Ecuaciones de Recurrencia la Complejidad de los Algoritmos Divide y Vencerás

José Ortiz Bejar

Facultad de Ingeniería Eléctrica
Universidad Michoacana de San Nicolás de Hidalgo

Marzo 7, 2022

Contenido

Análisis de Complejidad

Ecuación de recurrencia

Teorema maestro

Árbol de recurrencia

Complejidad Divide y Vencerás

Solución de Recurrencias

Análisis de Complejidad

Análisis de Complejidad

- **Merge-Sort**

- **Merge-Sort**
  - dividir una secuencia *grande* de tamaño n
  - en 2 sub-secuencias de tamaño $n/2$
  - después combina los resultados parciales en $\Theta(n)$

- **Merge-Sort**
  - dividir una secuencia *grande* de tamaño n
  - en 2 sub-secuencias de tamaño $n/2$
  - después combina los resultados parciales en $\Theta(n)$

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

- **Merge-Sort**
  - dividir una secuencia *grande* de tamaño n
  - en 2 sub-secuencias de tamaño $n/2$
  - después combina los resultados parciales en $\Theta(n)$

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

- Inferimos que la complejidad de **MergeSort** es $\Theta(n \log n)$

- **Merge-Sort**
  - dividir una secuencia *grande* de tamaño n
  - en 2 sub-secuencias de tamaño $n/2$
  - después combina los resultados parciales en $\Theta(n)$

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

- Inferimos que la complejidad de **MergeSort** es $\Theta(n \log n)$
  - ¿Es correcto?
  - ¿Podemo generalizar?

Recurrencia

- *Algoritmo divide y vencerás genérico*

Recurrencia

- *Algoritmo divide y vencerás genérico Caso base:*
  - resolver un problema P de tamaño ***1*** directamente, en $\Theta(1)$ pasos

Recurrencia

- *Algoritmo divide y vencerás genérico Caso base:*
  - resolver un problema P de tamaño ***1*** directamente, en $\Theta(1)$ pasos

  Caso recursivo:
  - dividir un problema P de tamaño $n > 1$

Recurrencia

- *Algoritmo divide y vencerás genérico Caso base:*
  - resolver un problema P de tamaño $\mathbf{1}$ directamente, en $\Theta(1)$ pasos

  Caso recursivo:
  - dividir un problema P de tamaño $n > 1$
  - en a sub-problemas del mismo tipo que P

Recurrencia

- *Algoritmo divide y vencerás genérico Caso base:*
  - resolver un problema P de tamaño ***1*** directamente, en $\Theta(1)$ pasos

  Caso recursivo:
  - dividir un problema P de tamaño $n > 1$
  - en a sub-problemas del mismo tipo que P
  - cada sub-problema de tamaño n/b

Recurrencia

- *Algoritmo divide y vencerás genérico Caso base:*
  - resolver un problema P de tamaño ***1*** directamente, en $\Theta(1)$ pasos

  Caso recursivo:
  - dividir un problema P de tamaño $n > 1$
  - en a sub-problemas del mismo tipo que P
  - cada sub-problema de tamaño n/b
  - combinar la solución de los subproblemas, en $f(n)$ pasos

Recurrencia

- *Algoritmo divide y vencerás genérico Caso base:*
  - resolver un problema P de tamaño ***1*** directamente, en $\Theta(1)$ pasos

  Caso recursivo:
  - dividir un problema P de tamaño $n > 1$
  - en a sub-problemas del mismo tipo que P
  - cada sub-problema de tamaño n/b
  - combinar la solución de los subproblemas, en $f(n)$ pasos

$$T(n) = \begin{cases} \Theta(1) & \text{if } n = 1 \\ aT(n/b) + f(n) & \text{if } n > 1 \end{cases}$$

Recurrencia

- *Algoritmo divide y vencerás genérico Caso base:*
  - resolver un problema P de tamaño ***1*** directamente, en $\Theta(1)$ pasos

  Caso recursivo:
  - dividir un problema P de tamaño $n > 1$
  - en a sub-problemas del mismo tipo que P
  - cada sub-problema de tamaño n/b
  - combinar la solución de los subproblemas, en $f(n)$ pasos

$$T(n) = \begin{cases} \Theta(1) & \text{if } n = 1 \\ aT(n/b) + f(n) & \text{if } n > 1 \end{cases}$$

- El objetivo es obtener una forma cerrada para la fórmula de costo $T(n)$

Teorema maestro

Teorema maestro

- Por simplicidad asumimos $f(n) = O(n^d)$ con $d \geq 0$

Teorema maestro

- Por simplicidad asumimos $f(n) = O(n^d)$ con $d \geq 0$
- **Teorema:**

Teorema maestro

- Por simplicidad asumimos $f(n) = O(n^d)$ con $d \geq 0$
- **Teorema:**

$$T(n) = \begin{cases} \Theta(1) & \text{si } n = 1 \\ aT(n/b) + O(n^d) & \text{si } n > 1 \end{cases}$$

Teorema maestro

- Por simplicidad asumimos $f(n) = O(n^d)$ con $d \geq 0$
- **Teorema:**

$$T(n) = \begin{cases} \Theta(1) & \text{si } n = 1 \\ aT(n/b) + O(n^d) & \text{si } n > 1 \end{cases}$$

con $a > 0$, $b > 1$, $d \geq 0$ la complejidad asintótica será

$$T(n) = \begin{cases} O(n^d) & \text{si } d > \log_b a \\ O(n^d \log n) & \text{si } d = \log_b a \\ O(n^{\log_b a}) & \text{si } d < \log_b a \end{cases}$$

Árbol de recurrencia

tamaño

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

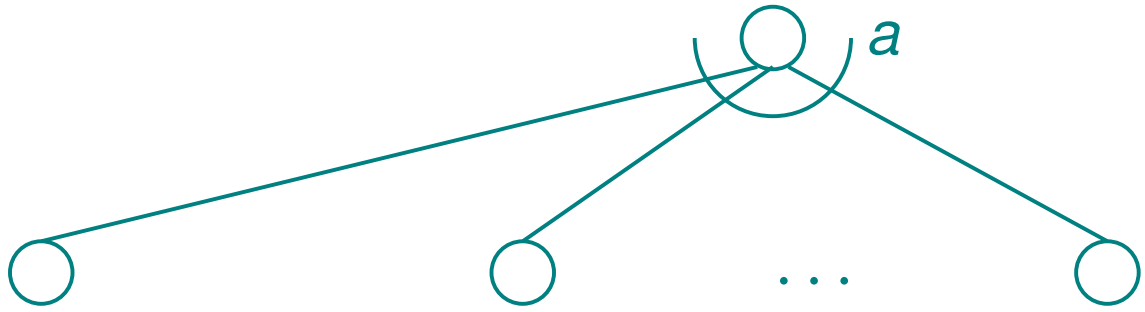
n

Árbol de recurrencia

tamaño

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

n

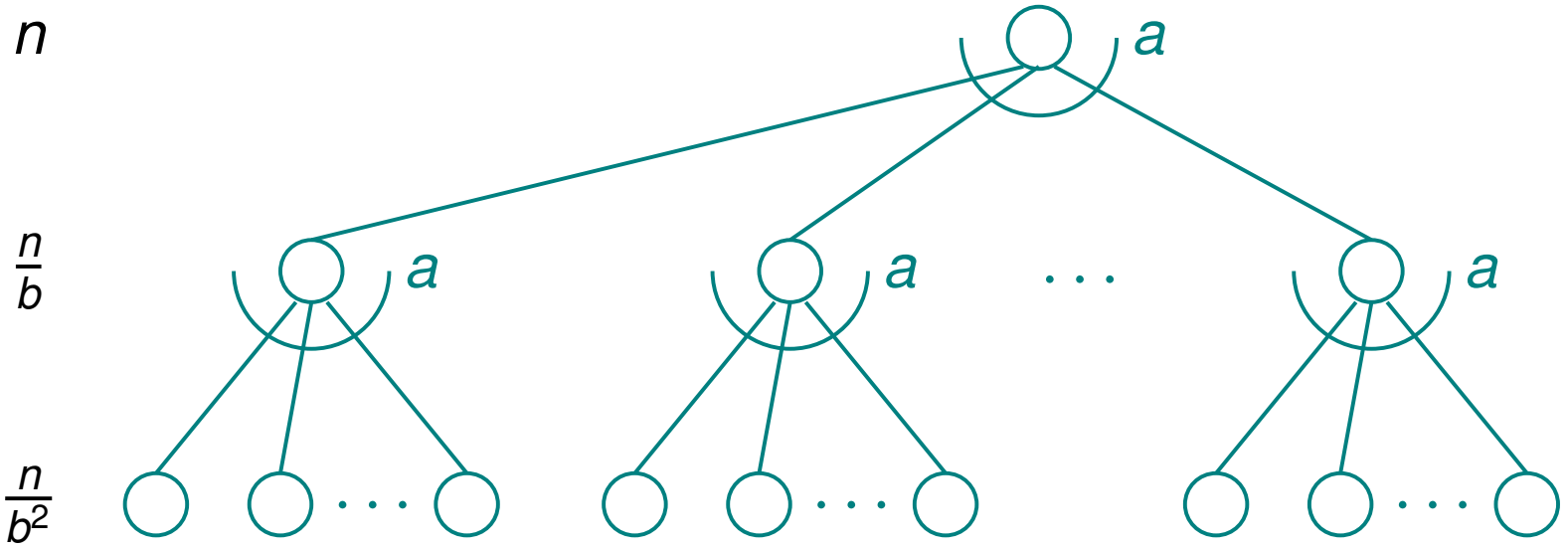


$\frac{n}{b}$

Árbol de recurrencia

tamaño

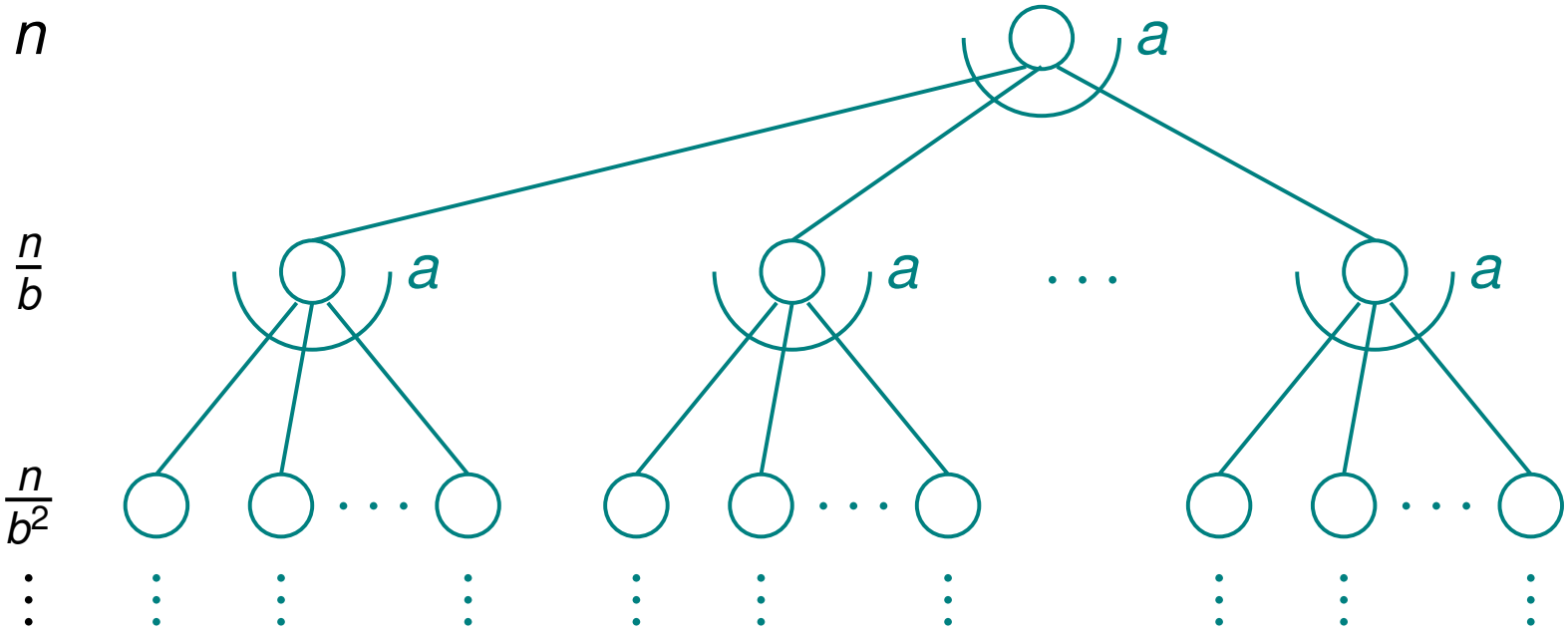
$$T(n) = aT(n/b) + f(n)$$



Árbol de recurrencia

tamaño

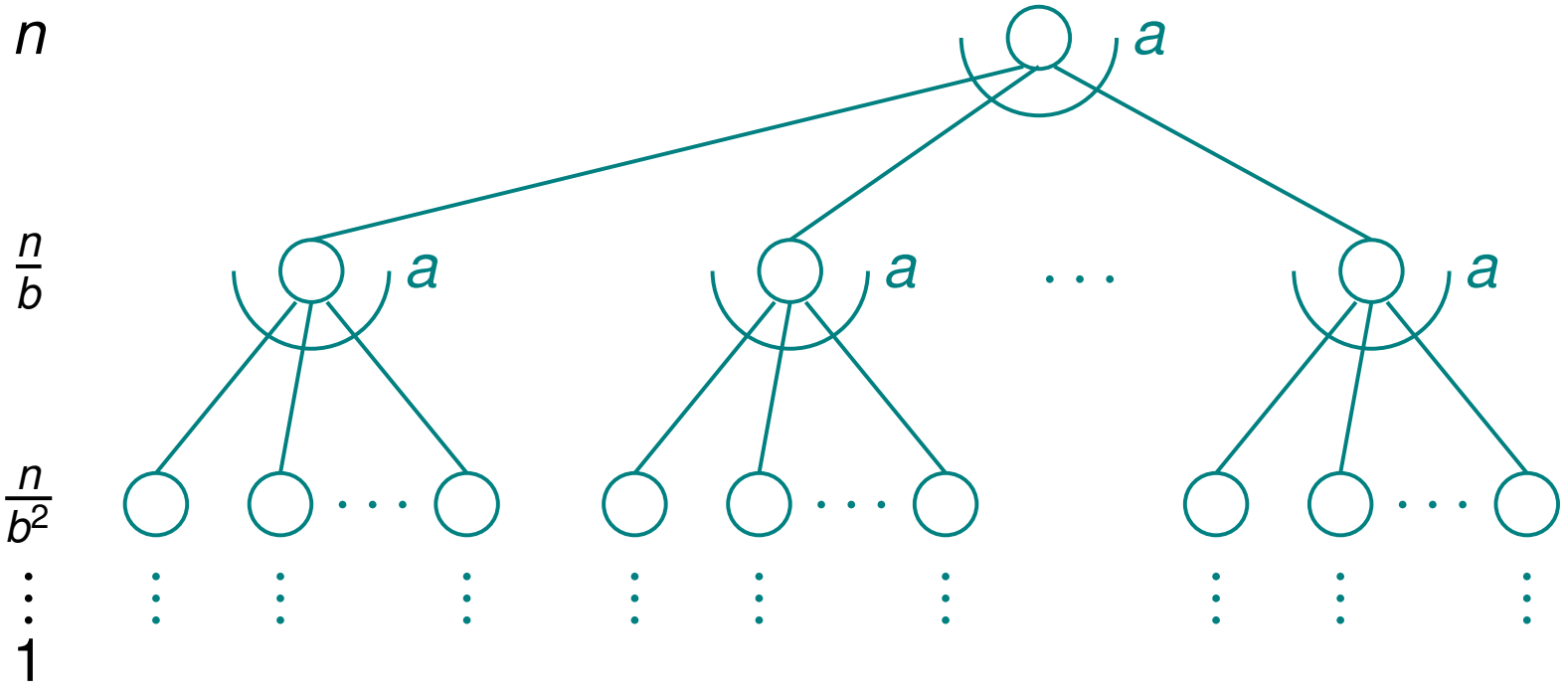
$$T(n) = aT(n/b) + f(n)$$



Árbol de recurrencia

tamaño

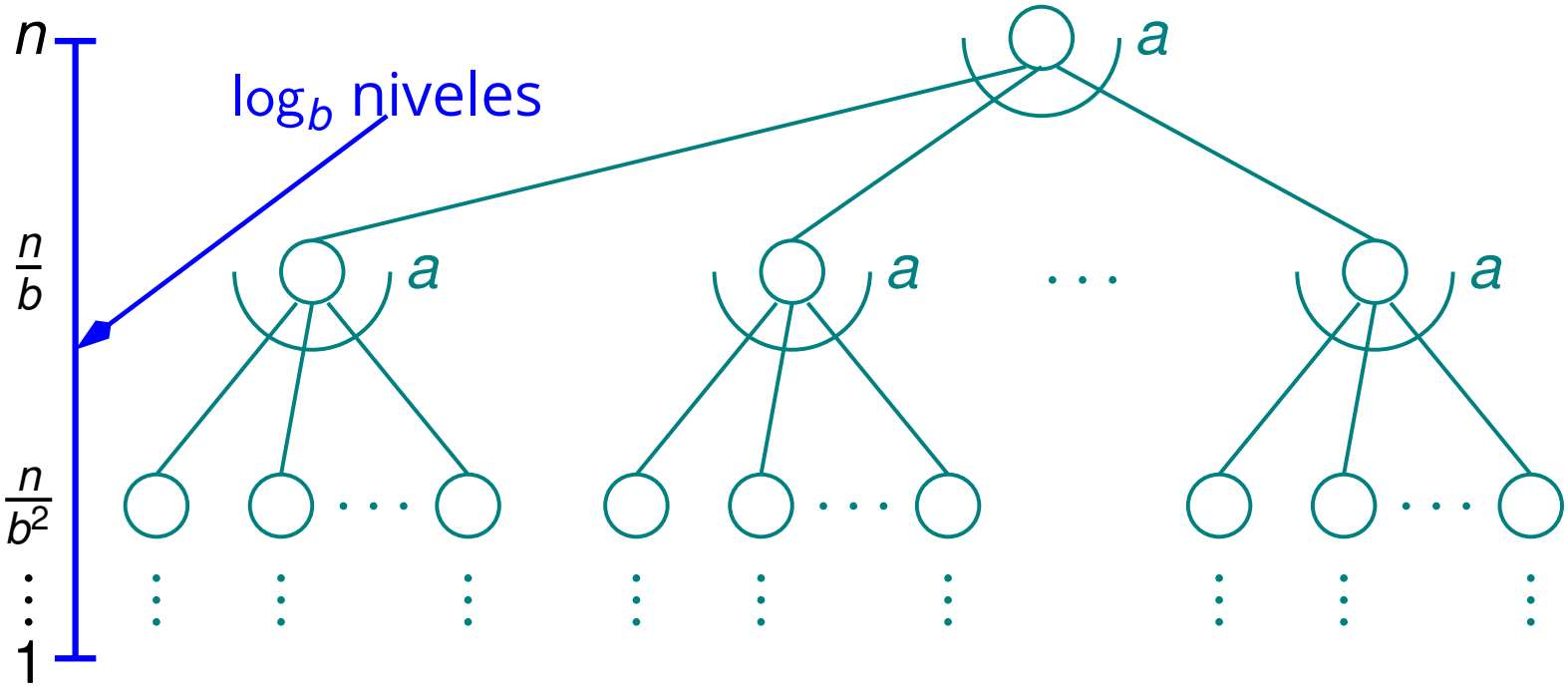
$$T(n) = aT(n/b) + f(n)$$



Árbol de recurrencia

tamaño

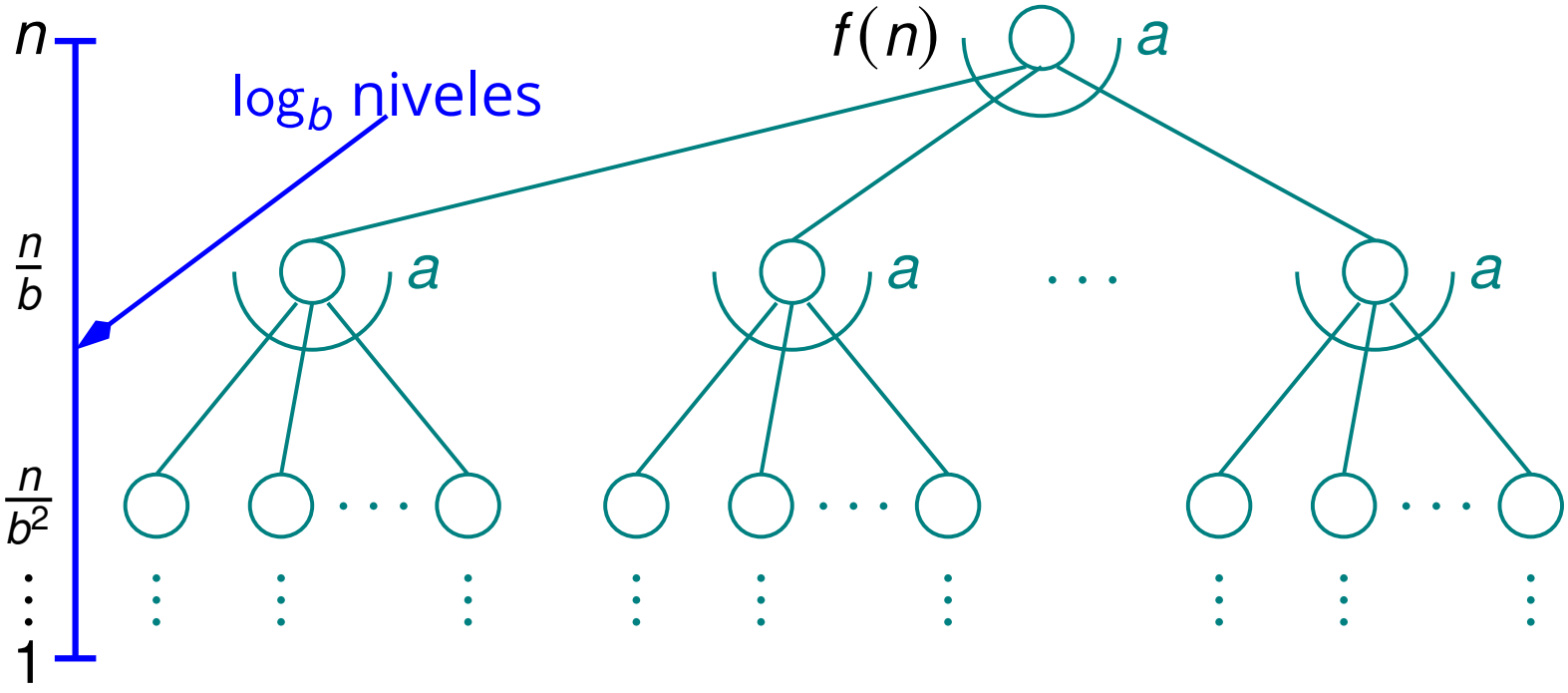
$$T(n) = aT(n/b) + f(n)$$



Árbol de recurrencia

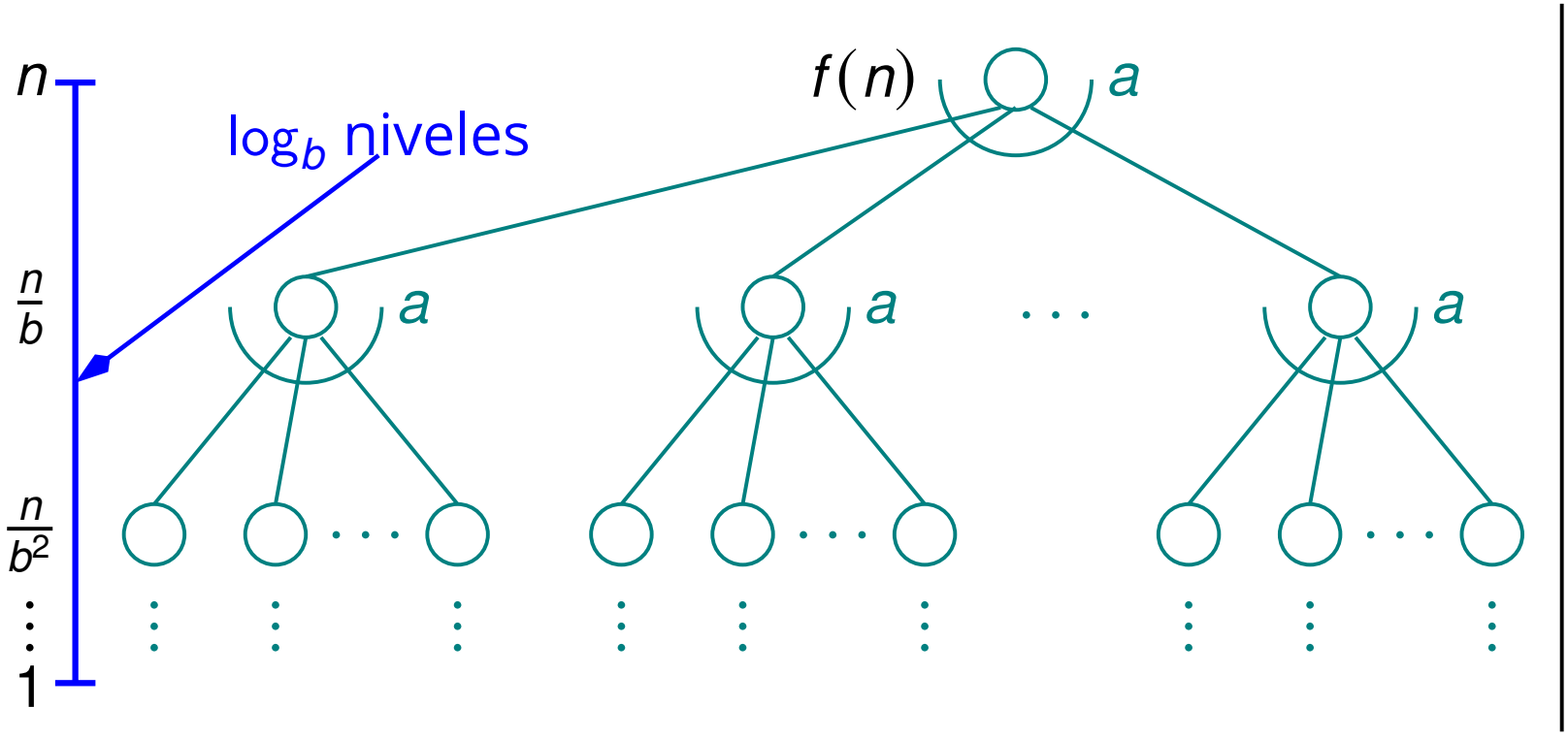
tamaño

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



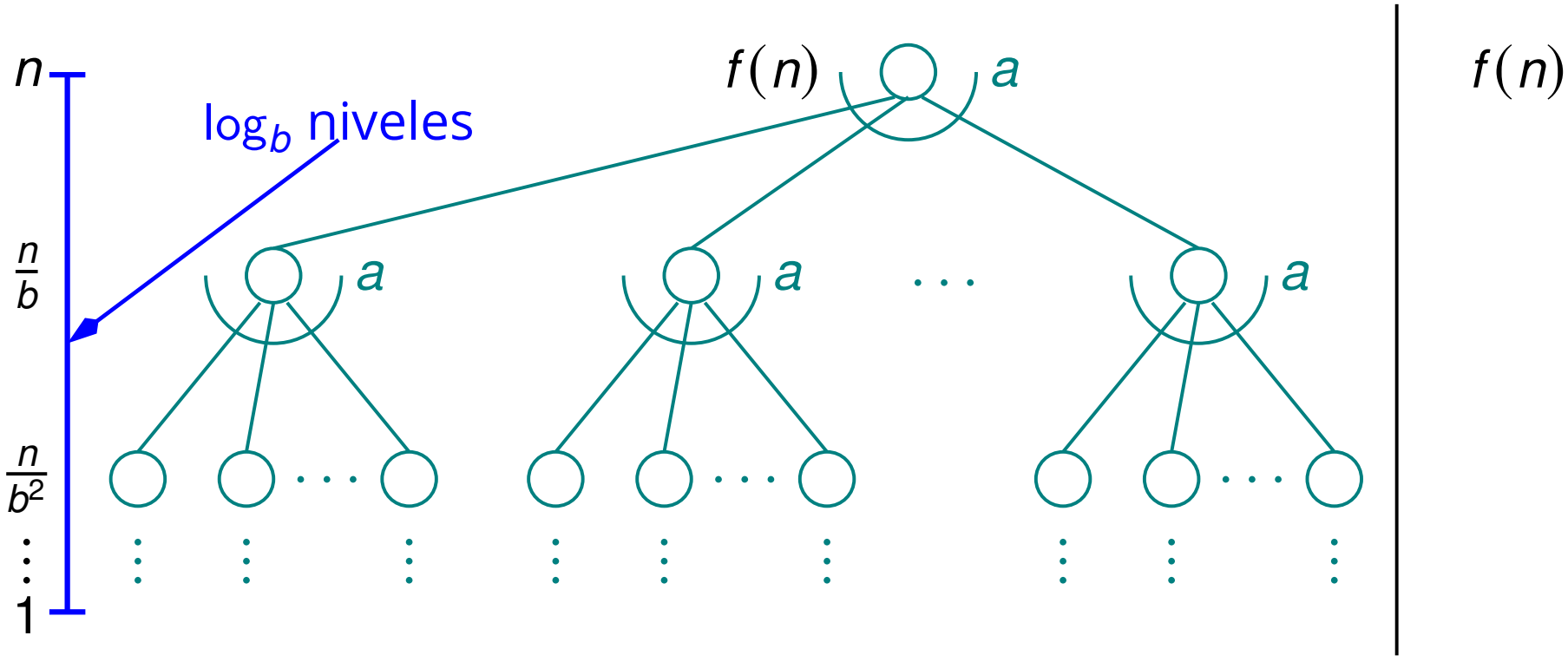
Árbol de recurrencia

tamaño $T(n) = aT(n/b) + f(n)$ costo



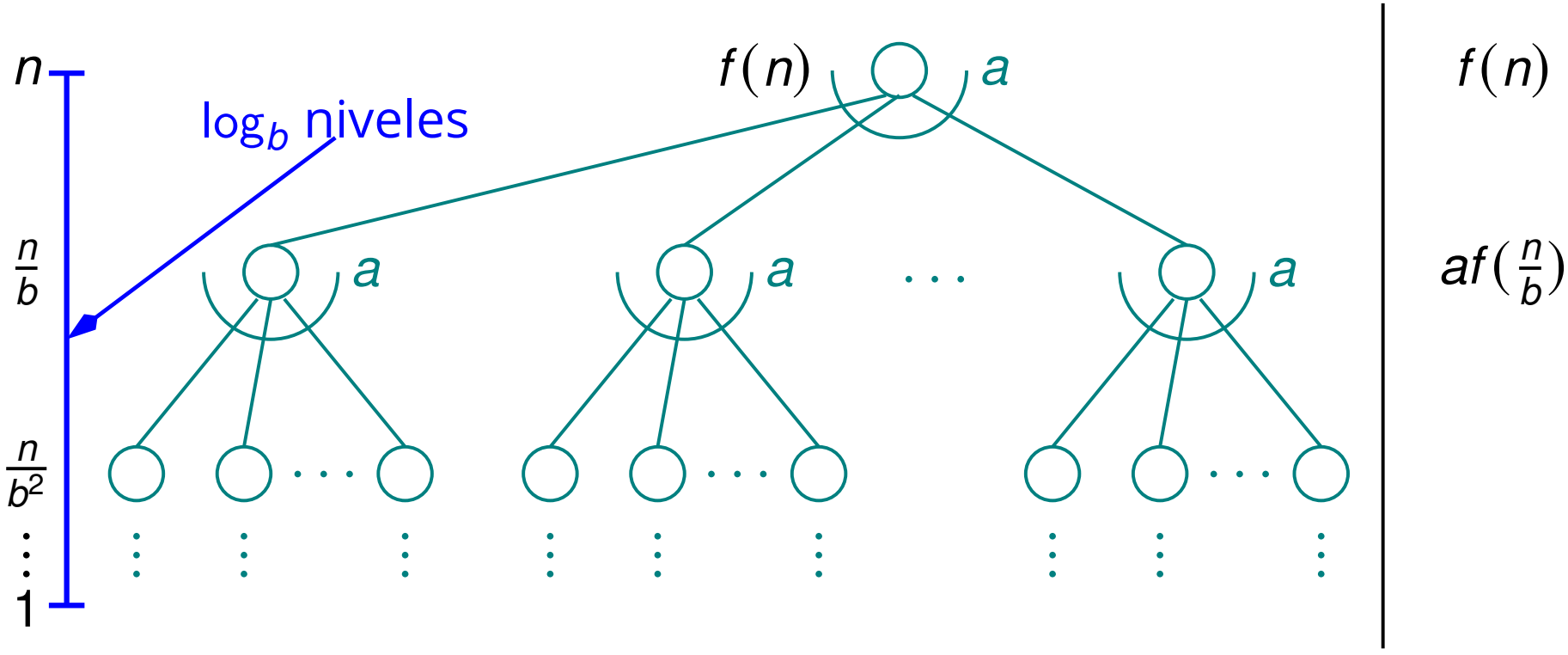
Árbol de recurrencia

tamaño $T(n) = aT(n/b) + f(n)$ costo

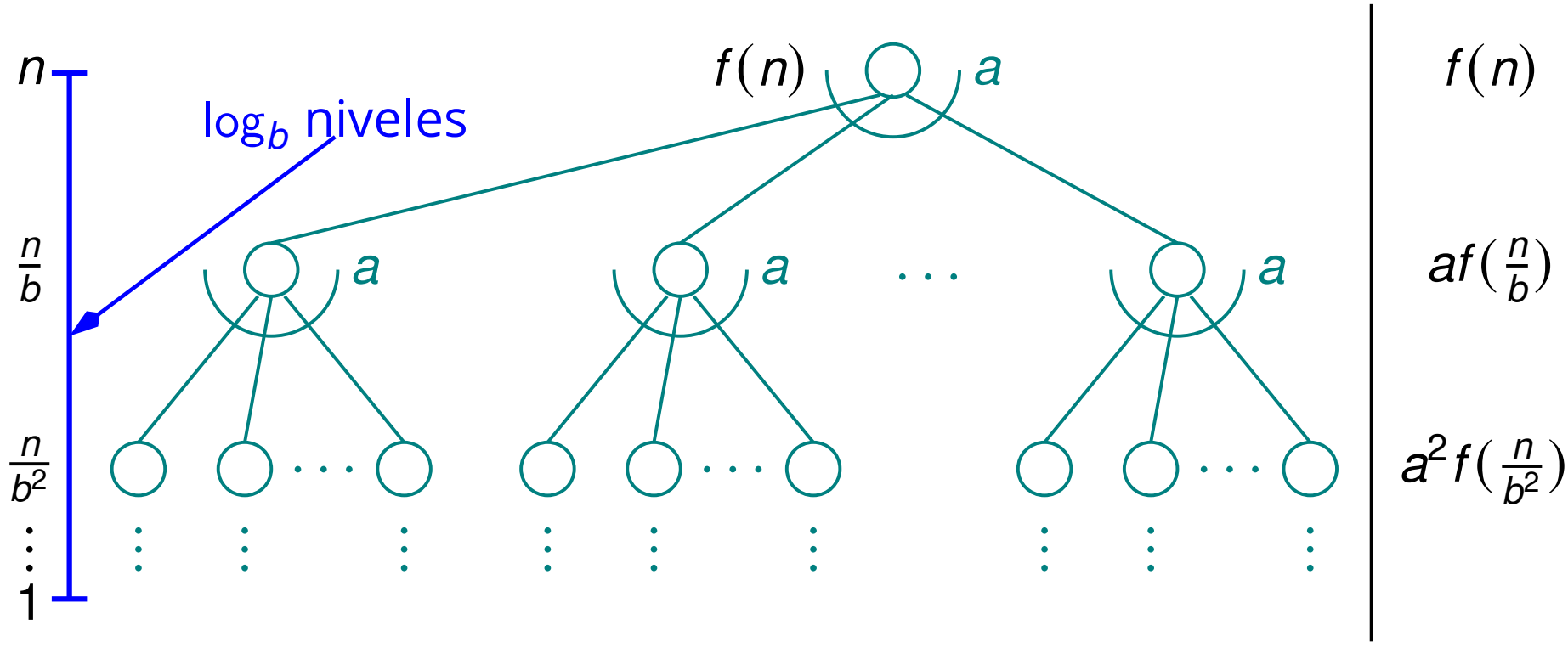


Árbol de recurrencia

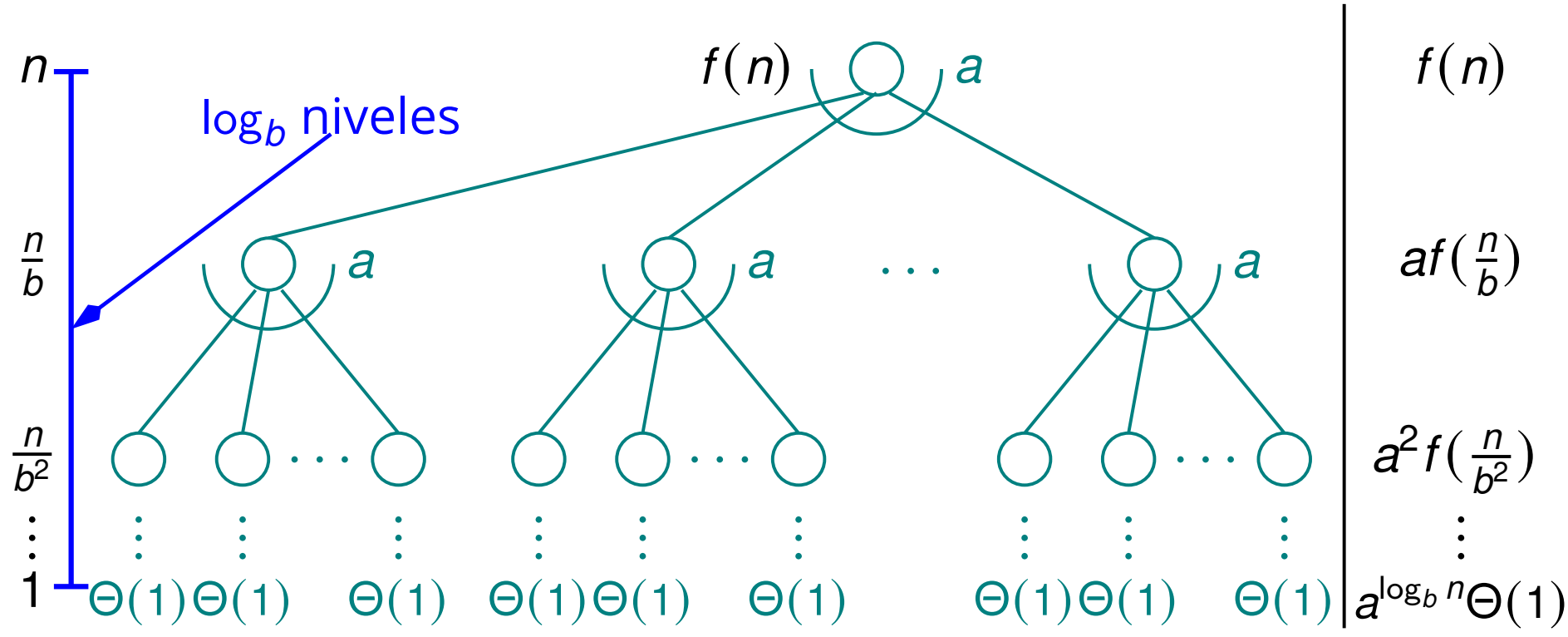
tamaño $T(n) = aT(n/b) + f(n)$ costo



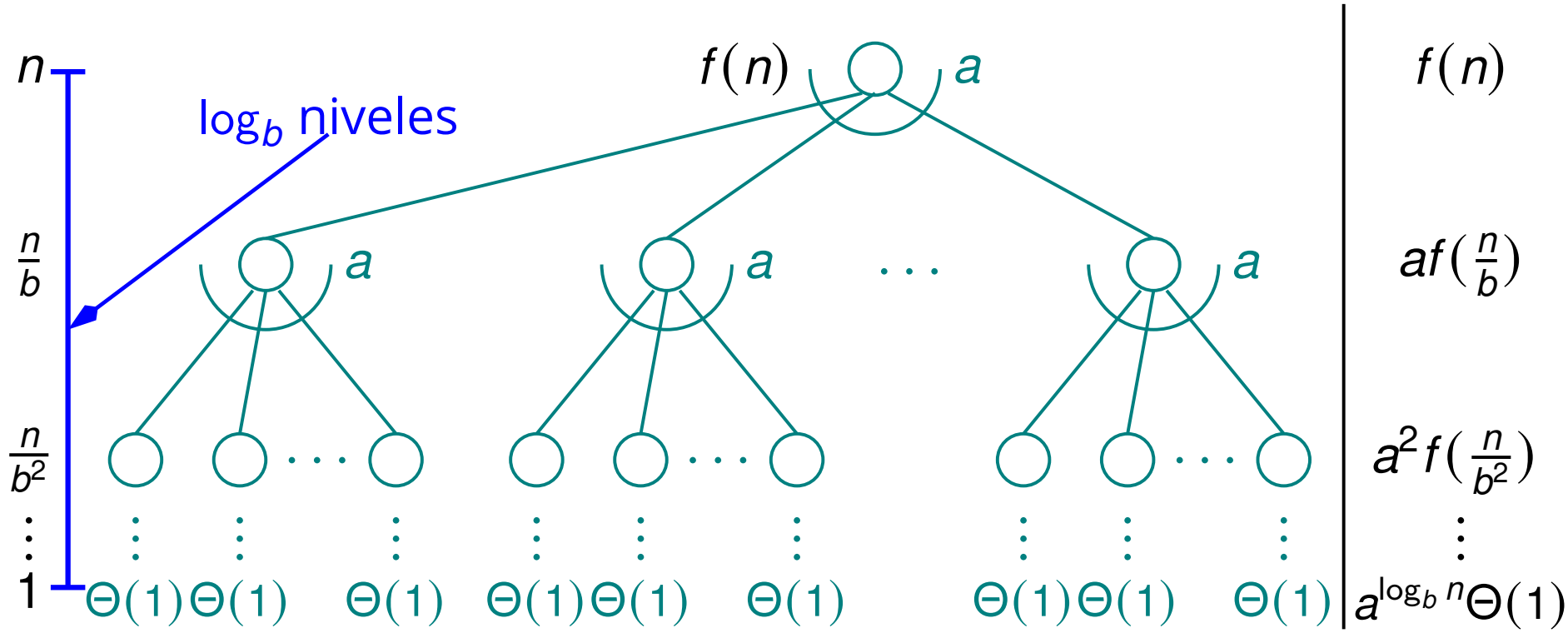
Árbol de recurrencia



Árbol de recurrencia



Árbol de recurrencia



$$T(n) = \Theta(n^{\log_b a}) + \sum_{i=0}^{\log_b n - 1} a^i f\left(\frac{n}{b^i}\right)$$

Complejidad Divide y Vencerás

$$T(n) = \Theta(n^{\log_b a}) + \sum_{i=0}^{\log_b n - 1} a^i f\left(\frac{n}{b^i}\right)$$

- Asunciones básicas

$$a \geq 1, b > 1$$

Complejidad Divide y Vencerás

$$T(n) = \Theta(n^{\log_b a}) + \sum_{i=0}^{\log_b n - 1} a^i f\left(\frac{n}{b^i}\right)$$

- Asunciones básicas

$$a \geq 1, b > 1$$

- Otras asunciones

$$f(n) = O(n^d) \text{con } d \geq 0$$

Complejidad Divide y Vencerás

$$T(n) = \Theta(n^{\log_b a}) + \sum_{i=0}^{\log_b n - 1} a^i f\left(\frac{n}{b^i}\right)$$

- Asunciones básicas

$$a \geq 1, b > 1$$

- Otras asunciones

$$f(n) = O(n^d) \text{con } d \geq 0$$

- Entonces

$$T(n) = \Theta(n^{\log_b a}) + \sum_{i=0}^{\log_b n - 1} a^i O(n^d / b^{id})$$

Complejidad Divide y Vencerás

$$T(n) = \Theta(n^{\log_b a}) + \sum_{i=0}^{\log_b n - 1} a^i f\left(\frac{n}{b^i}\right)$$

- Asunciones básicas

$$a \geq 1, b > 1$$

- Otras asunciones

$$f(n) = O(n^d) \text{con } d \geq 0$$

- Entonces

$$T(n) = \Theta(n^{\log_b a}) + \sum_{i=0}^{\log_b n - 1} a^i O(n^d / b^{id})$$

$$T(n) = \Theta(n^{\log_b a}) + \sum_{i=0}^{\log_b n - 1} \left(\frac{a}{b^d}\right)^i O(n^d)$$

Complejidad Divide y Vencerás (2)

$$T(n) = \Theta(n^{\log_b a}) + \sum_{i=0}^{\log_b n - 1} \left(\frac{a}{b^d}\right)^i O(n^d)$$

Complejidad Divide y Vencerás (2)

$$T(n) = \Theta(n^{\log_b a}) + \sum_{i=0}^{\log_b n - 1} \left(\frac{a}{b^d}\right)^i O(n^d)$$

$$T(n) = \Theta(n^{\log_b a}) + O(n^d) \sum_{i=0}^{\log_b n - 1} \left(\frac{a}{b^d}\right)^i$$

Complejidad Divide y Vencerás (2)

$$T(n) = \Theta(n^{\log_b a}) + \sum_{i=0}^{\log_b n - 1} \left(\frac{a}{b^d}\right)^i O(n^d)$$

$$T(n) = \Theta(n^{\log_b a}) + O(n^d) \sum_{i=0}^{\log_b n - 1} \left(\frac{a}{b^d}\right)^i$$

- Analizando el componente de de la serie geométrica; es posible inferir la complejidad de ese término

$$O(n^d) \sum_{i=0}^{\log_b n - 1} \left(\frac{a}{b^d}\right)^i$$

Complejidad Divide y Vencerás (3)

$$O(n^d) \sum_{i=0}^{\log_b n - 1} \left(\frac{a}{b^d}\right)^i$$

Complejidad Divide y Vencerás (3)

$$O(n^d) \sum_{i=0}^{\log_b n-1} \left(\frac{a}{b^d}\right)^i$$

- Generalizando

$$g(\ell) = \sum_{i=0}^{\ell} c^i = c^\ell + c^{\ell-1} + \cdots + c + 1$$

Complejidad Divide y Vencerás (3)

$$O(n^d) \sum_{i=0}^{\log_b n-1} \left(\frac{a}{b^d}\right)^i$$

- Generalizando

$$g(\ell) = \sum_{i=0}^{\ell} c^i = c^\ell + c^{\ell-1} + \cdots + c + 1$$

- ¿Cuál es el comportamiento asintótico de $g(\ell)$?

Complejidad Divide y Vencerás (3)

$$O(n^d) \sum_{i=0}^{\log_b n-1} \left(\frac{a}{b^d}\right)^i$$

- Generalizando

$$g(\ell) = \sum_{i=0}^{\ell} c^i = c^\ell + c^{\ell-1} + \cdots + c + 1$$

- ¿Cuál es el comportamiento asintótico de $g(\ell)$?

$$g(\ell) = \begin{cases} \Theta(1) & \text{si } c < 1 \\ \Theta(\ell) & \text{si } c = 1 \\ \Theta(c^\ell) & \text{si } c > 1 \end{cases}$$

Complejidad Divide y Vencerás (4)

- Retomemos la idea de Divide y Vencerás

$$O(n^d) \sum_{i=0}^{\log_b n - 1} \left(\frac{a}{b^d}\right)^i =$$

Complejidad Divide y Vencerás (4)

- Retomemos la idea de Divide y Vencerás

$$O(n^d) \sum_{i=0}^{\log_b n - 1} \left(\frac{a}{b^d}\right)^i = \begin{cases} O(n^d)\Theta(1) & \text{si } \frac{a}{b^d} < 1 \\ O(n^d)\Theta(\log_b n) & \text{si } \frac{a}{b^d} = 1 \\ O(n^d)\Theta\left(\left(\frac{a}{b^d}\right)^{\log_b n}\right) & \text{si } \frac{a}{b^d} > 1 \end{cases}$$

Complejidad Divide y Vencerás (4)

- Retomemos la idea de Divide y Vencerás

$$O(n^d) \sum_{i=0}^{\log_b n - 1} \left(\frac{a}{b^d}\right)^i = \begin{cases} O(n^d)\Theta(1) & \text{si } \frac{a}{b^d} < 1 \\ O(n^d)\Theta(\log_b n) & \text{si } \frac{a}{b^d} = 1 \\ O(n^d)\Theta\left(\left(\frac{a}{b^d}\right)^{\log_b n}\right) & \text{si } \frac{a}{b^d} > 1 \end{cases}$$

- En otras palabras, se prueba que

$$T(n) = \begin{cases} O(n^d) & \text{si } d > \log_b a \\ O(n^d \log n) & \text{si } d = \log_b a \\ O(n^{\log_b a}) & \text{si } d < \log_b a \end{cases}$$

Solución de recurrencias

- Supongamos la siguiente función de recurrencia

$$T(n) = 4T(n/2) + 100n$$

Solución de recurrencias

- Supongamos la siguiente función de recurrencia

$$T(n) = 4T(n/2) + 100n$$

- ¿Como la resolvemos?

Método de sustitución

- El método más general

Método de sustitución

- El método más general
  1. ***Guess*** suponga la forma de la solución

Método de sustitución

- El método más general
  1. ***Guess*** suponga la forma de la solución
  2. ***Verificar*** usando inducción

Método de sustitución

- El método más general
  1. ***Guess*** suponga la forma de la solución
  2. ***Verificar*** usando inducción
  3. ***Resolver*** determinar las constantes

Método de sustitución

- El método más general
  1. ***Guess*** suponga la forma de la solución
  2. ***Verificar*** usando inducción
  3. ***Resolver*** determinar las constantes

- **Ejemplo:** $T(n) = 4T(n/2) + 100n$

Método de sustitución

- El método más general
  1. ***Guess*** suponga la forma de la solución
  2. ***Verificar*** usando inducción
  3. ***Resolver*** determinar las constantes

- **Ejemplo:** $T(n) = 4T(n/2) + 100n$
  1. Asumimos la base $T(1) = O(1)$

Método de sustitución

- El método más general
  1. ***Guess*** suponga la forma de la solución
  2. ***Verificar*** usando inducción
  3. ***Resolver*** determinar las constantes

- **Ejemplo:** $T(n) = 4T(n/2) + 100n$
  1. Asumimos la base $T(1) = O(1)$
  2. Suponemos que es $O(n^3)$. Debemos probar Ω y O

Método de sustitución

- El método más general
  1. ***Guess*** suponga la forma de la solución
  2. ***Verificar*** usando inducción
  3. ***Resolver*** determinar las constantes

- **Ejemplo:** $T(n) = 4T(n/2) + 100n$
  1. Asumimos la base $T(1) = O(1)$
  2. Suponemos que es $O(n^3)$. Debemos probar Ω y O
  3. Asumimos $T(k) \leq ck^3$ para $k < n$

Método de sustitución

- El método más general
  1. ***Guess*** suponga la forma de la solución
  2. ***Verificar*** usando inducción
  3. ***Resolver*** determinar las constantes

- **Ejemplo:** $T(n) = 4T(n/2) + 100n$
  1. Asumimos la base $T(1) = O(1)$
  2. Suponemos que es $O(n^3)$. Debemos probar Ω y O
  3. Asumimos $T(k) \leq ck^3$ para $k < n$
  4. Probamos por inducción que $T(n) \leq cn^3$

Ejemplo

$$T(n) = 4T\left(\frac{n}{2}\right) + 100n$$

Ejemplo

$$
\begin{aligned}
T(n) &= 4T\left(\frac{n}{2}\right) + 100n \\
&\leq 4c\left(\frac{n}{2}\right)^3 + 100n
\end{aligned}
$$

Ejemplo

$$
\begin{aligned}
T(n) &= 4T\left(\frac{n}{2}\right) + 100n \\
&\leq 4c\left(\frac{n}{2}\right)^3 + 100n \\
&= \left(\frac{c}{2}\right)n^3 + 100n
\end{aligned}
$$

Ejemplo

$$
\begin{aligned}
T(n) &= 4T\left(\frac{n}{2}\right) + 100n \\
&\leq 4c\left(\frac{n}{2}\right)^3 + 100n \\
&= \left(\frac{c}{2}\right)n^3 + 100n \\
&= \underbrace{cn^3}_{\textit{objetivo}} - \underbrace{(c/2n^3 - 100n)}_{\textit{residual}}
\end{aligned}
$$

Ejemplo

$$
\begin{aligned}
T(n) &= 4T\left(\frac{n}{2}\right) + 100n \\
&\leq 4c\left(\frac{n}{2}\right)^3 + 100n \\
&= \left(\frac{c}{2}\right)n^3 + 100n \\
&= \underbrace{cn^3}_{\textit{objetivo}} - \underbrace{(c/2n^3 - 100n)}_{\textit{residual}} \\
&\leq cn^3
\end{aligned}
$$

Ejemplo

$$
\begin{aligned}
T(n) &= 4T\left(\frac{n}{2}\right) + 100n \\
&\leq 4c\left(\frac{n}{2}\right)^3 + 100n \\
&= \left(\frac{c}{2}\right)n^3 + 100n \\
&= \underbrace{cn^3}_{\textit{objetivo}} - \underbrace{(c/2n^3 - 100n)}_{\textit{residual}} \\
&\leq cn^3
\end{aligned}
$$

- Para que valores de n y c se cumple que

$$c/2n^3 - 100n \geq 0$$

Ejemplo

$$
\begin{aligned}
T(n) &= 4T\left(\frac{n}{2}\right) + 100n \\
&\leq 4c\left(\frac{n}{2}\right)^3 + 100n \\
&= \left(\frac{c}{2}\right)n^3 + 100n \\
&= \underbrace{cn^3}_{\textbf{objetivo}} - \underbrace{(c/2n^3 - 100n)}_{\textbf{residual}} \\
&\leq cn^3
\end{aligned}
$$

- Para que valores de n y c se cumple que

$$c/2n^3 - 100n \geq 0$$

- Por ejemplo para $c \geq 200$ y $n \geq 1$

Ejemplo (2)

- También debemos determinar las condiciones iniciales, es decir, determinar los casos base.

Ejemplo (2)

- También debemos determinar las condiciones iniciales, es decir, determinar los casos base.
- ***Base***: $T(n) = \Theta(1)$ para $n < n_0$, donde n_0 es una constante

Ejemplo (2)

- También debemos determinar las condiciones iniciales, es decir, determinar los casos base.
- ***Base***: $T(n) = \Theta(1)$ para $n < n_0$, donde n_0 es una constante
- Para $1 \leq n < n_0$, tenemos $\Theta(1) \leq cn^3$, si elegimos c lo suficientemente grande

Ejemplo (2)

- También debemos determinar las condiciones iniciales, es decir, determinar los casos base.
- ***Base***: $T(n) = \Theta(1)$ para $n < n_0$, donde n_0 es una constante
- Para $1 \leq n < n_0$, tenemos $\Theta(1) \leq cn^3$, si elegimos c lo suficientemente grande

No es un límite ajustado

¿Un cota más ajustada?

¿Un cota más ajustada?

- Probemos $T(n) = O(n^2)$

¿Un cota más ajustada?

- Probemos $T(n) = O(n^2)$
- Asumimos $T(k) \leq ck^2$:

¿Un cota más ajustada?

- Probemos $T(n) = O(n^2)$
- Asumimos $T(k) \leq ck^2$:

$$T(n) = 4T\left(\frac{n}{2}\right) + 100n$$

¿Un cota más ajustada?

- Probemos $T(n) = O(n^2)$
- Asumimos $T(k) \leq ck^2$:

$$T(n) = 4T\left(\frac{n}{2}\right) + 100n$$

$$= \underbrace{cn^2}_{\textit{objetivo}} + \underbrace{(\ 100n\ )}_{\textit{residual}}$$

¿Un cota más ajustada?

- Probemos $T(n) = O(n^2)$
- Asumimos $T(k) \leq ck^2$:

$$\begin{aligned} T(n) &= 4T\left(\frac{n}{2}\right) + 100n \\ &= \underbrace{cn^2}_{\textbf{objetivo}} + \underbrace{(\ 100n\ )}_{\textbf{residual}} \\ &\leq cn^3 \end{aligned}$$

¿Un cota más ajustada?

- Probemos $T(n) = O(n^2)$
- Asumimos $T(k) \leq ck^2$:

$$\begin{aligned} T(n) &= 4T\left(\frac{n}{2}\right) + 100n \\ &= \underbrace{cn^2}_{\textbf{objetivo}} + \underbrace{(\ 100n\ )}_{\textbf{residual}} \\ &\leq cn^3 \end{aligned}$$

- ***No*** es posible seleccionar $c > 0$.

Una cota ajustata

- **Idea:** Fortalecer la hipótesis de inducción

Una cota ajustata

- **Idea:** Fortalecer la hipótesis de inducción
- ***Sustraer*** el término de menor orden

Una cota ajustata

- **Idea:** Fortalecer la hipótesis de inducción
- ***Sustraer*** el término de menor orden
- Hipótesis de inducción: $T(k) \leq c_1 k^2 - c_2 k$ para $k < n$

Una cota ajustata

- **Idea:** Fortalecer la hipótesis de inducción
- ***Sustraer*** el término de menor orden
- Hipótesis de inducción: $T(k) \leq c_1 k^2 - c_2 k$ para $k < n$

$$T(n) = 4T\left(\frac{n}{2}\right) + 100n$$

Una cota ajustata

- **Idea:** Fortalecer la hipótesis de inducción
- ***Sustraer*** el término de menor orden
- Hipótesis de inducción: $T(k) \leq c_1 k^2 - c_2 k$ para $k < n$

$$
\begin{aligned}
T(n) &= 4T\left(\frac{n}{2}\right) + 100n \\
&\leq 4(c_1\left(\frac{n}{2}\right)^2 - c_2\left(\frac{n}{2}\right)) + 100n
\end{aligned}
$$

Una cota ajustata

- **Idea:** Fortalecer la hipótesis de inducción
- ***Sustraer*** el término de menor orden
- Hipótesis de inducción: $T(k) \leq c_1 k^2 - c_2 k$ para $k < n$

$$
\begin{aligned}
T(n) &= 4T\left(\frac{n}{2}\right) + 100n \\
&\leq 4(c_1\left(\frac{n}{2}\right)^2 - c_2\left(\frac{n}{2}\right)) + 100n \\
&= c_1 n^2 - 2c_2 n + 100n
\end{aligned}
$$

Una cota ajustata

- **Idea:** Fortalecer la hipótesis de inducción
- ***Sustraer*** el término de menor orden
- Hipótesis de inducción: $T(k) \leq c_1 k^2 - c_2 k$ para $k < n$

$$
\begin{aligned}
T(n) &= 4T\left(\frac{n}{2}\right) + 100n \\
&\leq 4(c_1\left(\frac{n}{2}\right)^2 - c_2\left(\frac{n}{2}\right)) + 100n \\
&= c_1 n^2 - 2c_2 n + 100n \\
&= c_1 n^2 - c_2 n - (c_2 n - 100n) \\
&\qquad \textit{objetivo} \qquad\quad \textit{residual}
\end{aligned}
$$

Una cota ajustata

- **Idea:** Fortalecer la hipótesis de inducción
- ***Sustraer*** el término de menor orden
- Hipótesis de inducción: $T(k) \leq c_1 k^2 - c_2 k$ para $k < n$

$$
\begin{aligned}
T(n) &= 4T\left(\frac{n}{2}\right) + 100n \\
&\leq 4(c_1\left(\frac{n}{2}\right)^2 - c_2\left(\frac{n}{2}\right)) + 100n \\
&= c_1 n^2 - 2c_2 n + 100n \\
&= \underset{\textit{objetivo}}{c_1 n^2 - c_2 n} - \underset{\textit{residual}}{(c_2 n - 100n)} \\
&\leq c_1 n^2 - c_2 n
\end{aligned}
$$

Es posible obtener el objetivo con $c_2 > 100$

Uso del árbol de recurrencia

- Un árbol de recurrencia modela los costos (tiempo) de un algoritmo recursivo.

Uso del árbol de recurrencia

- Un árbol de recurrencia modela los costos (tiempo) de un algoritmo recursivo.
- El método del árbol de recurrencia es bueno para generar conjeturas para el método de sustitución.

Uso del árbol de recurrencia

- Un árbol de recurrencia modela los costos (tiempo) de un algoritmo recursivo.
- El método del árbol de recurrencia es bueno para generar conjeturas para el método de sustitución.
- El método del árbol de recursión puede no ser confiable, al igual que cualquier método que usa puntos suspensivos (...).

Uso del árbol de recurrencia

- Un árbol de recurrencia modela los costos (tiempo) de un algoritmo recursivo.
- El método del árbol de recurrencia es bueno para generar conjeturas para el método de sustitución.
- El método del árbol de recursión puede no ser confiable, al igual que cualquier método que usa puntos suspensivos (. . .).
- Sin embargo, el método del árbol de recurrencia ayuda a generar una intuición.

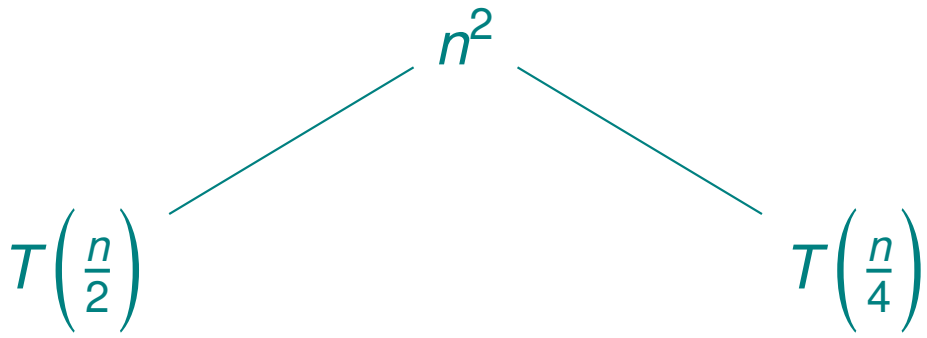
Ejemplo: Uso del árbol de recurrencia

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

$T(n)$

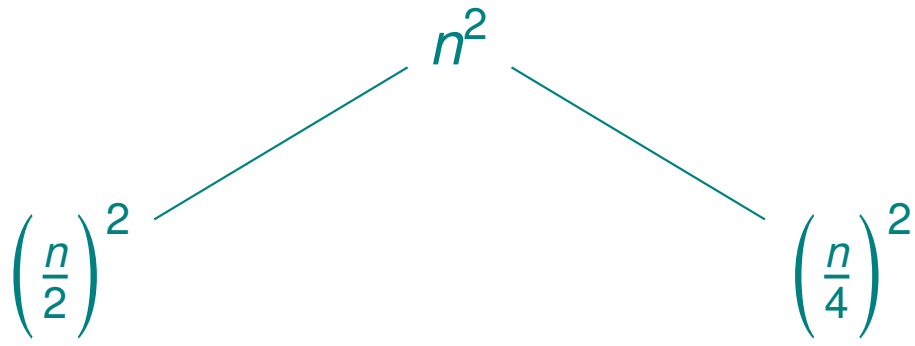
Ejemplo: Uso del árbol de recurrencia

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



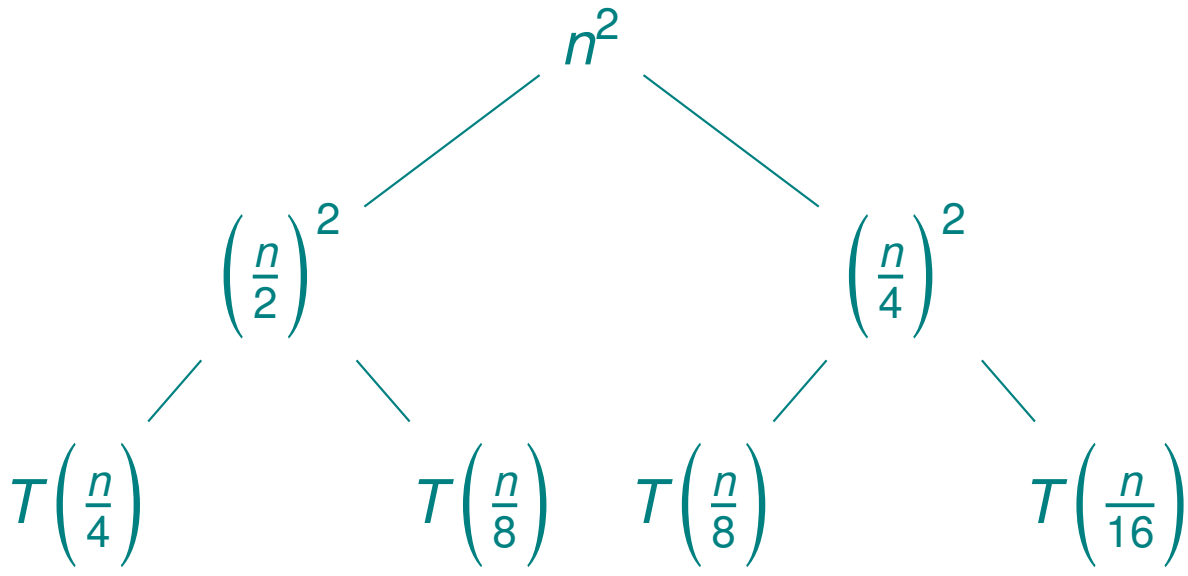
Ejemplo: Uso del árbol de recurrencia

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



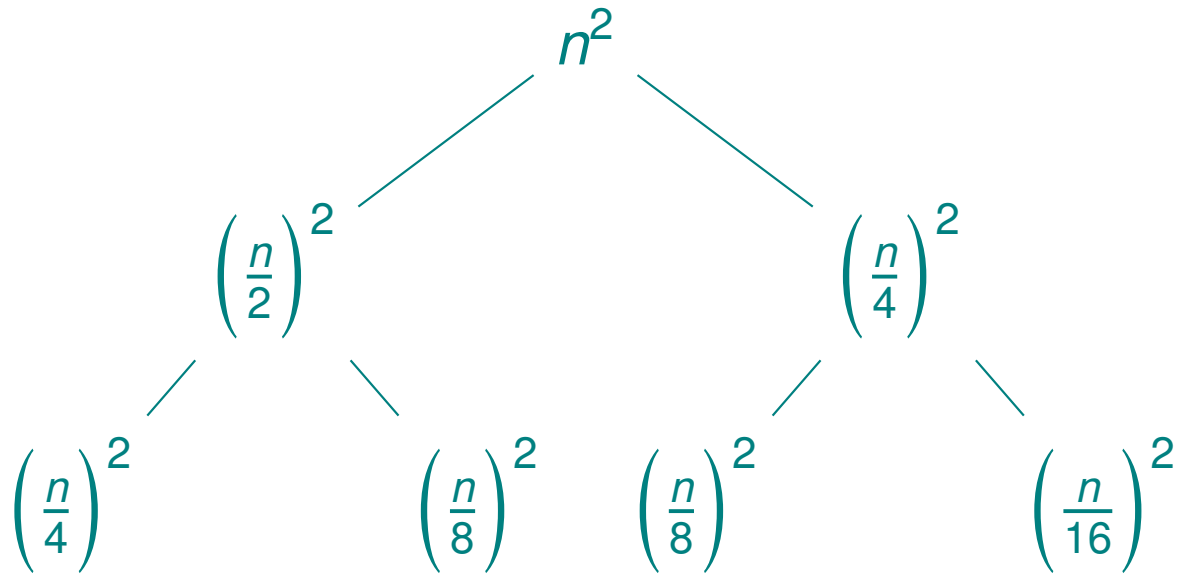
Ejemplo: Uso del árbol de recurrencia

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



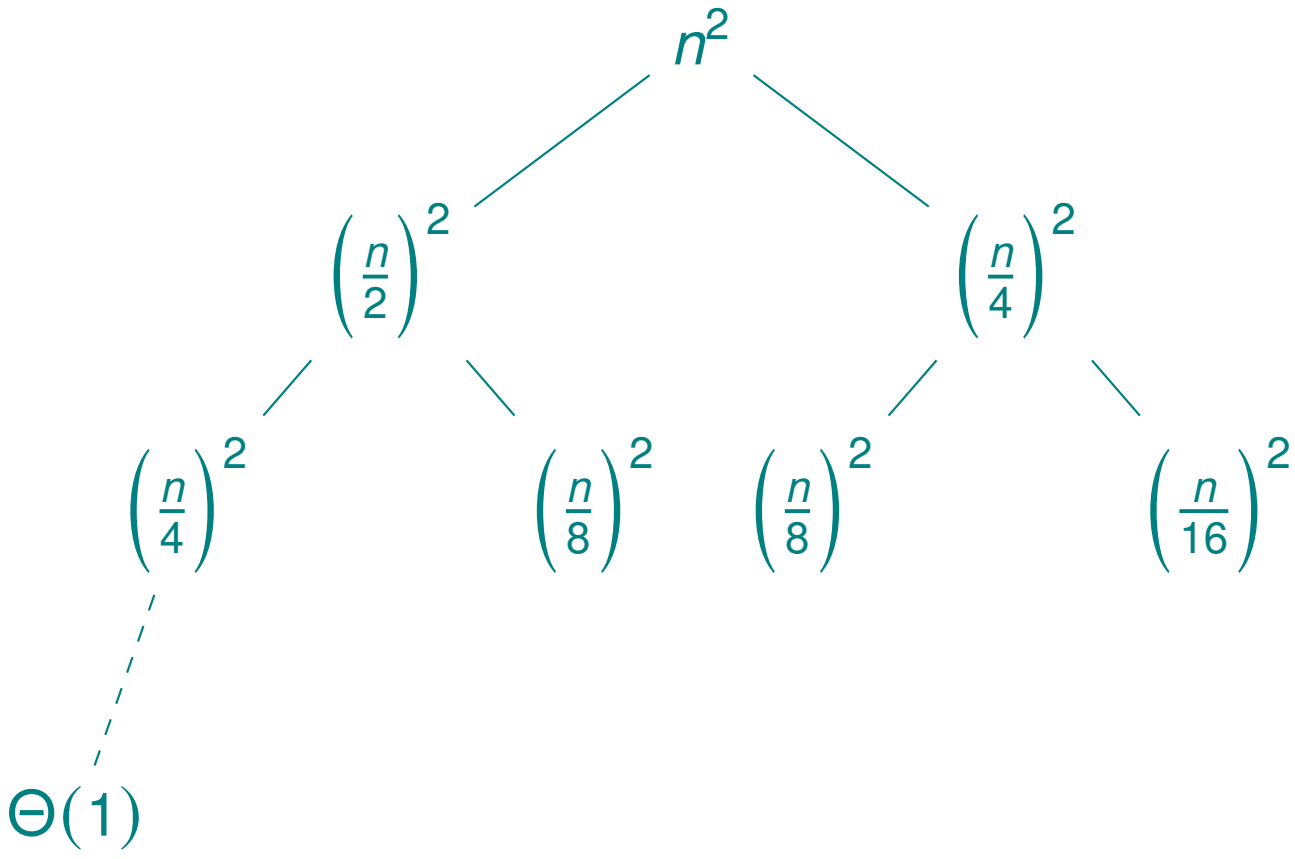
Ejemplo: Uso del árbol de recurrencia

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



Ejemplo: Uso del árbol de recurrencia

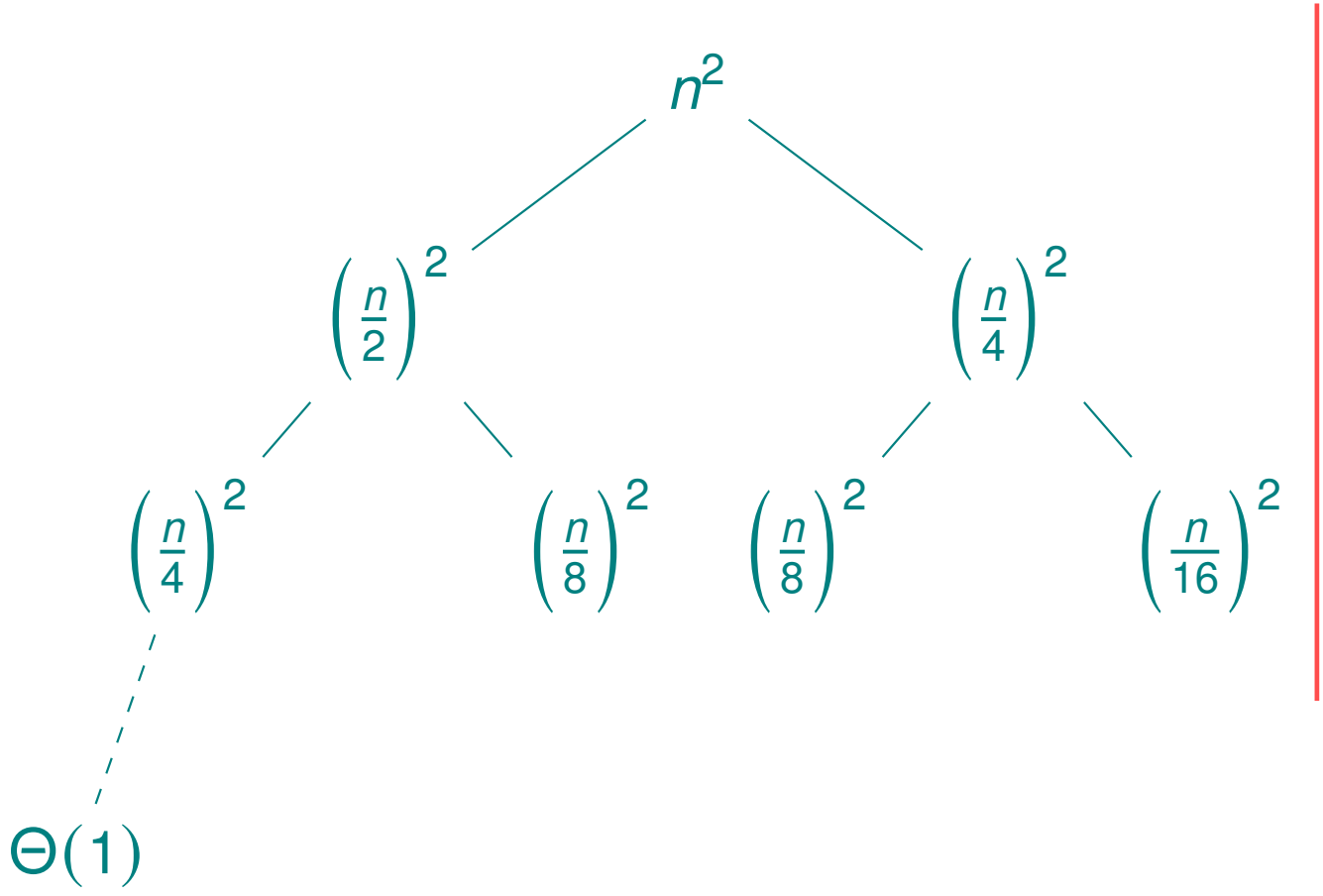
Resolver $T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{4}\right) + n^2$



Ejemplo: Uso del árbol de recurrencia

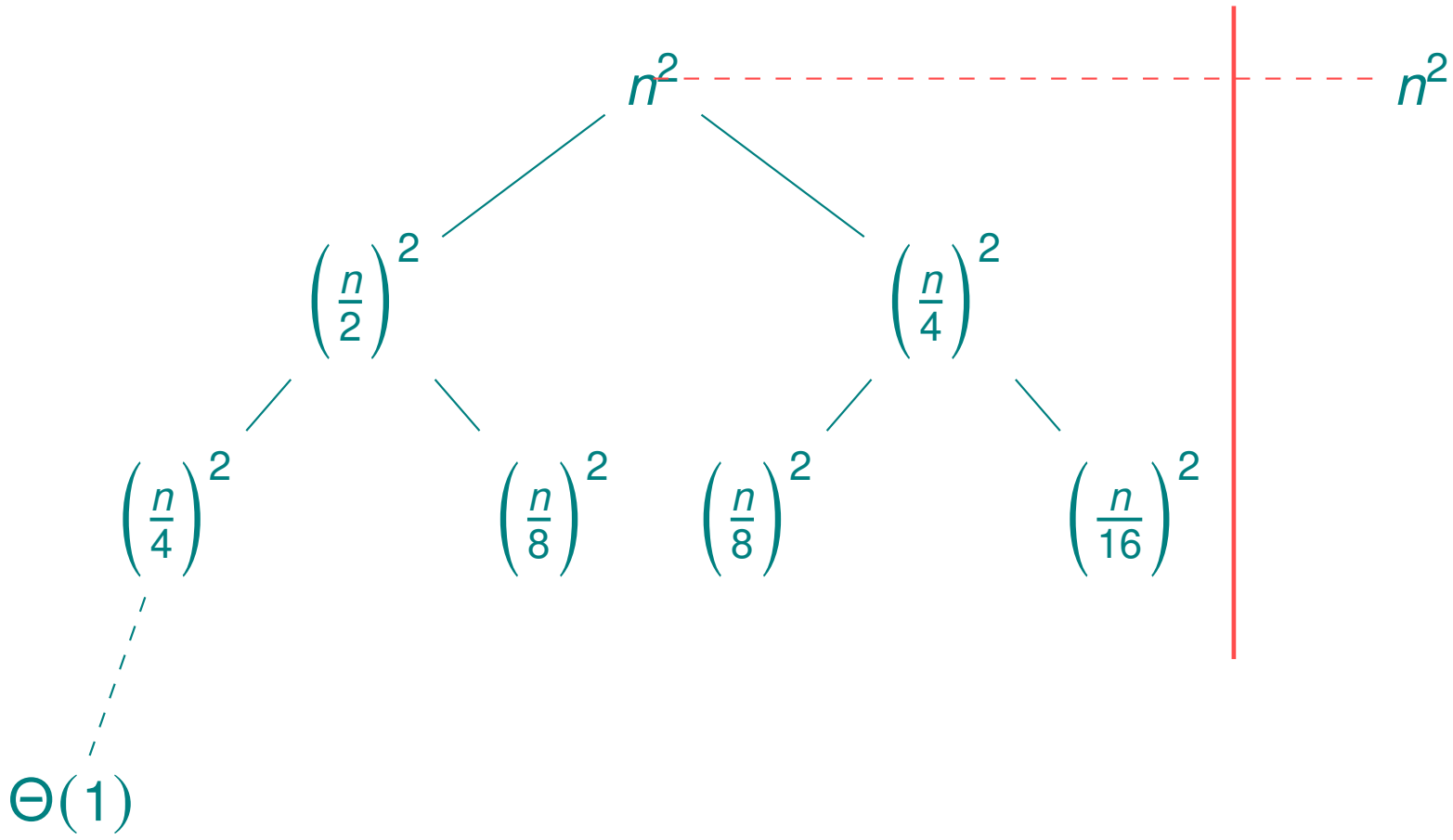
Resolver $T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{4}\right) + n^2$

Costo



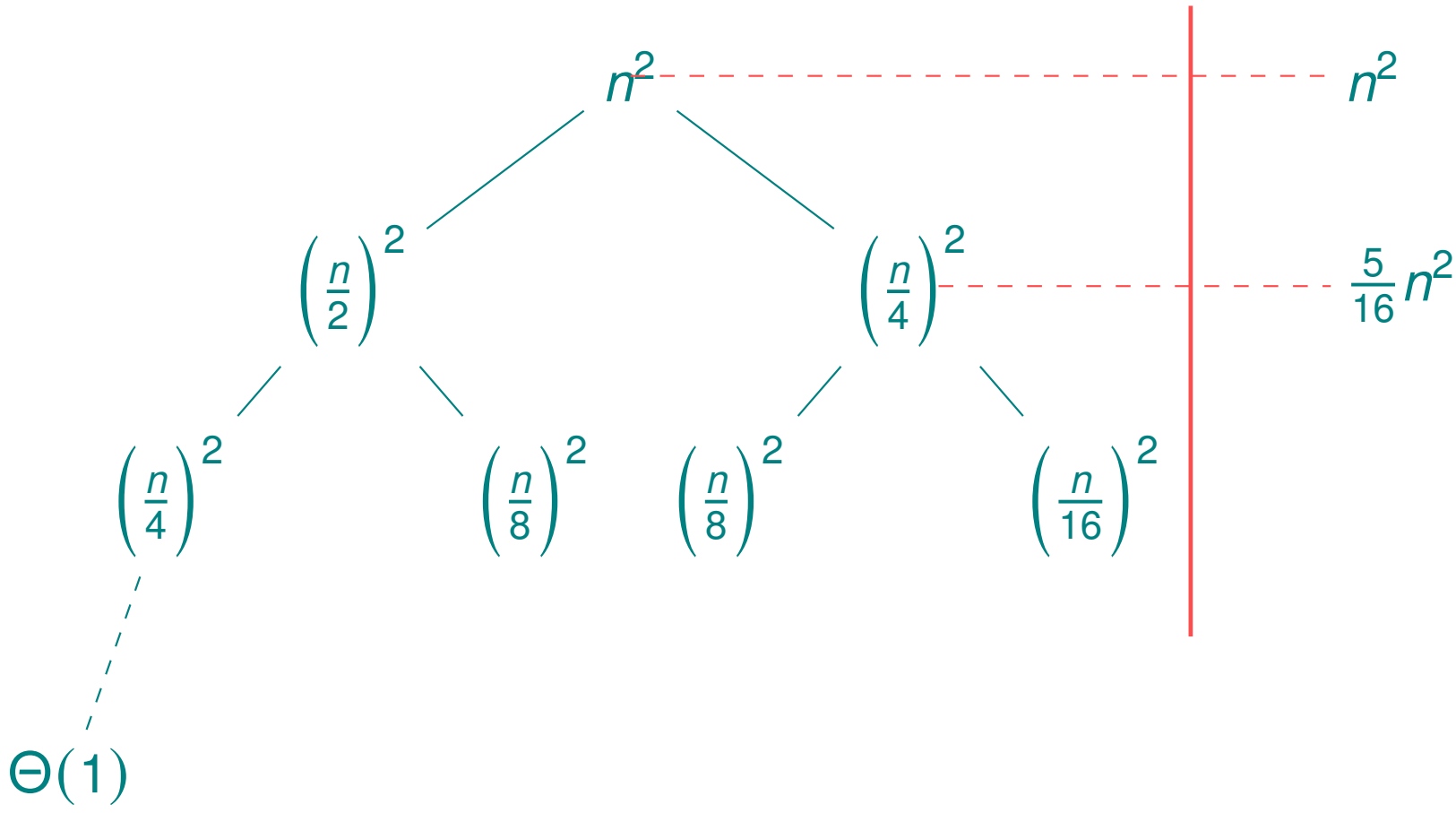
Ejemplo: Uso del árbol de recurrencia

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



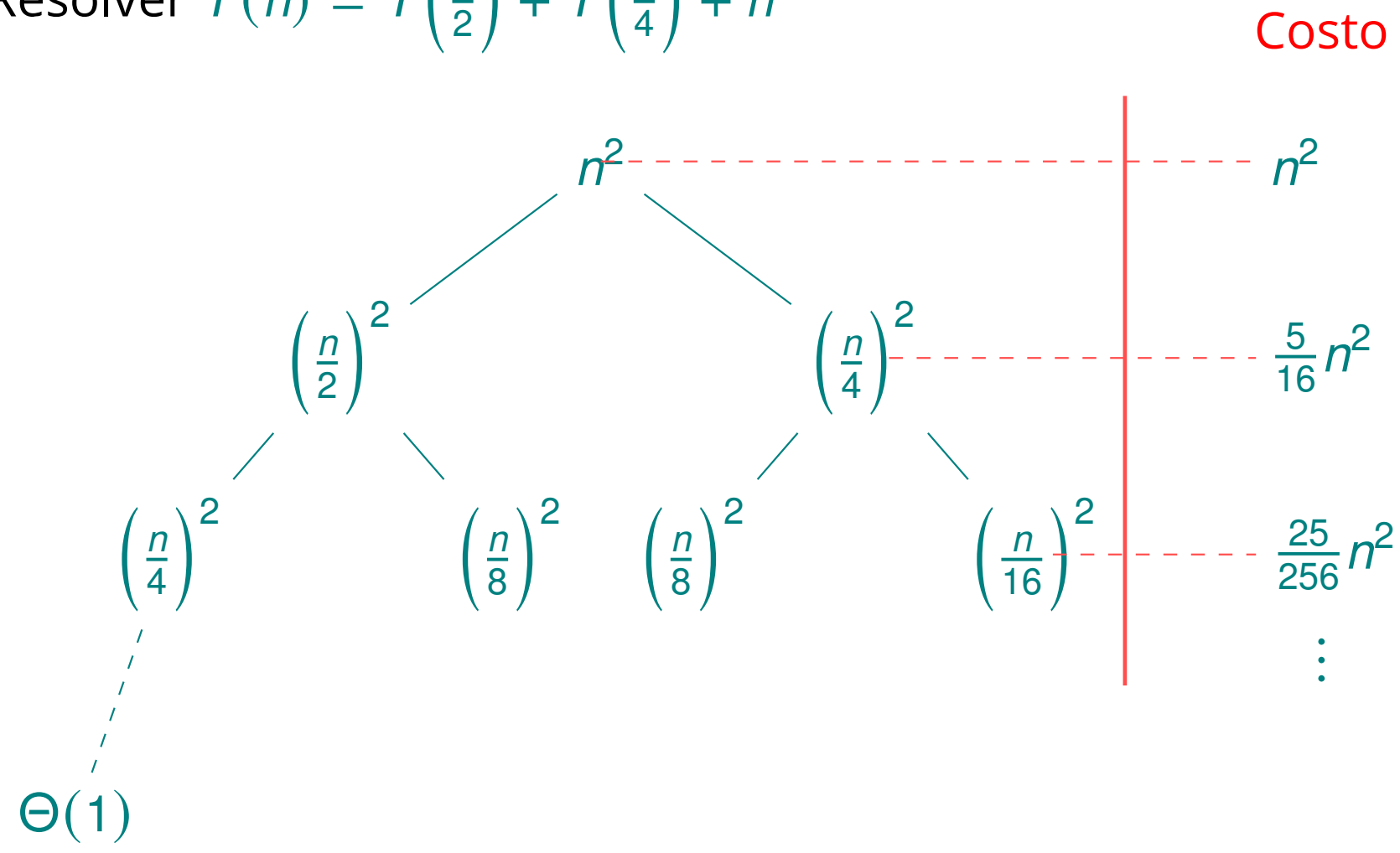
Ejemplo: Uso del árbol de recurrencia

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



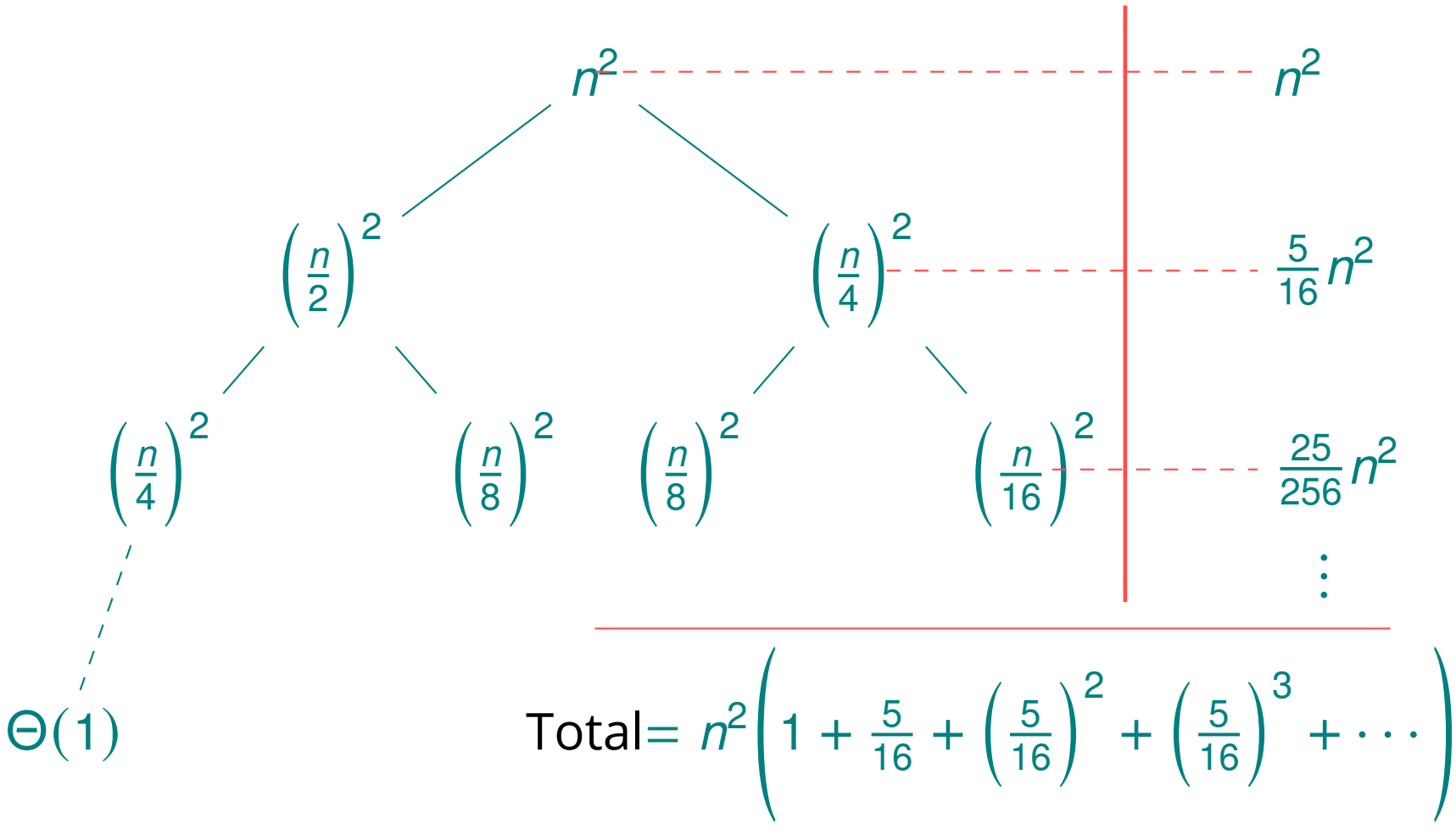
Ejemplo: Uso del árbol de recurrencia

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



Ejemplo: Uso del árbol de recurrencia

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



Ejemplo: Uso del árbol de recurrencia (2)

$$T(n) = n^2\left(1 + \frac{5}{16} + \left(\frac{5}{16}\right)^2 + \left(\frac{5}{16}\right)^3 + \cdots\right)$$

Ejemplo: Uso del árbol de recurrencia (2)

$$T(n) = n^2\left(1 + \frac{5}{16} + \left(\frac{5}{16}\right)^2 + \left(\frac{5}{16}\right)^3 + \cdots\right)$$

- Series geométricas

Ejemplo: Uso del árbol de recurrencia (2)

$$T(n) = n^2\left(1 + \frac{5}{16} + \left(\frac{5}{16}\right)^2 + \left(\frac{5}{16}\right)^3 + \cdots\right)$$

- Series geométricas
  1. para $x \neq 1$

$$1 + x + x^2 + \cdots x^n = \sum_{k=0}^{n} x^k = \frac{1 - x^{n+1}}{1 - x}$$

Ejemplo: Uso del árbol de recurrencia (2)

$$T(n) = n^2\left(1 + \frac{5}{16} + \left(\frac{5}{16}\right)^2 + \left(\frac{5}{16}\right)^3 + \cdots\right)$$

- Series geométricas
  1. para $x \neq 1$

$$1 + x + x^2 + \cdots x^n = \sum_{k=0}^{n} x^k = \frac{1 - x^{n+1}}{1 - x}$$

  2. para $|x| < 1$

$$1 + x + x^2 + \cdots = \sum_{k=0}^{\infty} x^k = \frac{1}{1 - x}$$

Ejemplo: Uso del árbol de recurrencia (2)

$$T(n) = n^2\left(1 + \frac{5}{16} + \left(\frac{5}{16}\right)^2 + \left(\frac{5}{16}\right)^3 + \cdots\right)$$

- Series geométricas
  1. para $x \neq 1$

$$1 + x + x^2 + \cdots x^n = \sum_{k=0}^{n} x^k = \frac{1 - x^{n+1}}{1 - x}$$

  2. para $|x| < 1$

$$1 + x + x^2 + \cdots = \sum_{k=0}^{\infty} x^k = \frac{1}{1 - x}$$

Usando la segunda serie tenemos: $T(n) = \Theta(n^2)$

Ejemplo: Uso del Teorema Maestro

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

Ejemplo: Uso del Teorema Maestro

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

- $a = 4$, $b = 2$, $f(n) = n^3$ y $d = 3$

Ejemplo: Uso del Teorema Maestro

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

- $a = 4$, $b = 2$, $f(n) = n^3$ y $d = 3$
- $\Longrightarrow \log_b(a) = \log_2(4) = 2$

Ejemplo: Uso del Teorema Maestro

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

- $a = 4$, $b = 2$, $f(n) = n^3$ y $d = 3$
- $\Longrightarrow \log_b(a) = \log_2(4) = 2$
- como $d > \log_b(a)$

Ejemplo: Uso del Teorema Maestro

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

- $a = 4$, $b = 2$, $f(n) = n^3$ y $d = 3$
- $\Longrightarrow \log_b(a) = \log_2(4) = 2$
- como $d > \log_b(a)$
- $T(n) = O(n^d) = O(n^3)$

Ejemplo: Uso del Teorema Maestro(2)

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

Ejemplo: Uso del Teorema Maestro(2)

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

- $a = 4$, $b = 2$, $f(n) = n^2$ y $d = 2$

Ejemplo: Uso del Teorema Maestro(2)

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

- $a = 4$, $b = 2$, $f(n) = n^2$ y $d = 2$
- $\Longrightarrow \log_b(a) = \log_2(4) = 2$

Ejemplo: Uso del Teorema Maestro(2)

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

- $a = 4$, $b = 2$, $f(n) = n^2$ y $d = 2$
- $\Longrightarrow \log_b(a) = \log_2(4) = 2$
- como $d = \log_b(a)$

Ejemplo: Uso del Teorema Maestro(2)

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

- $a = 4$, $b = 2$, $f(n) = n^2$ y $d = 2$
- $\Longrightarrow \log_b(a) = \log_2(4) = 2$
- como $d = \log_b(a)$
- $T(n) = O(n^d \log n) = O(n^2 \log n)$

Ejemplo: Uso del Teorema Maestro(3)

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

Ejemplo: Uso del Teorema Maestro(3)

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

- $a = 4$, $b = 2$, $f(n) = n$ y $d = 1$

Ejemplo: Uso del Teorema Maestro(3)

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

- $a = 4$, $b = 2$, $f(n) = n$ y $d = 1$
- $\Longrightarrow \log_b(a) = \log_2(4) = 2$

Ejemplo: Uso del Teorema Maestro(3)

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

- $a = 4$, $b = 2$, $f(n) = n$ y $d = 1$
- $\Longrightarrow \log_b(a) = \log_2(4) = 2$
- como $d < \log_b(a)$

Ejemplo: Uso del Teorema Maestro(3)

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

- $a = 4$, $b = 2$, $f(n) = n$ y $d = 1$
- $\Longrightarrow \log_b(a) = \log_2(4) = 2$
- como $d < \log_b(a)$
- $T(n) = O(n^{\log_b a}) = O(n^2)$

Ejemplo: Uso del Teorema Maestro(4)

Resolver $T(n) = 4T(n/2) + n^2/\log n$

Ejemplo: Uso del Teorema Maestro(4)

Resolver $T(n) = 4T(n/2) + n^2/\log n$

- $a = 4$, $b = 2$, $f(n) = n^2/\log n$

Ejemplo: Uso del Teorema Maestro(4)

Resolver $T(n) = 4T(n/2) + n^2/\log n$

- $a = 4$, $b = 2$, $f(n) = n^2/\log n$
- El teorema maestro no aplica, ya que $f(n)$ no tiene la forma $O(n^d)$

Ejercicios