Divide + Conquer Algorithms

- Brute force is generally inefficient
- Cut down work by pruning: eliminating infeasible solutions
- Even better: exploit some other structure to produce more efficient algorithms
- Divide + Conquer:
  - Generally presented as recursive
  - Cut the problem down
  - Combine solutions
  - Exploit structure

Review:

Divide + Conquer Algorithms

Merge Sort

Binary Search

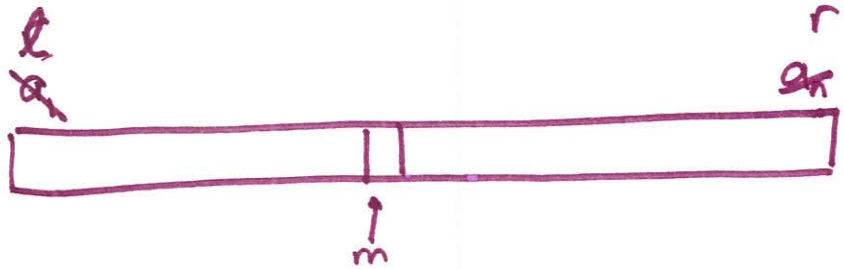
Quick Sort

Thinking Recursively

Linear Search: Given a collection $A = \{a_1, a_2, \cdots, a_n\}$

and a key element K

Output true if A contains K



$m = \left\lfloor \frac{l+r}{2} \right\rfloor$

$= k$ yes → stop, output true

no → recursively search the left half

recursively search the right half

RecLinearSearch (A, l, r, k)

if $(l = r)$

 └ return $A[l] = k$

else $m \leftarrow \left\lfloor \frac{l+r}{2} \right\rfloor$

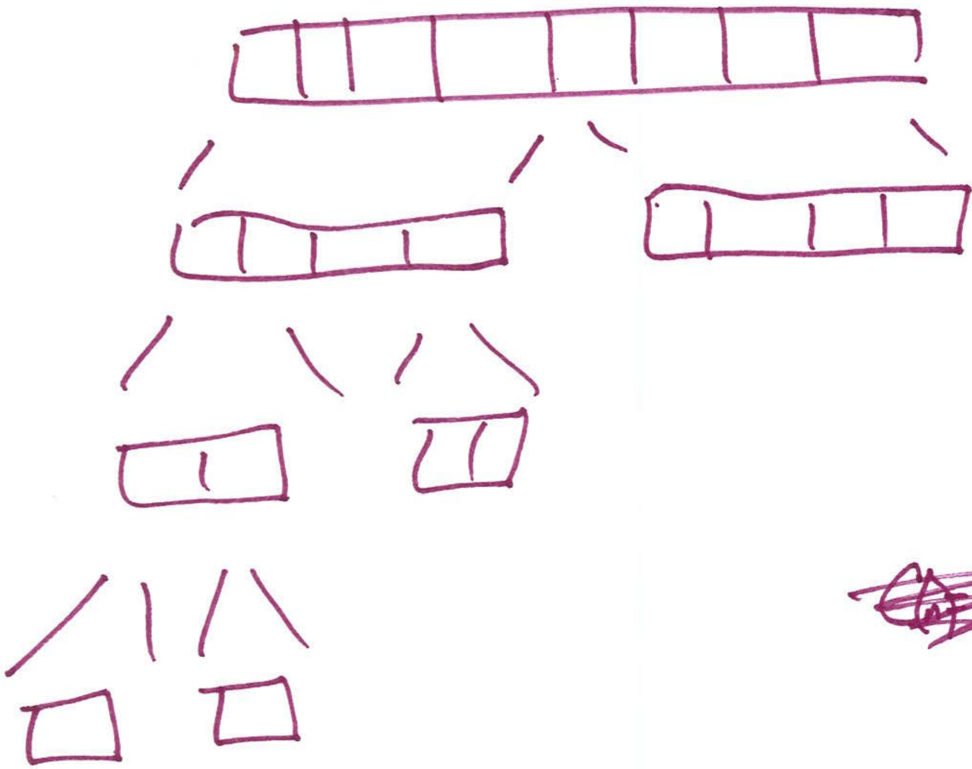
 if $A[m] = k$

 └ return true

 else

 └ return RecLinearSearch $(A, l, m-1, k)$ $\vee$ RecLinearSearch$(A, m+1, r, k)$

$C(1) = 1$



$C(n)$ = number of comparisons RLS on a collection of size n.

$$C(n) = C(n/2) + C(n/2) + 1$$

$$= 2C(n/2) + 1$$

$C(n) \in \Theta(n)$

$a = 2$

$b = 2$

$d = 0$

$1 \in \Theta(n^{0})$

$\log_2(2) = 1$

$a \;\_\_\; b^d$

$2 \geq 2^0$

Master Theorem

$$\text{Let} \quad T(n) = aT(n/b) + \overbrace{f(n)}^{\text{non-recursive work}}$$

how much the input is cut

# of recursive calls

$$\text{Let} \quad f(n) \in \Theta(n^d)$$

then

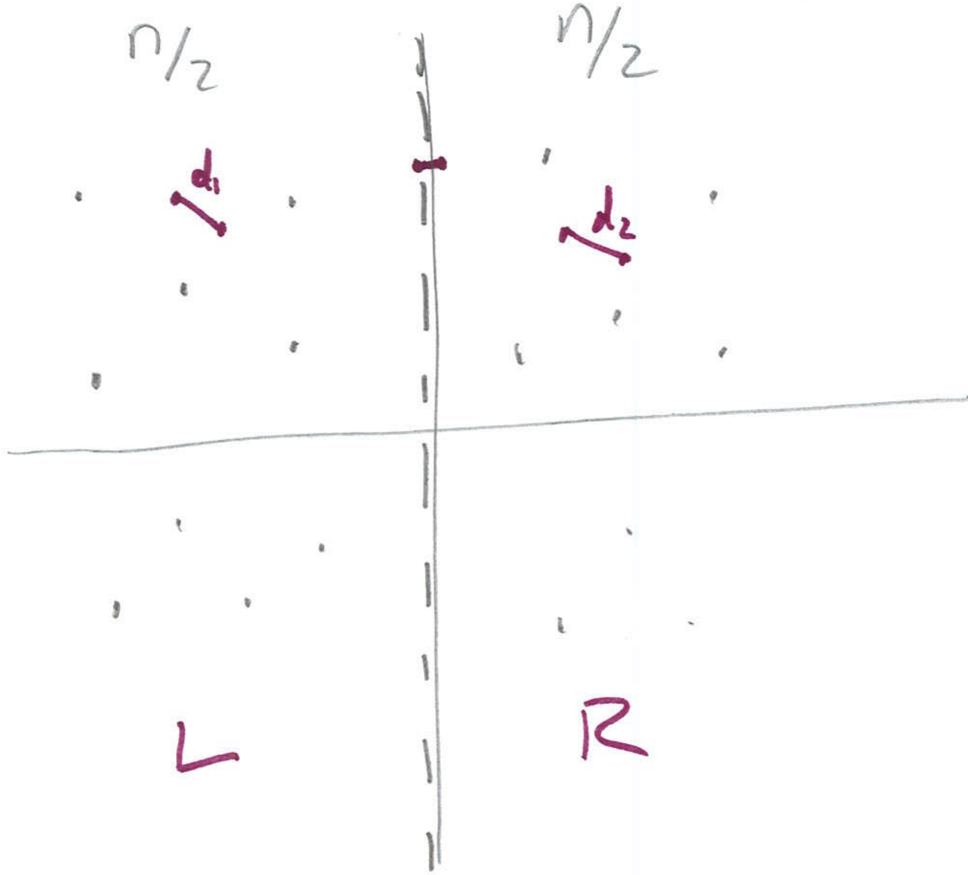
$$T(n) \in \begin{cases} \Theta(n^d) & a < b^d \\ \Theta(n^d \log(n)) & a = b^d \\ \Theta(n^{\log_b(a)}) & a > b^d \end{cases}$$

Closest Pair of Points

Given : $A = \{ (x_1 y_1), (x_2 y_2) \dots (x_n y_n) \}$

Output : the 2 closest points

$dist(p_1, p_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$



Divide : separate A into 2 partitions, L, R

Conquer : recursively find the 2 closest points in L, and the 2 closest in R

d_1 = min dist. of points in L

d_2 = min dist. of pts in R

$\delta = \min\{d_1, d_2\}$

Problem: there may be 2 closer points such that one is in L, and the other is in R

Naive Combine:

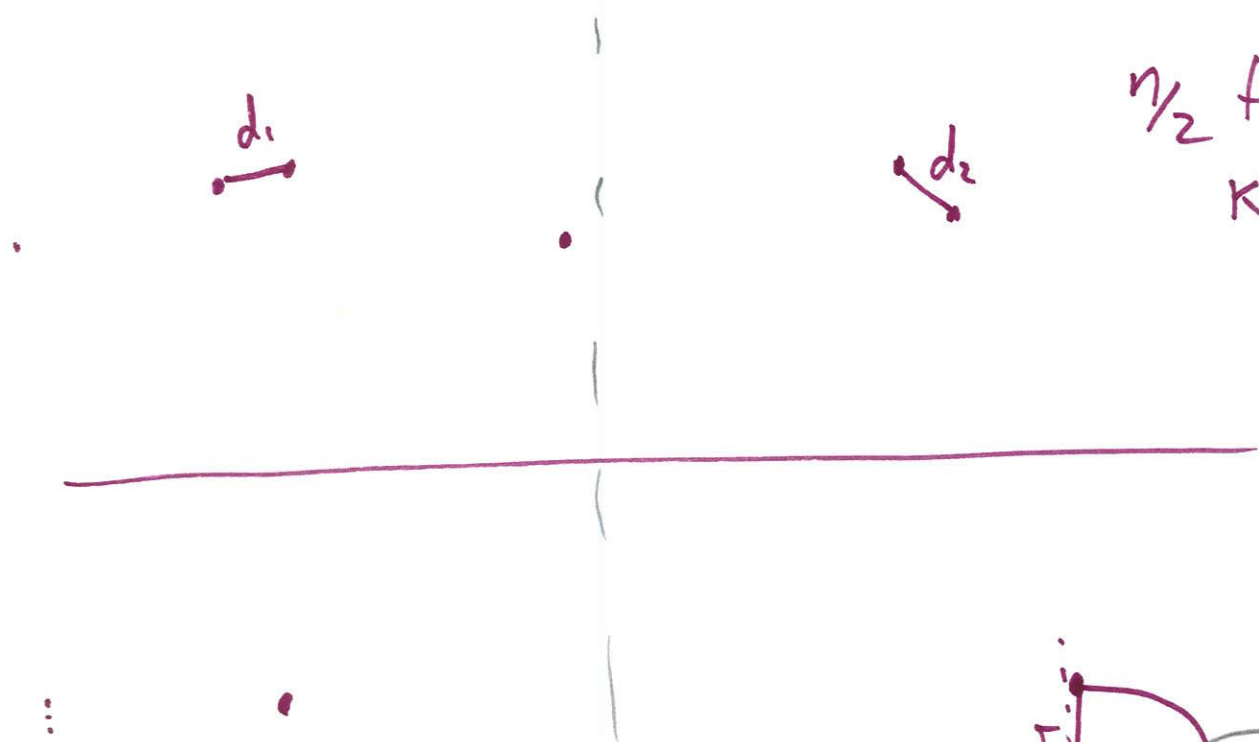
for each $p \in L$ — $n/2$ — $\frac{n}{2} \cdot \frac{n}{2} \in \Theta(n^2)$

  for each $q \in R$ — $n/2$

    if $(dist(p, q) < \delta)$

      update δ

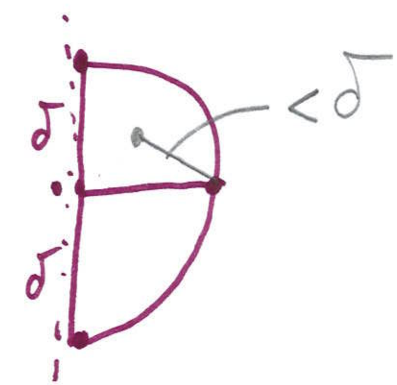
$\delta = \min\{d_1, d_2\}$

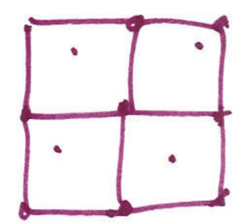


$n/2$ for each pt p in L:

K compare p to at most 4 points in R

$K \cdot n/2 \in \Theta(n)$





$C(n)$ = # of comparisons by
Divide + Conquer Closest Pair

$$= 2\,C(n/2) + cn$$

Master Theorem:

$a = 2$ $\qquad$ $a \_\_ b^d$

$b = 2$ $\qquad$ $2 \underline{=} 2^1$

$d = \underline{1}$

Case 2: $\Theta(n \log(n))$ quasilinear

Compute: $a^n \bmod m$

Input: a, n, m

```
int a, n, m;
 ...
int result = 1;
for(int i=0; i<n; i++){
    result = (result * a) % m;
}
```

$a = 3,\ m = 7$

$n = 77$

number of mults: 77

$n = 1{,}000{,}512$

mults: 1 million+

⋮

n mults: $n \in O(n)$

$= O(2^N)$

$n \approx 2^{1024}$

mults: 2^{1024}

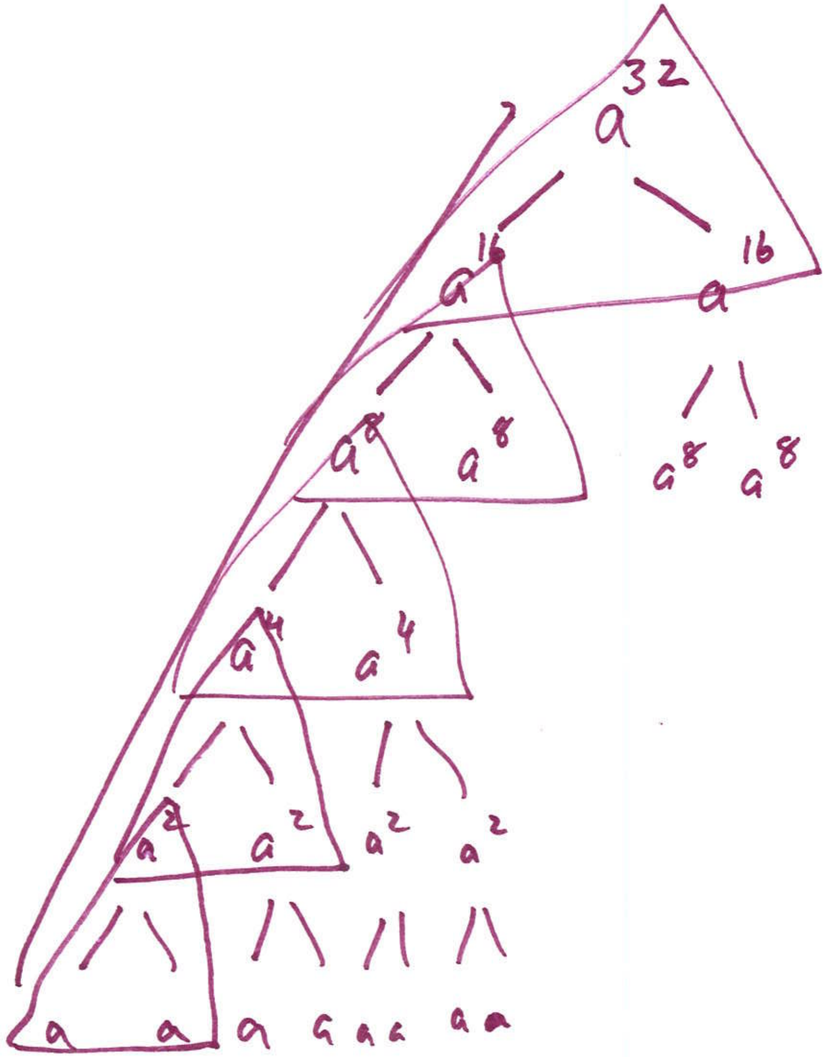
$\log_{10}(x)$

$\log_{10}(1000) = 3\ (+1)$

① input: n

② input size: # of bits to represent n

$N = \log(n)$



1 $a \cdot a = a^2$

2 $(a^2)^2 = a^4$

3 $(a^4)^2 = a^8$

4 $(a^8)^2 = a^{16}$

5 $(a^{16})^2 = a^{32}$

a^{32}

a^{16} a^2

$$a^{77} = a^{64} \cdot a^{13}$$

$$= a^{64} \cdot a^8 \cdot a^4 \cdot a^1$$

Repeated Squaring

$$\left(a^{2^k} \cdots a^4 a^2 a^1\right)$$

Input: a, n, m with $n = b_k b_{k-1} \cdots b_1 b_0$

Output: $a^n \bmod m$

```
term ← a
if (b_0 = 1)
  L prod ← a
else
  L prod ← 1
for i = 1...k
|   term ← (term × term) mod m    // (a^x)^2
|   if (b_i = 1)
L   L prod ← (prod × term) mod m

output prod.
```

$O(k)$, k = number of bits of n

$= \log(n)$

$O(\log(n))$?

input: n

input size:

$N = \log(n)$

$\therefore \Theta(N)$

(linear)

$12^{26} \bmod 17 = \underline{9}$

$a = 12$

$n = 26 = 11010$

$m = 17$

$\leftarrow$

1	1	0	1	0	(26)
4	3	2	1	-	i (k)
1	16	13	8	12	term
9	9	8	$\leftarrow$ 8	1	product

$12^2 \bmod 17$

$144 \bmod 17$

$64 \bmod 17$

$13^2 = 169 \bmod 17$

~~12~~ $128 \bmod 17$

$16^2 \bmod 17$

```
   3
   47
2     2
×  51
------
   47
+ 2350
------
```

235.0 vs 2350

×10

235.0

~~x · 2~~

7 × 2 = 14

1110 1110

```
int x;
⋮
x = x * 2;
```

~~x = x + x~~

```
x = x << 1;
```

shift

```
x >> 1
x >>> 1
x << 2
x << 3
```

$$47 \cdot 51 = (\overset{a}{4 \times 10^1} + \overset{b}{7}) \cdot (\overset{c}{5 \times 10^1} + \overset{d}{1})$$

$$= (a + b) \cdot (c + d)$$

FOIL

$$= \underset{1}{ac} + (\underset{2}{ad} + \underset{3}{bc}) + \underset{4}{bd}$$

$$(ad + bc) = \underset{1}{(a + b) \cdot (c + d)} - \underset{2}{ac} - \underset{3}{bd}$$

4 mults $\longrightarrow$ 3 mults

$$a = a_1 \cdot 10^{n/2} + a_0 \;,\quad b = b_1 \cdot 10^{n/2} + b_0$$

$\cancel{a \cdot b = (}$ 2. $a = \overbrace{\underbrace{\boxed{\;n/2\;}}_{a_1}\underbrace{\boxed{\;n/2 \text{ digits}\;}}_{a_0}}^{n \text{ digits}}$

$$a \cdot b = (a_1 \cdot 10^{n/2} + a_0)(b_1 \cdot 10^{n/2} + b_0)$$

$$= a_1 \cdot b_1 \cdot 10^n + a_1 \cdot b_0 \cdot 10^{n/2} + a_0 b_1 \cdot 10^{n/2} + a_0 b_0$$

$$= \underset{(1)}{a_1 b_1 10^n} + \Big[\underset{(2)}{(a_1 + a_0)(b_1 + b_0)} - a_1 b_1 - \underset{(3)}{a_0 b_0} \Big] 10^{n/2} + a_0 b_0$$

2 n-digit numbers:

Naive (Elem. school) Method:

$$\begin{array}{r} a_1 \, a_2 \, a_3 \cdots a_n \\ \times \quad b_1 \, b_2 \, b_3 \cdots b_n \\ \hline \end{array} \Big] \longrightarrow O(n^2) \text{ multiplications}$$

Karatsuba Multiplication:

split the input in half,

recursively make 3 multiplications

perform 0 non-recursive mults

2 shifts (free)

6 additions

$M(n)$ = number of mults by Karatsuba on 2 n-digit numbers

$= 3M(n/2) + 0$

$a = 3$ $\quad 3 \geq 2^0 = 1$

$b = 2$ $\quad$ case 3:

$d = 0$

$$M(n) = \Theta(n^{\log_2(3)})$$

$$\approx \Theta(n^{1.58\ldots})$$

Karatsuba : 1960

Toom - Cook : 1963 : $O(n^{1.465})$

Schönhage - Strassen 1968 : Fast Fourier Transform $O(n \cdot \log(n) \cdot \log\log(n))$

Fürer 2007 $O(n \log(n) 2^{\log^*(n)})$

$O(n)$

$\log_2(64) = 6$

$\log_2(6) = 2. \ldots$

$\log(2\ldots) = 1. \ldots$

$\log(1\ldots) \leq 1$

Matrix Multiplication

2×2: 8 mults

4 additions

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \cdot \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} ae + bg & af + bh \\ ce + dg & cf + dh \end{bmatrix}$$

$n \times n$ matrix multiplication

$A \cdot B = C$

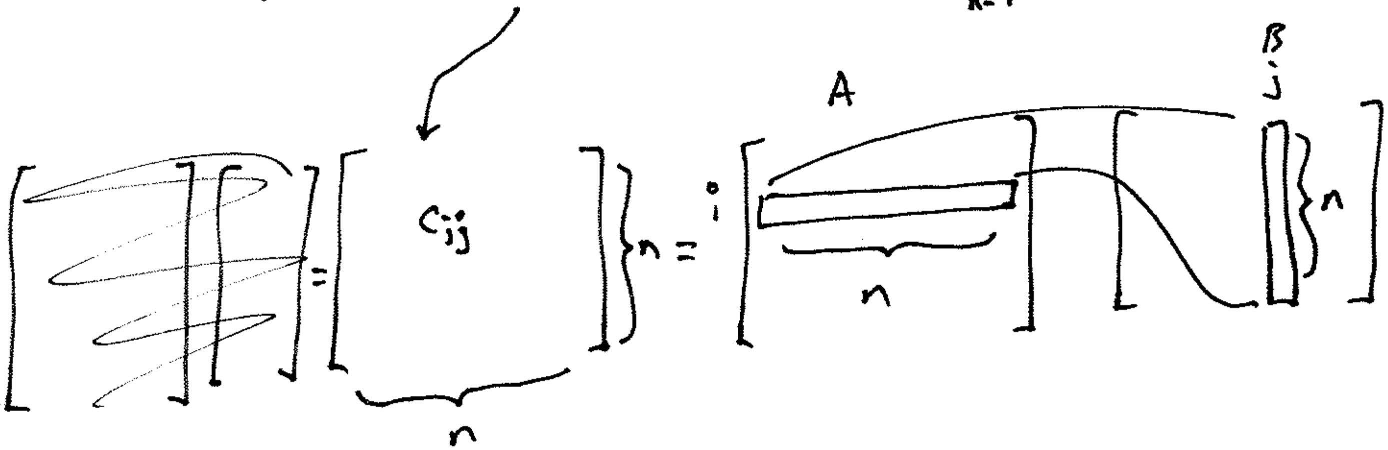
$$c_{ij} = \sum_{k=1}^{n} a_{ik} \cdot b_{kj}$$

1 entry:

<u>n</u> multiplications

<u>n-1</u> addition

n^2 total entries



in total: $\left.\begin{array}{l} n^3 \text{ multiplications} \\ n^2(n-1) \text{ additions} \end{array}\right\} O(n^3)$

trivial lower bound: $O(n^2)$

input: 2 matrices

input size: n^2 "linear" w.r.t. input size

$$A \left\{ \overbrace{\left[\begin{array}{c|c} A_{00} & A_{01} \\ \hline A_{10} & A_{11} \end{array}\right]}^{n} \right. \qquad B \overbrace{\left[\begin{array}{c|c} \phantom{A_{00}} & \phantom{A_{01}} \\ \hline \phantom{A_{10}} & \phantom{A_{11}} \end{array}\right]}^{n}$$

A, B: $n \times n$ matrices $\qquad n^2$

A_{00} -- etc : $\frac{n}{2} \times \frac{n}{2}$ $\qquad \frac{n^2}{4}$ ← half as big $\left(\frac{n}{2}\right)^2$

half as big

Strassen's Matrix Multiplication

$$\begin{bmatrix} c_{00} & c_{01} \\ c_{10} & c_{11} \end{bmatrix} = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix} \begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \end{bmatrix}$$

$$= \begin{bmatrix} m_1 + m_4 - m_5 + m_7 & m_3 + m_5 \\ m_2 + m_4 & m_1 + m_3 - m_2 + m_6 \end{bmatrix}$$

	mults	adds	
$m_1 = (a_{00} + a_{11}) \cdot (b_{00} + b_{11})$	1	2	
$m_2 = (a_{10} + a_{11}) \cdot b_{00}$	1	1	
$m_3 = a_{00} \cdot (b_{01} - b_{11})$	1	1	
$m_4 = a_{11} \cdot (b_{10} - b_{00})$	1	1	
$m_5 = (a_{00} + a_{01}) \cdot b_{11}$	1	1	
$m_6 = (a_{10} - a_{00}) \cdot (b_{00} + b_{01})$	1	2	
$m_7 = (a_{01} - a_{11}) \cdot (b_{10} + b_{11})$	1	2	+8

total: 7 mults.
18 additions

- in practice:

  - non-square matrices: $\begin{bmatrix} a & d \\ b & e \\ c & f \end{bmatrix} \cdot \begin{bmatrix} g & h & i \\ j & k & l \end{bmatrix}$

you can padout with zeros:

$$\begin{bmatrix} a & d & 0 & 0 \\ b & e & 0 & 0 \\ c & f & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} g & h & i & 0 \\ j & k & l & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} =$$

$$\begin{bmatrix} x_1 & y_2 & z_3 & 0 \\ x_2 & y_3 & z & 0 \\ x & y & z & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$ unpad the solution.

Observations:

Let $M(n) =$ number of multiplications by Strassen on 2 $n \times n$ matrices

$= 7M(n/2) + 0$ (non recursive multiplications)

$M(1) = 1$

$a = 7$

$b = 2$

$d = 0$

$a \_ b^d$

$7 \geq 2^0$

case 3 of the Master Theorem:

$M(n) \in \Theta(n^{\log_b(a)})$

$= \Theta(n^{\log_2(7)})$

$= \Theta(n^{2.807\ldots})$

$A(n) =$ number of additions by Strassen on ~~2~~ 2 $n \times n$ Matrices

$$= 7A(n/2) + \underbrace{18}_{\text{additions/subtractions}}\underbrace{\left(\frac{n}{2}\right)^2}_{\text{\# of entries}} \text{ — } O(n^2)$$

$a = 7$

$b = 2$

$d = 2$

$a \_ b^d$

$7 \geq 2^2 = 4$

case 3:

$A(n) \in \Theta(n^{2.807})$

Winogard : $O(n^{2.375477})$

$\downarrow$

Williams: $O(n^{2.3727})$

$\downarrow$

? $O(n^2)$

$\Theta(n\log(n))$ upper bound

$\parallel$

$\Theta(n\log(n))$ lower bound

~~$\Theta(kn)$~~