Chapter 1.4
Análisis y Diseño de Algoritmos (Algorítmica III)
-División y Conquista I (Recurrencias)-

Profesores:
Herminio Paucar.
Luis Guerra.

Contenido

- Algoritmos Recursivos
- Métodos de Recursión
    - Iteración
    - Sustitución
    - Árbol de recursión
    - Teorema del maestro

Las recurrencias y el Tiempo de Ejecución

- Una ecuación o inecuación que describe una función en términos de su valor sobre pequeñas entradas

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

- **Las recurrencias aparecen cuando un algoritmo contiene llamadas recursivas a sí mismo**
- ¿Cómo conocer el tiempo de ejecución de un algoritmo de estas características?
  - Es necesario resolver la recurrencia
    - Encontrar una fórmula explícita de una expresión
    - Limitar la recurrencia por una expresión que implique **n**

Ejemplos de recurrencias

- **$T(n) = T(n-1) + n$ $\Theta(n^2)$**

  Algoritmo recursivo que en cada loop examina la entrada y elimina un elemento

- **$T(n) = T(n/2) + c$ $\Theta(\lg n)$**

  Algoritmo recursivo que divide la entrada en cada paso

- **$T(n) = T(n/2) + n$ $\Theta(n)$**

  Algoritmo recursivo que divide la entrada, pero necesita examinar cada elemento en la entrada

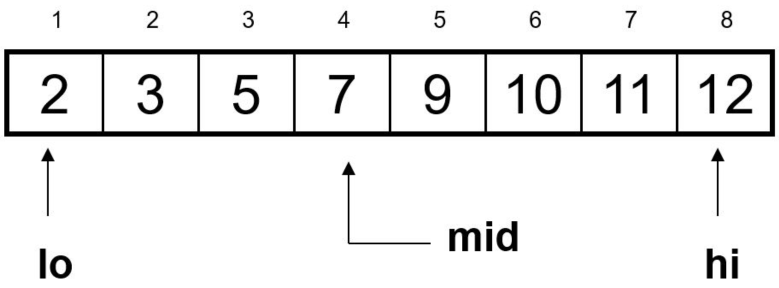
- **$T(n) = 2T(n/2) + 1$ $\Theta(n)$**

  Algoritmo recursivo que divide la entrada en dos mitades y ejecuta una cantidad constante de operaciones

Algoritmos recursivos (BINARY-SEARCH)

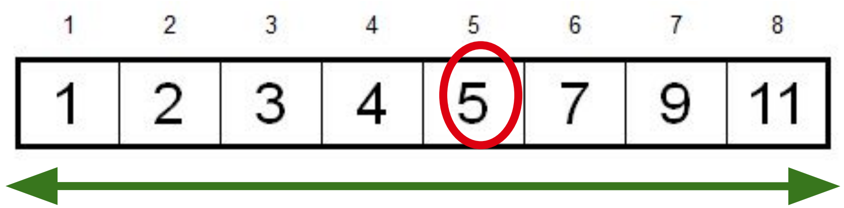
Para un vector ordenado A, verificar si x pertenece al vector A [lo...hi]

```
BINARY-SEARCH(A, lo, hi, x)
    if(lo > hi)
        return FALSE
    mid← [(lo + hi) / 2]
    if x = A[mid]
        return TRUE
    if (X < A[mid])
        BINARY-SEARCH(A, lo, mid-1, x)
    if (X > A [mid])
        BINARY-SEARCH(A, mid+1, hi, x)
```

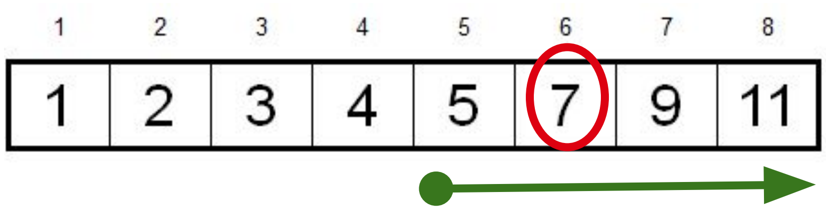


Ejemplo

A[8] = {1, 2, 3, 4, 5, 7, 9, 11}, lo = 1, hi = 8, x = 7



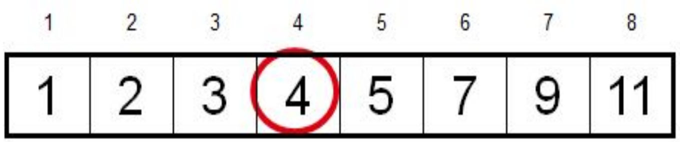
mid = 4, lo = 5, hi = 8



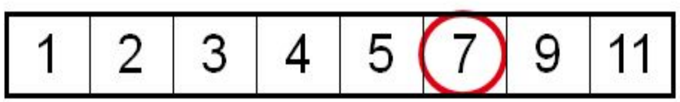
mid = 6, A[mid] = x
Encontrado!!!

Otro ejemplo

- A[8] = {1, 2, 3, 4, 5, 7, 9, 11}, lo = 1, hi = 8, x = 6



mid = 4, lo = 5, hi = 8



mid = 6, A[6] = 7, lo = 5, hi = 5

mid = 5, A[5] = 5, lo = 6, hi = 5

No encontrado!

Búsqueda binaria-Análisis

BINARY-SEARCH(A, lo, hi, x)
    if(lo > hi) ← tiempo constante: c_1
        return FALSE
    mid← [(lo + hi) / 2] ← tiempo constante: c_2
    if x = A[mid] ← tiempo constante: c_3
        return TRUE
    if (X < A[mid])
        BINARY-SEARCH(A, lo, mid-1, x) ← mismo problema de tamaño n/2
    if (X > A [mid])
        BINARY-SEARCH(A, mid+1, hi, x) ← mismo problema de tamaño n/2

- $T(n) = c + T(n/2)$
  - T(n): El tiempo de ejecución para un vector de longitud ***n***

Métodos para resolver las recurrencias

- Iteración
- Sustitución
- Árbol de recursión
- Teorema del maestro

El método de iteración

- Convertir la recurrencia en una suma y tratar de limitarlo usando una serie conocida
  - Iterar la recurrencia hasta alcanzar la condición inicial.
  - Usar la retro-Sustitución para expresar la recurrencia en términos de n y la condición inicial.

El método de iteración

$T(n) = c + T(n/2)$

$$
\begin{aligned}
T(n) &= c + T(n/2) \\
&= c + c + T(n/4) \\
&= c + c + c + T(n/8)
\end{aligned}
$$

$T(n/2) = c + T(n/4)$
$T(n/4) = c + T(n/8)$
...
$T(n/2^k) = c + T(1)$

Assume $(n = 2^k)$

$$
\begin{aligned}
T(n) &= \underbrace{c + c + \ldots + c}_{\text{k veces}} + T(1) \\
&= c\lg n + T(1) \\
&= \Theta(\lg n)
\end{aligned}
$$

Procedimiento del ejemplo Iteración

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

$$
\begin{aligned}
T(n) &= n + 2T(n/2) \\
&= n + 2(n/2 + 2T(n/4)) \\
&= n + n + 4T(n/4) \\
&= n + n + 4(n/4 + 2T(n/8)) \\
&= \quad \ldots \\
&= \underbrace{n + n + n +}_{\text{k veces}} \; 2^k T(n/2^k) \\
&= kn + 2^k T(1) \\
&= n\lg n + nT(1) = \boldsymbol{\Theta(n\lg n)}
\end{aligned}
$$

Asume: $n = 2^k$

$T(n/2) = n/2 + 2T(n/4)$

El método de sustitución

1. Adivine una solución
2. Utilizar la inducción para demostrar que la solución es correcta

Método de sustitución

- Adivinar una solución (una sola vez)
  - $T(n) = O(g(n))$
  - Propósito de inducción: aplicar la definición de la notación asintótica.
    - $T(n) \leq d\, g(n)$ para algunos $d>0$ e $n \geq n_0$
  - Hipótesis de inducción: $T(k) \leq d\, g(k)$ para toda $k<n$
- Pruebe la inducción
  - Use la **hipótesis inductiva** para encontrar algunos valores de constantes d y n_0 para los que la inducción es válida.

Ejemplo: búsqueda binaria

$T(n) = c + T(n/2)$

- Chute: $T(n) = O(\lg n)$
  - Inducción: $T(n) \le d \cdot \lg n$, para algún d y $n \ge n_0$
  - Hipótesis inductiva: $T(n/2) \le d \cdot \lg(n/2)$
- Prueba de la inducción:

$$T(n) = T(n/2) + c \le d \cdot \lg(n/2) + c$$

$$= d \cdot \lg n - d + c \le d \cdot \lg n$$

si: $-d + c \le 0$, $d \ge c$

- Caso base?

Ejemplo 2

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

- Chute: $T(n) = O(n^2)$
  - Inducción: $T(n) \leq cn^2$, para algún c y $n \geq n_0$
  - Hipótesis inductiva: $T(n-1) \leq c(n-1)^2$ para todo $k < n$
- Prueba de la inducción:

$$T(n) = T(n-1)+n \leq c(n-1)^2 + n$$

$$= cn^2 - (2cn-c-n) \leq cn^2$$

si: $2cn - c - n \geq 0 \leftrightarrow c \geq n/(2n-1) \leftrightarrow c \geq 1/(2 - 1/n)$

  - **Para $n \geq 1 \Rightarrow 2 - 1/n \geq 1$** $\Rightarrow$ qualquer $c \geq 1$ irá satisfazer.

Ejemplo 3

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

- Chute: $T(n) = O(n \lg n)$
  - Inducción: $T(n) \leq cn \lg n$ para algun c y $n \geq n_0$
  - hipótesis de inducción : $T(n/2) \leq cn/2 \lg(n/2)$
- Prueba de inducción:

$$T(n) = 2T(n/2) + n \leq 2c\,(n/2)\lg(n/2) + n$$
$$= cn \cdot \lg n - cn + n \leq cn \cdot \lg n$$
$$\text{si: } -cn + n \leq 0 \Rightarrow \mathbf{c \geq 1}$$

- caso base?

Cambio de variables

$$T(n) = 2T(\sqrt{n}) + \log n$$

- Hacemos: $m = \lg n \Rightarrow n = 2^m$

  $T(2^m) = 2T(2^{m/2}) + m$

- Tomando: $S(m) = T(2^m)$

  $S(m) = 2S(m/2) + m$

  $S(m) = O(m \lg m)$ (Como se ha visto anteriormente)

$T(n) = T(2^m) = S(m) = O(m \lg m) = O(\lg n \lg \lg n)$

Idea: transformar la recurrencia en una que es conocida

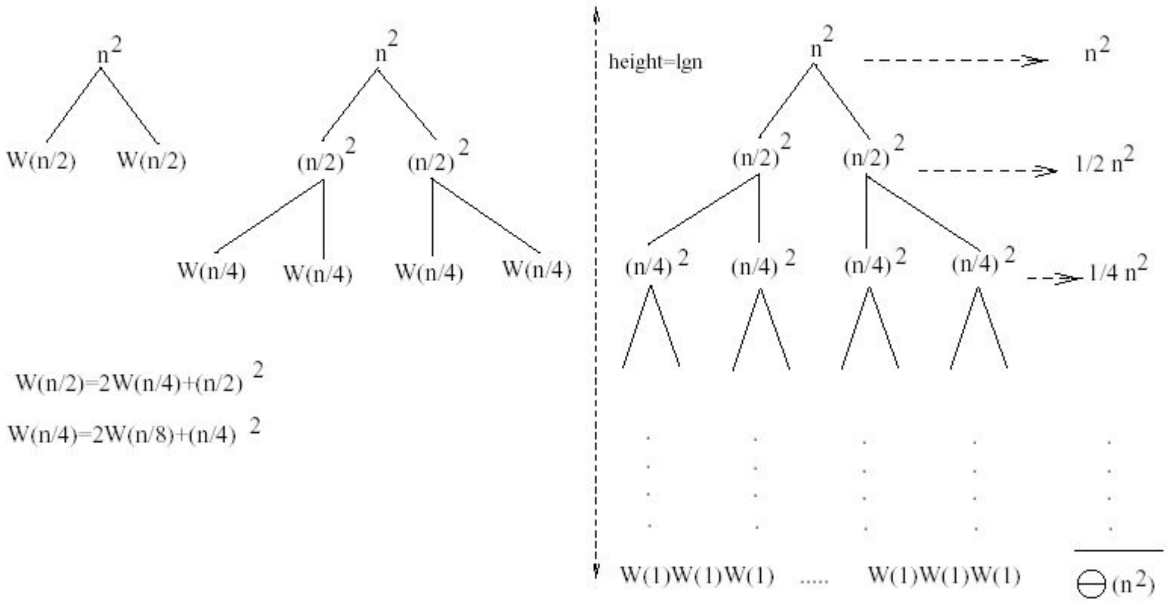
El método de árbol de recursión

- Convertir la recurrencia en un árbol:
    - Cada nodo representa el costo incurrido en los varios niveles de recursión
    - Sumar los costos de todos los niveles

Se utiliza para "adivinar" una solución a la recurrencia

Ejemplo 1

W(n) = 2W(n/2) + n^2

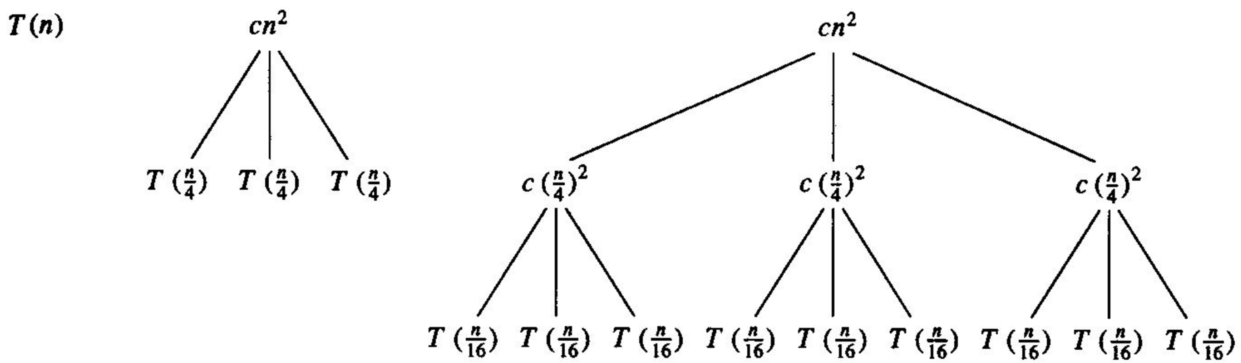


- Tamaño del nivel sub-problema i que es: $n/2^i$
- tamaño Subproblema alcanza 1 cuando 1 = $n/2^i$ ⇒ lgn
- Costo de un problema en el nivel i = $(n/2^i)^2$
- Número de nodos en el nivel i = 2^i
- Costo total: $$W(n)=\sum_{i=0}^{\lg n-1}\frac{n^2}{2^i}+2^{\lg n}W(1)=n^2\sum_{i=0}^{\lg n-1}\left(\frac{1}{2}\right)^i+n\le n^2\sum_{i=0}^{\infty}\left(\frac{1}{2}\right)^i+O(n)=n^2\frac{1}{1-\frac{1}{2}}+O(n)=2n^2$$

  ⇒ W (n) = (n2)

Ejemplo 2

$T(n) = 3T(n/4) + cn^2$



- Tamaño del nivel sub-problema ***i*** que es: $n/4^i$
- Tamaño subproblema alcanza 1 cuando $1 = n/4^i \Rightarrow i = \log_4 n$
- El costo de un nodo en el nivel ***i*** $= c(n/4^i)^2$
- Número de nodos en el nivel ***i*** $= 3^i \Rightarrow$ último nivel tiene $3^{\log_4 n} = n^{\log_4 3}$ nodos
- Costo total:

$$T(n) = \sum_{i=0}^{\log_4 n-1} \left(\frac{3}{16}\right)^i cn^2 + \Theta\left(n^{\log_4 3}\right) \leq \sum_{i=0}^{\infty} \left(\frac{3}{16}\right)^i cn^2 + \Theta\left(n^{\log_4 3}\right) = \frac{1}{1-\frac{3}{16}} cn^2 + \Theta\left(n^{\log_4 3}\right) = O(n^2)$$

- $\Rightarrow T(n) = O(n^2)$

Ejemplo 2 – Sustitución

$T(n) = 3T(n/4) + cn^2$

- Chute: $T(n) = O(n^2)$
  - Inducción: $T(n) \leq dn^2$ para algún d y $n \geq n_0$
  - hipótesis de inducción: $T(n/4) \leq d(n/4)^2$
- Prueba de inducción:

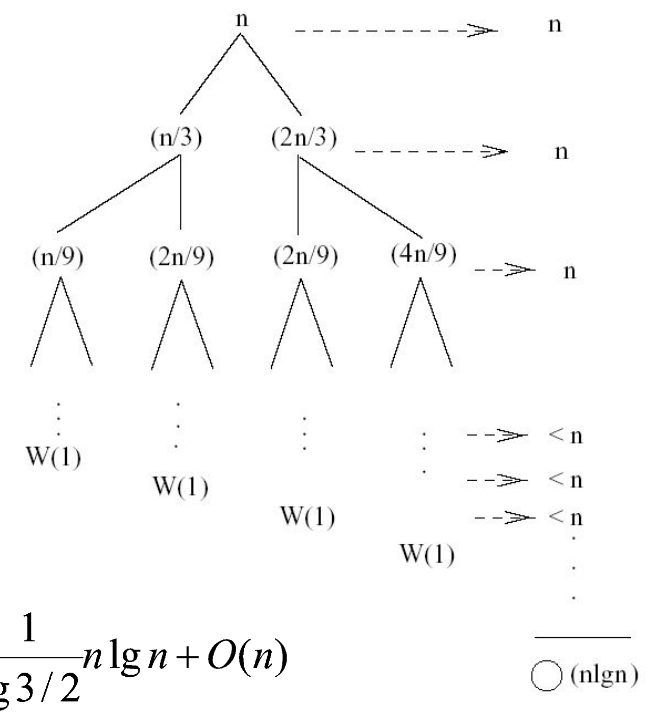
$$
\begin{aligned}
T(n) &= 3T(n/4) + cn^2 \\
&\leq 3d\,(n/4)^2 + cn^2 \\
&= (3/16)\,dn^2 + cn^2 \\
&\leq dn^2 \qquad \textbf{si: } d \geq (16/13)c
\end{aligned}
$$

- Por lo tanto: **$T(n) = (n^2)$**

Ejemplo 3

W(n) = W(n/3) + W(2n/3) + n

- El camino más largo desde la raíz a una hoja es:
  $n \to (2/3)\,n \to (2/3)^2\,n \to \ldots \to 1$
- tamaño Subproblema alcanza 1 cuando
  $1 = (2/3)^i n \Leftrightarrow i = \log_{3/2} n$
- Costo de un problema en el nivel i = n
- Costo total:



$$W(n) < n + n + \ldots = \sum_{i=0}^{(\log_{3/2} n)-1} n + 2^{(\log_{3/2} n)} W(1) <$$

$$< n \sum_{i=0}^{\log_{3/2} n} 1 + n^{\log_{3/2} 2} = n \log_{3/2} n + O(n) = n \frac{\lg n}{\lg 3/2} + O(n) = \frac{1}{\lg 3/2} n \lg n + O(n)$$

⇒ W(n) = O(nlgn)

Ejemplo 3 – Sustitución

$$W(n) = W(n/3) + W(2n/3) + O(n)$$

- Chute: $W(n) = O(n\lg n)$
  - Inducción: $W(n) \leq d\ n\lg n$ para algunos d y n $n \geq n_0$
  - Hipótesis de inducción: $W(k) \leq d\ k\lg k$ para cualquier $K<n(n/3, 2n/3)$
- Prueba de inducción:

  Queda como ejercicio!

- **$T(n) = O(n\lg n)$**

Teorema del Maestro

- "Receta de la torta" para resolver las recurrencias de la forma:

$$\boxed{T(n) = aT\left(\frac{n}{b}\right) + f(n)}$$

donde, $a \geq 1$, $b > 1$ y $f(n) > 0$

idea: comparar f(n) con $\boldsymbol{n^{\log_b a}}$

- f(n) Es asintóticamente menor o mayor que $\boldsymbol{n^{\log_b a}}$ por un factor polinomial $\boldsymbol{n^{\varepsilon}}$
- f(n) es asintóticamente igual a $\boldsymbol{n^{\log_b a}}$

Teorema maestro

"Receta de la torta" para resolver las recurrencias de la forma:

$$\boxed{T(n) = aT\left(\frac{n}{b}\right) + f(n)}$$

donde, $a \geq 1$, $b > 1$ y $f(n) > 0$

Caso 1: si $f(n) = O(n^{\log_b a - \varepsilon})$ para algún $\varepsilon > 0$ entonces: $T(n) = \Theta(n^{\log_b a})$

Caso 2: si $f(n) = \Theta(n^{\log_b a})$, entonces: $T(n) = \Theta(n^{\log_b a} \lg n)$

Caso 3: si $f(n) = \Omega(n^{\log_b a + \varepsilon})$ para algún $\boldsymbol{\varepsilon > 0}$, y si $\boldsymbol{f(n/b) \leq cf(n)}$ para algunos $\boldsymbol{c < 1}$ y para todo $\boldsymbol{n}$ suficientemente grande, entonces: $T(n) = \Theta(f(n))$

¿Por qué $n^{\log_b a}$?

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

$$\underbrace{T(n) = aT\left(\frac{n}{b}\right)}$$
$$\underbrace{a^2T\left(\frac{n}{b^2}\right)}$$
$$a^3T\left(\frac{n}{b^3}\right)$$
$$\vdots$$
$$T(n) = a^iT\left(\frac{n}{b^i}\right) \quad \forall i$$

- tomando $n = b^k \Rightarrow k = \log_b n$
- Al final de la iteración $i = k$:

- **Caso 1:**
  - si f (n) Está dominado por $n^{\log_b a}$:
    - $T(n) = \Theta(n^{\log_b n})$
- **Caso 3:**
  - si f (n) domina $n^{\log_b a}$:
    - $T(n) = \Theta(f(n))$
- **Caso 2:**
  - si $f(n) = \Theta(n^{\log_b a})$:
    - $T(n) = \Theta(n^{\log_b a} \log n)$

$$T(n) = a^{\log_b n}T\left(\frac{b^i}{b^i}\right) = a^{\log_b n}T(1) = \Theta\left(a^{\log_b n}\right) = \Theta\left(n^{\log_b a}\right)$$

Ejemplos

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

$a = 2$, $b = 2$, $\log_2 2 = 1$

comparar $n^{\log_2 2}$ con $f(n) = n$

$\Rightarrow f(n) = \Theta(n) \Rightarrow$ caso 2

$\Rightarrow T(n) = \Theta(n\lg n)$

Ejemplos

$$T(n) = 2T(n/2) + n^2$$

$a = 2$, $b = 2$, $\log_2 2 = 1$

comparar n con $f(n) = n^2$

$\Rightarrow$ **$f(n) = \Omega(n^{1+\varepsilon})$ caso 3** $\Rightarrow$ comprobar la condición de regularidad

$$\mathbf{f\,(n/b) \leq cf(n)}$$

$\Leftrightarrow 2n^2/4 \leq cn^2 \Rightarrow c = ½$ Es una solución $(c<1)$

$\Rightarrow$ **$T(n) = \Theta(n^2)$**

Ejemplos (cont.)

$$T(n) = 2T(n/2) + \sqrt{n}$$

a = 2, b = 2, $\log_2 2 = 1$

comparar n con $f(n) = n^{1/2}$

$\Rightarrow f(n) = (n^{1-\varepsilon})$ caso 1

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

Ejemplos

$$T(n) = 3T(n/4) + n\lg n$$

$a = 3$, $b = 4$, $\log_4 3 = 0{,}793$

comparar $n^{0{,}793}$ con $f(n) = n\lg n$

$f(n) = \Omega(n^{\log_4 3 + \varepsilon})$ caso 3

Verificando la condición de regularidad:

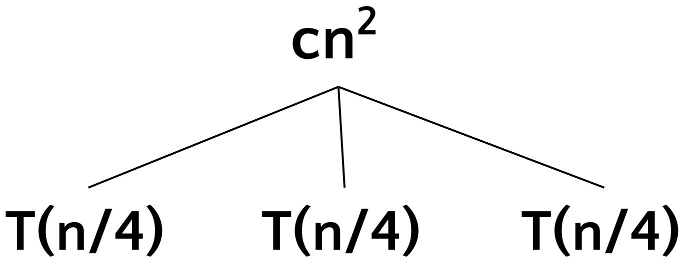
$$3*(n/4)\lg(n/4) \leq (3/4)n\lg n = c*f(n),\ c = 3/4$$

$\Rightarrow T(n) = \Theta(n\lg n)$

Àrbol de recurrencia

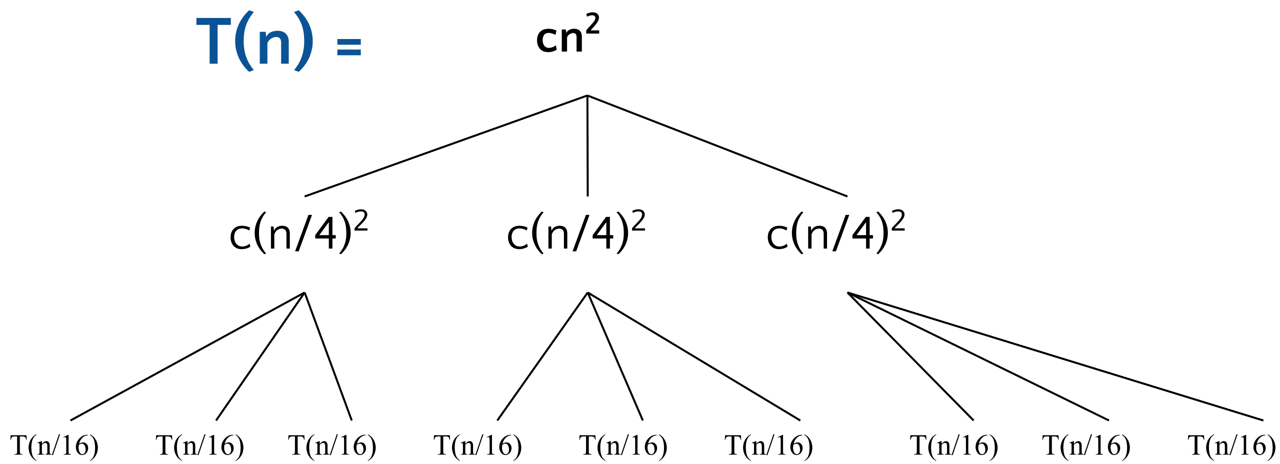
Sea $T(n) = 3T(n/4) + cn^2$

$T(n) =$



Árbol de recurrencia

Sea: $T(n) = 3T(n/4) + cn^2$



Árbol de recurrencia

Sea: $T(n) = 3T(n/4) + cn^2$

$T(n) =$