Q. How many $h$’s cause $x$ and $y$ to collide?
A. There are $m$ choices for each of $a_1, a_2, \ldots, a_r$, but once these are chosen, exactly one choice for $a_0$ causes $x$ and $y$ to collide, namely

$$
\sum_{i=1}^{r} (x_i - y_i) = \left( x_0 - y_0 \right) 
$$

Thus, the number of $h$’s that cause $x$ and $y$ to collide is $m^r = |H|/m$. 

October 5, 2005 Copyright © 2001-5 by Erik D. Demaine and Charles E. Leiserson
The divide-and-conquer design paradigm

1. **Divide** the problem (instance) into subproblems.

2. **Conquer** the subproblems by solving them recursively.

3. **Combine** subproblem solutions.
Proof (completed)

Q. How many h's cause x and y to collide?

A. There are m choices for each of a_1, a_2, …, a_r, but once these are chosen, exactly one choice for a_0 causes x and y to collide, namely

\[ \sum_{i=1}^{r} \left( (x_i - y_i) \cdot a_i \right) \mod m. \]

Thus, the number of h's that cause x and y to collide is m^r = |H| / m.

October 5, 2005
Copyright © 2001-5 by Erik D. Demaine and Charles E. Leiserson
L7.15

Merge sort

1. **Divide**: Trivial.
2. **Conquer**: Recursively sort 2 subarrays.
3. **Combine**: Linear-time merge.
Merge sort

1. **Divide:** Trivial.
2. **Conquer:** Recursively sort 2 subarrays.
3. **Combine:** Linear-time merge.

\[ T(n) = 2T(n/2) + \Theta(n) \]

- # subproblems
- subproblem size
- work dividing and combining

October 5, 2005

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

L7.15
Proof (completed)

Q. How many $h$'s cause $x$ and $y$ to collide?

A. There are $m$ choices for each of $a_1, a_2, \ldots, a_r$, but once these are chosen, exactly one choice for $a_0$ causes $x$ and $y$ to collide, namely

$$a_0 \sum_{i=1}^{r} \cdot (x_i - y_i) \mod m.$$

Thus, the number of $h$'s that cause $x$ and $y$ to collide is $m^r = |H|/m$.

October 5, 2005
Copyright © 2001-5 by Erik D. Demaine and Charles E. Leiserson

---

Master theorem (reprise)

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

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

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

**Case 2:** $f(n) = \Theta(n^{log_b a \cdot \log k n})$, constant $k \geq 0$

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

**Case 3:** $f(n) = \Omega(n^{log_b a + \varepsilon})$, constant $\varepsilon > 0$, and regularity condition

$\Rightarrow T(n) = \Theta(f(n)).$
Master theorem (reprise)

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

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

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

**Case 2:** \( f(n) = \Theta(n^{\log_b a} \log^k n) \), constant \( k \geq 0 \)

\[ \Rightarrow T(n) = \Theta(n^{\log_b a} \log^{k+1} n) . \]

**Case 3:** \( f(n) = \Omega(n^{\log_b a + \varepsilon}) \), constant \( \varepsilon > 0 \),
and regularity condition

\[ \Rightarrow T(n) = \Theta(f(n)) . \]

**Merge sort:** \( a = 2, \ b = 2 \) \( \Rightarrow \ n^{\log_b a} = n^{\log_2 2} = n \)

\[ \Rightarrow \text{Case 2 } (k = 0) \Rightarrow T(n) = \Theta(n \log n) . \]
Proof (completed) 
Q. How many h'a's cause x and y to collide? 
A. There are m choices for each of a_1, a_2, …, a_r, but once these are chosen, exactly one choice for a_0 causes x and y to collide, namely
\[
\left\{ \begin{array}{l}
\sum_{i=1}^{r} \cdot (x_i - y_i) \\
= -a_0 \mod m
\end{array} \right.
\]
Thus, the number of h'a's that cause x and y to collide is m^r \cdot 1 = m^r = |H|/m. 
October 5, 2005 Copyright © 2001-5 by Erik D. Demaine and Charles E. Leiserson

**Binary search**

Find an element in a sorted array:

1. **Divide:** Check middle element.
2. **Conquer:** Recursively search 1 subarray.
3. **Combine:** Trivial.
Binary search

Find an element in a sorted array:

1. **Divide:** Check middle element.
2. **Conquer:** Recursively search 1 subarray.
3. **Combine:** Trivial.

**Example:** Find 9

3 5 7 8 9 12 15
Binary search

Find an element in a sorted array:

1. **Divide:** Check middle element.
2. **Conquer:** Recursively search 1 subarray.
3. **Combine:** Trivial.

**Example:** Find 9

| 3 | 5 | 7 | 8 | 9 | 12 | 15 |
Binary search

Find an element in a sorted array:

1. **Divide:** Check middle element.
2. **Conquer:** Recursively search 1 subarray.
3. **Combine:** Trivial.

**Example:** Find 9

\[
\begin{array}{ccccccc}
3 & 5 & 7 & 8 & 9 & 12 & 15 \\
\end{array}
\]
Binary search

Find an element in a sorted array:

1. *Divide:* Check middle element.
2. *Conquer:* Recursively search 1 subarray.
3. *Combine:* Trivial.

*Example:* Find 9

3 5 7 8 9 12 15
Binary search

Find an element in a sorted array:

1. Divide: Check middle element.
2. Conquer: Recursively search 1 subarray.

Example: Find 9

3 5 7 8 9 12 15

Proof (completed) 
Q. How many $h$'s cause $x$ and $y$ to collide? 
A. There are $m$ choices for each of $a_1, a_2, \ldots, a_r$, but once these are chosen, exactly one choice for $a_0$ causes $x$ and $y$ to collide, namely 

$\begin{bmatrix} r \\ \sum_{i=1}^{r} \cdot (x_i - y_i) \end{bmatrix} \mod m.$

Thus, the number of $h$'s that cause $x$ and $y$ to collide is $m^r \cdot 1 = m^r = |H|/m$. 

October 5, 2005 Copyright © 2001-5 by Erik D. Demaine and Charles E. Leiserson L7.15
Binary search

Find an element in a sorted array:

1. Divide: Check middle element.
2. Conquer: Recursively search 1 subarray.

Example: Find 9

3  5  7  8  9  12  15
Recurrence for binary search

\[ T(n) = \begin{cases} 1 & \text{if } n = 1 \\ T(n/2) + \Theta(1) & \text{otherwise} \end{cases} \]
Recurrence for binary search

\[ T(n) = 1 \cdot T(n/2) + \Theta(1) \]

- # subproblems
- subproblem size
- work dividing and combining

\[ n^{\log_b a} = n^{\log_2 1} = n^0 = 1 \Rightarrow \text{CASE 2} \ (k = 0) \]

\[ \Rightarrow T(n) = \Theta(\lg n) . \]
Powering a number

**Problem:** Compute $a^n$, where $n \in \mathbb{N}$.

**Naive algorithm:** $\Theta(n)$.
Powering a number

Problem: Compute \( a^n \), where \( n \in \mathbb{N} \).

Naive algorithm: \( \Theta(n) \).

Divide-and-conquer algorithm:

\[
\begin{align*}
  a^n &= \begin{cases} 
    a^{n/2} \cdot a^{n/2} & \text{if } n \text{ is even;} \\
    a^{(n-1)/2} \cdot a^{(n-1)/2} \cdot a & \text{if } n \text{ is odd.}
  \end{cases}
\end{align*}
\]
Powering a number

**Problem:** Compute \( a^n \), where \( n \in \mathbb{N} \).

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

**Divide-and-conquer algorithm:**

\[
a^n = \begin{cases} 
  a^{n/2} \cdot a^{n/2} & \text{if } n \text{ is even;} \\
  a^{(n-1)/2} \cdot a^{(n-1)/2} \cdot a & \text{if } n \text{ is odd.}
\end{cases}
\]

\[
T(n) = T(n/2) + \Theta(1) \implies T(n) = \Theta(\lg n).
\]
Fibonacci numbers

Recursive definition:

\[
F_n = \begin{cases} 
0 & \text{if } n = 0; \\
1 & \text{if } n = 1; \\
F_{n-1} + F_{n-2} & \text{if } n \geq 2.
\end{cases}
\]

0 1 1 2 3 5 8 13 21 34 …
Fibonacci numbers

Recursive definition:

\[ F_n = \begin{cases} 
0 & \text{if } n = 0; \\
1 & \text{if } n = 1; \\
F_{n-1} + F_{n-2} & \text{if } n \geq 2.
\end{cases} \]

0 1 1 2 3 5 8 13 21 34 \ldots

Naive recursive algorithm: \( \Omega(\phi^n) \)
(exponential time), where \( \phi = (1 + \sqrt{5})/2 \) is the \textit{golden ratio}. 

Proof (completed)

Q. How many \( h \)'s cause \( x \) and \( y \) to collide?

A. There are \( m \) choices for each of \( a_1, a_2, \ldots, a_r \), but once these are chosen, exactly one choice for \( a_0 \) causes \( x \) and \( y \) to collide, namely

\[
\begin{aligned}
\left( \begin{array}{c}
\sum_i \cdot (x_0 - y_0) \cdot (x_i - y_i) \\
\end{array} \right) & = \\
\left( \begin{array}{c}
\sum_i \cdot (x_0 - y_0) \\
\end{array} \right) & \mod m.
\end{aligned}
\]

Thus, the number of \( h \)'s that cause \( x \) and \( y \) to collide is \( m \cdot 1 = m \cdot \frac{|H|}{m} \).

October 5, 2005  
Copyright © 2001-5 by Erik D. Demaine and Charles E. Leiserson
Computing Fibonacci numbers

**Bottom-up:**

- Compute $F_0, F_1, F_2, \ldots, F_n$ in order, forming each number by summing the two previous.
- Running time: $\Theta(n)$. 

Proof (completed)

Q. How many $h$'s cause $x$ and $y$ to collide?

A. There are $m$ choices for each of $a_1, a_2, \ldots, a_r$, but once these are chosen, exactly one choice for $a_0$ causes $x$ and $y$ to collide, namely $a_0 = \sum_{i=1}^{r} (x_i - y_i) \mod m$. Thus, the number of $h$'s that cause $x$ and $y$ to collide is $m^r = |H|/m$. 

October 5, 2005 Copyright © 2001-5 by Erik D. Demaine and Charles E. Leiserson
Computing Fibonacci numbers

Bottom-up:

• Compute $F_0, F_1, F_2, \ldots, F_n$ in order, forming each number by summing the two previous.
• Running time: $\Theta(n)$.

Naive recursive squaring:

\[ F_n = \phi^n/\sqrt{5} \]

rounded to the nearest integer.

• Recursive squaring: $\Theta(\lg n)$ time.
• This method is unreliable, since floating-point arithmetic is prone to round-off errors.
Recursive squaring

Theorem: \[
\begin{pmatrix}
F_{n+1} & F_n \\
F_n & F_{n-1}
\end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^n.
\]
Recursive squaring

**Theorem:** \[
\begin{pmatrix}
F_{n+1} & F_n \\
F_n & F_{n-1}
\end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^n.
\]

**Algorithm:** Recursive squaring.
Time = \(\Theta(\lg n)\).
Recursive squaring

Theorem: \[
\begin{bmatrix}
F_{n+1} & F_n \\
F_n & F_{n-1}
\end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^n.
\]

Algorithm: Recursive squaring. 
Time = \Theta(\lg n).

Proof of theorem. (Induction on \( n \).)

Base (\( n = 1 \)): \[
\begin{bmatrix} F_2 & F_1 \\ F_1 & F_0 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^1.
\]
Recursive squaring

Inductive step \((n \geq 2)\):

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

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

\[
= \begin{bmatrix}
1 & 1
\end{bmatrix}^{n}
\]

\[
= \begin{bmatrix}
1 & 1 \\
1 & 0
\end{bmatrix}
\]
**Matrix multiplication**

**Input:** \( A = [a_{ij}] \), \( B = [b_{ij}] \)  

**Output:** \( C = [c_{ij}] = A \cdot B \) \[i, j = 1, 2, \ldots, n.\]

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

\[c_{ij} = \sum_{k=1}^{n} a_{ik} \cdot b_{kj}\]
Standard algorithm

\[
\text{for } i \leftarrow 1 \text{ to } n \\
\quad \text{do for } j \leftarrow 1 \text{ to } n \\
\quad \quad \text{do } c_{ij} \leftarrow 0 \\
\quad \quad \text{for } k \leftarrow 1 \text{ to } n \\
\quad \quad \quad \text{do } c_{ij} \leftarrow c_{ij} + a_{ik} \cdot b_{kj}
\]
Standard algorithm

for $i \leftarrow 1$ to $n$
  do for $j \leftarrow 1$ to $n$
    do $c_{ij} \leftarrow 0$
      for $k \leftarrow 1$ to $n$
        do $c_{ij} \leftarrow c_{ij} + a_{ik} \cdot b_{kj}$

Running time $= \Theta(n^3)$
**Divide-and-conquer algorithm**

**Idea:**

An $n \times n$ matrix is a $2 \times 2$ matrix of $(n/2) \times (n/2)$ submatrices:

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

\[
C = A \cdot B
\]

\[
\begin{align*}
  r &= ae + bg \\
  s &= af + bh \\
  t &= ce + dg \\
  u &= cf + dh
\end{align*}
\]

- 8 multiplications of $(n/2) \times (n/2)$ submatrices
- 4 additions of $(n/2) \times (n/2)$ submatrices
Divide-and-conquer algorithm

**Idea:**
n×n matrix = 2×2 matrix of (n/2)×(n/2) submatrices:

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

\[
C = A \cdot B
\]

\[
\begin{aligned}
r &= ae + bg \\
s &= af + bh \\
t &= ce + dh \\
u &= cf + dg
\end{aligned}
\]

*Recursive*

8 mults of (n/2)×(n/2) submatrices

4 adds of (n/2)×(n/2) submatrices
Proof (completed) 

Q. How many $h$'s cause $x$ and $y$ to collide? 

A. There are $m$ choices for each of $a_1, a_2, \ldots, a_r$, but once these are chosen, exactly one choice for $a_0$ causes $x$ and $y$ to collide, namely 

$$a_0 = \left( x^0 - y^0 \right) \cdot \left( \sum_{i=1}^{r} (x_i - y_i) \right) \mod m.$$

Thus, the number of $h$'s that cause $x$ and $y$ to collide is $m^r \cdot 1 = m^r = |H|/m$. 

October 5, 2005 Copyright © 2001-5 by Erik D. Demaine and Charles E. Leiserson L7.15
Proof (completed)

Q. How many h's cause x and y to collide?

A. There are m choices for each of \(a_1, a_2, \ldots, a_r\), but once these are chosen, exactly one choice for \(a_0\) causes x and y to collide, namely

\[
\begin{pmatrix}
\vdots \\
\end{pmatrix}
\sum_i \cdot (x_i - y_i) \mod m.
\]

Thus, the number of h's that cause x and y to collide is

\[m \cdot 1 = m = |H|/m.\]

October 5, 2005

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

L7.15

Analysis of D&C algorithm

\[T(n) = 8T(n/2) + \Theta(n^2)\]

\(\# \text{ submatrices} \quad \text{submatrix size} \quad \text{work adding submatrices}\)

\[n^{\log_b a} = n^{\log_2 8} = n^3 \Rightarrow \text{CASE 1} \Rightarrow T(n) = \Theta(n^3).\]
Proof (completed) 
Q. How many $h$'s cause $x$ and $y$ to collide?
A. There are $m$ choices for each of $a_1, a_2, \ldots, a_r$, but once these are chosen, exactly one choice for $a_0$ causes $x$ and $y$ to collide, namely

$$\begin{bmatrix}
\end{bmatrix}\begin{bmatrix}
\end{bmatrix}\begin{bmatrix}
\end{bmatrix}$$

Thus, the number of $h$'s that cause $x$ and $y$ to collide is $m \cdot 1 = m = |H|/m$. 

October 5, 2005 Copyright © 2001-5 by Erik D. Demaine and Charles E. Leiserson L7.15

Analysis of D&C algorithm

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

# submatrices

submatrix size

work adding submatrices

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

No better than the ordinary algorithm.
Strassen’s idea

- Multiply $2 \times 2$ matrices with only 7 recursive multiplications.
Strassen’s idea

- Multiply $2 \times 2$ matrices with only 7 recursive mults.

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

October 5, 2005  
Copyright © 2001-5 by Erik D. Demaine and Charles E. Leiserson
Strassen’s idea

- Multiply $2 \times 2$ matrices with only 7 recursive mults.

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

\[
r = P_5 + P_4 - P_2 + P_6
\]
\[
s = P_1 + P_2
\]
\[
t = P_3 + P_4
\]
\[
u = P_5 + P_1 - P_3 - P_7
\]
Strassen’s idea

- Multiply $2 \times 2$ matrices with only 7 recursive mults.

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

\[
\begin{align*}
r &= P_5 + P_4 - P_2 + P_6 \\
s &= P_1 + P_2 \\
t &= P_3 + P_4 \\
u &= P_5 + P_1 - P_3 - P_7
\end{align*}
\]

7 mults, 18 adds/subs.

**Note:** No reliance on commutativity of mult!
Strassen’s idea

- Multiply $2 \times 2$ matrices with only 7 recursive mults.

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

\[ r = P_5 + P_4 - P_2 + P_6 \]
\[ = (a + d)(e + h) \]
\[ + d(g - e) - (a + b)h \]
\[ + (b - d)(g + h) \]
\[ = ae + ah + de + dh \]
\[ + dg - de - ah - bh \]
\[ + bg + bh - dg - dh \]
\[ = ae + bg \]
Strassen’s algorithm

1. **Divide:** Partition $A$ and $B$ into $(n/2) \times (n/2)$ submatrices. Form terms to be multiplied using $+$ and $-$. 

2. **Conquer:** Perform 7 multiplications of $(n/2) \times (n/2)$ submatrices recursively.

3. **Combine:** Form $C$ using $+$ and $-$ on $(n/2) \times (n/2)$ submatrices.
Strassen’s algorithm

1. **Divide:** Partition $A$ and $B$ into $(n/2) \times (n/2)$ submatrices. Form terms to be multiplied using $+$ and $-$.

2. **Conquer:** Perform 7 multiplications of $(n/2) \times (n/2)$ submatrices recursively.

3. **Combine:** Form $C$ using $+$ and $-$ on $(n/2) \times (n/2)$ submatrices.

$$T(n) = 7 T(n/2) + \Theta(n^2)$$
Analysis of Strassen

\[ T(n) = 7 T(n/2) + \Theta(n^2) \]
Analysis of Strassen

\[ T(n) = 7 T(n/2) + \Theta(n^2) \]

\[ n \log ba = n \log_2 7 \approx n^{2.81} \implies \text{CASE 1} \implies T(n) = \Theta(n^{\log_2 7}). \]
Analysis of Strassen

\[ T(n) = 7 T(n/2) + \Theta(n^2) \]

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

The number 2.81 may not seem much smaller than 3, but because the difference is in the exponent, the impact on running time is significant. In fact, Strassen’s algorithm beats the ordinary algorithm on today’s machines for \( n \geq 32 \) or so.
Analysis of Strassen

\[ T(n) = 7 \ T(n/2) + \Theta(n^2) \]

\[ n^{\log_b a} = n^{\log_2 7} \approx n^{2.81} \quad \Rightarrow \quad \text{CASE 1} \quad \Rightarrow \quad T(n) = \Theta(n^{\log_2 7}). \]

The number 2.81 may not seem much smaller than 3, but because the difference is in the exponent, the impact on running time is significant. In fact, Strassen’s algorithm beats the ordinary algorithm on today’s machines for \( n \geq 32 \) or so.

**Best to date** (of theoretical interest only): \( \Theta(n^{2.3728...}) \).
VLSI layout

**Problem:** Embed a complete binary tree with \( n \) leaves in a grid using minimal area.
**VLSI layout**

**Problem:** Embed a complete binary tree with \( n \) leaves in a grid using minimal area.

\[ W(n) \]

\[ H(n) \]
Problem: Embed a complete binary tree with \( n \) leaves in a grid using minimal area.

\[
H(n) = H(n/2) + \Theta(1) = \Theta(\lg n)
\]
Proof (completed) 

Q. How many $h_a$'s cause $x$ and $y$ to collide? 
A. There are $m$ choices for each of $a_1, a_2, \ldots, a_r$, but once these are chosen, exactly one choice for $a_0$ causes $x$ and $y$ to collide, namely 

$$a_0 \sum_{i=1}^r (x_i - y_i) \mod m.$$ 

Thus, the number of $h_a$'s that cause $x$ and $y$ to collide is $m^r = |H|/m.$

---

VLSI layout

**Problem:** Embed a complete binary tree with $n$ leaves in a grid using minimal area.

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

\[ W(n) = 2W(n/2) + \Theta(1) \]
\[ = \Theta(n) \]
VLSI layout

**Problem:** Embed a complete binary tree with \( n \) leaves in a grid using minimal area.

\[
H(n) = H(n/2) + \Theta(1) = \Theta(\log n)
\]

\[
W(n) = 2W(n/2) + \Theta(1) = \Theta(n)
\]

Area = \( \Theta(n \log n) \)
H-tree embedding

\[ L(n) \]

Proof (completed)

Q. How many \( h \)'s cause \( x \) and \( y \) to collide?

A. There are \( m \) choices for each of \( a_1, a_2, \ldots, a_r \), but once these are chosen, exactly one choice for \( a_0 \) causes \( x \) and \( y \) to collide, namely

\[
\begin{bmatrix}
1 & 1 & \cdots & 1 \\
0 & 0 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & 0
\end{bmatrix} \cdot \begin{bmatrix}
x_0 - y_0 \\
x_1 - y_1 \\
\vdots \\
x_r - y_r
\end{bmatrix} \mod m.
\]

Thus, the number of \( h \)'s that cause \( x \) and \( y \) to collide is \( m^r \cdot 1 = m^r = |H|/m. \)

October 5, 2005
Copyright © 2001-5 by Erik D. Demaine and Charles E. Leiserson
Proof (completed)

Q. How many $h$'s cause $x$ and $y$ to collide?

A. There are $m$ choices for each of $a_1, a_2, \ldots, a_r$, but once these are chosen, exactly one choice for $a_0$ causes $x$ and $y$ to collide, namely

$$
\begin{bmatrix}
\sum_i a_i (x_i - y_i) \\
\end{bmatrix}
\mod m.
$$

Thus, the number of $h$'s that cause $x$ and $y$ to collide is $m r = |H|/m$.

October 5, 2005  Copyright © 2001-5 by Erik D. Demaine and Charles E. Leiserson

H-tree embedding

$L(n)$

$L(n)$

$L(n/4)$  $\Theta(1)$  $L(n/4)$
Q. How many $h$'s cause $x$ and $y$ to collide?

A. There are $m$ choices for each of $a_1, a_2, \ldots, a_r$, but once these are chosen, exactly one choice for $a_0$ causes $x$ and $y$ to collide, namely

\[
\begin{pmatrix}
\sum \cdot (x_0 - y_0)
\end{pmatrix}
\]

Thus, the number of $h$'s that cause $x$ and $y$ to collide is $m^r \cdot 1 = m^r = \frac{|H|}{m}$.

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

Area $= \Theta(n)$
Conclusion

- Divide and conquer is just one of several powerful techniques for algorithm design.
- Divide-and-conquer algorithms can be analyzed using recurrences and the master method (so practice this math).
- The divide-and-conquer strategy often leads to efficient algorithms.