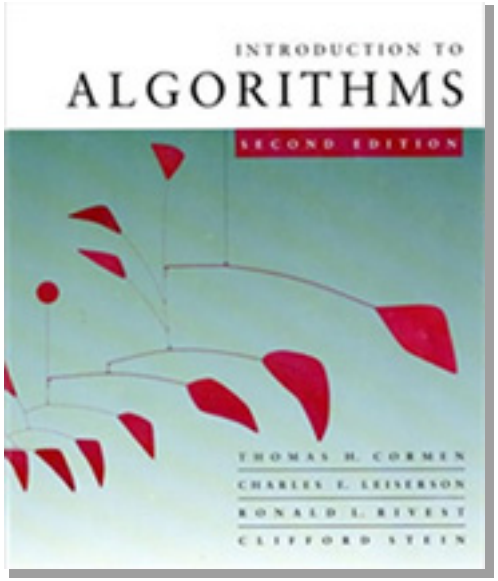
# *Algoritmalara Giriş*

## 6.046J/18.401J



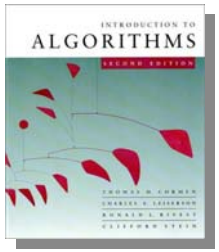

## DERS 3

**Böl ve Fethet**

- İkili arama
- Sayı üstelleri
- Fibonacci sayıları
- Matriks çarpımı
- Strassen'in algoritması
- VLSI ağaç yerleşimi

**Prof. Erik D. Demaine**

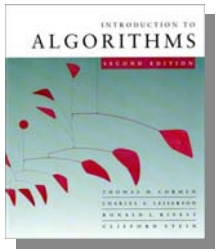

*Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson*




# Böl-ve-hükmet tasarım paradigması

1. Problemi (anlık durumu) alt problemlere **böl**.
2. Altproblemleri özyinelemeli olarak çözüp, onları **fethet**.
3. Altproblem çözümlerini **birleştir**.

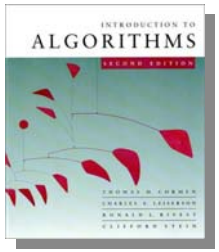
 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# Birleştirme sıralaması

1. ***Bölmek:*** Kolay.
2. ***Hükmetmek:*** 2 altdizilimi özyinelemeli sıralama.
3. ***Birleştirmek:*** Doğrusal-zamanda birleştirme.

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# Birleştirme sıralaması

***1. Bölmek:*** Kolay.

***2. Hükmetmek:*** 2 altdizilimi özyinelemeli sıralama.

***3. Birleştirmek:*** Doğrusal-zamanda birleştirme.

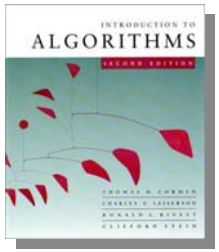
$$T(n) = 2T(n/2) + \Theta(n)$$

*altproblem sayısı*

*altproblem boyutu*

*bölme ve birleştirme işi*

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# Master teoremi (hatırlatma)

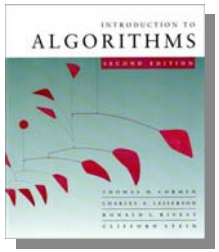
$$T(n) = a\,T(n/b) + f(n)$$

**DURUM 1**: $f(n) = O(n^{\log_b a - \varepsilon})$, sabit $\varepsilon > 0$
$\Rightarrow T(n) = \Theta(n^{\log_b a})$.

**DURUM 2**: $f(n) = \Theta(n^{\log_b a} \lg^k n)$, sabit $k \geq 0$
$\Rightarrow T(n) = \Theta(n^{\log_b a} \lg^{k+1} n)$.

**DURUM 3**: $f(n) = \Omega(n^{\log_b a + \varepsilon})$, sabit $\varepsilon > 0$,
ve düzenleyici koşul (regularity condition).
$\Rightarrow T(n) = \Theta(f(n))$.

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# Master teoremi (hatırlatma)

$$T(n) = a\,T(n/b) + f(n)$$

**DURUM 1**: $f(n) = O(n^{\log_b a - \varepsilon})$, sabit $\varepsilon > 0$

$\Rightarrow T(n) = \Theta(n^{\log_b a})$ .

**DURUM 2**: $f(n) = \Theta(n^{\log_b a} \lg^k n)$, sabit $k \geq 0$

$\Rightarrow T(n) = \Theta(n^{\log_b a} \lg^{k+1} n)$ .

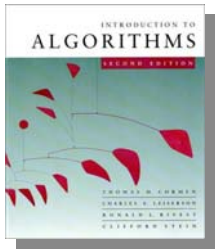
**DURUM 3**: $f(n) = \Omega(n^{\log_b a + \varepsilon})$, sabit $\varepsilon > 0$,
ve düzenleyici koşul (regularity condition).

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

***Birleştirme sıralaması:*** $a = 2, b = 2 \Rightarrow n^{\log_b a} = n^{\log_2 2} = n$

$\Rightarrow$ DURUM 2 ($k = 0$) $\Rightarrow T(n) = \Theta(n \lg n)$ .


*Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson*

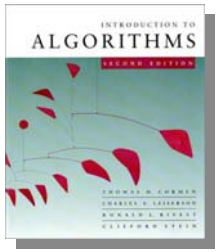



# İkili arama

Sıralı dizilimin bir elemanını bulma:

1. ***Böl:*** Orta elemanı belirle.
2. ***Hükmet:*** 1 altdizilimde özyinelemeli arama yap.
3. ***Birleştir:*** Kolay.

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# İkili arama

Sıralı dizilimin bir elemanını bulma:

***1. Böl:*** Orta elemanı belirle.

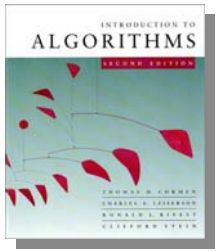
***2. Hükmet:*** 1altdizilimde özyinelemeli arama yap.

***3.Birleştir:*** Kolay.

***Örnek:*** 9' u bul.

| 3 | 5 | 7 | 8 | 9 | 12 | 15 |
|---|---|---|---|---|---|---|

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# İkili arama

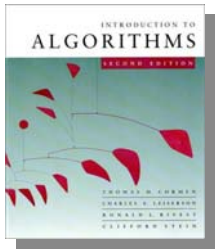
Sıralı dizilimin bir elemanını bulma:

1. ***Böl:*** Orta elemanı belirle.
2. ***Hükmet:*** 1altdizilimde özyinelemeli arama yap.
3. ***Birleştir:*** Kolay.

***Örnek:*** 9'u bul.

| 3 | 5 | 7 | 8 | 9 | 12 | 15 |
|---|---|---|---|---|---|---|

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# İkili arama

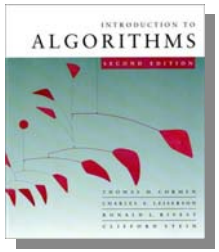
Sıralı dizilimin bir elemanını bulma:

1. ***Böl:*** Orta elemanı belirle.
2. ***Hükmet:*** 1altdizilimde özyinelemeli arama yap.
3. ***Birleştir:*** Kolay.

***Örnek:*** 9'u bul.

3 5 7 8 9 12 15

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# İkili arama

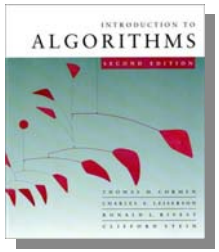
Sıralı dizilimin bir elemanını bulma:

1. ***Böl:*** Orta elemanı belirle.
2. ***Hükmet:*** 1 altdizilimde özyinelemeli arama yap.
3. ***Birleştir:*** Kolay.

***Örnek:*** 9'u bul.

3 5 7 8 9 12 15

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# İkili arama

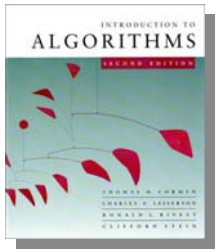
Sıralı dizilimin bir elemanını bulma:

1. ***Böl:*** Orta elemanı belirle.
2. ***Hükmet:*** 1 altdizilimde özyinelemeli arama yap.
3. ***Birleştir:*** Kolay.

***Örnek:*** 9'u bul

3 5 7 8 9 12 15

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# İkili arama

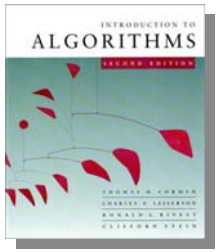
Sıralı dizilimin bir elemanını bulma:

1. ***Böl:*** Orta elemanı belirle.
2. ***Hükmet:*** 1 altdizilimde özyinelemeli arama yap.
3. ***Birleştir:*** Kolay.

***Örnek:*** 9'u bul.

3 5 7 8 9 12 15

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# İkili arama için yineleme

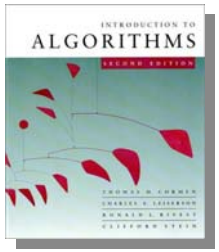
$$T(n) = 1\,T(n/2) + \Theta(1)$$

*altproblem sayısı*

*altproblem boyutu*

*bölme ve birleştime işi*

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# İkili arama için yineleme

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

*altproblem sayısı*

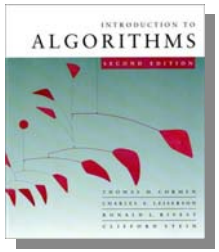
*altproblem boyutu*

*bölme ve birleştime işi*

$$n^{\log_b a} = n^{\log_2 1} = n^0 = 1 \Rightarrow \text{DURUM 2 } (k = 0)$$

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


*Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson*

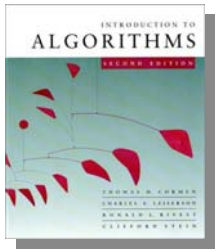



# Bir sayının üstellenmesi

**Problem:** $a^n$'yi, $n \in \mathbb{N}$ iken hesaplama.

**Saf (Naive) algoritma:** $\Theta(n)$.

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# Bir sayının üstellenmesi

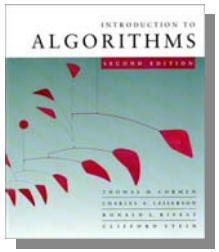
**Problem:** $a^n$'yi, $n \in \mathbb{N}$ iken hesaplama.

**Saf (Naive) algoritma:** $\Theta(n)$.

**Böl-ve-fethet algoritması:**

$$a^n = \begin{cases} a^{n/2} \cdot a^{n/2} & n \text{ çift sayıysa;} \\ a^{(n-1)/2} \cdot a^{(n-1)/2} \cdot a & n \text{ tek sayıysa.} \end{cases}$$

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# Bir sayının üstellenmesi

**Problem:** $a^n$ 'yi $n \in \mathbb{N}$ iken hesaplama.

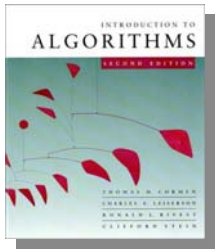
**Saf (Naive) algorithm:** $\Theta(n)$.

**Böl-ve-fethet algoritması:**

$$a^n = \begin{cases} a^{n/2} \cdot a^{n/2} & n \text{ çift sayıysa;} \\ a^{(n-1)/2} \cdot a^{(n-1)/2} \cdot a & n \text{ tek sayıysa.} \end{cases}$$

$$T(n) = T(n/2) + \Theta(1) \Rightarrow T(n) = \Theta(\lg n) .$$

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 



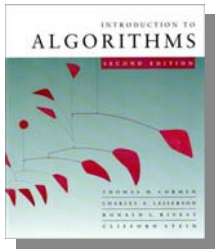

# Fibonacci sayıları

**Özyinelemeli tanım:**

$$F_n = \begin{cases} 0 & \text{eğer } n = 0 \text{ ise;} \\ 1 & \text{eğer } n = 1 \text{ ise;} \\ F_{n-1} + F_{n-2} & \text{eğer } n \geq 2 \text{ ise.} \end{cases}$$

0 1 1 2 3 5 8 13 21 34 ⋯

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# Fibonacci sayıları

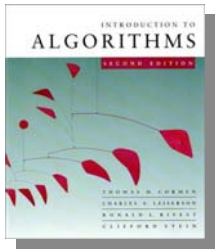
**Özyinelemeli tanım:**

$$F_n = \begin{cases} 0 & \text{eğer } n = 0; \\ 1 & \text{eğer } n = 1; \\ F_{n-1} + F_{n-2} & \text{eğer } n \geq 2 \text{ ise.} \end{cases}$$

$$0 \quad 1 \quad 1 \quad 2 \quad 3 \quad 5 \quad 8 \quad 13 \quad 21 \quad 34 \quad \cdots$$

**Saf özyinelemeli algoritma:** $\Omega(\phi^n)$ (üstel zaman), buradaki $\phi = (1+\sqrt{5})/2$ **altın oran**'dır (***golden ratio***).


*Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson*

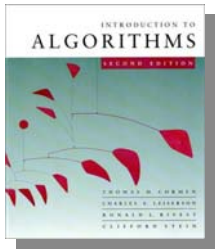



# Fibonacci sayılarını hesaplama

**Aşağıdan yukarıya:**

- $F_0, F_1, F_2, \ldots, F_n$'i sırayla, her sayı iki öncekinin toplamı olacak şekilde hesaplayın.
- Yürütüm süresi: $\Theta(n)$.

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# Fibonacci sayılarını hesaplama

**Aşağıdan yukarıya:**

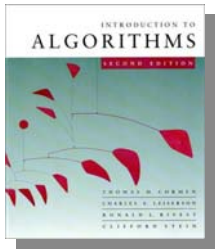
- $F_0, F_1, F_2, \ldots, F_n$'i sırayla, her sayı iki öncekinin toplamı olacak şekilde hesaplayın.
- Yürütüm süresi: $\Theta(n)$.

**Saf özyinelemeli kare alma (Naive recursive squaring):**

$F_n = \phi^n / \sqrt{5}$ yakın tamsayı yuvarlaması.

- Özyinelemeli kare alma: $\Theta(\lg n)$ zamanı.
- Bu yöntem güvenilir değildir, çünkü yüzer-nokta aritmatiği yuvarlama hatalarına gebedir.

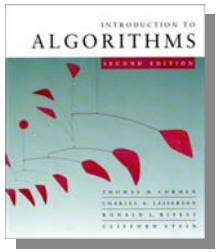

*Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson*




# Özyineleme ile kare alma (Recursive squaring)

**Teorem:** $$\begin{bmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^n.$$

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 



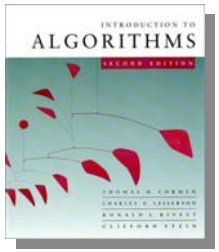

# Özyineleme ile kare alma

**Teorem:** $$\begin{bmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^n .$$

**Algoritma:** Özyineleme ile kare alma.

Süre $= \Theta(\lg n)$ .

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# Özyineleme ile kare alma

**Teorem:** $\begin{bmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^n$.

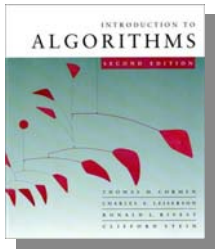
**Algoritma:** Özyineleme ile kare alma.

Süre = $\Theta(\lg n)$ .

*Teoremin ispatı.* ( $n$ 'de tümevarım)

Taban ($n = 1$): $\begin{bmatrix} F_2 & F_1 \\ F_1 & F_0 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^1$ .

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# Özyineleme ile kare alma

Tümevarım adımı ($n \geq 2$):

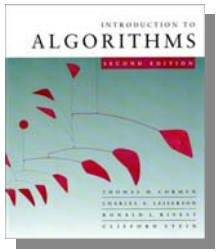
$$\begin{bmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{bmatrix} = \begin{bmatrix} F_n & F_{n-1} \\ F_{n-1} & F_{n-2} \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^{n-1} \cdot \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^{n}$$

■

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# Matrislerde çarpma

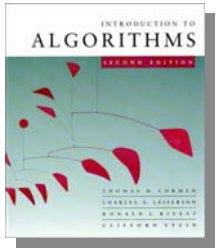
**Girdi:** $= [a_{ij}], B = [b_{ij}].$
**Çıktı:** $C = [c_{ij}] = A \cdot B.$

$\Big\}\ i, j = 1, 2, \ldots, n.$

$$\begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n1} & c_{n2} & \cdots & c_{nn} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \cdot \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{nn} \end{bmatrix}$$

$$c_{ij} = \sum_{k=1}^{n} a_{ik} \cdot b_{kj}$$

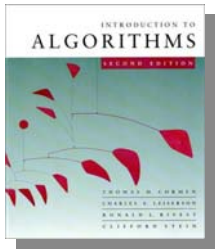

*Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson*




# Standart algoritma

**for** $i \leftarrow 1$ **to** $n$ *(i 1'den n'ye kadar)*
  **do for** $j \leftarrow 1$ **to** $n$ *(j 1'den n'ye kadar)*
    **do** $c_{ij} \leftarrow 0$
      **for** $k \leftarrow 1$ **to** $n$
        **do** $c_{ij} \leftarrow c_{ij} + a_{ik} \cdot b_{kj}$

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 

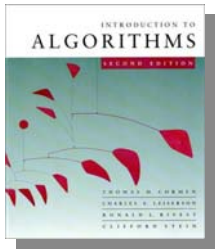



# Standart algoritma

**for** $i \leftarrow 1$ **to** $n$ *(i 1'den n'ye kadar)*
  **do for** $j \leftarrow 1$ **to** $n$ *(j 1'den n'ye kadar)*
    **do** $c_{ij} \leftarrow 0$
      **for** $k \leftarrow 1$ **to** $n$
        **do** $c_{ij} \leftarrow c_{ij} + a_{ik} \cdot b_{kj}$

Koşma süresi = $\Theta(n^3)$

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# Böl-ve-fethet algoritması

**FİKİR**

$n \times n$ matris = $(n/2)\times(n/2)$ altmatrisin $2\times 2$ matrisi:

$$\begin{bmatrix} r & s \\ t & u \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \cdot \begin{bmatrix} e & f \\ g & h \end{bmatrix}$$

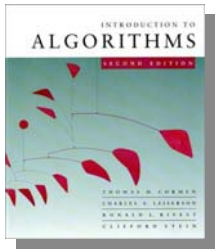
$$C = A \cdot B$$

$$\left.\begin{aligned} r &= ae + bg \\ s &= af + bh \\ t &= ce + dg \\ u &= cf + dh \end{aligned}\right\}$$

8 çarpma $(n/2)\times(n/2)$ altmatriste

4 toplama $(n/2)\times(n/2)$ altmatriste


*Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson*




# Böl-ve-fethet algoritması

**FİKİR:**

$n \times n$ matris = $(n/2)\times(n/2)$ altmatrisin $2\times 2$ matrisi:

$$\begin{bmatrix} r & s \\ t & u \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \cdot \begin{bmatrix} e & f \\ g & h \end{bmatrix}$$

$$C = A \cdot B$$

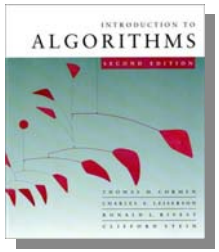
$$\left.\begin{aligned} r &= ae + bg \\ s &= af + bh \\ t &= ce + dh \\ u &= cf + dg \end{aligned}\right\}$$

*recursive (özyinelemeli)*

8 çarpma $(n/2)\times(n/2)$ altmatriste,

4 toplama $(n/2)\times(n/2)$ altmatriste.


*Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson*




# Böl-ve-Fethet algoritmasının çözümlemesi

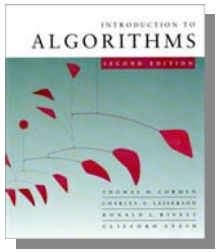
$$T(n) = 8\,T(n/2) + \Theta(n^2)$$

*altmatris sayısı*

*altmatris boyutu*

*altmatrisleri toplama işi*


*Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson*




# Böl-ve-Fethet algoritmasının çözümlemesi

$$T(n) = 8\,T(n/2) + \Theta(n^2)$$

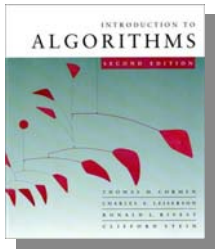
*altmatris sayısı*

*altmatris boyutu*

*altmatrisleri toplama işi*

$$n^{\log_b a} = n^{\log_2 8} = n^3 \Rightarrow \text{DURUM 1} \Rightarrow T(n) = \Theta(n^3).$$


*Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson*




# Böl-ve-Fethet algoritmasının çözümlemesi

$$T(n) = 8\,T(n/2) + \Theta(n^2)$$

*altmatris sayısı*

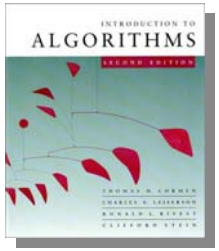
*altmatris boyutu*

*altmatrisleri toplama işi*

$$n^{\log_b a} = n^{\log_2 8} = n^3 \Rightarrow \text{DURUM 1} \Rightarrow T(n) = \Theta(n^3).$$

***Sıradan algoritmadan daha iyi değil.***

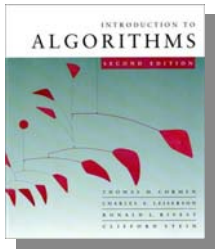

*Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson*




# Strassen’in fikri

- $2 \times 2$ matrisleri yalnız $7$ özyinelemeli çarpmayla çöz.

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 

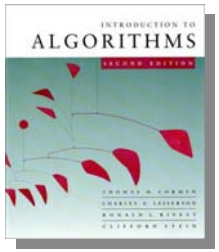



# Strassen's fikri

- 2×2 matrisleri yalnız 7 özyinelemeli çarpmayla çöz.

$$P_1 = a \cdot (f - h)$$
$$P_2 = (a + b) \cdot h$$
$$P_3 = (c + d) \cdot e$$
$$P_4 = d \cdot (g - e)$$
$$P_5 = (a + d) \cdot (e + h)$$
$$P_6 = (b - d) \cdot (g + h)$$
$$P_7 = (a - c) \cdot (e + f)$$

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 



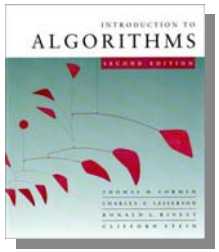

# Strassen’in fikri

- $2\times2$ matrisleri yalnız $7$ özyinelemeli çarpmayla çöz.

$$P_1 = a \cdot (f - h)$$
$$P_2 = (a + b) \cdot h$$
$$P_3 = (c + d) \cdot e$$
$$P_4 = d \cdot (g - e)$$
$$P_5 = (a + d) \cdot (e + h)$$
$$P_6 = (b - d) \cdot (g + h)$$
$$P_7 = (a - c) \cdot (e + f)$$

$$r = P_5 + P_4 - P_2 + P_6$$
$$s = P_1 + P_2$$
$$t = P_3 + P_4$$
$$u = P_5 + P_1 - P_3 - P_7$$

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# Strassen'in fikri

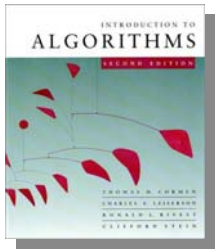
- 2×2 matrisleri yalnız 7 özyinelemeli çarpmayla çöz.

$$P_1 = a \cdot (f - h)$$
$$P_2 = (a + b) \cdot h$$
$$P_3 = (c + d) \cdot e$$
$$P_4 = d \cdot (g - e)$$
$$P_5 = (a + d) \cdot (e + h)$$
$$P_6 = (b - d) \cdot (g + h)$$
$$P_7 = (a - c) \cdot (e + f)$$

$$r = P_5 + P_4 - P_2 + P_6$$
$$s = P_1 + P_2$$
$$t = P_3 + P_4$$
$$u = P_5 + P_1 - P_3 - P_7$$

7 çarp., 18 topl. /çıkar.
**Not:** Çarpma işleminde **sırabağımsızlık yok!**


*Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson*



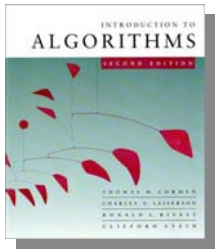

# Strassen's fikri

- $2\times2$ matrislerin yalnız $7$ özyinelemeli çarpmayla çözümü

$$
\begin{aligned}
P_1 &= a \cdot (f - h) \\
P_2 &= (a + b) \cdot h \\
P_3 &= (c + d) \cdot e \\
P_4 &= d \cdot (g - e) \\
P_5 &= (a + d) \cdot (e + h) \\
P_6 &= (b - d) \cdot (g + h) \\
P_7 &= (a - c) \cdot (e + f)
\end{aligned}
$$

$$
\begin{aligned}
r &= P_5 + P_4 - P_2 + P_6 \\
&= (a + d)(e + h) \\
&\quad + d(g - e) - (a + b)h \\
&\quad + (b - d)(g + h) \\
&= ae + ah + de + dh \\
&\quad + dg - de - ah - bh \\
&\quad + bg + bh - dg - dh \\
&= ae + bg
\end{aligned}
$$


*Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson*



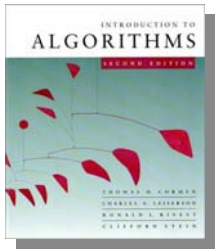

# Strassen'in algoritması

***1. Böl:*** $A$ ve $B$'yi $(n/2)\times(n/2)$ altmatrislere böl. $+$ ve $-$ kullanarak çarpılabilecek terimler oluştur.

***2. Fethet:*** $(n/2)\times(n/2)$ altmatrislerde özyinelemeli $7$ çarpma yap.

***3. Birleştir:*** $+$ ve $-$ kullanarak $(n/2)\times(n/2)$ altmatrislerde $C$ 'yi oluştur.

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 

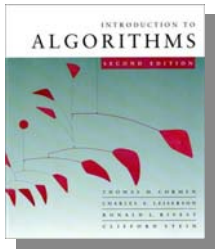



# Strassen'in algoritması

1. ***Böl:*** A ve B'yi $(n/2)\times(n/2)$ altmatrislere böl. + ve − kullanarak çarpılabilecek terimler oluştur.
2. ***Fethet:*** $(n/2)\times(n/2)$ altmatrislerde özyinelemeli 7 çarpma yap.
3. ***Birleştir:*** +ve − kullanarak $(n/2)\times(n/2)$ altmatrislerde $C$ 'yi oluştur.

$$T(n) = 7\,T(n/2) + \Theta(n^2)$$

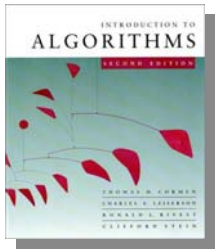
 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# Strassen'in çözümlenmesi

$$T(n) = 7\,T(n/2) + \Theta(n^2)$$

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 

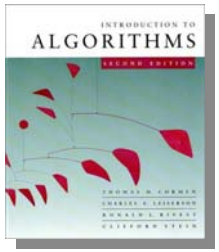



# Strassen'in çözümlenmesi

$$T(n) = 7\,T(n/2) + \Theta(n^2)$$

$n^{\log_b a} = n^{\log_2 7} \approx n^{2.81} \Rightarrow$ DURUM 1 $\Rightarrow T(n) = \Theta(n^{\lg 7})$.


*Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson*



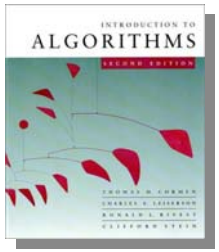

# Strassen'in çözümlenmesi

$$T(n) = 7\,T(n/2) + \Theta(n^2)$$

$n^{\log_b a} = n^{\log_2 7} \approx n^{2.81} \Rightarrow$ DURUM 1 $\Rightarrow T(n) = \Theta(n^{\lg 7})$.

2.81 değeri 3'den çok küçük görünmeyebilir ama, fark üstelde olduğu için, koşma süresine etkisi kayda değerdir. Aslında, $n \geq 32$ değerlerinde, Strassen'in algoritması günün makinelerinde normal algoritmadan daha hızlı çalışır.


*Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson*




# Strassen'in çözümlenmesi

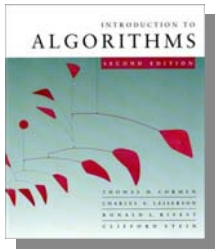
$$T(n) = 7\,T(n/2) + \Theta(n^2)$$

$n^{\log_b a} = n^{\log_2 7} \approx n^{2.81} \Rightarrow$ CASE 1 $\Rightarrow T(n) = \Theta(n^{\lg 7})$.

2.81 değeri 3' den çok küçük görünmeyebilir ama, fark üstelde olduğu için, yürütüm süresine etkisi kayda değerdir. Aslında, $n \geq 32$ değerlerinde Strassen'in algoritması günün makinelerinde normal algoritmadan daha hızlı çalışır.

**Bugünün en iyi değeri** (teorik merak açısından): $\Theta(n^{2.376\cdots})$.

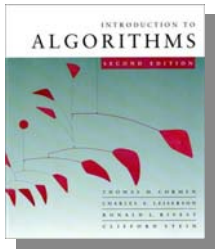

*Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson*




# VLSI yerleşimi

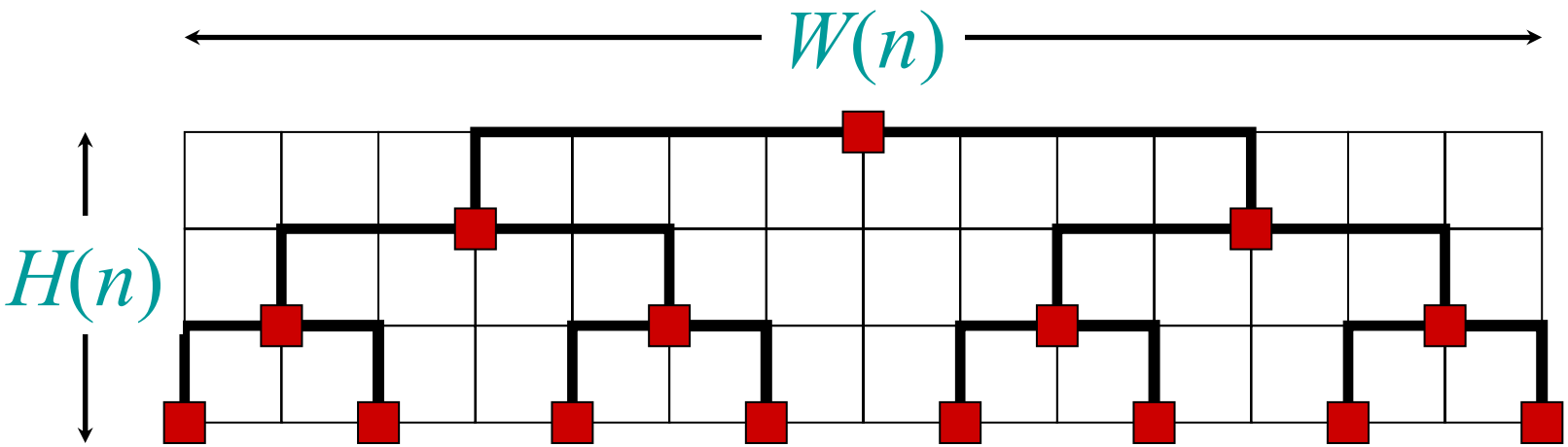
**Problem:** $n$ yaprağı olan tam bir ikili ağacı en az alan kullanarak bir ızgaraya gömmek.

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 

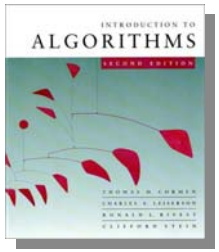



# VLSI yerleşimi

**Problem:** $n$ yaprağı olan tam bir ikili ağacı en az alan kullanarak bir ızgaraya gömmek.

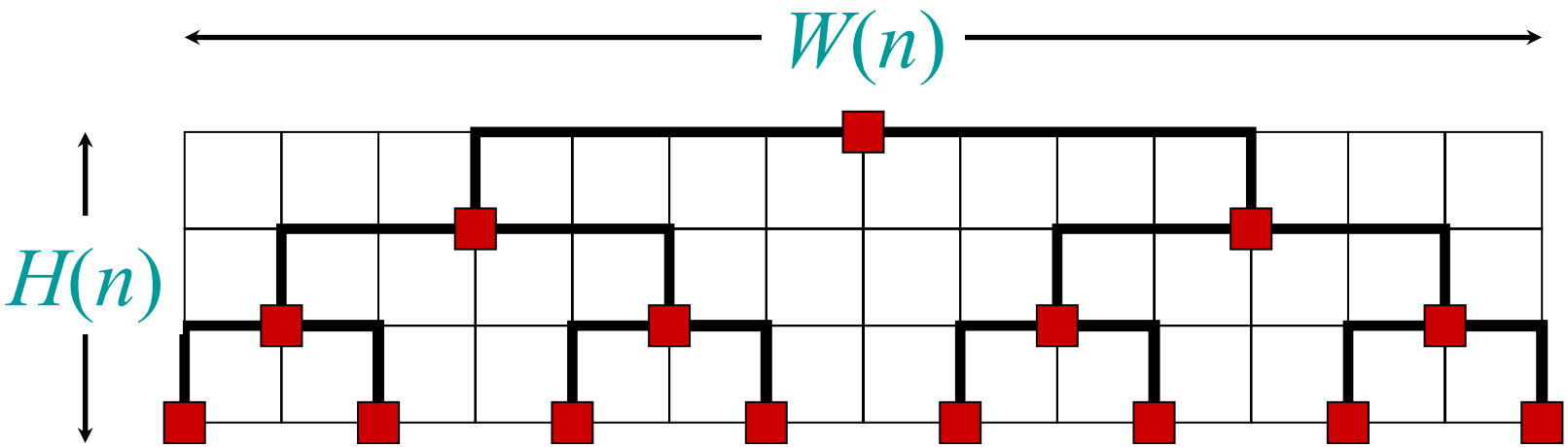



 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 



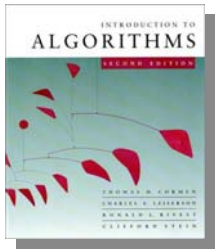

# VLSI yerleşimi

**Problem:** *n*yaprağı olan tam bir ikili ağacı en az alan kullanarak bir ızgaraya gömmek.



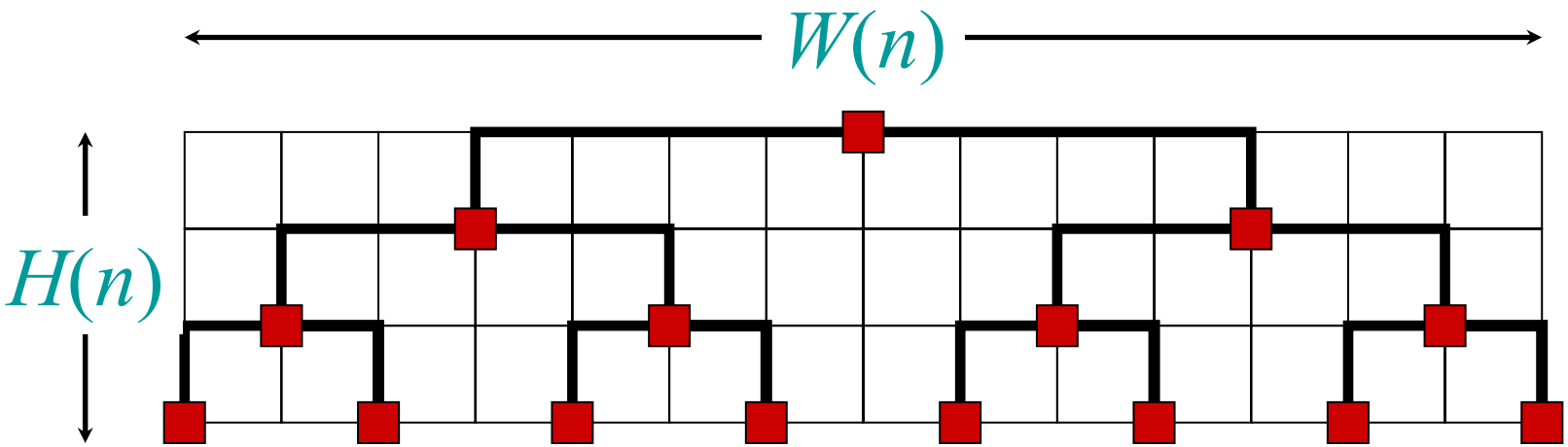

$$H(n) = H(n/2) + \Theta(1)$$
$$= \Theta(\lg n)$$

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# VLSI yerleşimi

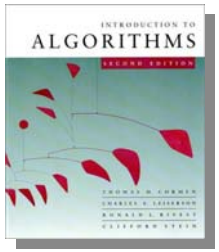
**Problem:** $n$ yaprağı olan tam bir ikili ağacı en az alan kullanarak bir ızgaraya gömmek.




$$H(n) = H(n/2) + \Theta(1) = \Theta(\lg n)$$

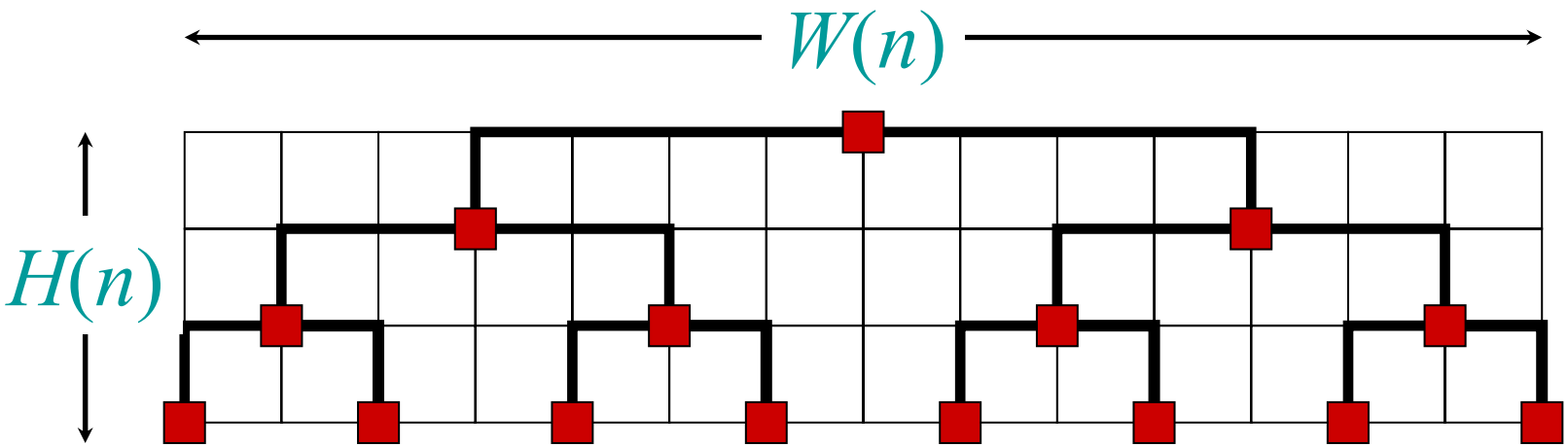
$$W(n) = 2\,W(n/2) + \Theta(1) = \Theta(n)$$

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 




# VLSI yerleşimi

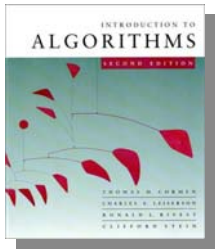
**Problem:** $n$ yaprağı olan tam bir ikili ağacı en az alan kullanarak bir ızgaraya gömmek.



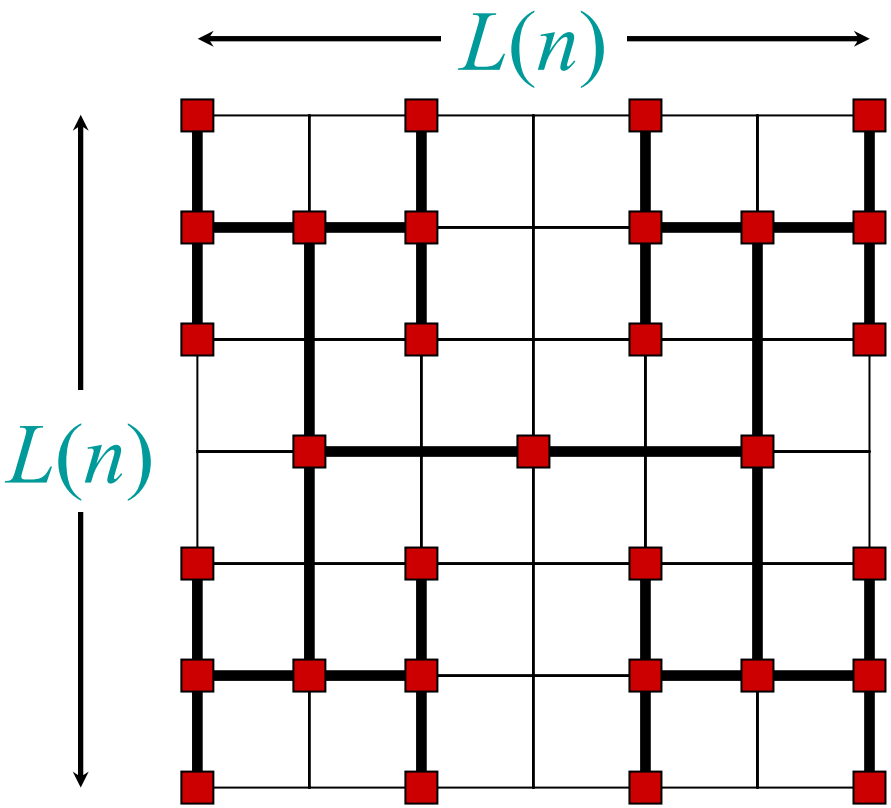

$$H(n) = H(n/2) + \Theta(1) = \Theta(\lg n) \qquad W(n) = 2\,W(n/2) + \Theta(1) = \Theta(n)$$

$$\text{Alan} = \Theta(n \lg n)$$

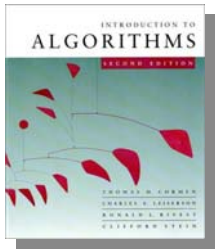

*Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson*



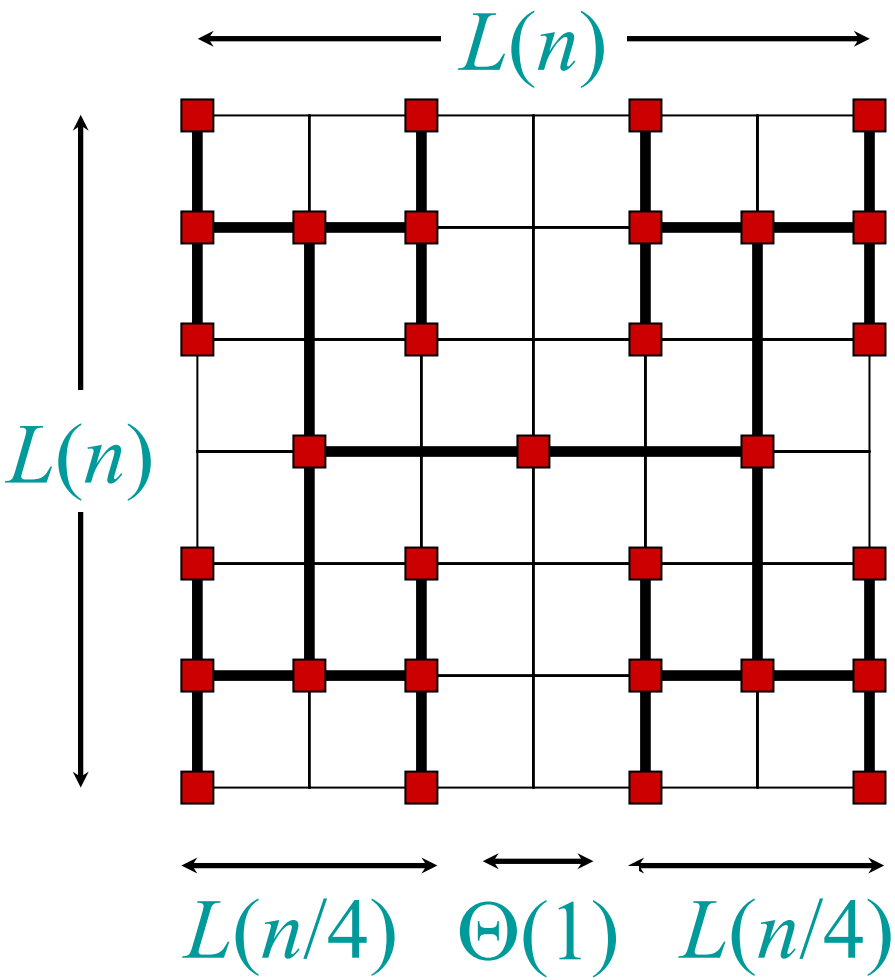

# H-ağacını gömme




 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 

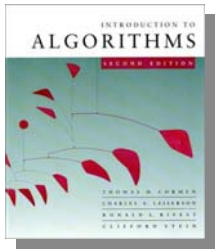



# H-ağacını gömme

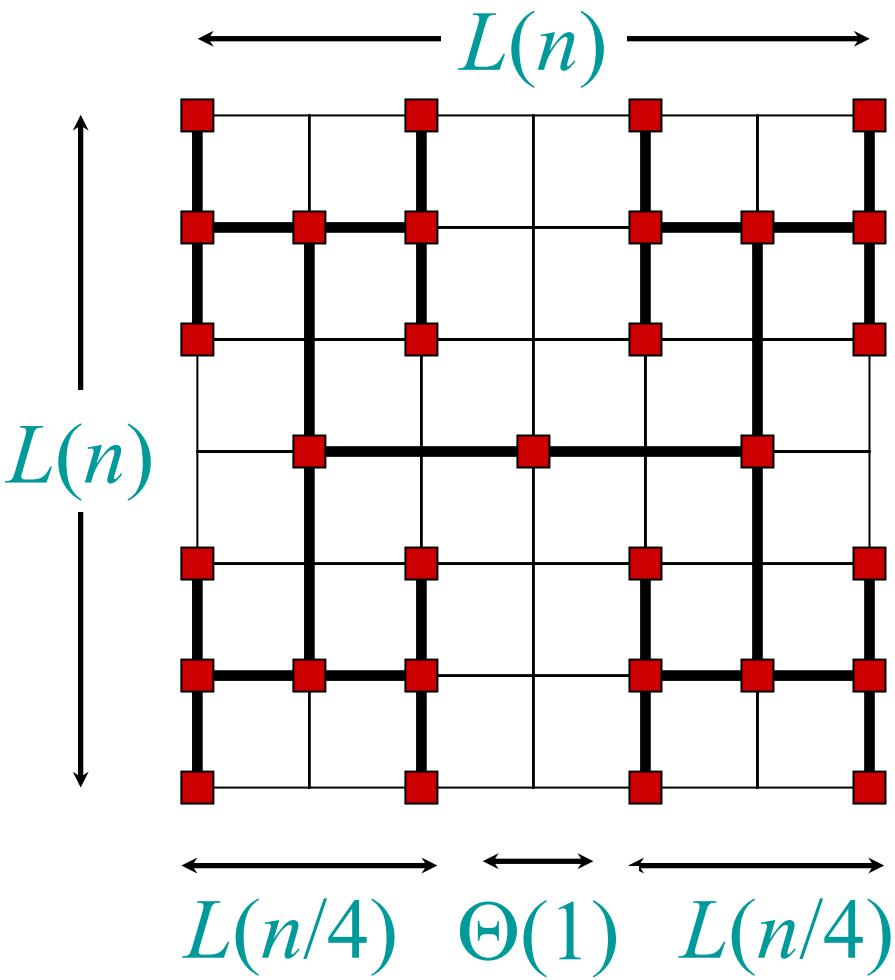




*Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson*



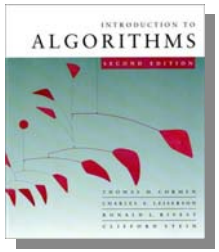

# H-ağacını gömme




$$L(n) = 2L(n/4) + \Theta(1)$$
$$= \Theta(\sqrt{n})$$

Alan $= \Theta(n)$


*Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson*




# Sonuç

- Böl ve Fethet algoritma tasarımının güçlü tekniklerinden sadece biridir.
- Böl ve Fethet algoritmaları yinelemeler ve Ana (Master) metot kullanarak çözümlenebilir. (bu nedenle bu matematiğin pratiğini yapın).
- Böl ve Fethet stratejisi genellikle verimli algoritmalara götürür.

 *Copyright © 2001-5 Erik D. Demaine and Charles E. Leiserson* 