MONASH University

MONASH
INFORMATION
TECHNOLOGY

FIT2004
Algorithms and Data Structures

Ian Wern Han Lim
lim.wern.han@monash.edu

Referencing materials by
Rafael Dowsley, Nathan Companez, Aamir Cheema, Arun Konagurthu and Lloyd Allison

GROUP OF EIGHT AUSTRALIA

Faculty of Information Technology, Monash University

Ready?

Agenda

- Grade School Integer Multiplication
- Quick Integer Multiplication
  - Karatsuba algorithm (1960, 1962)
  - Schönhage–Strassen algorithm (1971)
    - Popular for matrix multiplication
  - … and many more including a 2019 one in UNSW

Agenda

- Grade School Integer Multiplication
- Quick Integer Multiplication
  - Karatsuba algorithm (1960, 1962)
  - Schönhage–Strassen algorithm (1971)
    - Popular for matrix multiplication
  - … and many more including a 2019 one in UNSW

- Divide and Conquer
  - Simple recap on MergeSort
  - Simple recap on QuickSort

MONASH University

Let us begin!

Grade School Multiplication

An algorithm we all know

- Recall how you multiply from grade school

Grade School Multiplication

An algorithm we all know

- Recall how you multiply from grade school
  - It is an algorithm
  - It is something that you do use till today, right?

Grade School Multiplication

An algorithm we all know

- Recall how you multiply from grade school
  - It is an algorithm
  - It is something that you do use till today, right?

- Consider the following 2 inputs
  - $x = 123$
  - $y = 345$
  - Multiple x with y, what is the answer?

Grade School Multiplication

An algorithm we all know

- Recall how you multiply from grade school
  - It is an algorithm
  - It is something that you do use till today, right?
  - What is your algorithm?
    1. Loop through y from right to left.
    2. For each integer of y, multiply with each integer in x from right to left.
    3. If overflow exist, add to the left value

- Consider the following 2 inputs
  - x = 123
  - y = 345
  - Multiple x with y, what is the answer?

Grade School Multiplication

An algorithm we all know

- Recall how you multiply from grade school
  - It is an algorithm
  - It is something that you do use till today, right?
  - What is your algorithm?
    1. Loop through y from right to left.
    2. For each integer of y, multiply with each integer in x from right to left.
    3. If overflow exist, add to the left value
    4. …

    You can write it in a better way, and code it

- Consider the following 2 inputs
  - $x = 123$
  - $y = 345$
  - Multiple *x* with *y*, what is the answer?

Grade School Multiplication

An algorithm we all know

- Consider the following 2 inputs
  - *x* = 123
  - *y* = 345
  - Multiple *x* with *y*, what is the answer?

- Have your teacher ever…

$$
\begin{array}{rccccc}
 & & & 1 & 2 & 3 \\
\times & & & 3 & 4 & 5 \\
\hline
 & & & 6 & 1 & 5 \\
 & & 4 & 9 & 2 & \\
 & 3 & 6 & 9 & & \\
\hline
 & 4 & 2 & 4 & 3 & 5
\end{array}
$$

Grade School Multiplication

An algorithm we all know

- Consider the following 2 inputs
  - x = 123
  - y = 345
  - Multiple x with y, what is the answer?

$$
\begin{array}{cccccc}
 & & & 1 & 2 & 3 \\
\times & & & 3 & 4 & 5 \\
\hline
 & & & 6 & 1 & 5 \\
 & & 4 & 9 & 2 & \\
 & 3 & 6 & 9 & & \\
\hline
 & 4 & 2 & 4 & 3 & 5
\end{array}
$$

- Have your teacher ever…
  - Explain why it work?

Grade School Multiplication
An algorithm we all know

- Consider the following 2 inputs
  - x = 123
  - y = 345
  - Multiple x with y, what is the answer?

$$
\begin{array}{rccccc}
 & & & 1 & 2 & 3 \\
\times & & & 3 & 4 & 5 \\
\hline
 & & & 6 & 1 & 5 \\
 & & 4 & 9 & 2 & \\
 & 3 & 6 & 9 & & \\
\hline
 & 4 & 2 & 4 & 3 & 5
\end{array}
$$

- Have your teacher ever…
  - Explain why it work? For every possible number combinations?

Grade School Multiplication

An algorithm we all know

- Consider the following 2 inputs
  - x = 123
  - y = 345
  - Multiple x with y, what is the answer?

$$
\begin{array}{rccccc}
 & & & 1 & 2 & 3 \\
\times & & & 3 & 4 & 5 \\
\hline
 & & & 6 & 1 & 5 \\
 & & 4 & 9 & 2 & \\
 & 3 & 6 & 9 & & \\
\hline
 & 4 & 2 & 4 & 3 & 5
\end{array}
$$

- Have your teacher ever…
  - Explain why it work? For every possible number combinations?
  - How efficient it is?

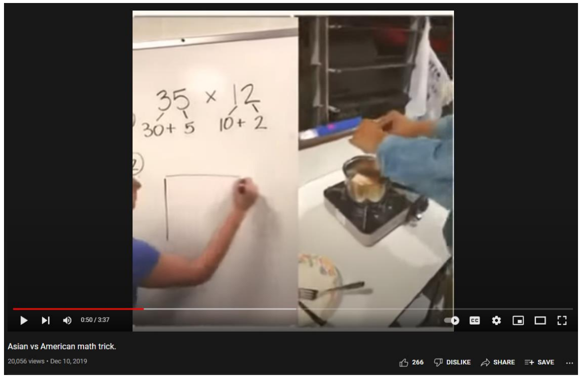
Grade School Multiplication

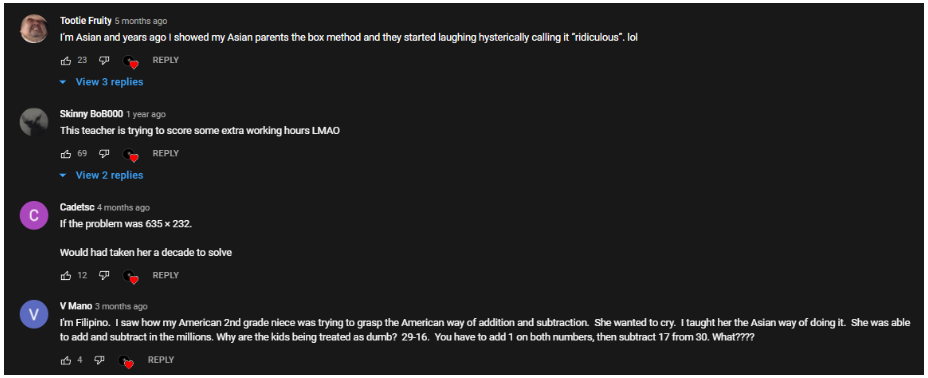
An algorithm we all know

- Consider the following 2 inputs
  - $x = 123$
  - $y = 345$
  - Multiple x with y, what is the answer?

$$
\begin{array}{r}
1\;2\;3 \\
\times \quad 3\;4\;5 \\
\hline
6\;1\;5 \\
4\;9\;2\; \\
3\;6\;9\;\; \\
\hline
4\;2\;4\;3\;5
\end{array}
$$

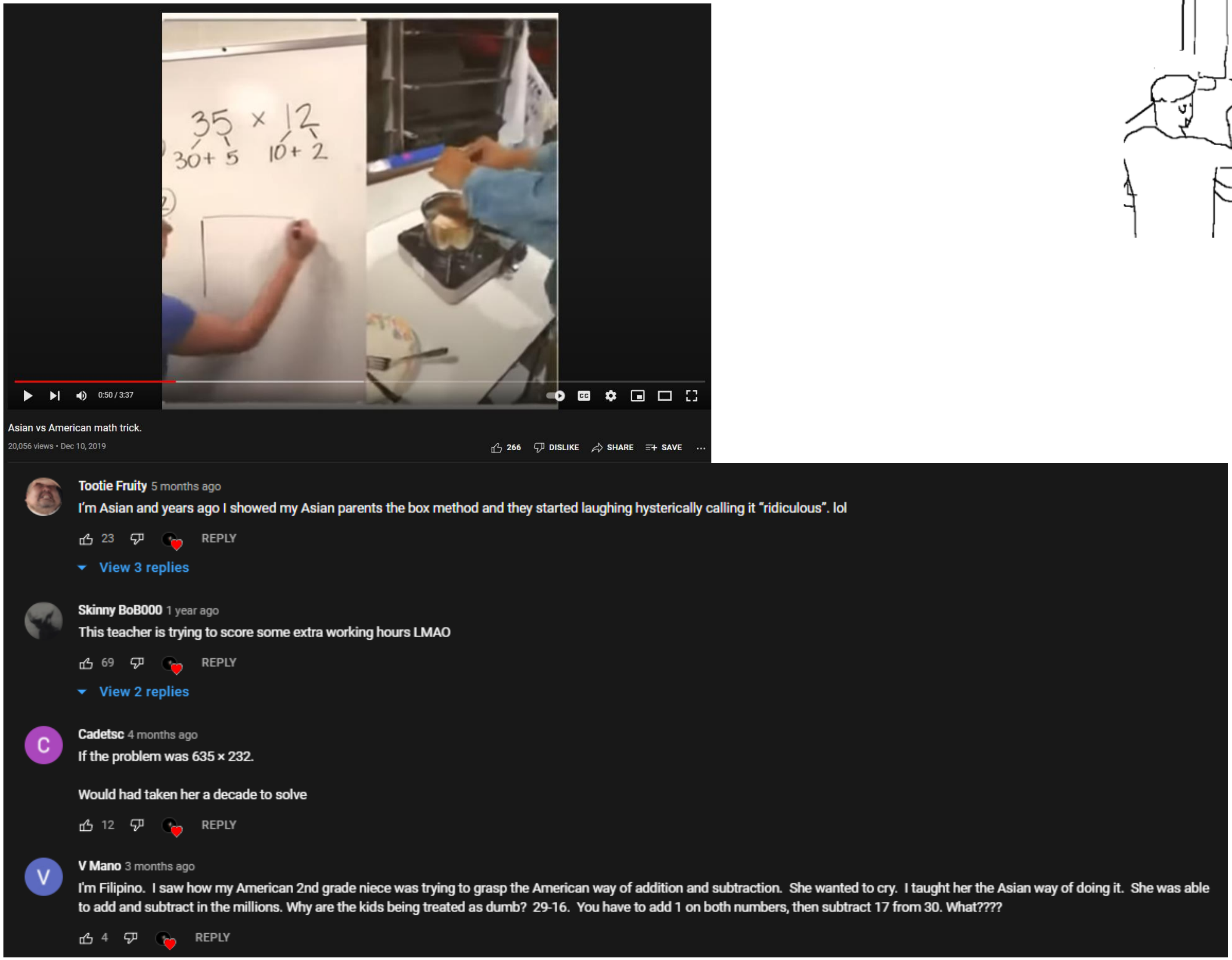
- Have your teacher ever…
  - Explain why it work? For every possible number combinations?
  - How efficient it is? Probably not… that is why….





Grade School Multiplication

An algorithm we all know



Grade School Multiplication

An algorithm we all know

- Consider the following 2 inputs
  - *x* = 123
  - *y* = 345
  - Multiple *x* with *y*, what is the answer?

$$
\begin{array}{r}
123 \\
\times \quad 345 \\
\hline
615 \\
492 \\
369 \\
\hline
42435
\end{array}
$$

- Have your teacher ever…
  - Explain why it work? For every possible number combinations?
  - How efficient it is?
    - This is not efficient…
    - As the numbers become bigger you need more steps!

Grade School Multiplication

An algorithm we all know

$$
\begin{array}{rccccc}
 & & & 1 & 2 & 3 \\
\times & & & 3 & 4 & 5 \\
\hline
 & & & 6 & 1 & 5 \\
 & & 4 & 9 & 2 & \\
 & 3 & 6 & 9 & & \\
\hline
 & 4 & 2 & 4 & 3 & 5
\end{array}
$$

How many operations to produce this?

Grade School Multiplication

An algorithm we all know

$$
\begin{array}{rccccc}
 & & & 1 & 2 & 3 \\
\times & & & 3 & 4 & 5 \\
\hline
 & & & 6 & 1 & 5 \\
 & & 4 & 9 & 2 & \\
 & 3 & 6 & 9 & & \\
\hline
 & 4 & 2 & 4 & 3 & 5
\end{array}
$$

How many operations to produce this?
Multiply 5x3
Note the overflow
Multiply 5x2
Add the overflow (from prev)
Note the overflow
Multiply 5x1
Add the overflow (from prev)

Grade School Multiplication

An algorithm we all know

$$
\begin{array}{r}
1\;2\;3 \\
\times \quad 3\;4\;5 \\
\hline
6\;1\;5 \\
4\;9\;2 \\
3\;6\;9 \\
\hline
4\;2\;4\;3\;5
\end{array}
$$

Now repeat for this…

Grade School Multiplication

An algorithm we all know

$$
\begin{array}{r}
1\;2\;3 \\
\times \quad 3\;4\;5 \\
\hline
6\;1\;5 \\
4\;9\;2\; \\
3\;6\;9\;\; \\
\hline
4\;2\;4\;3\;5
\end{array}
$$

And this this…

Grade School Multiplication

An algorithm we all know

$$
\begin{array}{cccccc}
 & & & 1 & 2 & 3 \\
\times & & & 3 & 4 & 5 \\
\hline
 & & & 6 & 1 & 5 \\
 & & 4 & 9 & 2 & \\
 & 3 & 6 & 9 & & \\
 & 4 & 2 & 4 & 3 & 5
\end{array}
$$

Then add them all up

Grade School Multiplication

An algorithm we all know

$$
\begin{array}{rccccc}
 & & & 1 & 2 & 3 \\
\times & & & 3 & 4 & 5 \\
\hline
 & & & 6 & 1 & 5 \\
 & & 4 & 9 & 2 & \\
 & 3 & 6 & 9 & & \\
\hline
 & 4 & 2 & 4 & 3 & 5
\end{array}
$$

It is a lot of operations!

… and it become worse as the number becomes bigger!

Grade School Multiplication

An algorithm we all know

$$\begin{array}{rrrrr} & & 1 & 2 & 3 \\ \times & & 3 & 4 & 5 \\ \hline & & 6 & 1 & 5 \\ & 4 & 9 & 2 & \\ 3 & 6 & 9 & & \\ \hline 4 & 2 & 4 & 3 & 5 \end{array}$$

It is a lot of operations!
… and it become worse as the number becomes bigger!

Each digit in *x*, increases actions per row
Each digit in *y*, increases a row!

Grade School Multiplication

An algorithm we all know

$$
\begin{array}{rrrrrr}
 & & & 1 & 2 & 3 \\
\times & & & 3 & 4 & 5 \\
\hline
 & & & 6 & 1 & 5 \\
 & & 4 & 9 & 2 & \\
 & 3 & 6 & 9 & & \\
\hline
 & 4 & 2 & 4 & 3 & 5
\end{array}
$$

It is a lot of operations!
… and it become worse as the number becomes bigger!

Each digit in *x*, increases actions per row
Each digit in *y*, increases a row!

And in the end, you need to add them all up!!!

Grade School Multiplication

An algorithm we all know

$$
\begin{array}{r}
1\;2\;3 \\
\times \quad 3\;4\;5 \\
\hline
\boxed{\begin{array}{r} 6\;1\;5 \\ 4\;9\;2\; \\ 3\;6\;9\;\; \end{array}} \\
4\;2\;4\;3\;5
\end{array}
$$

And don't forget, you need to store all of these somewhere…
What if they are really big? You are wasting memory….

Grade School Multiplication

An algorithm we all know

$$
\begin{array}{rccccc}
 & & & 1 & 2 & 3 \\
\times & & & 3 & 4 & 5 \\
\hline
 & & & 6 & 1 & 5 \\
 & & 4 & 9 & 2 & \\
 & 3 & 6 & 9 & & \\
\hline
 & 4 & 2 & 4 & 3 & 5
\end{array}
$$

This entire thing?

- 9 multiplications
- 3 noting of overflow
- 3 addition for overflow
- The final addition process that deals with a lot of integers and overflow

Grade School Multiplication

An algorithm we all know

- That is why students are taught the box method… which provides the foundation for Karatsuba!

Questions?

Karatsuba

Simple Quick Integer Multiplication

Karatsuba

Simple Quick Integer Multiplication

- A method to multiply 2 integers
- Able to do so quickly, even as the integers scales to be bigger integers

- A method to multiply 2 integers
- Able to do so quickly, even as the integers scales to be bigger integers

- By breaking large numbers into smaller ones
  - x = 1234
  - y = 6789
  - x * y = ?

- A method to multiply 2 integers
- Able to do so quickly, even as the integers scales to be bigger integers

- By breaking large numbers into smaller ones
  - x = 1234 = 12 * 10^2 + 34 * 10^0
  - y = 6789 = 67 * 10^2 + 89 * 10^0
  - x * y = ?

Karatsuba

Simple Quick Integer Multiplication

- A method to multiply 2 integers
- Able to do so quickly, even as the integers scales to be bigger integers

- By breaking large numbers into smaller ones
  - x = 1234 = 12 * 10^2 + 34 * 10^0
  - y = 6789 = 67 * 10^2 + 89 * 10^0
  - x * y = (12 * 10^2 + 34 * 10^0) * (67 * 10^2 + 89 * 10^0)

Karatsuba

Simple Quick Integer Multiplication

- A method to multiply 2 integers
- Able to do so quickly, even as the integers scales to be bigger integers

- By breaking large numbers into smaller ones
  - x = 1234 = 12 * 10^2 + 34 * 10^0
  - y = 6789 = 67 * 10^2 + 89 * 10^0
  - x * y = (12 * 10^2 + 34 * 10^0) * (67 * 10^2 + 89 * 10^0)
    = (12 * 10^2 * 67 * 10^2) + (12 * 10^2 * 89 * 10^0)
    + (34 * 10^0 * 67 * 10^2) + (34 * 10^0 * 89 * 10^0)

- A method to multiply 2 integers
- Able to do so quickly, even as the integers scales to be bigger integers

- By breaking large numbers into smaller ones
  - x = 1234 = 12 * 10^2 + 34 * 10^0
  - y = 6789 = 67 * 10^2 + 89 * 10^0
  - x * y = (12 * 10^2 + 34 * 10^0) * (67 * 10^2 + 89 * 10^0)
    = (12 * 10^2 * 67 * 10^2) + (12 * 10^2 * 89 * 10^0)
    + (34 * 10^0 * 67 * 10^2) + (34 * 10^0 * 89 * 10^0)
    = (12 * 67 * 10^4) + (12 * 89 * 10^2)
    + (34 * 67 * 10^2) + (34 * 89)

Karatsuba

Simple Quick Integer Multiplication

- By breaking large numbers into smaller ones
  - x = 1234 = 12 * 10^2 + 34 * 10^0
  - y = 6789 = 67 * 10^2 + 89 * 10^0
  - x * y = (12 * 10^2 + 34 * 10^0) * (67 * 10^2 + 89 * 10^0)
    = (12 * 10^2 * 67 * 10^2) + (12 * 10^2 * 89 * 10^0)
    + (34 * 10^0 * 67 * 10^2) + (34 * 10^0 * 89 * 10^0)
    = (12 * 67 * 10^4) + (12 * 89 * 10^2)
    + (34 * 67 * 10^2) + (34 * 89)

Karatsuba
Simple Quick Integer Multiplication

- By breaking large numbers into smaller ones
  - x = 1234 = 12 * 10^2 + 34 * 10^0
  - y = 6789 = 67 * 10^2 + 89 * 10^0
  - x * y = (12 * 10^2 + 34 * 10^0) * (67 * 10^2 + 89 * 10^0)
    = (12 * 10^2 * 67 * 10^2) + (12 * 10^2 * 89 * 10^0)
    + (34 * 10^0 * 67 * 10^2) + (34 * 10^0 * 89 * 10^0)
    = (12 * 67 * 10^4) + (12 * 89 * 10^2)
    + (34 * 67 * 10^2) + (34 * 89)
  - Can you see it?

Karatsuba

Simple Quick Integer Multiplication

- By breaking large numbers into smaller ones
  - x = 1234 = 12 * 10^2 + 34 * 10^0
  - y = 6789 = 67 * 10^2 + 89 * 10^0
  - x * y = (12 * 10^2 + 34 * 10^0) * (67 * 10^2 + 89 * 10^0)
    = (12 * 10^2 * 67 * 10^2) + (12 * 10^2 * 89 * 10^0)
    + (34 * 10^0 * 67 * 10^2) + (34 * 10^0 * 89 * 10^0)
    = (12 * 67 * 10^4) + (12 * 89 * 10^2)
    + (34 * 67 * 10^2) + (34 * 89)
  - Can you see it?

- By breaking large numbers into smaller ones
  - x = 1234 = 12 * 10^2 + 34 * 10^0
  - y = 6789 = 67 * 10^2 + 89 * 10^0
  - x * y = (12 * 10^2 + 34 * 10^0) * (67 * 10^2 + 89 * 10^0)
    = (12 * 10^2 * 67 * 10^2) + (12 * 10^2 * 89 * 10^0)
    + (34 * 10^0 * 67 * 10^2) + (34 * 10^0 * 89 * 10^0)
    = (12 * 67 * 10^4) + (12 * 89 * 10^2)
    + (34 * 67 * 10^2) + (34 * 89)

- So why do this matter?

Karatsuba

Simple Quick Integer Multiplication

- By breaking large numbers into smaller ones
  - x = 1234 = 12 * 10^2 + 34 * 10^0
  - y = 6789 = 67 * 10^2 + 89 * 10^0
  - x * y = (12 * 10^2 + 34 * 10^0) * (67 * 10^2 + 89 * 10^0)
    = (12 * 10^2 * 67 * 10^2) + (12 * 10^2 * 89 * 10^0)
    + (34 * 10^0 * 67 * 10^2) + (34 * 10^0 * 89 * 10^0)
    = (12 * 67 * 10^4) + (12 * 89 * 10^2)
    + (34 * 67 * 10^2) + (34 * 89)

- So why do this matter?
  - Quicker to multiply small integers
  - Quicker to multiple simple numbers
    - Example: 123*100 = 12300

Karatsuba

Simple Quick Integer Multiplication

- By breaking large numbers into smaller ones
  - x = 1234 = 12 * 10^2 + 34 * 10^0
  - y = 6789 = 67 * 10^2 + 89 * 10^0
  - x * y = (12 * 10^2 + 34 * 10^0) * (67 * 10^2 + 89 * 10^0)
    = (12 * 10^2 * 67 * 10^2) + (12 * 10^2 * 89 * 10^0)
    + (34 * 10^0 * 67 * 10^2) + (34 * 10^0 * 89 * 10^0)
    = (12 * 67 * 10^4) + (12 * 89 * 10^2)
    + (34 * 67 * 10^2) + (34 * 89)

- So why do this matter? If you count the operations???
  - Quicker to multiply small integers
  - Quicker to multiple simple numbers
    - Example: 123*100 = 12300

- By breaking large numbers into smaller ones
  - x = 1234 = 12 * 10^2 + 34 * 10^0
  - y = 6789 = 67 * 10^2 + 89 * 10^0
  - x * y = (12 * 10^2 + 34 * 10^0) * (67 * 10^2 + 89 * 10^0)
    = (12 * 10^2 * 67 * 10^2) + (12 * 10^2 * 89 * 10^0)
    + (34 * 10^0 * 67 * 10^2) + (34 * 10^0 * 89 * 10^0)
    = (12 * 67 * 10^4) + (12 * 89 * 10^2)
    + (34 * 67 * 10^2) + (34 * 89)
    = (12 * 67 * 10^4) + (12 * 89 + 34 * 67) * 10^2 + (34 * 89)
- And we can do even better!
  - Add small numbers, then only multiply!

Karatsuba

Simple Quick Integer Multiplication

- With that, we can generalize:
  - Given integer x with n-digits
  - Given integer y with n-digits

Karatsuba

Simple Quick Integer Multiplication

- With that, we can generalize:
  - Given integer x with n-digits
  - Given integer y with n-digits
  - x can be broken down into
    - The most significant half x_l
    - The less significant half x_r

12345678

1234 | 5678

x_l x_r

Karatsuba

Simple Quick Integer Multiplication

- With that, we can generalize:
  - Given integer x with n-digits
  - Given integer y with n-digits
  - x can be broken down into
    - The most significant half x_l
    - The less significant half x_r

123456789

01234 | 56789

x_l x_r

Karatsuba

Simple Quick Integer Multiplication

- With that, we can generalize:
  - Given integer x with n-digits
  - Given integer y with n-digits
  - x can be broken down into
    - The most significant half x_l
    - The less significant half x_r
  - y can be broken down the same

    put 0 at the front to make all even digit number into odd digit number

Karatsuba

Simple Quick Integer Multiplication

- With that, we can generalize:
  - Given integer x with n-digits
  - Given integer y with n-digits
  - x can be broken down into
    - The most significant half x_l
    - The less significant half x_r
  - y can be broken down the same
  - Therefore $x * y = x_l * y_l * 10^n + (x_l * y_r + x_r * y_l) * 10^{n/2} + x_r * y_r$

Karatsuba

Simple Quick Integer Multiplication

- With that, we can generalize:
  - Given integer x with n-digits
  - Given integer y with n-digits
  - x can be broken down into
    - The most significant half x_l
    - The less significant half x_r
  - y can be broken down the same
  - Therefore $x * y = \underbrace{x_l * y_l}_{} * 10^n + (\underbrace{x_l * y_r}_{} + \underbrace{x_r * y_l}_{}) * 10^{n/2} + \underbrace{x_r * y_r}_{}$

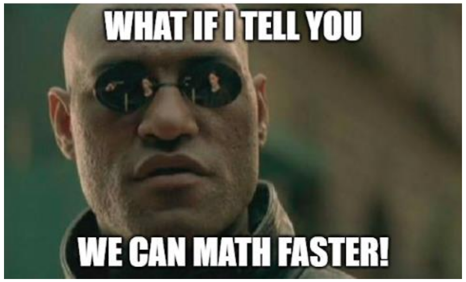
Questions?

Karatsuba

Simple Quick Integer Multiplication

- By breaking large numbers into smaller ones
  - x = 1234 = 12 * 10^2 + 34 * 10^0
  - y = 6789 = 67 * 10^2 + 89 * 10^0
  - x * y = (12 * 10^2 + 34 * 10^0) * (67 * 10^2 + 89 * 10^0)
    = (12 * 10^2 * 67 * 10^2) + (12 * 10^2 * 89 * 10^0)
    + (34 * 10^0 * 67 * 10^2) + (34 * 10^0 * 89 * 10^0)
    = (12 * 67 * 10^4) + (12 * 89 * 10^2)
    + (34 * 67 * 10^2) + (34 * 89)
    = (12 * 67 * 10^4) + (12 * 89 + 34 * 67) * 10^2 + (34 * 89)

- And we can do even better!
  - Add small numbers, then only multiply!
  - What if I tell you we can do even better?



Karatsuba

Simple Quick Integer Multiplication

- Recall we stopped at the following
  - Therefore $x * y = \underbrace{x_l * y_l}_{} * 10^n + (\underbrace{x_l * y_r}_{} + \underbrace{x_r * y_l}_{}) * 10^{n/2} + \underbrace{x_r * y_r}_{}$

Karatsuba

Simple Quick Integer Multiplication

- Recall we stopped at the following
  - Therefore $x * y = x_l * y_l * 10^n + (x_l * y_r + x_r * y_l) * 10^{n/2} + x_r * y_r$

Karatsuba

Simple Quick Integer Multiplication

MONASH University

- Recall we stopped at the following
  - Therefore $x * y = x_l * y_l * 10^n + \underbrace{(x_l * y_r + x_r * y_l)}_{} * 10^{n/2} + x_r * y_r$
  - Gauss introduce a trick for us

Karatsuba

Simple Quick Integer Multiplication

- Recall we stopped at the following
  - Therefore $x * y = x_l * y_l * 10^n + \underbrace{(x_l * y_r + x_r * y_l)} * 10^{n/2} + x_r * y_r$
  - Gauss introduce a trick for us
    - $(x_l + x_r) * (y_l + y_r) = x_l * y_l + x_l * y_r + x_r * y_l + x_r * y_r$

Karatsuba

Simple Quick Integer Multiplication

- Recall we stopped at the following
  - Therefore $x * y = x_l * y_l * 10^n + (x_l * y_r + x_r * y_l) * 10^{n/2} + x_r * y_r$
  - Gauss introduce a trick for us
    - $(x_l + x_r) * (y_l + y_r) = x_l * y_l + x_l * y_r + x_r * y_l + x_r * y_r$
    - Then we rearrange the above…
    - $x_l * y_r + x_r * y_l \quad = (x_l + x_r) * (y_l + y_r) - x_l * y_l - x_r * y_r$

Karatsuba

Simple Quick Integer Multiplication

- Recall we stopped at the following
  - Therefore $x * y = x_l * y_l * 10^n + \underbrace{(x_l * y_r + x_r * y_l)} * 10^{n/2} + x_r * y_r$
  - Gauss introduce a trick for us
    - $(x_l + x_r) * (y_l + y_r) = x_l * y_l + x_l * y_r + x_r * y_l + x_r * y_r$
    - Then we rearrange the above…
    - $\underbrace{x_l * y_r + x_r * y_l} = (x_l + x_r) * (y_l + y_r) - x_l * y_l - x_r * y_r$
    - Why are we doing this?

Karatsuba

Simple Quick Integer Multiplication

- Recall we stopped at the following
  - Therefore $x * y = x_l * y_l * 10^n + \underbrace{(x_l * y_r + x_r * y_l)}_{} * 10^{n/2} + x_r * y_r$
  - Gauss introduce a trick for us
    - $(x_l + x_r) * (y_l + y_r) = x_l * y_l + x_l * y_r + x_r * y_l + x_r * y_r$
    - Then we rearrange the above…
    - $\underbrace{x_l * y_r + x_r * y_l}_{} \quad = (x_l + x_r) * (y_l + y_r) - x_l * y_l - x_r * y_r$
    - Why are we doing this?

Karatsuba

Simple Quick Integer Multiplication

- Recall we stopped at the following
  - Therefore $x * y = x_l * y_l * 10^n + (x_l * y_r + x_r * y_l) * 10^{n/2} + x_r * y_r$
  - Gauss introduce a trick for us
    - $(x_l + x_r) * (y_l + y_r) = x_l * y_l + x_l * y_r + x_r * y_l + x_r * y_r$
    - Then we rearrange the above…
    - $x_l * y_r + x_r * y_l \quad = (x_l + x_r) * (y_l + y_r) - x_l * y_l - x_r * y_r$
    - Why are we doing this?

Karatsuba

Simple Quick Integer Multiplication

- Recall we stopped at the following
  - Therefore $x * y = x_l * y_l * 10^n + \underbrace{(x_l * y_r + x_r * y_l)} * 10^{n/2} + x_r * y_r$
  - Gauss introduce a trick for us
    - $(x_l + x_r) * (y_l + y_r) = x_l * y_l + x_l * y_r + x_r * y_l + x_r * y_r$
    - Then we rearrange the above…
    - $x_l * y_r + x_r * y_l \quad = \underbrace{(x_l + x_r) * (y_l + y_r)} - x_l * y_l - x_r * y_r$

      Sub this back upper red underlined part to have a quicker computation
    - Why are we doing this?
      - 1 multiplication instead of 2 multiplication
      - Note that it is slower to multiply than it is to add/ subtract in general

Questions?

In summary

- Given 2 large numbers
- Divide and conquer the large number into 2 halves
  - Smaller numbers are faster to operate on
  - Only need 3 multiplications, on smaller numbers
- Then combine the result

Karatsuba
In summary

- Given 2 large numbers
- Divide and conquer the large number into 2 halves
  - Smaller numbers are faster to operate on
  - Only need 3 multiplications, on smaller numbers

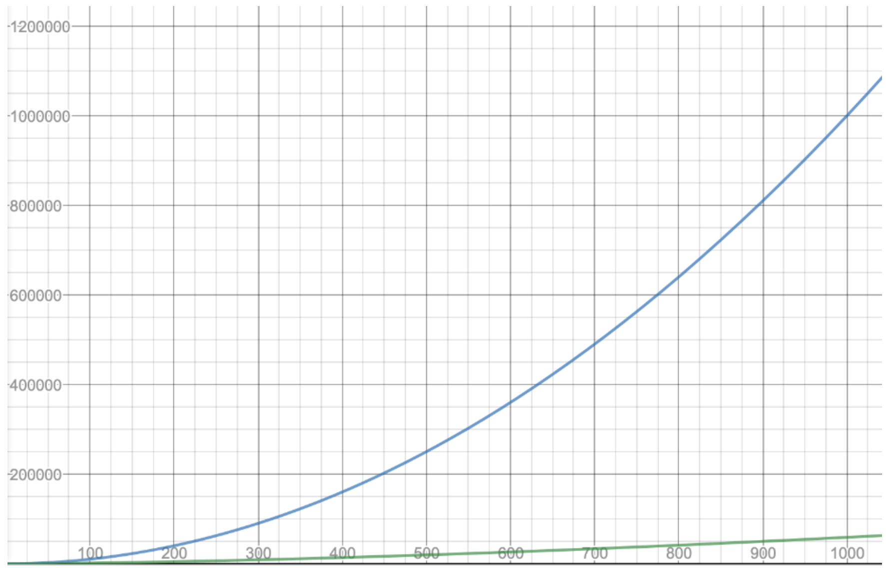
We can follow Karatsuba again
for the 3 multiplications!

- Then combine the result

Karatsuba

In summary

- To multiply 2 large numbers of n-digits,
  Karatsuba can do so in O(N^1.59), which is much more scalable than O(N^2)



Questions?

Divide and Conquer

A Recap

- Take a problem
- Divide the problem into smaller subproblems
- Conquer the smaller subproblems, getting the solution
- Combine the smaller solutions to obtain the bigger solution

Divide and Conquer

A Recap

- Take a problem
- Divide the problem into smaller subproblems
- Conquer the smaller subproblems, getting the solution
- Combine the smaller solutions to obtain the bigger solution

- And you done that with:
  - Karatsuba earlier
  - MergeSort and QuickSort from earlier your prerequisite(s)

Divide and Conquer

A Recap

- Take a problem
- Divide the problem into smaller subproblems
  - Karatsuba: Split a number into most-significant digits (MSD) and least-siginificant digits (LSD)
  - MergeSort and QuickSort: Split a list into left-partition and right-partition
- Conquer the smaller subproblems, getting the solution
- Combine the smaller solutions to obtain the bigger solution

Divide and Conquer

A Recap

- Take a problem
- Divide the problem into smaller subproblems
- Conquer the smaller subproblems, getting the solution
  - Karatsuba: Multiply the smaller digits.
  - MergeSort and QuickSort: Sort the partitions
- Combine the smaller solutions to obtain the bigger solution

Divide and Conquer
A Recap

- Take a problem
- Divide the problem into smaller subproblems
- Conquer the smaller subproblems, getting the solution
- Combine the smaller solutions to obtain the bigger solution
  - Karatsuba: Add and subtract the values together.
  - MergeSort and QuickSort: Combine the partitions in sorted order.

Divide and Conquer
A Recap

- Take a problem
- Divide the problem into smaller subproblems
- Conquer the smaller subproblems, getting the solution
- Combine the smaller solutions to obtain the bigger solution

- You would notice that many of them are done in recursively as well; we will explore how to analyze recursive complexity in a later lecture.

Questions?

Other DnC Algorithms?

- Finding closest pair of points in a plane in O(n log n).
- Counting inversions in O(n log n), see you Studio question.
- Improving matrix multiplication (Strassen's algorithm).
- Fast Fourier Transform: this algorithm published by James Cooley and John Tukey in 1965 is one of the most influential algorithms, with a wide range of applications in engineering, music, science, mathematics, etc.
  - In fact, it can be traced back to unpublished work by Gauss.

Questions?

Thank You