Escuela Profesional de
Ciencia de la Computación

ICC Fase 1

Análisis y Diseño de Algoritmos

Adaptado del material de Paulo Feofilof y aportes de Coelho, Cris y Alair

Mg. Carlos Eduardo Atencio Torres

Universidad Nacional de San Agustín de Arequipa

2021/Semestre Impar

Índice

catencio@unsa.edu.pe

Recordando las primeras aulas de análisis asintótico

Notación $O(f(n))$

significa	la familia de funciones que no crecen más que $f(n)$.
incluye	el conjunto de funciones que son menores que $f(n)$.
indica	una cota superior

catencio@unsa.edu.pe

Recordando las primeras aulas de análisis asintótico

Notación $O(f(n))$

significa	la familia de funciones que no crecen más que $f(n)$.
incluye	el conjunto de funciones que son menores que $f(n)$.
indica	una cota superior

Notación

$T(n) = O(f(n))$ se lee "$T(n)$ es O de $(f(n))$".

- $T(n) \leq cf(n)$, en que c es una constante positiva, y $n \geq n_0$.

catencio@unsa.edu.pe

Recordando las primeras aulas de análisis asintótico

Notación $O(f(n))$

significa	la familia de funciones que no crecen más que $f(n)$.
incluye	el conjunto de funciones que son menores que $f(n)$.
indica	una cota superior

Notación

$T(n) = O(f(n))$ se lee "$T(n)$ es O de $(f(n))$".

- $T(n) \leq cf(n)$, en que c es una constante positiva, y $n \geq n_0$.

Ejemplo

$3n^2$ es $O(n^2)$. Probamos:

- Consideramos $T(n) = 3n^2$ y $f(n) = n^2$.
- Según definición, $3n^2 \leq cn^2$, para una constante positiva c y un $n \geq n_0$
- Para la prueba, consideramos $c = 3$ y $n_0 = 1$.

Recordando las primeras aulas de análisis asintótico

Notación $\Omega(f(n))$

significa	la familia de funciones que no crecen menos que $f(n)$.
incluye	el conjunto de funciones que son mayores que $f(n)$.
indica	una cota inferior

catencio@unsa.edu.pe

Recordando las primeras aulas de análisis asintótico

Notación $\Omega(f(n))$

significa	la familia de funciones que no crecen menos que $f(n)$.
incluye	el conjunto de funciones que son mayores que $f(n)$.
indica	una cota inferior

Notación

$T(n) = \Omega(f(n))$ se lee "$T(n)$ es *omega* de $(f(n))$".

- $T(n) \geq cf(n)$, en que c es una constante positiva, y $n \geq n_0$.

catencio@unsa.edu.pe

Recordando las primeras aulas de análisis asintótico

Notación $\Omega(f(n))$

significa	la familia de funciones que no crecen menos que $f(n)$.
incluye	el conjunto de funciones que son mayores que $f(n)$.
indica	una cota inferior

Notación

$T(n) = \Omega(f(n))$ se lee "$T(n)$ es *omega* de $(f(n))$".

- $T(n) \geq cf(n)$, en que c es una constante positiva, y $n \geq n_0$.

Ejemplo

$0.01n^2$ es $\Omega(n^2)$. Probamos:

- Consideramos $T(n) = 0.01n^2$ y $f(n) = n^2$.
- Según definición, $0.01n^2 \geq cn^2$, para una constante positiva c y un $n \geq n_0$
- Para la prueba, consideramos $c = 0.001$ y $n_0 = 1$.

catencio@unsa.edu.pe

Recordando las primeras aulas de análisis asintótico

Notación $\Theta(f(n))$

significa	familia de funciones que crecen en un orden igual a $f(n)$.
conforma	las familias pertenecientes a $O(f(n))$ y $\Omega(f(n))$

Recordando las primeras aulas de análisis asintótico

Notación $\Theta(f(n))$

significa	familia de funciones que crecen en un orden igual a $f(n)$.
conforma	las familias pertenecientes a $O(f(n))$ y $\Omega(f(n))$

Notación

$T(n) = \Theta(f(n))$ se lee "$T(n)$ es *theta* de $(f(n))$".

- $c_1 f(n) \leq T(n) \leq c_2 f(n)$, en que c_1 y c_2 es una constante positiva, y $n \geq n_0$.

catencio@unsa.edu.pe

Recordando las primeras aulas de análisis asintótico

Notación $\Theta(f(n))$

significa	familia de funciones que crecen en un orden igual a $f(n)$.
conforma	las familias pertenecientes a $O(f(n))$ y $\Omega(f(n))$

Notación

$T(n) = \Theta(f(n))$ se lee "$T(n)$ es *theta* de $(f(n))$".

- $c_1 f(n) \leq T(n) \leq c_2 f(n)$, en que c_1 y c_2 es una constante positiva, y $n \geq n_0$.

Ejemplo

$(3/2)n^2$ es $\Theta(n^2)$. Probamos:

- Consideramos $T(n) = (3/2)n^2$ y $f(n) = n^2$.
- Según definición, $c_1 n^2 \leq (3/2)n^2 \leq c_2 n^2$, para las constantes positivas c_1 y c_2, además $n \geq n_0$
- Para la prueba, consideramos $c_1 = (1/2)$, $c_2 = 2$ y $n_0 = 1$.

catencio@unsa.edu.pe

Análisis asintótico del algoritmo de INSERCIÓN

ORDENA-POR-INSERCION(A,p,r)

Análisis asintótico del algoritmo de INSERCIÓN

ORDENA-POR-INSERCION(A,p,r)

para $j \leftarrow p + 1$ hasta r **hacer**
    $clave \leftarrow A[j]$
    $i \leftarrow j - 1$
    mientras $i \geq p$ AND $A[i] > clave$ **hacer**
        $A[i+1] \leftarrow A[i]$, $\triangleright$ Haciendo campo
        $i \leftarrow i - 1$
    $A[i+1] \leftarrow clave$, $\triangleright$ Insertando

¿Cuánto tiempo consume el algoritmo?

Supongamos que $n = r - p + 1$

catencio@unsa.edu.pe

Consumo de tiempo

línea	Consumo de tiempo
1	$O(n)$
2	$O(n)$
3	$O(n)$
4	$nO(n)$
5	$nO(n)$
6	$nO(n)$
7	$O(n)$
Total	$O(3n^2 + 4n) = \mathrm{O}(n^2)$

Observaciones

- Las líneas 4-6 son ejecutadas $\leq n$ veces. Cada ejecución consume $O(n)$. Todas juntas consumen $nO(n)$.
- Probar $nO(n) = O(n^2)$.
- Probar $O(n^2) + O(n^2) + O(n^2) = O(3n^2)$.
- Probar $O(3n^2 + 4n) = O(n^2)$.

Algoritmo de Intercalación

Problema

Para los segmentos A[p. . . q] y A[q+1 . . . r], ambos crecientes, deseamos ordenar A[p. . . r] de modo que quede en orden creciente.

catencio@unsa.edu.pe

Algoritmo de Intercalación

Problema

Para los segmentos A[p. . . q] y A[q+1 . . . r], ambos crecientes, deseamos ordenar A[p. . . r] de modo que quede en orden creciente.

Entra

p				q						r
10	35	38	60	25	40	45	50	65	77	99

catencio@unsa.edu.pe

Algoritmo de Intercalación

Problema

Para los segmentos A[p. . . q] y A[q+1 . . . r], ambos crecientes, deseamos ordenar A[p. . . r] de modo que quede en orden creciente.

Entra

p				q						r
10	35	38	60	25	40	45	50	65	77	99

Sale

1										n
10	25	35	38	40	45	50	60	65	77	99

catencio@unsa.edu.pe

Algoritmo de Intercalación

Vaciamos los valores en un arreglo temporal B. Asegurarse de dejar el segundo bloque con los valores invertidos.

catencio@unsa.edu.pe

Algoritmo de Intercalación

Vaciamos los valores en un arreglo temporal B. Asegurarse de dejar el segundo bloque con los valores invertidos.

	p					q					r
B	20	25	35	40	44	50	99	65	55	38	10
A											

catencio@unsa.edu.pe

Algoritmo de Intercalación

Vaciamos los valores en un arreglo temporal B. Asegurarse de dejar el segundo bloque con los valores invertidos.

	i										j
B	20	25	35	40	44	50	99	65	55	38	10
	k										
A											

catencio@unsa.edu.pe

Algoritmo de Intercalación

Vaciamos los valores en un arreglo temporal B. Asegurarse de dejar el segundo bloque con los valores invertidos.

	i									j	
B	20	25	35	40	44	50	99	65	55	38	10
		k									
A	10										

catencio@unsa.edu.pe

Algoritmo de Intercalación

Vaciamos los valores en un arreglo temporal B. Asegurarse de dejar el segundo bloque con los valores invertidos.

		i								j	
B	20	25	35	40	44	50	99	65	55	38	10
			k								
A	10	20									

catencio@unsa.edu.pe

Algoritmo de Intercalación

Vaciamos los valores en un arreglo temporal B. Asegurarse de dejar el segundo bloque con los valores invertidos.

			i							j	
B	20	25	35	40	44	50	99	65	55	38	10
				k							
A	10	20	25								

catencio@unsa.edu.pe

Algoritmo de Intercalación

Vaciamos los valores en un arreglo temporal B. Asegurarse de dejar el segundo bloque con los valores invertidos.

				i						j	
B	20	25	35	40	44	50	99	65	55	38	10
					k						
A	10	20	25	35							

catencio@unsa.edu.pe

Algoritmo de Intercalación

Vaciamos los valores en un arreglo temporal B. Asegurarse de dejar el segundo bloque con los valores invertidos.

				i					j		
B	20	25	35	40	44	50	99	65	55	38	10
						k					
A	10	20	25	35	38						

Algoritmo de Intercalación

Vaciamos los valores en un arreglo temporal B. Asegurarse de dejar el segundo bloque con los valores invertidos.

					i				j		
B	20	25	35	40	44	50	99	65	55	38	10
							k				
A	10	20	25	35	38	40					

catencio@unsa.edu.pe

Algoritmo de Intercalación

Vaciamos los valores en un arreglo temporal B. Asegurarse de dejar el segundo bloque con los valores invertidos.

						i			j		
B	20	25	35	40	44	50	99	65	55	38	10
								k			
A	10	20	25	35	38	40	44				

catencio@unsa.edu.pe

Algoritmo de Intercalación

Vaciamos los valores en un arreglo temporal B. Asegurarse de dejar el segundo bloque con los valores invertidos.

							i		j		
B	20	25	35	40	44	50	99	65	55	38	10
									k		
A	10	20	25	35	38	40	44	50			

catencio@unsa.edu.pe

Algoritmo de Intercalación

Vaciamos los valores en un arreglo temporal B. Asegurarse de dejar el segundo bloque con los valores invertidos.

							i	j			
B	20	25	35	40	44	50	99	65	55	38	10
										k	
A	10	20	25	35	38	40	44	50	55		

Algoritmo de Intercalación

Vaciamos los valores en un arreglo temporal B. Asegurarse de dejar el segundo bloque con los valores invertidos.

i=j

B	20	25	35	40	44	50	99	65	55	38	10
											k
A	10	20	25	35	38	40	44	50	55	65	

catencio@unsa.edu.pe

Algoritmo de Intercalación

Vaciamos los valores en un arreglo temporal B. Asegurarse de dejar el segundo bloque con los valores invertidos.

i=j

B	20	25	35	40	44	50	99	65	55	38	10
A	10	20	25	35	38	40	44	50	55	65	99

catencio@unsa.edu.pe

Algoritmo de Intercalación

INTERCALA(A,p,q,r)

$\triangleright$ $B[p..r]$ es un vector auxiliar
para $i \leftarrow p$ **hasta** q **hacer**
    $B[i] \leftarrow A[i]$
para $j \leftarrow q + 1$ **hasta** r **hacer**
    $B[r + q + 1 - j] \leftarrow A[j]$
$i \leftarrow p$
$j \leftarrow r$
para $k \leftarrow p$ **hasta** r **hacer**
    si $B[i] \leq B[j]$ **entonces**
        $A[k] \leftarrow B[i]$
        $i \leftarrow i + 1$
    si no
        $A[k] \leftarrow B[j]$
        $j \leftarrow j - 1$

catencio@unsa.edu.pe

Ejercicios

- Si cada línea de código consume 1 unidad de tiempo. Cuál sería el consumo total?. $n = r - p + 1$
- Cómo sería el análisis asintótico por cada línea?.
- Demuestre que el algoritmo Intercalación es $\Omega(n)$.

catencio@unsa.edu.pe

Soluciones

Si cada línea de código consume 1 unidad de tiempo...

línea		consumo
1	=	1
2	=	$q - p + 2 = n - r + q + 1$
3	=	$q - p + 1 = n - r + q$
4	=	$r - (q+1) + 2 = n - q + p$
5	=	$r - (q+1) + 1 = n - q + p - 1$
6	=	1
7	=	1
8	=	$r - p+2 = n+1$
9	=	$r - p+1 = n$
10-14	=	$2(r - p+1) = 2n$
Total	=	$8n - 2\ (r-p+1)+5 = 6n+5$

Soluciones

Si cada línea de código consume 1 unidad de tiempo...

línea		**consumo**
1	=	1
2	=	$q - p + 2 = n - r + q + 1$
3	=	$q - p + 1 = n - r + q$
4	=	$r - (q+1) + 2 = n - q + p$
5	=	$r - (q+1) + 1 = n - q + p - 1$
6	=	1
7	=	1
8	=	$r - p+2 = n+1$
9	=	$r - p+1 = n$
10-14	=	$2(r - p+1) = 2n$
Total	=	$8n - 2\ (r-p+1)+5 = \underline{6n+5}$

catencio@unsa.edu.pe

Soluciones

Análisis asintótico...

línea	**consumo**
1-4	$O(n)$
5-6	$O(1)$
7	$nO(1) = O(n)$
8	$nO(1) = O(n)$
9-14	$nO(1) = O(n)$
Total	$O(4n + 1) = O(n)$

El algoritmo por lo tanto consume $O(n)$.

catencio@unsa.edu.pe

Soluciones

Demuestre que el algoritmo de intercalación es $\Omega(n)$

- Sabemos que el algoritmo consume $6n + 5$ unidades de tiempo, suponiendo que cada línea toma 1 unidad de tiempo.
- Probaremos por lo tanto que $6n + 5$ es $\Omega(n)$
- Según definición de Ω, para $c > 0$ y $n > n_0$

$$
\begin{aligned}
6n + 5 &\geq cn \\
6n + 5 &\geq 6n \\
c &= 6 \\
n_0 &= 1 \qquad \ldots \textit{Probado!}
\end{aligned}
$$

Podemos decir más aún..

- El algoritmo de intercalación es $O(n)$

catencio@unsa.edu.pe

Soluciones

Demuestre que el algoritmo de intercalación es $\Omega(n)$

- Sabemos que el algoritmo consume $6n + 5$ unidades de tiempo, suponiendo que cada línea toma 1 unidad de tiempo.
- Probaremos por lo tanto que $6n + 5$ es $\Omega(n)$
- Según definición de Ω, para $c > 0$ y $n > n_0$

$$
\begin{aligned}
6n + 5 &\geq cn \\
6n + 5 &\geq 6n \\
c &= 6 \\
n_0 &= 1 \qquad \qquad \ldots \textit{Probado}!
\end{aligned}
$$

Podemos decir más aún..

- El algoritmo de intercalación es $O(n)$
- El algoritmo de intercalación es $\Omega(n)$

Soluciones

Demuestre que el algoritmo de intercalación es $\Omega(n)$

- Sabemos que el algoritmo consume $6n + 5$ unidades de tiempo, suponiendo que cada línea toma 1 unidad de tiempo.
- Probaremos por lo tanto que $6n + 5$ es $\Omega(n)$
- Según definición de Ω, para $c > 0$ y $n > n_0$

$$
\begin{aligned}
6n + 5 &\geq cn \\
6n + 5 &\geq 6n \\
c &= 6 \\
n_0 &= 1 \qquad \ldots \textit{Probado}!
\end{aligned}
$$

Podemos decir más aún..

- El algoritmo de intercalación es $O(n)$
- El algoritmo de intercalación es $\Omega(n)$
- Por lo tanto el algoritmo de intercalación es $\Theta(n)$.

Divide y vencerás (Divide and Conquer)

- Estos algoritmos tienen 3 pasos:
  1. **Dividir**: Generación de subproblemas.
  2. **Conquistar**: Resolver cada subproblemas de forma recursiva.
  3. **Combinar**: Cada solución de los subproblemas es combinada.

ntender el concepto de subproblema es importante para probar los problemas de Programación Dinámica.

catencio@unsa.edu.pe

Divide y vencerás (Divide and Conquer)

- Estos algoritmos tienen 3 pasos:
  1. **Dividir**: Generación de subproblemas.
  2. **Conquistar**: Resolver cada subproblema de forma recursiva.
  3. **Combinar**: Cada solución de los subproblema es combinada.

Entender el concepto de subproblema es importante para probar los problemas de Programación Dinámica.

catencio@unsa.edu.pe

Merge Sort

Problema

Reordenar $A[p..r]$ de modo que esté en orden creciente.

Entra

	p								r
A	55	33	66	44	99	11	77	22	88

Sale

	p								r
A	11	22	33	44	55	66	77	88	99

catencio@unsa.edu.pe

Merge-Sort

	p				q				r
A	55	33	66	44	99	11	77	22	88

catencio@unsa.edu.pe

Merge-Sort

	p				q				r
A	55	33	66	44	99	11	77	22	88

	p		q		r				
A	55	33	66	44	99				

catencio@unsa.edu.pe

Merge-Sort

	p				q				r
A	55	33	66	44	99	11	77	22	88

	p		q		r				
A	55	33	66	44	99				

	p	q	r						
A	55	33	66						

Algoritmos – p.127/106

catencio@unsa.edu.pe

Merge-Sort

	p				q				r
A	55	33	66	44	99	11	77	22	88

	p		q		r				
A	55	33	66	44	99				

	p	q	r						
A	55	33	66						

	p	r							
A	55	33							

catencio@unsa.edu.pe

Merge-Sort

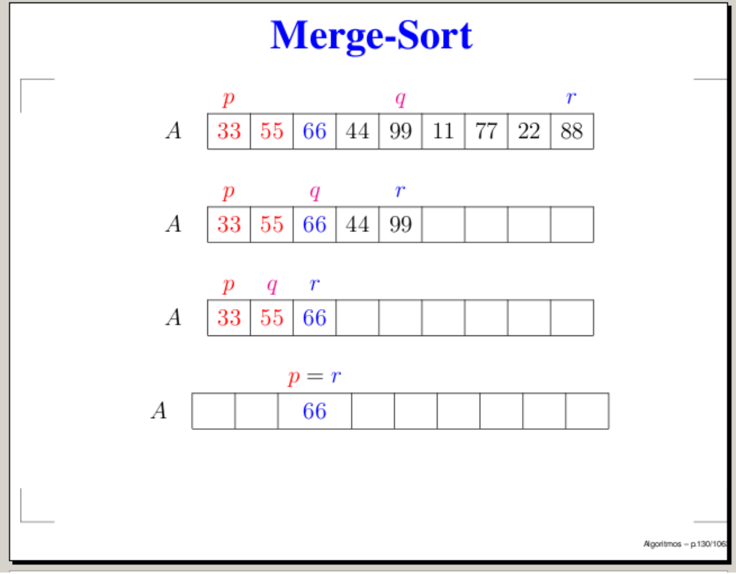
	p				q				r
A	33	55	66	44	99	11	77	22	88

	p		q		r				
A	33	55	66	44	99				

	p	q	r						
A	33	55	66						

	p	r							
A	33	55							

Algoritmos – p.129/106



catencio@unsa.edu.pe

Merge-Sort

	p				q				r
A	33	55	66	44	99	11	77	22	88

	p		q		r				
A	33	55	66	44	99				

	p	q	r						
A	33	55	66						

Algoritmos – p.131/106?

catencio@unsa.edu.pe

Merge-Sort

	p				q				r
A	33	55	66	44	99	11	77	22	88

	p		q		r				
A	33	55	66	44	99				

Algoritmos – p.132/106

catencio@unsa.edu.pe

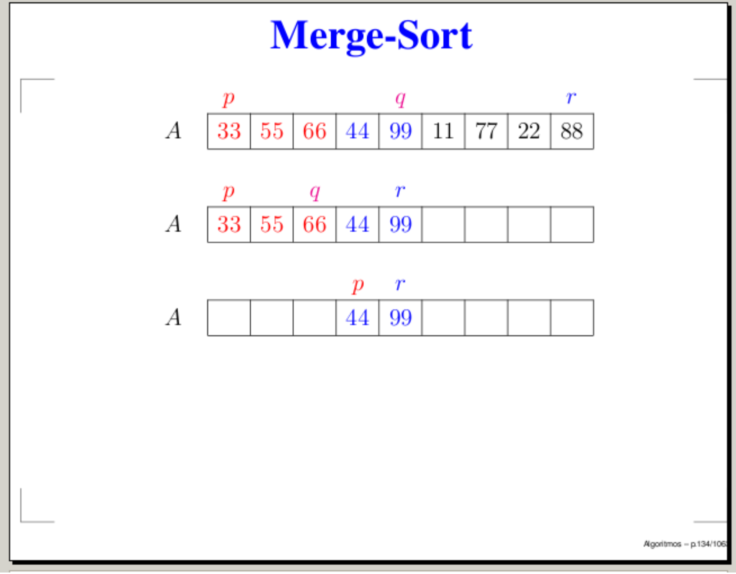
Merge-Sort

	p				q				r
A	33	55	66	44	99	11	77	22	88

	p		q		r				
A	33	55	66	44	99				

				p	r				
A				44	99				

catencio@unsa.edu.pe



catencio@unsa.edu.pe

Merge-Sort

	p				q				r
A	33	55	66	44	99	11	77	22	88

	p		q		r				
A	33	55	66	44	99				

Algoritmos – p.135/106

Merge-Sort

	p				q				r
A	33	44	55	66	99	11	77	22	88

	p		q		r				
A	33	44	55	66	99				

catencio@unsa.edu.pe

Merge-Sort

	p				q				r
A	33	44	55	66	99	11	77	22	88

catencio@unsa.edu.pe

Merge-Sort

	p				q				r
A	33	44	55	66	99	11	77	22	88

						p			r
A						11	77	22	88

catencio@unsa.edu.pe

Merge-Sort

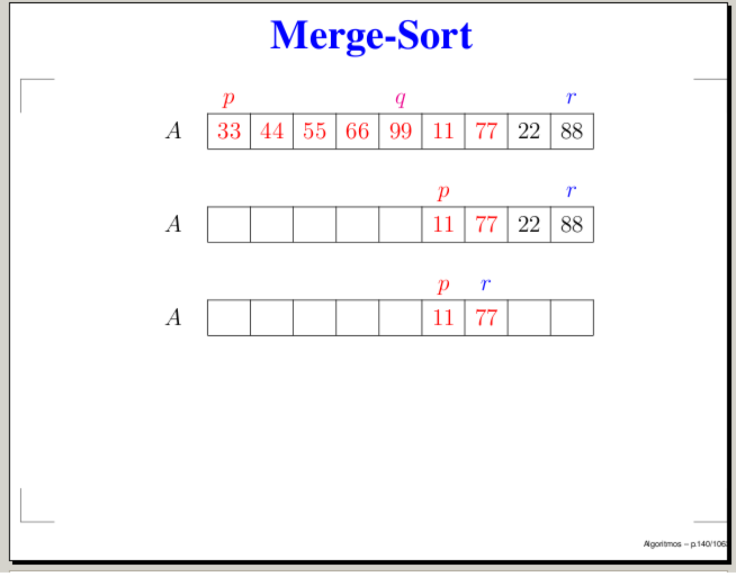
	p				q				r
A	33	44	55	66	99	11	77	22	88

						p			r
A						11	77	22	88

						p	r		
A						11	77		

Algoritmos – p.139/106

catencio@unsa.edu.pe



catencio@unsa.edu.pe

Merge-Sort

	p				q				r
A	33	44	55	66	99	11	77	22	88

						p			r
A						11	77	22	88

catencio@unsa.edu.pe

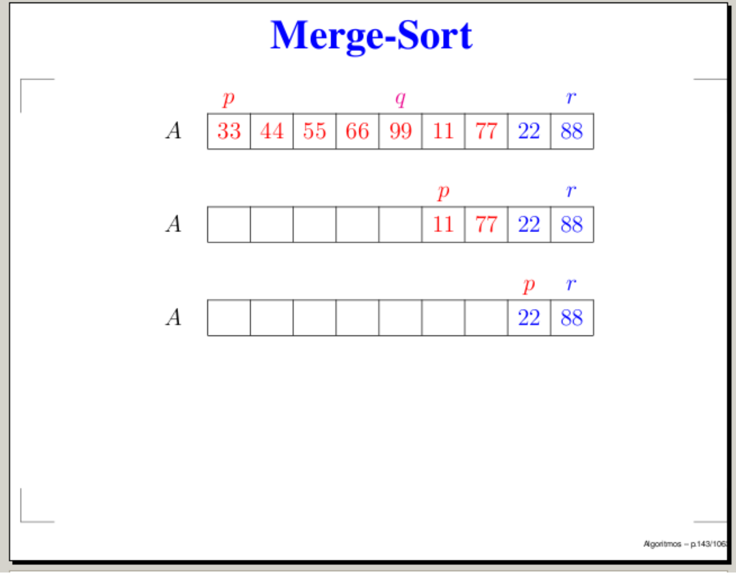
Merge-Sort

	p				q				r
A	33	44	55	66	99	11	77	22	88

						p			r
A						11	77	22	88

								p	r
A								22	88

catencio@unsa.edu.pe



catencio@unsa.edu.pe

Merge-Sort

	p				q				r
A	33	44	55	66	99	11	77	22	88

						p			r
A						11	77	22	88

Algoritmos – p.144/106

Merge-Sort

	p				q				r
A	33	44	55	66	99	11	22	77	88

						p			r
A						11	22	77	88

catencio@unsa.edu.pe

Merge-Sort

	p				q				r
A	33	44	55	66	99	11	22	77	88

catencio@unsa.edu.pe

Merge-Sort

	p				q				r
A	11	22	33	44	55	66	77	88	99

catencio@unsa.edu.pe

Algoritmo de MergeSort

MERGE-SORT(A,p,r)

1: **si** $p < r$ **entonces**
2: $q \leftarrow \lfloor (p+r)/2 \rfloor$
3: MERGE-SORT(A,p,q-1)
4: MERGE-SORT(A,q+1,r)
5: INTERCALA(A,p,q,r)

Invariantes

- i1 Al final de la línea 3, los elementos entre p y $q-1$ están ordenados.
- i2 Al final de la línea 4, los elementos entre $q+1$ y r están ordenados.
- i3 Al final de la línea 5, los elementos entre p y r están ordenados.

catencio@unsa.edu.pe

Algoritmo de MergeSort

MERGE-SORT(A,p,r)

1: **si** $p < r$ **entonces**
2: $q \leftarrow \lfloor (p + r)/2 \rfloor$
3: MERGE-SORT(A,p,q-1)
4: MERGE-SORT(A,q+1,r)
5: INTERCALA(A,p,q,r)

Corrección del algoritmo (Prueba inductiva)

- Para el primer caso, cuando el arreglo tiene tamaño $n = 1$, es decir, $p = r$, el algoritmo sólo trabaja hasta la línea 1 dejando el arreglo intacto. Este arreglo por tener 1 elemento ya está ordenado.
- Para el caso de $n > 1$, el problema será subdivido en subproblemas y al final de las llamadas recursivas, el algoritmo de intercala retornará un arreglo ordenado de tamaño n.
- Note que el algoritmo de intercalación no puede ser probado a no ser que las invariantes i1 e i2 sean falsas.

catencio@unsa.edu.pe

Consumo de Tiempo del Merge-Sort

línea		Consumo en la línea
1	$=$	$\theta(1)$
2	$=$	$\theta(1)$
3	$=$	$T(\lceil n/2 \rceil)$
4	$=$	$T(\lfloor n/2 \rfloor)$
5	$=$	$\theta(n)$
T(n)	$=$	$T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \theta(n+2)$

catencio@unsa.edu.pe

Consumo de Tiempo del Merge-Sort

línea		Consumo en la línea
1	$=$	$\theta(1)$
2	$=$	$\theta(1)$
3	$=$	$T(\lceil n/2 \rceil)$
4	$=$	$T(\lfloor n/2 \rfloor)$
5	$=$	$\theta(n)$
T(n)	$=$	$T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \theta(n+2)$

Conclusión

Por la segunda regla del Teorema Maestro, obtenemos que la complejidad del Merge-Sort queda en $\theta(nlg(n))$

catencio@unsa.edu.pe

Teorema Maestro

Suponga:

$$T(n) = aT(n/b) + f(n)$$

para algún $a \geq 1$ y $b > 1$ y donde n/b significa $\lceil n/b \rceil$ o $\lfloor n/b \rfloor$ entonces en general:

$$\begin{array}{lll} \text{Si } f(n) = O(n^{log_b a - e}) & \text{entonces} & T(n) = \theta(n^{log_b a}) \\ \text{Si } f(n) = \Theta(n^{log_b a}) & \text{entonces} & T(n) = \theta(n^{log_b a} lg\, n) \\ \text{Si } f(n) = \Omega(n^{log_b a + e}) & \text{entonces} & T(n) = \theta(f(n)) \end{array}$$

para un $e > 0$

catencio@unsa.edu.pe

Teorema Maestro Simplificado

Suponga

$$T(n) = aT(n/b) + cn^k$$

para algun $a \geq 1$ y $b > 1$ y donde n/b significa $\lceil n/b \rceil$ o $\lfloor n/b \rfloor$. Entonces en general:

Si $a > b^k$	entonces	$T(n) = \theta(n^{\log_b a})$
Si $a = b^k$	entonces	$T(n) = \theta(n^k \lg n)$
Si $a < b^k$	entonces	$T(n) = \theta(n^k)$

catencio@unsa.edu.pe