right) + 2cn\left(\left(\frac{3}{2}\right)^i - 1\right).$$

Sustituyendo en la hoja

Tomamos $i = k$ tal que $\frac{n}{2^k} = 1 \implies k = \log_2 n$.

$$3^k = 3^{\log_2 n} = n^{\log_2 3}, \quad \left(\frac{3}{2}\right)^k = n^{\log_2(3/2)} = n^{\log_2 3 - 1}.$$

Con $T(1) = 4$:

$$\begin{aligned} T(n) &= 3^k T(1) + 2cn \left(n^{\log_2 3 - 1} - 1 \right) \\ &= 4 n^{\log_2 3} + 2c n^{\log_2 3} - 2c n. \end{aligned}$$

Conclusión

Dado que $\log_2 3 \approx 1,585$, el término dominante es $n^{\log_2 3}$.

$$T(n) = \Theta(n^{\log_2 3}) \approx \Theta(n^{1,585})$$

Problema 3: Multiplicación de matrices (Parte I)

Multiplicación de matrices por divide y vencerás

```
1: function MULT_DIVIDE_CONQUER(A, B)
2:    $n \leftarrow$  tamaño de  $A$ 
3:   if  $n = 1$  then
4:     return  $A_{11} \cdot B_{11}$ 
5:   end if
6:    $(A_{11}, A_{12}, A_{21}, A_{22}) \leftarrow \text{Split\_Matrix}(A)$ 
7:    $(B_{11}, B_{12}, B_{21}, B_{22}) \leftarrow \text{Split\_Matrix}(B)$ 
```

Problema 3: Multiplicación de matrices (Parte II)

Multiplicación de matrices por divide y vencerás (cont.)

8:	C_{11}	\leftarrow	$\text{Mult_Divide_Conquer}(A_{11}, B_{11})$	+
	$\text{Mult_Divide_Conquer}(A_{12}, B_{21})$			
9:	C_{12}	\leftarrow	$\text{Mult_Divide_Conquer}(A_{11}, B_{12})$	+
	$\text{Mult_Divide_Conquer}(A_{12}, B_{22})$			

Problema 3: Multiplicación de matrices (Parte III)

Multiplicación de matrices por divide y vencerás (cont.)

```
10:    $C_{21} \leftarrow \text{Mult\_Divide\_Conquer}(A_{21}, B_{11})$    +  
       $\text{Mult\_Divide\_Conquer}(A_{22}, B_{21})$   
11:    $C_{22} \leftarrow \text{Mult\_Divide\_Conquer}(A_{21}, B_{12})$    +  
       $\text{Mult\_Divide\_Conquer}(A_{22}, B_{22})$   
12:   Combinar  $C_{11}, C_{12}, C_{21}, C_{22}$  en  $C$  de tamaño  $n \times n$   
13:   return  $C$   
14: end function
```

Problema 3: Multiplicación de matrices

Multiplicación de matrices por divide y vencerás(Parte II)

```
1: function ADD_MATRIX( $A$ ,  $B$ )
2:    $n \leftarrow$  tamaño de  $A$ 
3:   for  $i \leftarrow 1$  hasta  $n$  do
4:     for  $j \leftarrow 1$  hasta  $n$  do
5:        $C[i][j] \leftarrow A[i][j] + B[i][j]$ 
6:     end for
7:   end for
8:   return  $C$ 
9: end function
```

Problema 3: Multiplicación de matrices

Multiplicación de matrices por divide y vencerás(Parte II)

```
1: function SUB_MATRIX( $A$ ,  $B$ )
2:    $n \leftarrow$  tamaño de  $A$ 
3:   for  $i \leftarrow 1$  hasta  $n$  do
4:     for  $j \leftarrow 1$  hasta  $n$  do
5:        $C[i][j] \leftarrow A[i][j] - B[i][j]$ 
6:     end for
7:   end for
8:   return  $C$ 
9: end function
```

Problema 4: Multiplicacion de matrices

Dividir y combinar matrices

```
1: function SPLIT_MATRIX(A)
2:    $n \leftarrow$  tamaño de  $A$ 
3:    $m \leftarrow n/2$ 
4:    $A_{11} \leftarrow$  submatriz superior izquierda
5:    $A_{12} \leftarrow$  submatriz superior derecha
6:    $A_{21} \leftarrow$  submatriz inferior izquierda
7:    $A_{22} \leftarrow$  submatriz inferior derecha
8:   return ( $A_{11}, A_{12}, A_{21}, A_{22}$ )
9: end function
```

Problema 4: Multiplicacion de matrices

Dividir y combinar matrices

```
1: function COMBINE_MATRIX( $C_{11}$ ,  $C_{12}$ ,  $C_{21}$ ,  $C_{22}$ )  
2:   Formar matriz  $C$  uniendo los cuatro cuadrantes  
3:   return  $C$   
4: end function
```

Recurrencia de la multiplicación de matrices

La recurrencia asociada al algoritmo es:

$$T(n) = \begin{cases} 3, & n = 1, \\ 8T\left(\frac{n}{2}\right) + cn^2, & n > 1, \end{cases}$$

donde $c > 0$ es una constante que representa el coste de sumar y combinar matrices.

Expansión iterativa

Expandimos la recurrencia paso a paso:

$$T(n) = 8T\left(\frac{n}{2}\right) + cn^2$$

$$T(n) = 8\left(8T\left(\frac{n}{2^2}\right) + c\left(\frac{n}{2}\right)^2\right) + cn^2 = 8^2T\left(\frac{n}{2^2}\right) + 8c\frac{n^2}{4} + cn^2$$

$$T(n) = 8^2T\left(\frac{n}{2^2}\right) + 3cn^2$$

En general, tras i expansiones:

$$T(n) = 8^iT\left(\frac{n}{2^i}\right) + cn^2 \sum_{j=0}^{i-1} 2^j$$

porque cada nivel de recursión agrega un coste proporcional a $n^2 \cdot 2^j$.

Evaluación en la hoja

Tomamos $i = k$ tal que $\frac{n}{2^k} = 1 \implies k = \log_2 n$.

$$8^k = 8^{\log_2 n} = n^{\log_2 8} = n^3$$

La suma geométrica:

$$\sum_{j=0}^{k-1} 2^j = \frac{2^k - 1}{2 - 1} = \frac{n^{\log_2 2} - 1}{1} = n - 1$$

Por tanto:

$$T(n) = n^3 T(1) + cn^2 \cdot (n - 1)$$

Conclusión

Dado que $T(1) = 3$:

$$T(n) = 3n^3 + c(n^3 - n^2)$$

El término dominante es n^3 , proveniente de $n^{\log_2 8}$.

$$T(n) = \Theta(n^3)$$

Problema 4: Maxima subsecuencia (Parte I)

Función MAX_SUM

```
1: function MAX_SUM(A, izquierda, medio, derecha)
2:   suma_izq  $\leftarrow -\infty$ 
3:   total  $\leftarrow 0$ 
4:   for  $i \leftarrow$  medio hasta izquierda do
5:     total  $\leftarrow$  total + A[i]
6:     suma_izq  $\leftarrow$   $\text{máx}(\text{suma\_izq}, \text{total})$ 
7:   end for
8:   suma_der  $\leftarrow -\infty$ 
9:   total  $\leftarrow 0$ 
10:  for  $i \leftarrow$  medio+1 hasta derecha do
11:    total  $\leftarrow$  total + A[i]
12:    suma_der  $\leftarrow$   $\text{máx}(\text{suma\_der}, \text{total})$ 
13:  end for
14:  return suma_izq + suma_der
15: end function
```


Problema 4: Maxima subsecuencia (Parte II)

Función MAX_SUBARRAY

```
1: function MAX_SUBARRAY(A, izquierda, derecha)
2:   if izquierda = derecha then
3:     return A[izquierda]
4:   end if
5:   medio  $\leftarrow \lfloor \frac{\text{izquierda} + \text{derecha}}{2} \rfloor$ 
6:   max_izq  $\leftarrow$  Max_Subarray(A, izquierda, medio)
7:   max_der  $\leftarrow$  Max_Subarray(A, medio+1, derecha)
8:   max_cruz  $\leftarrow$  Max_Sum(A, izquierda, medio, derecha)
9:   return máx(max_izq, max_der, max_cruz)
10: end function
```

Consideramos la recurrencia

$$T(n) = \begin{cases} 2, & n = 1, \\ 2 T\left(\frac{n}{2}\right) + c n, & n > 1, \end{cases}$$

donde $c > 0$ es una constante que representa el coste lineal de combinar resultados y calcular la suma cruzada.

Expansión iterativa

Desarrollamos la recurrencia por sustitución:

$$\begin{aligned}T(n) &= 2T\left(\frac{n}{2}\right) + cn \\&= 2\left(2T\left(\frac{n}{2^2}\right) + c\frac{n}{2}\right) + cn = 2^2T\left(\frac{n}{2^2}\right) + 2 \cdot c\frac{n}{2} + cn \\&= 2^2T\left(\frac{n}{2^2}\right) + cn + cn = 2^2T\left(\frac{n}{2^2}\right) + 2cn.\end{aligned}$$

Tras i expansiones obtenemos la forma general

$$T(n) = 2^i T\left(\frac{n}{2^i}\right) + i \cdot cn.$$

(Observación: en cada nivel de la recursión aparece un término cn , por eso la suma es $i \cdot cn$.)

Evaluación en la hoja y conclusión

Tomamos $i = k$ tal que $\frac{n}{2^k} = 1$, es decir $k = \log_2 n$. Sustituyendo:

$$2^k = 2^{\log_2 n} = n, \quad T(1) = 2.$$

Entonces

$$T(n) = 2^k T(1) + k \, cn = n \cdot 2 + cn \log_2 n.$$

Como término dominante queda $cn \log_2 n$, por lo que

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