Universidad Politécnica de Madrid
Escuela Técnica Superior de Ingeniería de Sistemas Informáticos

Tema 4. Recursividad

Algorítmica y Complejidad

Introducción

Algoritmo Recursivo

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
      …
      algoritmoRecursivo(…)
      …
}
```


Introducción

Resolución de la complejidad de un algoritmo recursivo

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
      …
      algoritmoRecursivo(…)
      …
}
```

Ecuación de Recurrencia

$T(n)$ = ...
(Ecuación en Diferencias Finitas)

Cálculo del Orden

$T(n) \in \Theta(...)$

Asignatura:
Análisis Matemático

Resolución de ecuaciones en diferencias finitas:

- ***Ecuaciones lineales con coeficientes constantes***

Introducción

Resolución de la complejidad de un algoritmo recursivo

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
      …
}
```

Ecuación de Recurrencia

$T(n)$ = ...
(Ecuación en Diferencias Finitas)

Cálculo del Orden

$T(n) \in \Theta(...)$

Ejemplos:

```
int factorial (int n){…}
```

```
int fibonacci (int n){…}
```

Introducción

Ejemplo: factorial

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
      …
}
```

Ecuación de Recurrencia

$T(n)$ = ...

(Ecuación en Diferencias Finitas)

Cálculo del Orden

$T(n) \in \Theta(...)$

Ejemplos:

```
int factorial (int n){…}
```

```
int fibonacci (int n){…}
```

1. Introducción

Introducción

Ejemplo: factorial

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
      …
}
```

Ecuación de Recurrencia

$T(n)$ = ...

(Ecuación en Diferencias Finitas)

Cálculo del Orden

$T(n) \in \Theta(...)$

```
int factorial (int n){
      if (n==0)
         return 1;
      else
         return n* factorial(n-1);
}
```

1. Introducción

Introducción

Ejemplo: factorial

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
      …
}
```

Ecuación de Recurrencia

$T(n)$ = ...

(Ecuación en Diferencias Finitas)

Cálculo del Orden

$T(n) \in \Theta(...)$

```
int factorial (int n){
      if (n==0)
         return 1;
      else
         return n* factorial(n-1);
}
```

Ecuación de Recurrencia

$$T(n) = \begin{cases} a & n = 0 \\ T(n-1) + b & n > 0 \end{cases}$$

1. Introducción

Introducción

Ejemplo: factorial

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
      …
}
```

Ecuación de Recurrencia

$T(n)$ = ...

(Ecuación en Diferencias Finitas)

Cálculo del Orden

$T(n) \in \Theta(...)$

```
int factorial (int n){
      if (n==0)
         return 1;          } a ∈ Θ(1)
      else
         return n* factorial(n-1);
}
```

Ecuación de Recurrencia

$$T(n) = \begin{cases} a & n = 0 \\ T(n-1) + b & n > 0 \end{cases}$$

1. Introducción

Introducción

Ejemplo: factorial

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
    …
}
```

Ecuación de Recurrencia

$T(n)$ = ...

(Ecuación en Diferencias Finitas)

Cálculo del Orden

$T(n) \in \Theta(...)$

```
int factorial (int n){
    if (n==0)
        return 1;
    else
        return n* factorial(n-1);
}
```

$T(n-1)$ + b → $\Theta(1)$

Ecuación de Recurrencia

$$T(n) = \begin{cases} a & n = 0 \\ T(n-1) + b & n > 0 \end{cases}$$

Introducción

Ejemplo: factorial

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
    …
}
```

Ecuación de Recurrencia

$T(n)$ = ...

(Ecuación en Diferencias Finitas)

Cálculo del Orden

$T(n) \in \Theta(...)$

```
int factorial (int n){
    if (n==0)
        return 1;
    else
        return n* factorial(n-1);
}
```

Condiciones iniciales

Ecuación de Recurrencia

$$T(n) = \begin{cases} a & n = 0 \\ T(n-1) + b & n > 0 \end{cases}$$

Introducción

Ejemplo: factorial

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
    …
}
```

Ecuación de Recurrencia

$T(n)$ = ...

(Ecuación en Diferencias Finitas)

Cálculo del Orden

$T(n) \in \Theta(...)$

```
int factorial (int n){
    if (n==0)
        return 1;
    else
        return n* factorial(n-1);
}
```

Ecuación de Recurrencia

$$T(n) = T(n-1) + b$$

Introducción

Ejemplo: factorial

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
    …
}
```

Ecuación de Recurrencia

$T(n)$ = ...

(Ecuación en Diferencias Finitas)

Cálculo del Orden

$T(n) \in \Theta(...)$

```
int factorial (int n){
    if (n==0)
        return 1;
    else
        return n* factorial(n-1);
}
```

Ecuación de Recurrencia

$$T(n) = T(n-1) + b$$

Ecuación lineal con coeficientes constantes

Resolución de ecuaciones en diferencias finitas

Introducción

Ejemplo: factorial

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
      …
}
```

Ecuación de Recurrencia

$T(n)$ = ...

(Ecuación en Diferencias Finitas)

Cálculo del Orden

$T(n) \in \Theta(...)$

```
int factorial (int n){
      if (n==0)
         return 1;
      else
         return n* factorial(n-1);
}
```

Ecuación de Recurrencia

$$T(n) = T(n-1) + b$$

$T(n) = b \cdot n + a \in \Theta(n)$

Introducción

Ejemplo: factorial

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
      …
}
```

Ecuación de Recurrencia

$T(n)$ = ...

(Ecuación en Diferencias Finitas)

Cálculo del Orden

$T(n) \in \Theta(...)$

Ejemplos:

```
int factorial (int n){…}
```

```
int fibonacci (int n){…}
```


Introducción

Ejemplo: Fibonacci

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
      …
}
```

Ecuación de Recurrencia

$T(n)$ = ...

(Ecuación en Diferencias Finitas)

Cálculo del Orden

$T(n) \in \Theta(...)$

```
int fibonacci (int n){
 if (n<=1)
   return 1;
 else
   return fibonacci(n-1)+fibonacci(n-2);
}
```


Introducción

Ejemplo: Fibonacci

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
      …
}
```

Ecuación de Recurrencia

$T(n)$ = ...

(Ecuación en Diferencias Finitas)

Cálculo del Orden

$T(n) \in \Theta(...)$

```
int fibonacci (int n){
 if (n<=1)
   return 1;
 else
   return fibonacci(n-1)+fibonacci(n-2);
}
```

Ecuación de Recurrencia

$$T(n) = \begin{cases} a & n \leq 1 \\ T(n-1) + T(n-2) + b & n > 0 \end{cases}$$

Introducción

Ejemplo: Fibonacci

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
    …
}
```

Ecuación de Recurrencia

$T(n)$ = ...

(Ecuación en Diferencias Finitas)

Cálculo del Orden

$T(n) \in \Theta(...)$

```
int fibonacci (int n){
 if (n<=1)
   return 1;
 else
   return fibonacci(n-1)+fibonacci(n-2);
}
```

$a \in \Theta(1)$

Ecuación de Recurrencia

$$T(n) = \begin{cases} a & n \leq 1 \\ T(n-1) + T(n-2) + b & n > 0 \end{cases}$$

1. Introducción

1 2 3

Introducción

Ejemplo: Fibonacci

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
    …
}
```

Ecuación de Recurrencia

$T(n)$ = ...

(Ecuación en Diferencias Finitas)

Cálculo del Orden

$T(n) \in \Theta(...)$

```
int fibonacci (int n){
 if (n<=1)
   return 1;     T(n-1)      T(n-2)
 else
   return fibonacci(n-1)+fibonacci(n-2);
}
```

$T(n-1) + T(n-1) + b$

$\Theta(1)$

Ecuación de Recurrencia

$$T(n) = \begin{cases} a & n \leq 1 \\ T(n-1) + T(n-2) + b & n > 1 \end{cases}$$

1. Introducción

1 2 3

Introducción

Ejemplo: Fibonacci

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
    …
}
```

Ecuación de Recurrencia

$T(n)$ = ...

(Ecuación en Diferencias Finitas)

Cálculo del Orden

$T(n) \in \Theta(...)$

```
int fibonacci (int n){
 if (n<=1)
   return 1;
 else
   return fibonacci(n-1)+fibonacci(n-2);
}
```

Condiciones iniciales

Ecuación de Recurrencia

$$T(n) = \begin{cases} a & n \leq 1 \\ T(n-1) + T(n-2) + b & n > 0 \end{cases}$$

1. Introducción

1 2 3

Introducción

Ejemplo: Fibonacci

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
    …
}
```

Ecuación de Recurrencia

$T(n)$ = ...

(Ecuación en Diferencias Finitas)

Cálculo del Orden

$T(n) \in \Theta(...)$

```
int fibonacci (int n){
 if (n<=1)
   return 1;
 else
   return fibonacci(n-1)+fibonacci(n-2);
}
```

Ecuación de Recurrencia

$$T(n) = T(n-1) + T(n-2) + b$$

1. Introducción

1 2 3

Introducción

Ejemplo: Fibonacci

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
    …
}
```

Ecuación de Recurrencia

$T(n)$ = ...

(Ecuación en Diferencias Finitas)

Cálculo del Orden

$T(n) \in \Theta(...)$

```
int fibonacci (int n){
 if (n<=1)
   return 1;
 else
   return fibonacci(n-1)+fibonacci(n-2);
}
```

Ecuación de Recurrencia

$$T(n) = T(n-1) + T(n-2) + b$$

Ecuación lineal con coeficientes constantes

?

Resolución de ecuaciones en diferencias finitas

Introducción

Ejemplo: Fibonacci

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
    …
}
```

Ecuación de Recurrencia

$T(n)$ = ...

(Ecuación en Diferencias Finitas)

Cálculo del Orden

$T(n) \in \Theta(...)$

```
int fibonacci (int n){
 if (n<=1)
   return 1;
 else
   return fibonacci(n-1)+fibonacci(n-2);
}
```

Ecuación de Recurrencia

$$T(n) = T(n-1) + T(n-2) + b$$

$$T(n) = c_1 \cdot \left(\frac{1+\sqrt{5}}{2}\right)^n + c_2 \cdot \left(\frac{1-\sqrt{5}}{2}\right)^n + c_3$$

$c_1 > 0$

$$\Theta\left(\left(\frac{1+\sqrt{5}}{2}\right)^n\right)$$

Introducción

Ecuaciones de Recurrencias

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
    …
}
```

Ecuación de Recurrencia

$T(n)$ = ...

(Ecuación en Diferencias Finitas)

?

Cálculo del Orden

$T(n) \in \Theta(...)$

- $T(n) = T(n-1) + f(n)$
- $T(n) = p \cdot T\left(\frac{n}{q}\right) + f(n)$

Ecuación de Recurrencia Simple

Recurrencia Simple

$$T(0) = f(0)$$

$$T(n) = T(n-1) + f(n)$$

$$T(n) = \sum_{i=0}^{n} f(i)$$

Ejemplo 1

$T(n) = T(n\text{-}1) + b$ $\quad$ $T(0) = b$

$$T(n) = \sum_{i=0}^{n} f(i) = (n+1)b \in \Theta(n)$$

Ecuación de Recurrencia Simple

Recurrencia Simple

$T(0) = f(0)$

$T(n) = T(n-1) + f(n)$

$$T(n) = \sum_{i=0}^{n} f(i)$$

Ejemplo 2

$T(n) = T(n\text{-}1) + n$ $\quad$ $T(0) = 0$

$$T(n) = \sum_{i=0}^{n} f(i) = \sum_{i=0}^{n} i = \frac{n(n+1)}{2} \in \Theta(n^2)$$

Ecuación de Recurrencia Simple

Complejidad Algorítmica

Actividad 4.1. Supongamos que tenemos la siguiente ecuación de recurrencia (donde *a* es una constante):

T(0)=5

T(N)=*T*(N-1) + 2*N*+5

Calcula la complejidad de *T*(*N*)

$$T(N) = \sum_{i=0}^{N} (2i+5) = 5(N+1) + 2\sum_{i=0}^{N} i = 5N + 5 + 2\frac{N(N+1)}{2} \in \Theta(N^2)$$

Teorema Maestro

Algoritmo

```
tipoSalida algoritmoRecursivo(…){
    …
}
```

Ecuación de Recurrencia

T(*n*) = ...

(Ecuación en Diferencias Finitas)

Cálculo del Orden

T(*n*) ∈ **Θ**(...)

- $T(n) = T(n-1) + f(n)$
- $T(n) = p \cdot T\left(\frac{n}{q}\right) + f(n)$

Muy útil para analizar la complejidad en algoritmos basados en el **esquema Divide y Vencerás**

Teorema Maestro

Casos

$$T(n) = p \cdot T\left(\frac{n}{q}\right) + f(n)$$ $p \geq 1$, $q > 1$

1er Caso

$f(n) \in \mathbf{O}(n^a)$ donde $a < \log_q(p)$

$T(n) \in \Theta\left(n^{\log_q(p)}\right)$

2º Caso

$f(n) \in \mathbf{\Theta}(n^a)$ donde $a = \log_q(p)$

$T(n) \in \Theta\left(n^{\log_q(p)} \cdot \ln n\right)$

3er Caso

$f(n) \in \mathbf{\Omega}(n^a)$ donde $a > \log_q(p)$

Además se cumple que $\exists c<1\ \forall n>n_0$: $p \cdot f(n/q) \leq c \cdot f(n)$

$T(n) \in \Theta(f(n))$

Teorema Maestro

1er Caso

$T(n) = p \cdot T\left(\frac{n}{q}\right) + f(n)$ $p \geq 1$, $q > 1$

1er Caso

$f(n) \in \mathbf{O}(n^a)$ donde $a < \log_q(p)$

$T(n) \in \Theta\left(n^{\log_q(p)}\right)$

$T(n) = 4 \cdot T\left(\frac{n}{2}\right) + 5n + \log n$

$\log_2(4)=2$ $\quad 5n + \log n \in O(n^1)$

1er Caso $1 < \log_2(4)=2$

$T(n) \in \Theta(n^2)$

3. Teorema Maestro

Teorema Maestro

2º Caso

$T(n) = p \cdot T\left(\frac{n}{q}\right) + f(n)$ $p \geq 1$, $q > 1$

2º Caso

$f(n) \in \mathbf{\Theta}(n^a)$ donde $a = \log_q(p)$

$T(n) \in \Theta\left(n^{\log_q(p)} \cdot \ln n\right)$

$T(n) = 4 \cdot T\left(\frac{n}{2}\right) + n^2 + 5n$

$\log_2(4)=2$ $\quad n^2 + 5n \in \Theta(n^2)$

2º Caso $2 = \log_2(4)=2$

$T(n) \in \Theta(n^2 \cdot \ln n)$

3. Teorema Maestro

Teorema Maestro

3er Caso

$T(n) = p \cdot T\left(\frac{n}{q}\right) + f(n)$ $p \geq 1$, $q > 1$

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

$\log_4(4)=1$ $\quad \frac{n^2}{4} \in \Omega(n^2)$

3er Caso $2 > \log_4(4)=1$ $\quad 4 \cdot f(N/4) = f(N)$

$T(n) \in \Theta(n^2)$

3er Caso

$f(n) \in \mathbf{\Omega}(n^a)$ donde $a > \log_q(p)$

Además se cumple que $\exists c<1\ \forall n>n_0$: $p \cdot f(n/q) \leq c \cdot f(n)$

$T(n) \in \Theta(f(n))$

3. Teorema Maestro

Teorema Maestro

Complejidad Algorítmica

Actividad 4.2. Supongamos que tenemos la siguiente ecuación de recurrencia (donde a es una constante):

$$T(N)=2 \cdot T(N/2) + a$$

Calcula la complejidad de $T(N)$

$T(N) = 2 \cdot T\left(\frac{N}{2}\right) + a$

$\log_2(2)=1$ $\quad a \in O(N^0)$

1er Caso $0 < \log_2(2)=1$

$T(N) \in \Theta(N)$

3. Teorema Maestro

Teorema Maestro

Complejidad Algorítmica

Actividad 4.3. Supongamos que tenemos la siguiente ecuación de recurrencia (donde a es una constante):

$$T(N)=T(N/2)+a$$

Calcula la complejidad de $T(N)$

$$T(N) = 1 \cdot T\left(\frac{N}{2}\right) + a$$

$\log_2(1)=0 \qquad a \in \Theta(N^0)$

2º Caso $0 = \log_2(1)=0$

$$T(N) \in \Theta(\log N)$$

3. Teorema Maestro

Teorema Maestro

Complejidad Algorítmica

Actividad 4.4. Supongamos que tenemos la siguiente ecuación de recurrencia:

$$T(N)=T(N/2)+N$$

Calcula la complejidad de $T(N)$

$$T(N) = 1 \cdot T\left(\frac{N}{2}\right) + N$$

$\log_2(1)=0 \qquad N \in \Omega(N^1)$

3er Caso $1 > \log_4(4)=0$ $\qquad \frac{N}{2} \leq \frac{1}{2}N$

$$T(n) \in \Theta(N)$$

3. Teorema Maestro

Teorema Maestro

Complejidad Algorítmica

Actividad 4.5. Supongamos que tenemos la siguiente ecuación de recurrencia:

$$T(N)=4 \cdot T(N/2)+N^2$$

Calcula la complejidad de $T(N)$

$$T(N) = 4 \cdot T\left(\frac{N}{2}\right) + N^2$$

$\log_2(4)=2 \qquad a \in \Theta(N^2)$

2º Caso $2 = \log_2(4)=2$

$$T(N) \in \Theta(N^2 \cdot \log N)$$

3. Teorema Maestro

Teorema Maestro

Complejidad Algorítmica

Actividad 4.6. Supongamos que tenemos la siguiente ecuación de recurrencia:

$$T(N)=8 \cdot T(N/2)+N^2+2N \cdot \ln N$$

Calcula la complejidad de $T(N)$

$$T(N) = 8 \cdot T\left(\frac{N}{2}\right) + N^2 + 2N \cdot \ln N$$

$\log_2(8)=3 \qquad N^2 + 2N \cdot \ln N \in \mathbf{O}(N^2)$

1er Caso $2 < \log_2(8)=3$

$$T(N) \in \Theta(N^3)$$

3. Teorema Maestro

Teorema Maestro

Complejidad Algorítmica

Actividad 4.7. Supongamos que tenemos la siguiente ecuación de recurrencia:

$$T(N)=4T(N/4)+N^2$$

Calcula la complejidad de $T(N)$

$T(N) = 4 \cdot T\left(\frac{N}{4}\right) + N^2$

$\log_4(4)=1 \qquad N^2 \in \Omega(N^2)$

3er Caso $2 > \log_4(4)=1 \qquad 4\left(\frac{N}{4}\right)^2 \leq \frac{4}{16}N^2$

$T(n) \in \Theta(N^2)$

3. Teorema Maestro

Teorema Maestro

Complejidad Algorítmica

Actividad 4.8. Supongamos que tenemos la siguiente ecuación de recurrencia:

$$T(N)=4\cdot T(N/2)+N+2$$

Calcula la complejidad de $T(N)$

$T(N) = 2 \cdot T\left(\frac{N}{2}\right) + N + 2$

$\log_2(4)=2 \qquad N+2 \in O(N^1)$

1er Caso $1 < \log_2(4)=2$

$T(N) \in \Theta(N^2)$

3. Teorema Maestro

Teorema Maestro

Complejidad Algorítmica

Actividad 4.9. Supongamos que tenemos la siguiente ecuación de recurrencia:

$$T(N)=3\cdot T(N/2)+3N+\ln N$$

Calcula la complejidad de $T(N)$

$T(N) = 3 \cdot T\left(\frac{N}{2}\right) + 3N + ln\, N$

$\log_2(3)=1.58 \qquad 3N + \ln N \in O(N^1)$

1er Caso $1 < \log_2(3)=1.58$

$T(N) \in \Theta(N^{1.58})$

3. Teorema Maestro

Teorema Maestro

Complejidad Algorítmica

Actividad 4.10. Supongamos que tenemos la siguiente ecuación de recurrencia:

$$T(N)=3\cdot T(N/2)+N^2$$

Calcula la complejidad de $T(N)$

$T(N) = 3 \cdot T\left(\frac{N}{2}\right) + N^2$

$\log_2(3)=1.58 \qquad N^2 \in \Omega(N^2)$

3er Caso $2 > \log_2(3)=1.58 \qquad 3\left(\frac{N}{2}\right)^2 \leq \frac{3}{4}N^2$

$T(N) \in \Theta(N^2)$

3. Teorema Maestro

Teorema Maestro

No exhaustividad

$$T(n) = p \cdot T\left(\frac{n}{q}\right) + f(n) \qquad p \geq 1,\ q > 1$$

1er Caso	2º Caso	3er Caso
$f(n) \in \mathbf{O}(n^a)$ donde $a < \log_q(p)$	$f(n) \in \mathbf{\Theta}(n^a)$ donde $a = \log_q(p)$	$f(n) \in \mathbf{\Omega}(n^a)$ donde $a > \log_q(p)$
$T(n) \in \Theta\left(n^{\log_q(p)}\right)$	$T(n) \in \Theta\left(n^{\log_q(p)} \cdot \ln n\right)$	Además se cumple que $\exists c<1\ \forall n>n_0$: $p \cdot f(n/q) \leq c \cdot f(n)$
		$T(n) \in \Theta(f(n))$

Hay casos que **NO** contempla el **TEOREMA MAESTRO** !!!

Teorema Maestro

No exhaustividad: Ejemplo 1

$$T(n) = p \cdot T\left(\frac{n}{q}\right) + f(n) \qquad p \geq 1,\ q > 1$$

1er Caso	2º Caso	3er Caso
$f(n) \in \mathbf{O}(n^a)$ donde $a < 1$	$f(n) \in \mathbf{\Theta}(n^a)$ donde $a = 1$	$f(n) \in \mathbf{\Omega}(n^a)$ donde $a > 1$
$T(n) \in \Theta\left(n^{\log_q(p)}\right)$	$T(n) \in \Theta\left(n^{\log_q(p)} \cdot \ln n\right)$	Además se cumple que $\exists c<1\ \forall n>n_0$: $p \cdot f(n/q) \leq c \cdot f(n)$
		$T(n) \in \Theta(f(n))$

Hay casos que **NO** contempla el **TEOREMA MAESTRO** !!!

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

Teorema Maestro

No exhaustividad: Ejemplo 1

$$T(n) = p \cdot T\left(\frac{n}{q}\right) + f(n) \qquad p \geq 1,\ q > 1$$

1er Caso: $f(n) \in \mathbf{O}(n^a)$ donde $a < 1$

$$\lim_{n\to\infty} \frac{\frac{n}{\log n}}{n^a} = \lim_{n\to\infty} \frac{n^{1-a}}{\log n} = \infty \Rightarrow f(n) \notin \mathbf{O}(n^a)$$

Hay casos que **NO** contempla el **TEOREMA MAESTRO** !!!

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

Teorema Maestro

No exhaustividad: Ejemplo 1

$$T(n) = p \cdot T\left(\frac{n}{q}\right) + f(n) \qquad p \geq 1,\ q > 1$$

2º Caso: $f(n) \in \mathbf{\Theta}(n^a)$ donde $a = 1$

$$\lim_{n\to\infty} \frac{\frac{n}{\log n}}{n} = \lim_{n\to\infty} \frac{1}{\log n} = 0 \Rightarrow f(n) \notin \mathbf{\Theta}(n)$$

Hay casos que **NO** contempla el **TEOREMA MAESTRO** !!!

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

Teorema Maestro

No exhaustividad: Ejemplo 1

$$T(n) = p \cdot T\left(\frac{n}{q}\right) + f(n)$$ $p \geq 1$, $q > 1$

3er Caso

$f(n) \in \Omega(n^a)$ donde $a > 1$

$$\lim_{n\to\infty} \frac{n}{\log n} = \lim_{n\to\infty} \frac{1}{n^{a-1}\log n} = 0 \Rightarrow f(n) \notin \Omega(n^a)$$

Hay casos que **NO** contempla el **TEOREMA MAESTRO** !!!

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

3. Teorema Maestro

Teorema Maestro

No exhaustividad: Ejemplo 2

$$T(n) = p \cdot T\left(\frac{n}{q}\right) + f(n)$$ $p \geq 1$, $q > 1$

1er Caso: $f(n) \in \mathbf{O}(n^a)$ donde $a < \log_q(p)$ → $T(n) \in \Theta\left(n^{\log_q(p)}\right)$

2º Caso: $f(n) \in \mathbf{\Theta}(n^a)$ donde $a = \log_q(p)$ → $T(n) \in \Theta\left(n^{\log_q(p)} \cdot \ln n\right)$

3er Caso: $f(n) \in \mathbf{\Omega}(n^a)$ donde $a > \log_q(p)$

Además se cumple que $\exists c<1\ \forall n>n_0$: $p \cdot f(n/q) \leq c \cdot f(n)$ → $T(n) \in \Theta(f(n))$

Hay casos que **NO** contempla el **TEOREMA MAESTRO** !!!

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

No es una constante

3. Teorema Maestro

Teorema Maestro

No exhaustividad: Ejemplo 3

$$T(n) = p \cdot T\left(\frac{n}{q}\right) + f(n)$$ $p \geq 1$, $q > 1$

1er Caso: $f(n) \in \mathbf{O}(n^a)$ donde $a < \log_q(p)$ → $T(n) \in \Theta\left(n^{\log_q(p)}\right)$

2º Caso: $f(n) \in \mathbf{\Theta}(n^a)$ donde $a = \log_q(p)$ → $T(n) \in \Theta\left(n^{\log_q(p)} \cdot \ln n\right)$

3er Caso: $f(n) \in \mathbf{\Omega}(n^a)$ donde $a > \log_q(p)$

Además se cumple que $\exists c<1\ \forall n>n_0$: $p \cdot f(n/q) \leq c \cdot f(n)$ → $T(n) \in \Theta(f(n))$

Hay casos que **NO** contempla el **TEOREMA MAESTRO** !!!

$$T(n) = \frac{1}{2} \cdot T\left(\frac{n}{2}\right) + \frac{n}{\log n}$$

3. Teorema Maestro

Teorema Maestro

No exhaustividad: Ejemplo 4

$$T(n) = p \cdot T\left(\frac{n}{q}\right) + f(n)$$ $p \geq 1$, $q > 1$

3er Caso

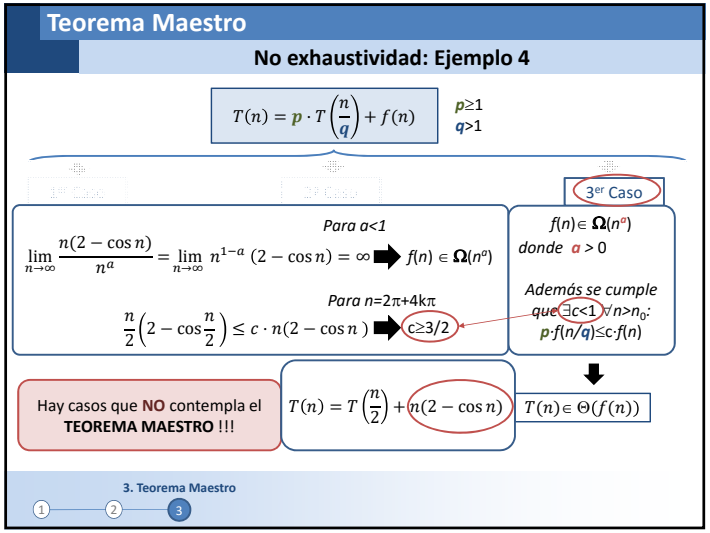
$f(n) \in \mathbf{\Omega}(n^a)$ donde $a > 0$

Además se cumple que $\exists c<1\ \forall n>n_0$: $p \cdot f(n/q) \leq c \cdot f(n)$ → $T(n) \in \Theta(f(n))$

Hay casos que **NO** contempla el **TEOREMA MAESTRO** !!!

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

3. Teorema Maestro


Teorema Maestro
No exhaustividad: Ejemplo 4
$T(n) = p \cdot T\left(\frac{n}{q}\right) + f(n)$
p≥1
q>1
3er Caso
Para a<1
$\lim_{n\to\infty} \frac{n(2-\cos n)}{n^a} = \lim_{n\to\infty} n^{1-a}(2-\cos n) = \infty$ f(n) ∈ Ω(n^a)
Para n=2π+4kπ
$\frac{n}{2}\left(2-\cos\frac{n}{2}\right) \le c \cdot n(2-\cos n)$ c≥3/2
f(n)∈ Ω(n^a)
donde a > 0
Además se cumple que ∃c<1 ∀n>n0: p·f(n/q)≤c·f(n)
Hay casos que NO contempla el TEOREMA MAESTRO !!!
$T(n) = T\left(\frac{n}{2}\right) + n(2-\cos n)$
T(n)∈ Θ(f(n))
3. Teorema Maestro
1
2
3