دانشگاه آزاد اسلامی واحد تبریز

نام درس: طراحی الگوریتم ها

بخش:

Logistics, introduction, and multiplication!

نام استاد: دکتر مسعود کارگر

Some final remarks about the master theorem

- Suppose $T(n) = a \cdot T\left(\frac{n}{b}\right) + O(n^d)$. Then

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

Three parameters:
a : number of subproblems
b : factor by which input size shrinks
d : need to do n^d work to create all the subproblems and combine their solutions.

Jedi master Yoda

Algorithms are fun!

- Algorithm design is both an art and a science.
- Many surprises!
- A young field, lots of exciting research questions!

- (Will help you get a job you like!)

Course goals

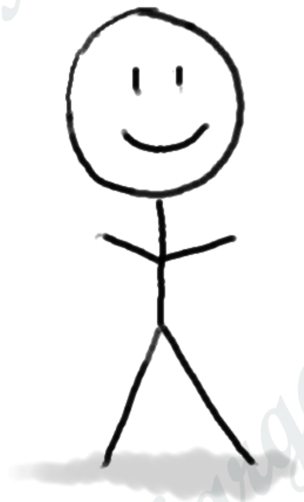
- Build an "algorithmic toolkit"
- Learn to think "algorithmically"

Today's goals

- Karatsuba Integer Multiplication
- Technique: Divide and conquer
- Meta points:
  - Algorithm designer's question
  - The role of rigor

The algorithm designer's question

Can I do better?



Algorithm designer

The algorithm designer’s internal monologue…

What exactly do we mean by better? And what about that corner case? Shouldn’t we be zero-indexing?

Can I do better?

Dude, this is just like that other time. If you do the thing and the stuff like you did then, it’ll totally work real fast!

Plucky the
Pedantic Penguin

Detail-oriented
Precise
Rigorous

Algorithm designer

Lucky the
Lackadaisical Lemur

Big-picture
Intuitive
Hand-wavey

We will feel this tension throughout the course

- In lecture, I will channel Lucky maybe a bit more than I should.
- On HW, you should lean a bit more towards Plucky.
  - See Homework Style Guidelines (on webpage) for more.

Course goals

- Build an “algorithmic toolkit”
- Learn to think “algorithmically”

Today’s goals

- Karatsuba Integer Multiplication
- Technique: Divide and conquer
- Meta points:
  - Algorithm designer’s question
  - The role of rigor

Integer Multiplication

A problem you all know how to solve:

Integer Multiplication

$$\begin{array}{r} 12 \\ \times\ 34 \\ \hline \end{array}$$

Integer Multiplication

A problem you all know how to solve:

Integer Multiplication

```
    12345678959031413
x   45638235520395533
_____________________
```

Integer Multiplication

A problem you all know how to solve:

n

1233925720752752384623764283568364918374523856298

x 4562323582342395285623467235019130750135350013753

???

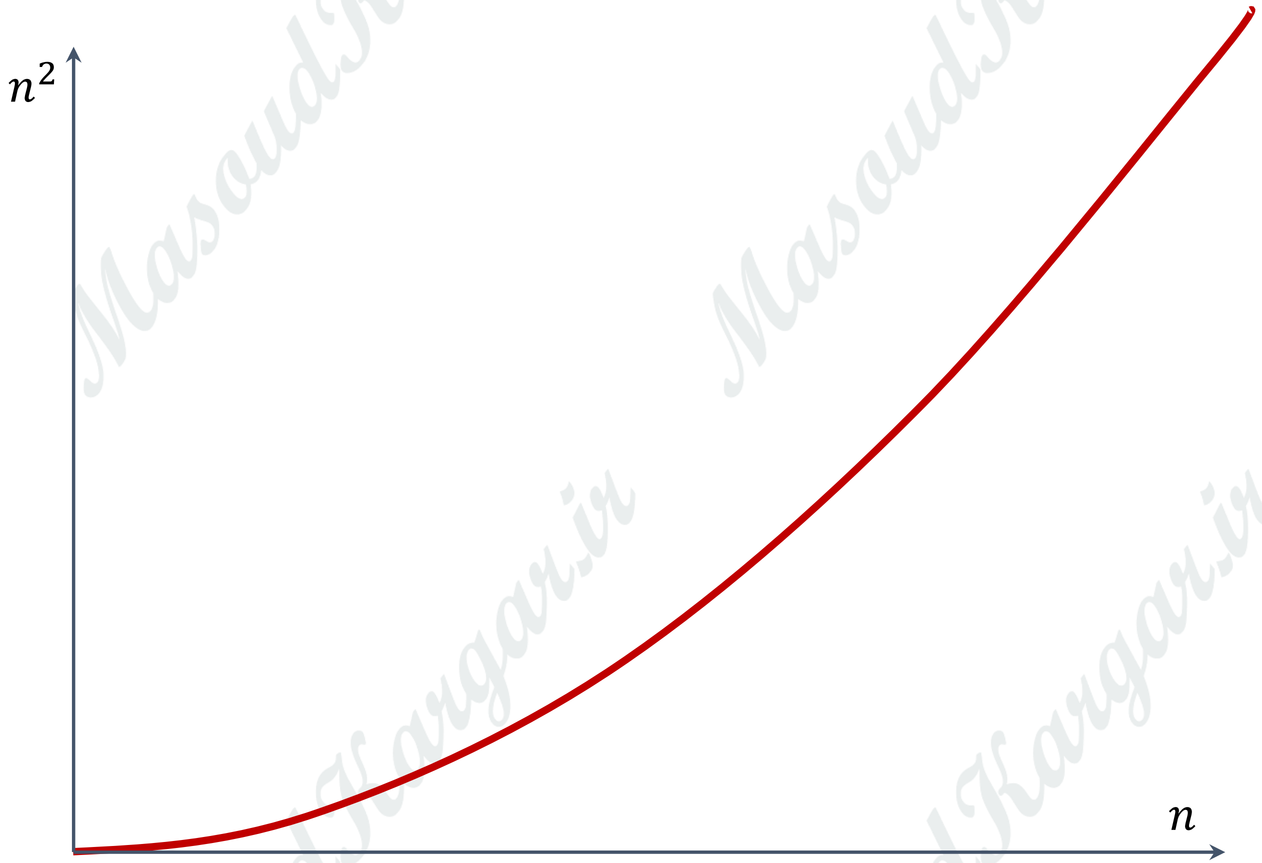
How would you solve this problem?
How long would it take you?

About n^2 one-digit operations

At most n^2 multiplications,
and then at most n^2 additions (for carries)
and then I have to add n different 2n-digit numbers…

And I take 1 second to multiply two one-digit numbers and .6 seconds to add, so…

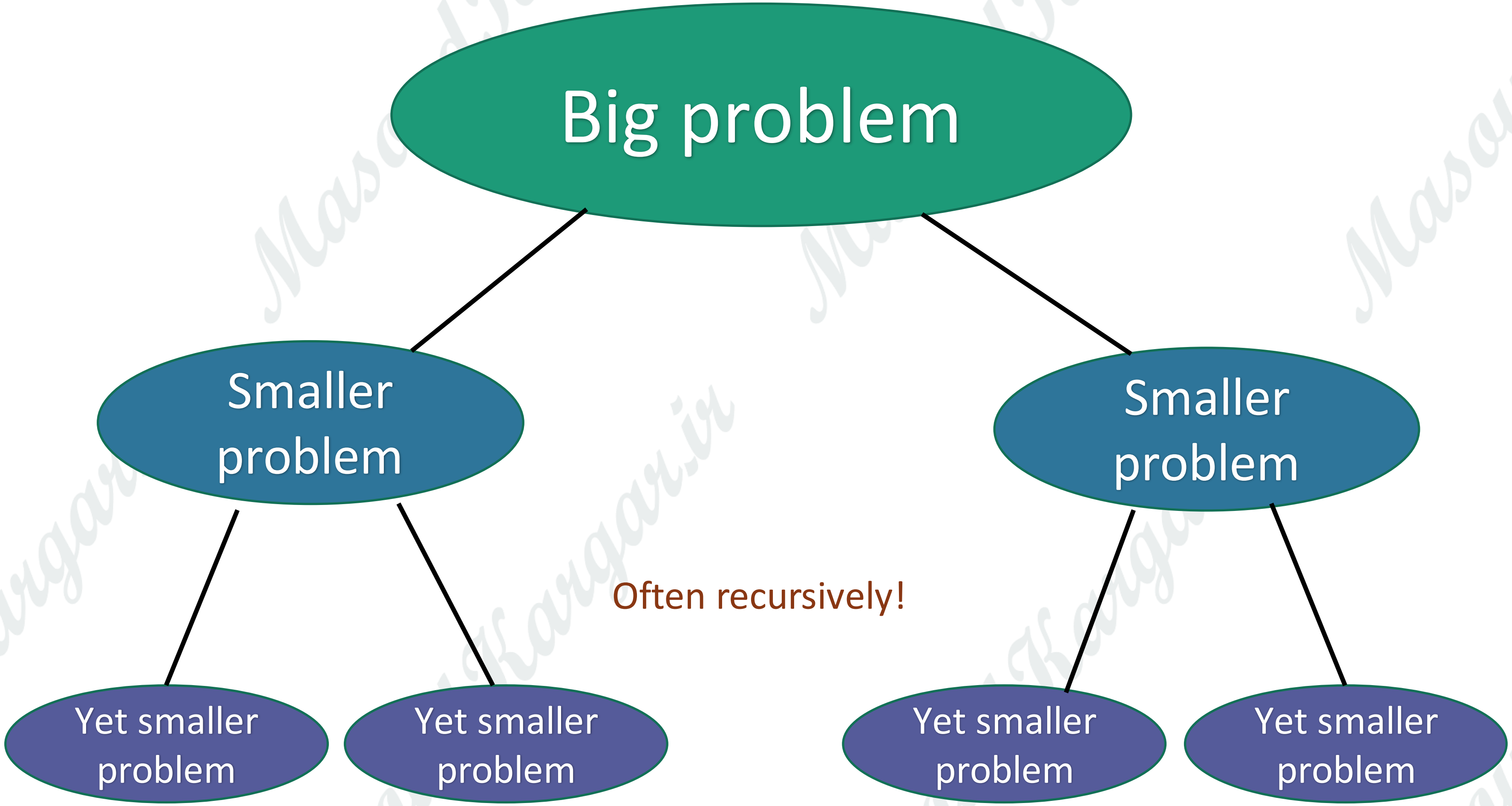
Can we do better?



Let’s dig in to our algorithmic toolkit…

Divide and conquer

Break problem up into smaller (easier) sub-problems



Divide and conquer for multiplication

Break up an integer:

$$1234 = 12\times 100 + 34$$

$$1234 \times 5678$$
$$= (12\times 100 + 34)(56\times 100 + 78)$$
$$= (12 \times 56)10000 + (34 \times 56 + 12 \times 78)100 + (34 \times 78)$$

1 (12×56) 2 (34×56) 3 (12×78) 4 (34×78)

One 4-digit multiply → Four 2-digit multiplies

More generally

Break up an n-digit integer:

$$[x_1 x_2 \cdots x_n] = [x_1 x_2 \cdots x_{n/2}] \times 10^{n/2} + [x_{n/2+1} x_{n/2+2} \cdots x_n]$$

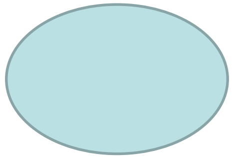
$$x \times y = (a \times 10^{n/2} + b)(c \times 10^{n/2} + d)$$

$$= \underbrace{(a \times c)}_{1} 10^n + (\underbrace{a \times d}_{2} + \underbrace{c \times b}_{3}) 10^{n/2} + \underbrace{(b \times d)}_{4}$$

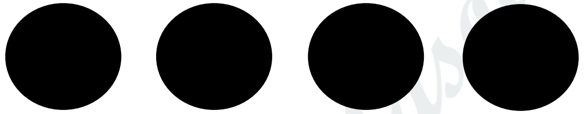
One n-digit multiply → Four (n/2)-digit multiplies

Another way to see this*

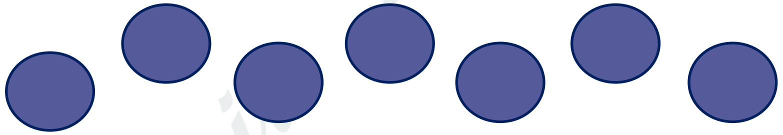
*we will come back to this sort of analysis later and still more rigorously.



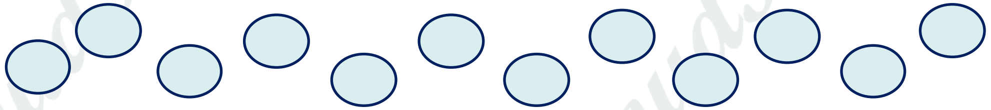
1 problem
of size n



4 problems
of size n/2



4^t problems
of size $n/2^t$



n^2 problems
of size 1

- If you cut n in half $\log_2(n)$ times, you get down to 1.
- So we do this $\log_2(n)$ times and get…

$4^{log\_2(n)} = n^2$

problems of size 1.

What about the work you actually do in the problems?

Yet another way to see this*

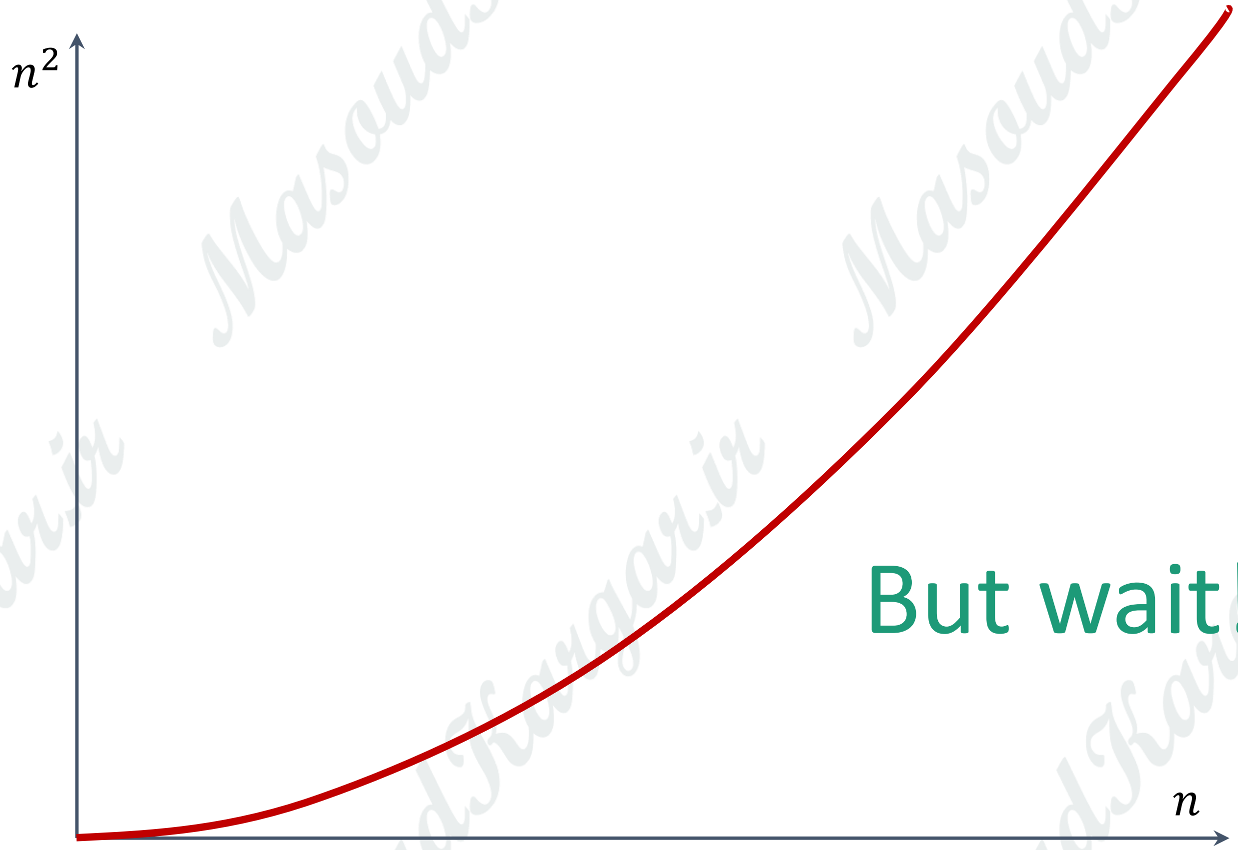
- Let T(n) be the time to multiply two n-digit numbers.
- Recurrence relation:
  - $T(n) = 4 \cdot T\left(\frac{n}{2}\right)$ + (about n to add stuff up)

Ignore this term for now…

$$T(n) = 4 \cdot T(n/2)$$
$$= 4 \cdot (4 \cdot T(n/4)) \qquad 4^2 \cdot T(n/2^2)$$
$$= 4 \cdot (4 \cdot (4 \cdot T(n/8))) \qquad 4^3 \cdot T(n/2^3)$$
$$\vdots$$
$$= 2^{2t} \cdot T(n/2^t) \qquad 4^t \cdot T(n/2^t)$$
$$\vdots$$
$$= n^2 \cdot T(1). \qquad 4^{\log_2(n)} \cdot T(n/2^{\log_2(n)})$$

That's a bit disappointing

All that work and still n^2…



But wait!!

Divide and conquer can actually make progress

- Karatsuba figured out how to do this better!

$$xy = (a \cdot 10^{n/2} + b)(c \cdot 10^{n/2} + d)$$
$$= ac \cdot 10^{n} + (ad + bc)10^{n/2} + bd$$

Need these three things

- If only we recurse three times instead of four…

Karatsuba integer multiplication

- Recursively compute
  - ac
  - bd
  - (a+b)(c+d)

Subtract these off

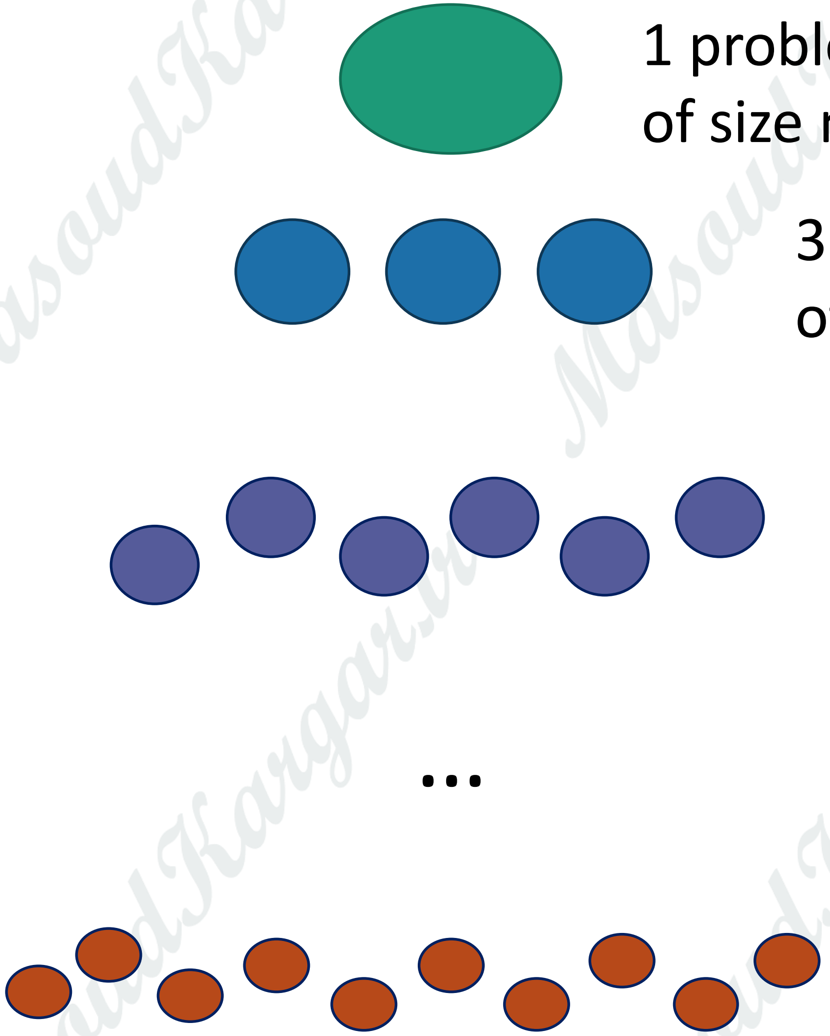
get this

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

- Assemble the product:

$$xy = (a \cdot 10^{n/2} + b)(c \cdot 10^{n/2} + d)$$
$$= ac \cdot 10^{n} + (ad + bc)10^{n/2} + bd$$

What's the running time?



1 problem
of size n

3 problems
of size n/2

3^t problems
of size $n/2^t$

...

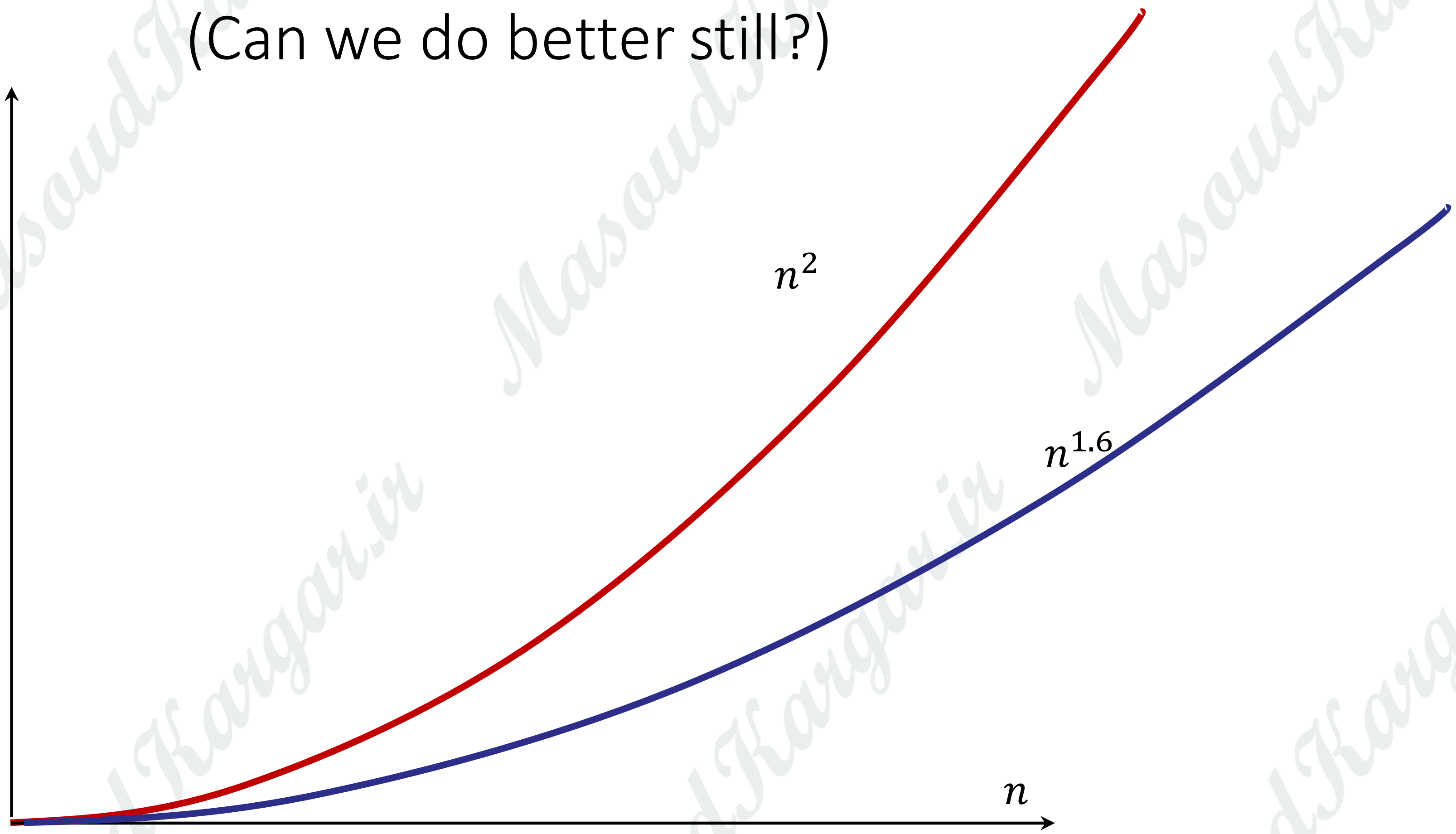
$n^{1.6}$ problems
of size 1

- If you cut n in half $\log_2(n)$ times, you get down to 1.
- So we do this $\log_2(n)$ times and get…

$3^{\log\_2(n)} = n^{\log\_2(3)} = n^{1.6}$
problems of size 1.

This is much better!

(Can we do better still?)



Karatsuba's algorithm was proposed in 1960. The Toom-Cook algorithm (1963 and 1966) works similarly but reduces 9 multiplications to 5, instead of 4 to 3. This runs in time about n^(1.465). The Schönhage–Strassen algorithm (1971) works in time O(nlog(n)loglog(n)) using FFT-like stuff. The state-of-the-art (in theory, not in practice) is Furer's algorithm (2007), which runs in time O(n*log(n)* 2^(\log^*(n))).

Course goals

- Build an “algorithmic toolkit”
- Learn to think “algorithmically”

Today’s goals

- Karatsuba Integer Multiplication
- Technique: Divide and conquer
- Meta points:
  - Algorithm designer’s question
  - The role of rigor
- End on a historical note…

actually pretty amazing

It's actually pretty amazing that you can big multiply numbers quickly at all

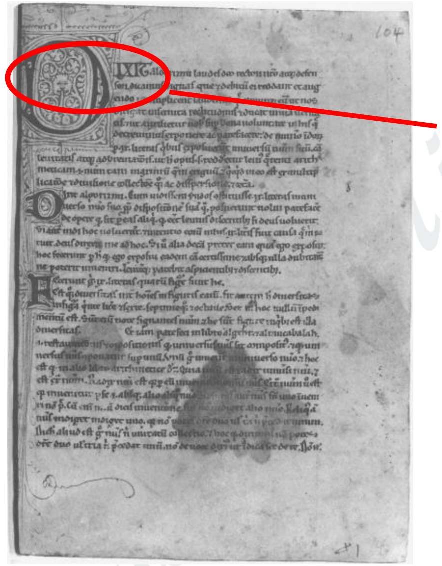
- You could do this when you were 8.
- It wasn't always so easy!

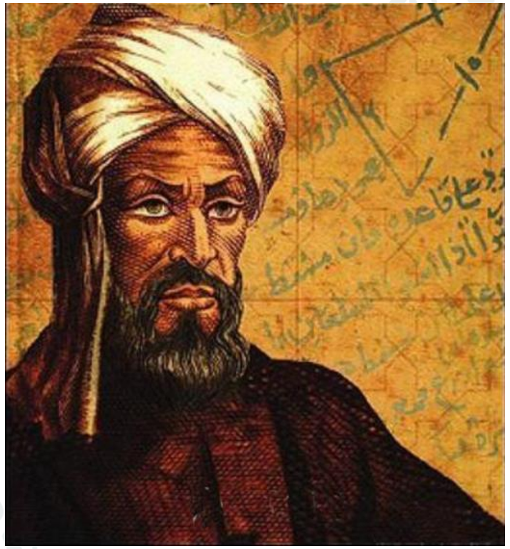
LXXXIX × CM = ?

Etymology of “Algorithm”

- Al-Khwarizmi (Persian mathematician, lived around 800AD) wrote a book about how to multiply with Arabic numerals.

- His ideas came to Europe in the 12^{th} century.





`Dixit algorizmi`
(so says Al-Khwarizmi)

- Originally, “Algorisme” [old French] referred to just the Arabic number system, but eventually it came to mean “Algorithm” as we know today.

Wrap up

- Algorithms are:
  - Fundamental, useful, and fun!
- In this course, we will develop both algorithmic intuition and algorithmic technical chops
- Karatsuba Integer Multiplication:
  - You can do better than grade school multiplication!
  - Example of divide-and-conquer in action

Next time

- Divide-and-conquer again
- Asymptotics and big-O notation

قدردانی