Lecture 7: Divide & Conquer 2

Integer Multiplication

& Matrix Multiplication

CS 341: Algorithms

Tuesday, Jan 29th 2019

Outline For Today

1. Integer Multiplication
2. Matrix Multiplication

Outline For Today

1. Integer Multiplication
2. Matrix Multiplication

Integer Multiplication

- Input: 2 n-digit integers, X, Y
- Output Z = XY
- E.g: X = 2345 and Y = 6789 then Z = 15920205
- Will work in base 10 b/c it is easier to think about base 10
- Same argument & analysis as in base 2 (done in handout)
- Warning: There is no intuition to the DC algorithm we'll see

Grade School Algorithm

$$
\begin{array}{r}
\boxed{2}\boxed{3}\boxed{4}\boxed{5} \\
678\boxed{9} \\
\times \quad \\
\hline
21105
\end{array}
$$

Grade School Algorithm

```
     2345
     6789
x
-----------
    21105
   18760   ← shift by 1 digit
```

Grade School Algorithm

$$
\begin{array}{r}
\boxed{2}\boxed{3}\boxed{4}\boxed{5} \\
6\boxed{7}89 \\
\times \quad\quad\quad \\
\hline
21105 \\
18760 \\
16415
\end{array}
$$

18760 ⟵ shift by 1 digit

16415 ⟵ shift by 2 digits

Grade School Algorithm

```
       2345
       6789
  x
  ----------
      21105
     18760   <- shift by 1 digit
    16415    <- shift by 2 digits
   14070     <- shift by 3 digits
 +
 ----------
   15920205
```

Observe that even the total work for shifting is $(1+2+3 + \ldots + n-1) = O(n^2)$!

- Multiplying two digits => 1 operation
- Addition of two digits => 1 operation
- Shift by 1 digit => 1 operation (shift by x digits, x ops)

TOTAL WORK: $O(n^2)$

Question: Can we do better?

- Upshot: A set of mysterious looking multiplications, additions and subtractions giving the correct answer!
- $X = (a10^{n/2} + b)$; $Y = (c10^{n/2} + d)$

$$a \leftarrow \boxed{23}\,\boxed{45} \rightarrow b$$

$$c \leftarrow \boxed{67}\,\boxed{89} \rightarrow d$$

each of a, b, c, d is of size n/2

- Ex: a=23, b=45, so $2345 = 23*10^{n/2} + 45$
- $XY = ac10^{n} + ad10^{n/2} + bc10^{n/2} + bd$
- $XY = \boxed{ac10^{n} + (ad + bc)10^{n/2} + bd}$

Call this expression (★)

DC-Multiplication-1

```
procedure DC-Mult1(X, Y both n digit numbers):
   Base Case: if (X or Y is single digit): …
   set a, b, c, d defined as before
   ac = DC-Mult(a, c)
   ad = DC-Mult(a, d)
   bc = DC-Mult(b, c)
   bd = DC-Mult(b, d)
   return ac10^n + (ad + bc)10^(n/2) + bd
```

$O(n)$

shifts: $O(n)$

3 additions of n digit numbers: $O(n)$

Total Work Outside of Recursive Calls: $O(n)$

Recurrence: $4T(n/2) + O(n)$

Total Runtime: $O(n^2)$

Not better than Grade School Algorithm

Observation

Observation: We care about only 3 quantities in (★):

$ac10^n + (ad + bc)10^{n/2} + bd$

(1) ac => with 10^n padding

(2) (ad + bc) => with $10^{n/2}$ padding

(3) bd => with 10 padding

Question: If we care about 3 quantities, can we get these quantities with only 3 recursive calls?

Karatsuba-Ofman Algorithm (1962)

$\boxed{ac}10^n + \boxed{(ad + bc)}10^{n/2} + \boxed{bd}$

(1) ac as before

(2) bd as before

(3) (a + b) (c + d) = (ac + ad + bc + bd)

Observation: (3) – (2) – (1) = (ac+ad+bc+bd)-ac–bd = ad + bc

```
procedure KO(X, Y both n digit numbers):
   Base Case: if (X or Y is single digit): …
   set a, b, c, d defined as before
   ac = KO(a, c)
   bd = KO(b, d)
   X = KO(a+b, c+d)
   return (ac)10^n + (X-ac-bd)10^{n/2} + bd
```

Karatsuba-Ofman Algorithm (1962)

```
procedure KO(X, Y both n digit numbers):
    Base Case: if (X or Y is single digit): …
    set a, b, c, d defined as before            O(n)
    ac = KO(a, c)
    bd = KO(b, d)                               O(n)
    X = KO(a+b, c+d)
    return (ac)10^n + (X – ac - bd)10^(n/2) + bd    O(n)
```

Total Work Outside of Recursive Calls: O(n)

Recurrence: $3T(n/2) + O(n)$

Total Runtime: $O(n^{\log\_2(3)}=n^{1.59})$

Facts About Multiplication

Fact 1: (By Toom & Stephen Cook): Can generalize KO to divide X and Y

into k pieces instead of 2 (Math gets very messy) and get

$O(n^{log\_k(2k-1)})$ => $O(n^{1+\varepsilon})$ for any $\varepsilon > 1$

Stephen Cook is a Canadian Computer Scientist currently @ UToronto.

Fact 2: Best known alg: O(nlog(n)loglog(n)) (Schonhage & Strassen)

Outline For Today

1. Integer Multiplication
2. Matrix Multiplication

Matrix Multiplication

- Input: 2 n x n matrices A, B
- Output: C = AxB

$$
\begin{array}{c} \\ \\ i \\ \\ \\ \end{array}
\begin{bmatrix}
a_{11} & a_{12} & \cdots & \cdots & a_{1n} \\
a_{21} & a_{22} & \cdots & \cdots & a_{2n} \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
a_{n1} & a_{n2} & \cdots & \cdots & a_{nn}
\end{bmatrix}
\;\mathrm{X}\;
\begin{array}{c}
\begin{array}{ccccc} \phantom{b_{11}} & \phantom{b_{12}} & j & & \phantom{b_{1n}} \end{array} \\
\begin{bmatrix}
b_{11} & b_{12} & \cdots & \cdots & b_{1n} \\
b_{21} & b_{22} & \cdots & \cdots & b_{2n} \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
b_{n1} & b_{n2} & \cdots & \cdots & b_{nn}
\end{bmatrix}
\end{array}
=
\begin{bmatrix}
c_{11} & c_{12} & \cdots & \cdots & c_{1n} \\
c_{21} & c_{22} & \cdots & \cdots & c_{2n} \\
\cdots & \cdots & \boxed{c_{ij}} & \cdots & \cdots \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
c_{n1} & c_{n2} & \cdots & \cdots & c_{nn}
\end{bmatrix}
$$

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj} = (a_{i1}b_{1j} + a_{i2}b_{2j} + \ldots + a_{in}b_{nj})$$

Q: By definition, how many multip. and additions needed to compute c_{ij}?

A: n multiplications, n-1 additions

Then there are $O(n^3)$ basic operations (by definition) to compute C.

- Warning: Again, no intuition to the DC algorithm we'll see!

Standard Algorithm

$$\begin{bmatrix} a_{11} & a_{12} & \dots & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & \dots & a_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & \dots & a_{nn} \end{bmatrix} \text{X} \begin{bmatrix} b_{11} & b_{12} & \dots & \dots & b_{1n} \\ b_{21} & b_{22} & \dots & \dots & b_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ b_{n1} & b_{n2} & \dots & \dots & b_{nn} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \dots & \dots & c_{1n} \\ c_{21} & c_{22} & \dots & \dots & c_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ c_{n1} & c_{n2} & \dots & \dots & c_{nn} \end{bmatrix}$$

Standard Algorithm

$$\begin{bmatrix} a_{11} & a_{12} & \dots & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & \dots & a_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & \dots & a_{nn} \end{bmatrix} \mathrm{X} \begin{bmatrix} b_{11} & b_{12} & \dots & \dots & b_{1n} \\ b_{21} & b_{22} & \dots & \dots & b_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ b_{n1} & b_{n2} & \dots & \dots & b_{nn} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \dots & \dots & c_{1n} \\ c_{21} & c_{22} & \dots & \dots & c_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ c_{n1} & c_{n2} & \dots & \dots & c_{nn} \end{bmatrix}$$

Standard Algorithm

$$\begin{bmatrix} a_{11} & a_{12} & \ldots & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & \ldots & a_{2n} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ a_{n1} & a_{n2} & \ldots & \ldots & a_{nn} \end{bmatrix} \text{ X } \begin{bmatrix} b_{11} & b_{12} & \ldots & \ldots & b_{1n} \\ b_{21} & b_{22} & \ldots & \ldots & b_{2n} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ b_{n1} & b_{n2} & \ldots & \ldots & b_{nn} \end{bmatrix} = \begin{bmatrix} c_{11} & \boxed{c_{12}} & \ldots & \ldots & c_{1n} \\ c_{21} & c_{22} & \ldots & \ldots & c_{2n} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ c_{n1} & c_{n2} & \ldots & \ldots & c_{nn} \end{bmatrix}$$

Standard Algorithm

$$
\begin{bmatrix}
a_{11} & a_{12} & \dots & \dots & a_{1n} \\
a_{21} & a_{22} & \dots & \dots & a_{2n} \\
\dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots \\
a_{n1} & a_{n2} & \dots & \dots & a_{nn}
\end{bmatrix}
\text{X}
\begin{bmatrix}
b_{11} & b_{12} & \dots & \dots & b_{1n} \\
b_{21} & b_{22} & \dots & \dots & b_{2n} \\
\dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots \\
b_{n1} & b_{n2} & \dots & \dots & b_{nn}
\end{bmatrix}
=
\begin{bmatrix}
c_{11} & c_{12} & \boxed{\dots} & \dots & c_{1n} \\
c_{21} & c_{22} & \dots & \dots & c_{2n} \\
\dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots \\
c_{n1} & c_{n2} & \dots & \dots & c_{nn}
\end{bmatrix}
$$

Standard Algorithm

$$\begin{bmatrix} a_{11} & a_{12} & \dots & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & \dots & a_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & \dots & a_{nn} \end{bmatrix} \times \begin{bmatrix} b_{11} & b_{12} & \dots & \dots & b_{1n} \\ b_{21} & b_{22} & \dots & \dots & b_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ b_{n1} & b_{n2} & \dots & \dots & b_{nn} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \dots & \boxed{\dots} & c_{1n} \\ c_{21} & c_{22} & \dots & \dots & c_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ c_{n1} & c_{n2} & \dots & \dots & c_{nn} \end{bmatrix}$$

Standard Algorithm

$$
\begin{bmatrix}
a_{11} & a_{12} & \dots & \dots & a_{1n} \\
a_{21} & a_{22} & \dots & \dots & a_{2n} \\
\dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots \\
a_{n1} & a_{n2} & \dots & \dots & a_{nn}
\end{bmatrix}
\text{X}
\begin{bmatrix}
b_{11} & b_{12} & \dots & \dots & b_{1n} \\
b_{21} & b_{22} & \dots & \dots & b_{2n} \\
\dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots \\
b_{n1} & b_{n2} & \dots & \dots & b_{nn}
\end{bmatrix}
=
\begin{bmatrix}
c_{11} & c_{12} & \dots & \dots & \boxed{c_{1n}} \\
c_{21} & c_{22} & \dots & \dots & c_{2n} \\
\dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots \\
c_{n1} & c_{n2} & \dots & \dots & c_{nn}
\end{bmatrix}
$$

Standard Algorithm

$$\begin{bmatrix} a_{11} & a_{12} & \dots & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & \dots & a_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & \dots & a_{nn} \end{bmatrix} \times \begin{bmatrix} b_{11} & b_{12} & \dots & \dots & b_{1n} \\ b_{21} & b_{22} & \dots & \dots & b_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ b_{n1} & b_{n2} & \dots & \dots & b_{nn} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \dots & \dots & c_{1n} \\ \boxed{c_{21}} & c_{22} & \dots & \dots & c_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ c_{n1} & c_{n2} & \dots & \dots & c_{nn} \end{bmatrix}$$

Standard Algorithm

$$\begin{bmatrix} a_{11} & a_{12} & \dots & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & \dots & a_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & \dots & a_{nn} \end{bmatrix} \times \begin{bmatrix} b_{11} & b_{12} & \dots & \dots & b_{1n} \\ b_{21} & b_{22} & \dots & \dots & b_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ b_{n1} & b_{n2} & \dots & \dots & b_{nn} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \dots & \dots & c_{1n} \\ c_{21} & \boxed{c_{22}} & \dots & \dots & c_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ c_{n1} & c_{n2} & \dots & \dots & c_{nn} \end{bmatrix}$$

Standard Algorithm

$$\begin{bmatrix} a_{11} & a_{12} & \dots & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & \dots & a_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & \dots & a_{nn} \end{bmatrix} \times \begin{bmatrix} b_{11} & b_{12} & \dots & \dots & b_{1n} \\ b_{21} & b_{22} & \dots & \dots & b_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ b_{n1} & b_{n2} & \dots & \dots & b_{nn} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \dots & \dots & c_{1n} \\ c_{21} & c_{22} & \dots & \dots & c_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ c_{n1} & c_{n2} & \dots & \dots & \boxed{c_{nn}} \end{bmatrix}$$

```
procedure StandardMM(A, B nxn matrices):
    C = int[n][n];
    for i = 1…n:
        for j = 1…n:
            C[i][j] = 0;
            for k=1…n:
                C[i][j] += a_ik b_kj
    return C
```

Runtime: $O(n^3)$

Question: Can we Divide & Conquer?

$$\begin{bmatrix} a_{11} & a_{12} & \dots & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & \dots & a_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & \dots & a_{nn} \end{bmatrix} \mathrm{X} \begin{bmatrix} b_{11} & b_{12} & \dots & \dots & b_{1n} \\ b_{21} & b_{22} & \dots & \dots & b_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ b_{n1} & b_{n2} & \dots & \dots & b_{nn} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \dots & \dots & c_{1n} \\ c_{21} & c_{22} & \dots & \dots & c_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ c_{n1} & c_{n2} & \dots & \dots & c_{nn} \end{bmatrix}$$

Question: Can we Divide & Conquer?

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \times \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

- Where A_{11}, …, A_{22}, B_{11}, …, B_{22}, & C_{11}, …, C_{22} are n/2 x n/2 matrices.

Fact: When you split matrices into blocks & multiply, the blocks behave as they are atomic elements.

$$C_{ij} = \sum_{k=1}^{n} A_{ik}B_{kj} = (A_{i1}B_{1j} \boxed{+} A_{i2}B_{2j} \boxed{+} .. \boxed{+} A_{in}B_{nj})$$

where + is matrix addition operation (coordinate-wise addition)

Matrix Addition

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

Matrix Addition

$$
\begin{bmatrix}
a_{11} & \boxed{a_{12}} & \dots & \dots & a_{1n} \\
a_{21} & a_{22} & \dots & \dots & a_{2n} \\
\dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots \\
a_{n1} & a_{n2} & \dots & \dots & a_{nn}
\end{bmatrix}
+
\begin{bmatrix}
b_{11} & \boxed{b_{12}} & \dots & \dots & b_{1n} \\
b_{21} & b_{22} & \dots & \dots & b_{2n} \\
\dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots \\
b_{n1} & b_{n2} & \dots & \dots & b_{nn}
\end{bmatrix}
=
\begin{bmatrix}
c_{11} & \boxed{c_{12}} & \dots & \dots & c_{1n} \\
c_{21} & c_{22} & \dots & \dots & c_{2n} \\
\dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots \\
c_{n1} & c_{n2} & \dots & \dots & c_{nn}
\end{bmatrix}
$$

Matrix Addition

$$\begin{bmatrix} a_{11} & a_{12} & \dots & \dots & \boxed{a_{1n}} \\ a_{21} & a_{22} & \dots & \dots & a_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & \dots & a_{nn} \end{bmatrix} + \begin{bmatrix} b_{11} & b_{12} & \dots & \dots & \boxed{b_{1n}} \\ b_{21} & b_{22} & \dots & \dots & b_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ b_{n1} & b_{n2} & \dots & \dots & b_{nn} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \dots & \dots & \boxed{c_{1n}} \\ c_{21} & c_{22} & \dots & \dots & c_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ c_{n1} & c_{n2} & \dots & \dots & c_{nn} \end{bmatrix}$$

$O(n^2)$ operation

Example of Block Multiplication

$$\left[\begin{array}{cc|cc} 2 & 4 & 3 & 2 \\ 3 & 5 & 4 & 2 \\ \hline 6 & 5 & 9 & 3 \\ 2 & 2 & 3 & 5 \end{array}\right] \times \left[\begin{array}{cc|cc} 3 & 2 & 5 & 2 \\ 2 & 4 & 8 & 4 \\ \hline 5 & 2 & 9 & 3 \\ 3 & 4 & 2 & 2 \end{array}\right] = \left[\begin{array}{c|c} & \\ \hline & \end{array}\right]$$

$2*3 + 4*2 + 3*5 + 2*3 = 35$

Example of Block Multiplication

$$\left[\begin{array}{cc|cc} 2 & 4 & 3 & 2 \\ 3 & 5 & 4 & 2 \\ \hline 6 & 5 & 9 & 3 \\ 2 & 2 & 3 & 5 \end{array}\right] \times \left[\begin{array}{cc|cc} 3 & 2 & 5 & 2 \\ 2 & 4 & 8 & 4 \\ \hline 5 & 2 & 9 & 3 \\ 3 & 4 & 2 & 2 \end{array}\right] = \left[\begin{array}{cc|cc} 35 & & & \\ & & & \\ \hline & & & \\ & & & \end{array}\right]$$

$2*2 + 4*4 + 3*2 + 2*4 = 34$

Example of Block Multiplication

2	4	3	2
3	5	4	2
6	5	9	3
2	2	3	5

X

3	2	5	2
2	4	8	4
5	2	9	3
3	4	2	2

=

35	34		

3*3 + 5*2 + 4*5 + 2*3 = 45

Example of Block Multiplication

$$\left[\begin{array}{cc|cc} 2 & 4 & 3 & 2 \\ 3 & 5 & 4 & 2 \\ \hline 6 & 5 & 9 & 3 \\ 2 & 2 & 3 & 5 \end{array}\right] \times \left[\begin{array}{cc|cc} 3 & 2 & 5 & 2 \\ 2 & 4 & 8 & 4 \\ \hline 5 & 2 & 9 & 3 \\ 3 & 4 & 2 & 2 \end{array}\right] = \left[\begin{array}{cc|cc} 35 & 34 & & \\ 45 & \boxed{} & & \\ \hline & & & \\ & & & \end{array}\right]$$

3*2 + 5*4 + 4*2 + 2*4 = 42

Example of Block Multiplication

$$\left[\begin{array}{cc|cc} \boxed{2 & 4 & 3 & 2} \\ 3 & 5 & 4 & 2 \\ \hline 6 & 5 & 9 & 3 \\ 2 & 2 & 3 & 5 \end{array}\right] \times \left[\begin{array}{cc|cc} 3 & 2 & 5 & 2 \\ 2 & 4 & 8 & 4 \\ \hline 5 & 2 & 9 & 3 \\ 3 & 4 & 2 & 2 \end{array}\right] = \left[\begin{array}{cc|c} 35 & 34 & \\ 45 & 42 & \\ \hline & & \end{array}\right]$$

$$A_{11}B_{11} = \begin{bmatrix} 2 & 4 \\ 3 & 5 \end{bmatrix} \times \begin{bmatrix} 3 & 2 \\ 2 & 4 \end{bmatrix} = \begin{bmatrix} 14 & 20 \\ 19 & 26 \end{bmatrix}$$

$$A_{12}B_{21} = \begin{bmatrix} 3 & 2 \\ 4 & 2 \end{bmatrix} \times \begin{bmatrix} 5 & 2 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} 21 & 14 \\ 26 & 16 \end{bmatrix}$$

$$\begin{bmatrix} 14 & 20 \\ 19 & 26 \end{bmatrix} + \begin{bmatrix} 21 & 14 \\ 26 & 16 \end{bmatrix} = \boxed{\begin{bmatrix} 35 & 34 \\ 45 & 42 \end{bmatrix}}$$

DC-1 Matrix Multiplication

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \text{X} \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

procedure DC1-MM(A, B nxn matrices):

    C_{11} = DC1-MM(A_{11}, B_{11}) + DC1-MM(A_{12}, B_{21})

DC-1 Matrix Multiplication

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \times \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

procedure DC1-MM(A, B nxn matrices):

  C_{11} = DC1-MM(A_{11}, B_{11}) + DC1-MM(A_{12}, B_{21})

  C_{12} = DC1-MM(A_{11}, B_{12}) + DC1-MM(A_{12}, B_{22})

DC-1 Matrix Multiplication

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \times \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

procedure DC1-MM(A, B nxn matrices):

$\quad C_{11}$ = DC1-MM(A_{11}, B_{11}) + DC1-MM(A_{12}, B_{21})

$\quad C_{21}$ = DC1-MM(A_{11}, B_{12}) + DC1-MM(A_{12}, B_{22})

$\quad C_{21}$ = DC1-MM(A_{21}, B_{11}) + DC1-MM(A_{22}, B_{21})

DC-1 Matrix Multiplication

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \mathrm{X} \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

procedure DC1-MM(A, B nxn matrices):

C_{11} = DC1-MM(A_{11}, B_{11}) + DC1-MM(A_{12}, B_{21})

C_{21} = DC1-MM(A_{11}, B_{12}) + DC1-MM(A_{12}, B_{22})

C_{12} = DC1-MM(A_{21}, B_{11}) + DC1-MM(A_{22}, B_{21})

C_{22} = DC1-MM(A_{21}, B_{12}) + DC1-MM(A_{22}, B_{22})

return C = $C_{11}C_{21}C_{12}C_{22}$

Outside recursion: 4 additions of n/2xn/2 matrices=$4n^2/4=n^2$ work

Runtime: $T(n) = 8T(n/2) + O(n^2) = O(n^3)$ (by Master Thm)

So not faster than standard method!

Strassen's (Mysterious) Algorithm (1969)

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \times \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} = \begin{bmatrix} C_{11} = Z_5 + Z_4 - Z_2 + Z_6 & \boxed{C_{12} = Z_1 + Z_2} \\ C_{21} = Z_3 + Z_4 & C_{22} = Z_5 + Z_1 - Z_3 - Z_7 \end{bmatrix}$$

$$\begin{aligned}
Z_1 &= A_{11}(B_{12} - B_{22}) \\
Z_2 &= (A_{11} + A_{12})B_{22} \\
Z_3 &= (A_{21} + A_{22})B_{11} \\
Z_4 &= A_{22}(B_{21} + B_{11}) \\
Z_5 &= (A_{11} + A_{22})(B_{11} + B_{22}) \\
Z_6 &= (A_{12} - A_{22})(B_{21} + B_{22}) \\
Z_7 &= (A_{11} - A_{21})(B_{11} + B_{12})
\end{aligned}$$

Exercise: Check the rest of the quadrants to believe Strassen!

Ex: $C_{12} = Z_1 + Z_2 = A_{11}B_{12} - A_{11}B_{22} + A_{11}B_{22} + A_{12}B_{22} = A_{11}B_{12} + A_{12}B_{22}$

Strassen's (Mysterious) Algorithm

A_{11}	A_{12}	X	B_{11}	B_{12}	=	$C_{11} = Z_5 + Z_4 - Z_2 + Z_6$	$C_{12}=Z_1 + Z_2$
A_{21}	A_{22}		B_{21}	B_{22}		$C_{21}=Z_3 + Z_4$	$C_{22}=Z_5 + Z_1 - Z_3 - Z_7$

```
procedure Strassen(A, B nxn matrices):
    Z_1 = Strassen(A_11, (B_12 – B_22)); Z_2 = Strassen((A_11 + A_12), B_22)
    Z_3 = Strassen((A_21 + A_22), B_11); Z_4 = Strassen(A_22, (B_21 + B_11))
    Z_5 = Strassen((A_11 + A_22), (B_11 + B_22))
    Z_6 = Strassen((A_12 - A_22), (B_21 + B_22))
    Z_7 = Strassen((A_11 – A_21), (B_11 + B_12))
    C_11=Z_5 + Z_4 -Z_2 + Z_6; C_12=Z_1 + Z_2; C_21=Z_3 + Z_4; C_22=Z_5 + Z_1 - Z_3 – Z_7
    return C = C_11C_21C_12C_22
```

Runtime: $T(n) = 7T(n/2) + O(n^2) = O(n^{\log\_2(7)} = n^{2.81})$ (by Master Thm)

(Asymptotically) much faster than standard algorithm!

History of Matrix Multiplication

Runtime	Year	Authors
$O(n^3)$	until 1969	N/A
$O(n^{2.81})$	1969	Strassen
$O(n^{2.796})$	1978	Pan
$O(n^{2.780})$	1979	Bini et. al.
$O(n^{2.522})$	1981	Schoenhage
$O(n^{2.517})$	1982	Romani
$O(n^{2.496})$	1983	Coppersmith & Winograd
$O(n^{2.479})$	1986	Strassen
$O(n^{2.376})$	1989	Coppersmith and Winograd
$O(n^{2.374})$	2011	Stothers
$O(n^{2.3728642})$	2012	V. Williams
$O(n^{2.3728639})$	2013	Le Gall