COMP3600/6466 Algorithms
Lecture 1

S2 2016

Dr. Hassan Hijazi
Prof. Weifa Liang

Course logistics - Part 1

- **Textbook:** Introduction to Algorithms
  T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein. The MIT Press, 3rd Edition, 2009 or later.
- **Lecturers:** Prof. Weifa Liang and Dr. Hassan Hijazi
- **Volume:** 28 lectures and 4 review sessions
- **Tute-Labs:** Six combined 2 hours long tute-labs starting week 2 and running once every fortnight.
  - Use **Wattle** to register yourself into one of the **five** groups.

Course logistics - Part 2

- There will be **FOUR** quizzes with due dates on Fridays.
  - The mark allocations of the four quizzes are 3%, 3.5%, 4.5% and 4% of the final mark (due dates: 29/07, 12/08, 23/09, and 14/10)
- There will be **TWO** assignments.
  - The first assignment (due date: 02/09), worth 15% of the final mark
  - The second assignment (due date: 24/10), worth 20% of the final mark
- There will be **TWO** exams.
  - The mid-semester exam worth 20% of the final mark
  - The final exam worth 30% of the final mark

Course logistics - Part 3

- The course assessment consists of:
  - Four Quizzes, 15% of the total
  - Two Assignments, 35% of the total - Mandatory
  - One Mid-Semester Exam, 20% of the total - Mandatory
  - One Final Exam, 30% of the total - Mandatory
- Answer sheets for all quizzes and assignments should be submitted electronically through **Wattle**.
- To pass the course you need at least 16/35 on assignments and 22/50 on exams.

Course logistics - Part 4

- **Discussion forums in Wattle:**
  - Announcements: for official announcements
  - Student forum: for discussions among students
  - Can be accessed through your Wattle account
  - Only used for issues related to this course
  - Avoid posting quiz or assignment solutions
  - Bonus questions posted on the forum are optional

Origins of the word

The word ’Algorithm' comes from the name:

Al-Khawarizmi

A Persian mathematician, astronomer, and geographer born 1300 years ago.

Definition

An algorithm is a set of step-by-step operations that takes a number of inputs, and produces a number of outputs.

- Example -

Input: Frozen Pizza

Output: Ready-to-eat Pizza

Algorithm:

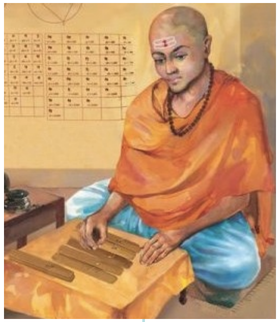
1. Preheat the oven to 180ºC
2. Remove pizza from all packaging
3. Place the frozen pizza on the middle rack
4. Wait for 20 minutes

Algorithms over time

- Babylonians — 3600

- Euclid — 2300

- Liu Hui — 1800



- Brahmagupta — 1400

- Al-Khawarizmi — 1300

- You! — 0

The Babylonians

An algorithm to approximate of the square root of a number

Euclid

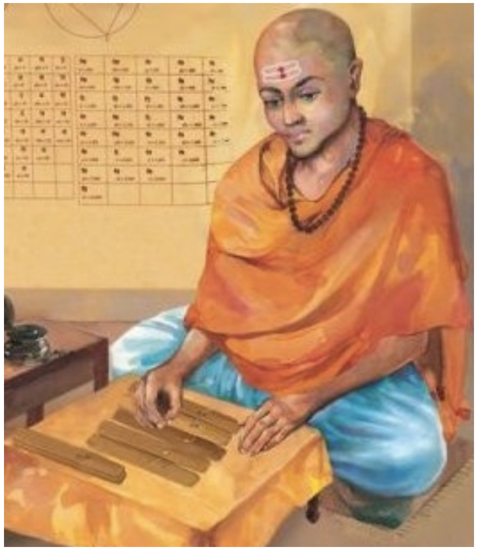
An algorithm to compute the greatest common divisor of two numbers

Liu Hui

An algorithm for solving a system of linear equations

(Gauss elimination)

Brahmagupta



An algorithm to solve quadratic equations with integers

$$x^2 = \boldsymbol{N} y^2 + 1,$$

x, y integer variables, $\boldsymbol{N}$ integer constant

Al-Khawarizmi

An algorithm to solve general quadratic equations

$$\boldsymbol{a}x^2 + \boldsymbol{b}x + \boldsymbol{c} = 0$$

$$x \text{ real variable}$$

Today’s algorithms can solve:

Face Recognition

Weather Forecasting

Financial Trading

Data Compression

Data Encryption

Data Routing

Power System Design

Integrated Circuit Design

Analyzing Genes and Protein sequencing

Video Games!

Algorithms rule your:

Phone

PC

TV

Car

Flight

Google search

GPS

Microwave

Ultrasound

Xbox!

Some basic algorithms

1. Sort n integers in increasing order.
   Input: a sequence of n integers, $a_1, a_2, \ldots, a_n$
   Output: a permutation (rearrangement) of the numbers as $a'_1, a'_2, \ldots, a'_n$, such that $a'_1 \leq a'_2 \leq \cdots \leq a'_n$.

2. Given an integer N, determine whether it is a prime number.
   Input: N
   Output: YES if N is a prime number; otherwise, NO.

3. Given an integer N, print out prime numbers no greater than N.
   Input: N
   Output: a sequence of prime numbers, and no number is larger than N.

Some basic algorithms: Linear search

Given a vector V of size n, find if x is in V.

INPUT: Vector V, value x.

OUTPUT: YES or NO.

Worst case number of comparisons: n

Best case number of comparisons: 1

Some basic algorithms: Binary search

How many times can you divide n by 2 before reaching 1?

$$100 = 10^2$$
$$\log(100) = 2$$

$$1000 = 10^3$$
$$\log(1000) = 3$$

$$4 = 2^2$$
$$\log(4)/\log(2) = \log_2(4) = 2$$

$$8 = 2^3$$
$$\log(8)/\log(2) = \log_2(8) = 3$$

Some basic algorithms: Binary search

How many times can you divide n by 2 before reaching 1?

$$\sim \lg n$$

$$\lg n = \log_2(n)$$

Some basic algorithms: Binary search

Given a **sorted** (increasing) vector V, find if x is in V.

1. $\mathrm{n} \leftarrow \mathrm{card}(\mathrm{V})$

   if $n = 1$, return YES if $\mathrm{V}[0] = x$, return NO otherwise.

Some basic algorithms: Binary search

Given a **sorted** (increasing) vector V, find if x is in V.

1. $\mathrm{n} \leftarrow \mathrm{card}(\mathrm{V})$

   if $n = 1$, return YES if $\mathrm{V}[0] = x$, return NO otherwise.

2. If $\mathrm{x} = \mathrm{V}\left[\left\lceil\frac{n}{2}\right\rceil - 1\right]$ return YES.

Some basic algorithms: Binary search

Given a **sorted** (increasing) vector V, find if x is in V.

1. $\mathrm{n} \leftarrow \mathrm{card(V)}$

if $n = 1$, return YES if $\mathrm{V}[0] = x$, return NO otherwise.

2. If $\mathrm{x} = \mathrm{V}\left[\left\lceil \frac{n}{2} \right\rceil - 1\right]$ return YES.

3. If $x > \mathrm{V}\left[\left\lceil \frac{n}{2} \right\rceil - 1\right], \mathrm{V} \leftarrow \left\{\mathrm{V}\left[\left\lceil \frac{n}{2} \right\rceil\right], \dots, \mathrm{V}[n-1]\right\}$ else

$$\mathrm{V} \leftarrow \left\{\mathrm{V}[0], \dots, \mathrm{V}\left[\left\lceil \frac{n}{2} \right\rceil - 1\right]\right\}$$

4. Go to 1.

Some basic algorithms: Binary search

How many times can you divide n by 2 before reaching 1?

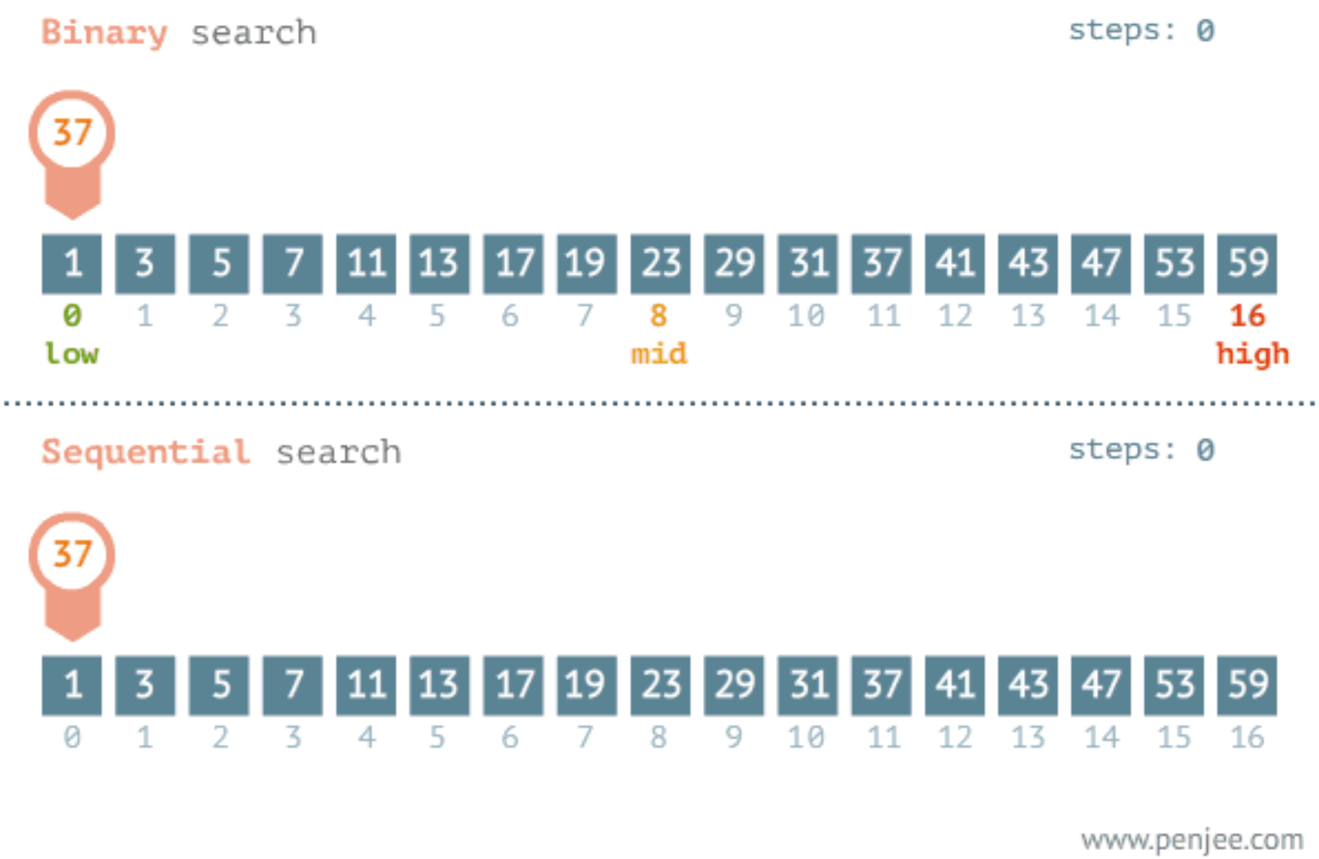
$$\sim \lg n$$

$$\Downarrow$$

Binary search worst case number of operations:

$$\sim \lg n$$

Linear vs. Binary Search



Linear vs. Binary Search

N	Time (seconds)
10000	1
20000	2
60000	6
600000	60
6000000	600

N	Time (seconds)
10000	1
20000	1
60000	2.5
600000	5.9
6000000	9.2

10 minutes vs. 10 seconds

Is time important?

Profits from High Frequency Trading = $1 Billion in the U.S. alone (2012).

Formalizing: Complexity of Algorithms

Assumption 1. A generic one-processor, random-access machine (**RAM**)

- Instructions are executed one after another
- No concurrent operations

Formalizing: Complexity of Algorithms

Assumption 2. Primitive operations take constant time:

operation	example	nanoseconds
variable declaration	$int\ a$	c_1
assignment statement	$a = b$	c_2
comparing numbers	$a < b$	c_3
array element access	$a[i]$	c_4
array length	$a.size()$	c_5
1D array allocation	$new\ int[N]$	$c_6\ N$
2D array allocation	$new\ int[N][N]$	$c_7\ N^2$

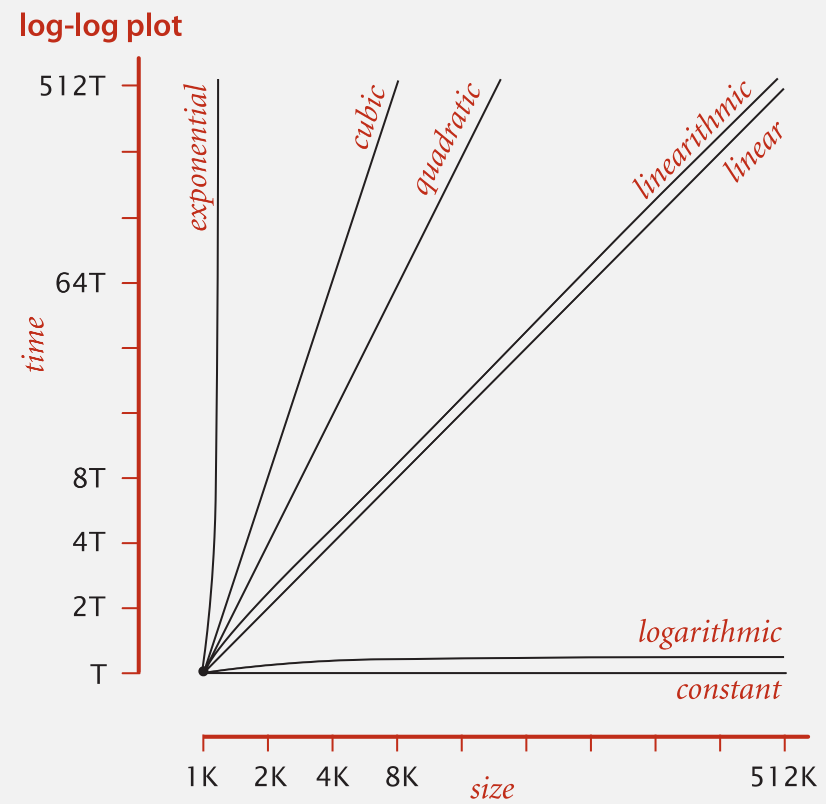
What comes after constant?

- Constant c
- Logarithmic $\lg n$
- Fractional Power $n^{\epsilon},\ 0 < \epsilon < 1$
- Linear n
- Quasilinear $n \lg^k n$
    - Linearithmic $n \lg n$
- Polynomial n^k
    - Quadratic n^2
- Quasi-polynomial $2^{\lg^k n}$
- Exponential 2^{n^k}

k positive integer

Increasing order

Order-of-growth



Typical orders of growth

Exercise 1.

1 operation = 10 milliseconds
How big n can be?

Number of Operations	1 second	1 minute	1 hour
$\lg n$			
n	100		
n^2			
2^n	6		

1 millisecond = 10^{-3} seconds
$\lg n = \log_2(n)$

Complexity of Algorithms

Assumption 3. Ignore constants and low order terms

$$2n^3 + 20n + 16 \equiv n^3$$

$$n^3 + 2n^2 + 5n + 10 \equiv n^3$$

$$100n^2 + n \equiv n^2$$

$$2n \lg n + n^2 \equiv n^2$$

$$2n \lg n + n \equiv n \lg n$$

Asymptotic behavior

- when N is large, low order terms are negligible
- when N is small, we don't care

Big-O Notation

$$f(n) = O(g(n))$$

$g(n)$ is an asymptotic upper bound for $f(n)$.

$f(n)$ grows at most as fast as $g(n)$.

Big-O Formal Definition

$$f(n) = O(g(n))$$

$$\Updownarrow$$

There **exist positive** c and n_0 such that $0 \leq f(n) \leq c \cdot g(n)$ for **all** $n \geq n_0$.

$$n^2 + n + 10 = O\left(n^2\right)?$$

Big-O Formal Definition

$$f(n) = O(g(n))$$

$$\Updownarrow$$

There **exist positive** c and n_0 such that $0 \le f(n) \le c \cdot g(n)$ for **all** $n \ge n_0$.

$$n^2 + n + 10 = O\left(n^2\right)$$

$$\Updownarrow$$

$$\exists\ c \text{ and } n_0 \text{ s.t. } 0 \le n^2 + n + 10 \le c \cdot n^2, \forall n \ge n_0$$

$$\Updownarrow$$

$$\exists\ c \text{ and } n_0 \text{ s.t. } 0 \le 1 + \frac{1}{n} + \frac{10}{n^2} \le c, \forall n \ge n_0$$

$$\Updownarrow$$

$$\text{Let } c = 12 \text{ and } n_0 = 1, \text{ then } 0 \le 1 + \frac{1}{n} + \frac{10}{n^2} \le c, \forall n \ge n_0$$

Exercise 2.

$$f(n) = O(g(n))$$

$$\Updownarrow$$

There **exist positive** c and n_0 such that $0 \leq f(n) \leq c \cdot g(n)$ for **all** $n \geq n_0$.

$$4n \lg n + 100n = O\left(n \lg n\right)?$$