Analysis of Algorithms - CS 323
Lecture #1 – February 4th, 2016

Notes by: Ateeb Khawaja

Announcements:

- The URL for the Course Website is https://groups.yahoo.com/group/qc-algorithms
- All pertinent grading information and topics to be covered is found in the syllabus, which can also be found at the above yahoo group.
- The following books were recommended by the professor during the lecture.
  1) Introduction to Algorithms by Cormen, Leiserson, Rivest and Stein
     ISBN 978-0262033848
  2) Introduction to the Design and Analysis of Algorithms by Levitin
     ISBN 978-0132316811
  3) Algorithms by Sedgewick and Wayne
     ISBN 978-0321573513
  4) An Introduction to the Analysis of Algorithms by Sedgewick and Flajolet
     ISBN 978-0321905758

Home Work:
Do assignment number 1, which has been posted on the yahoo group. The instructor reserved the right whether to collect the homework or not.

Purpose of Course: Not to code, but to compare the efficiency of solving the same problem using different methods.

What is an Algorithm?

- Step by step solution of a problem.
- Defined method to solve a particular problem.
- A process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer (Wikipedia.)

What is meant by analysis of an algorithm?

Are they **correct?** Do they **solve** the problem? How **efficient** is the algorithm?

Efficiency:

i) **Time Complexity**
   - **a.** Measured in number of iterations.
   - **b.** Provides comparability across different hardware or systems.

ii) **Space Complexity**
   - **a.** Measured in memory required (Ram, HDD, External.)
   - **b.** Two sub-groups:
     - **i.** Data: Unavoidable required space.
     - **ii.** Auxiliary: Depends on the algorithm and our main concern.

Correlation b/w Space and Time Complexities:

1) Can be independent
   E.g. an empty loop that runs forever. No impact on space, but time complexity is infinite.
2) Can be directly related.
   E.g. if we use an adjacency matrix to represent the graph - which is more expensive than an adjacency table, then usually the graph algorithm takes longer. More space -> More time.
3) Can be inversely related.
   E.g. in a database, if we use indexing (takes a lot of space), then we can considerably reduce the time for data retrievals.

Design Patters vs. Design Techniques:

Design Patterns: is a general reusable solution to a commonly occurring problem within a given context in software design as mentioned in Chapter 1 of the course book, and is not a focus in our course. These are two good books on the subject

i) http://www.amazon.com/Head-First-Design-Patterns-Freeman-ebook/dp/B00AA36RZY/
ii) http://www.amazon.com/Design-Patterns-Elements-Reusable-Object-Oriented-ebook/dp/B000SEIBB8

Design Techniques: This is the focus of the course. The following are some examples of design techniques, and given enough time we are expected to go over most, if not all of them by the end of the course.
E.g. Brute Force, Divide and Conquer, Decrease and Conquer, Backtracking, Greedy Algorithm, Parallel Algorithm, Dynamic Programming, Randomized Algorithm etc.

Examples relating to Algorithms of a certain Design Techniques:
Brute Force Example:
One student has not paid their bill. How can the registrar locate them on campus?
One Possible solution: Ask each student in each classroom, one by one.

How will be efficiency measured in this particular case?
Amount of time spent? Amount of steps walked?

Change the problem to **n-students**. How can the above solution be applied?
Use the same method for each student, start starting again after finding each one?
Use a list?

It is important to identify **<u>what</u>** is being measured - distance walked, checking individual students? The latter one - "comparisons" - is what is often measured in sorts.

Some Math stuff now:
Notations:

$\mathbb{Z}$ = The set of integers {…-2, -1, 0, 1, 2…}

$\mathbb{Q}$ = The set of rational numbers that can be expressed as fractions. They should also be in lowest terms. E.g. $\{\ldots\frac{1}{2},\frac{2}{3},\frac{5}{9}\ldots\}$

Formally $\{\frac{a}{b}$ $such\ that\ a \in \mathbb{Z}\ \&\&\ \mathrm{b} \in \mathbb{Z} - \{0\}\ \&\&\ \mathrm{gcd(a,b)} = 1\ \}$

$\mathbb{R}$ = The set of real numbers.

$\mathbb{N}$ = The set of Natural number. For this course, they include **ZERO.**
{0, 1, 2, 3…}

Sequence: a particular order in which related events, movements, or things follow each other. (Wikipedia).
Practically, a sequence can have any number of terms as the next term of formula for it.

Some Formulas:
Sum of First n Natural Numbers (1 + 2 + 3 +…+ n):

$$\frac{n(n+1)}{2}$$

Sum of First (n-1) Natural numbers (1 + 2 + 3 +…+ (n - 1)):

$$\frac{n(n-1)}{2}$$

Some Proofing Techniques:
Proof by Induction / Mathematical Induction / Weak Induction:
Proving the formula for sum of first n natural numbers.

Base Case: n = 1. Apply above formula, it yields 1, which is correct.
Inductive Hypothesis:

Assume that the function/formula is true. Let's call is P(k).
Prove that P(k) is true for all k.

Say we have the numbers:
1 +2 + 3 + … + k + (k + 1)
We already know that the sum of the first k natural number is $\frac{n(n+1)}{2}$, so we can re-write the above numbers as:

$$\frac{k(k+1)}{2} + (k+1)$$

Which should give us the sum for the first (k+1) natural numbers.
Simplifying the above expression yields:

$$\frac{(k+1)\big((k+1)+1\big)}{2}$$

This shows that the formula is the same for k+1 as it was for k, thus concluding the proof.

Geometric Proof of Sum of Natural Numbers:
This site describes is much better than I can draw here:
http://www.9math.com/book/sum-first-n-natural-numbers

Another Method:
Using lower and upper bounds.
Say we want to sum the numbers: 1 + 2 + 3 +… + n
Find Lower Bound: has to be greater than or equal to at least n, so LB $\geq n$
Find Upper Bound: has to be less or equal to the square of the largest number, so UB $\leq n^2$

The general form a polynomial equation is:

$$P(x) = a_0 + a_1x^1 + a_2x^2 + \cdots + a_nx^n$$

For a quadratic function:

$$P(x) = ax^2 + bx + c$$

Using any three test values, we can get three equation from the above polynomial, and solve them to get the values of a, b, and c.

We took values: n = 0 (sum 0), 1 (sum 1), and 2 (sum 3)
We get equations:

$$0a + 0b + 1c = 0$$
$$1a + 1b + 1c = 1$$
$$4a + 2b + 1c = 3$$

Solving these equations give $a = ½, b = ½, and\ c = 0.$
Putting it back in the original quadratic equation gives:

$$P(x) = \frac{1}{2}x^2 + \frac{1}{2}x + 0 => \frac{x(x+1)}{2}$$

Which is the original formula we were trying to prove and simply requires replacing x with n.

Proof by Contradiction:
In logic, proof by contradiction is a form of proof, and more specifically a form of indirect proof, that establishes the truth or validity of a proposition by showing that the proposition's being false would imply a contradiction. (Wikipedia)

Prove that $\sqrt{2}$ is irrational.
Assume that $\sqrt{2}$ is rational. Which means it can be written as a/b in lowest terms.

Then we can write	$\frac{a}{b} = 2$	(By definition of rational numbers.)
	$\frac{a^2}{b^2} = 2$	(Squaring both sides.)
	$a^2 = 2b^2$	(a is even due to being a multiple of 2)
	$(2k)^2 = 2b^2$	(Even number a = 2k)
	$2k^2 = b^2$	(Since RHS is the square of an even number, it
	$2k^2 = (2l)^2$	(must also be even, so b = 2l)
	$2k^2 = 4l^2$	
	$k^2 = 2l^2$	

We have shown that both a and b are even numbers, and that $\sqrt{2}$ is equal to this fraction. But even numbers can be further reduced, which contradicts the fact that a and b are in lowest terms. Thus our initial assumption that $\sqrt{2}$ is rational is false, which proves by contradiction that it must be irrational.

Exponents: are a shorthand way to show how many times a number, called the base, is multiplied times itself.

Notation: $x^y = z$

$\mathbf{2 * 2 * 2 => 2^3 = 8}$

Logarithms: In mathematics, the **logarithm** is the inverse operation to exponentiation. That means the **logarithm** of a number is the exponent to which another fixed value, the base, must be raised to produce that number.

Notation: $log_x z = y$

$log_2 8 = 3$

The base, 2, must be raised to 3 to get the log base 2 of the value 8.

On most calculators, generally only the functions of $\log_{10}$ is given which has the base 10 and the lg, the natural log, which has base *e (sometimes referred to as Euler's Number, constant value approximately 2.71828).*

Log with bases can be broken down as follows:

$$log_x Z = \frac{log_c Z}{log_c x}$$

Using the calculator, say we wanted to calculate $log_2 8$ we would type $\frac{\log(8)}{\log(2)}$ which has the common base of 10.

Exponents and logarithms are **inverses** of each other.

Rules of Logarithms: [*http://www.sosmath.com/algebra/logs/log4/log44/log44.html*](http://www.sosmath.com/algebra/logs/log4/log44/log44.html)

Asymptotic Notation:

i) Big Oh (O)
ii) Little Oh (o)
iii) Big Omega (Ω)
iv) Little Omega (ω)
v) Theta (Θ)

Detailed information on definitions of these notations:
[*https://en.wikibooks.org/wiki/Data_Structures/Asymptotic_Notation*](https://en.wikibooks.org/wiki/Data_Structures/Asymptotic_Notation)

Relation b/w functions: $f(x) = \_\_g(x)$

Consider the two functions to be polynomial functions, then we can use L' Hospital's Rule to find which notation fits. The reason or this is that sometimes it is difficult to find the ratio of the functions themselves, which is why we can use this rule to compare the ratio of the derivatives, which is the same as the ratio of the functions. Possible outcomes:

$$\lim_{n\to\infty}\left(\frac{f'(x)}{g'(x)}\right) = \begin{cases} \infty & \begin{cases} f(x) = \Omega(\mathrm{g(x)}) \\ f(x) = \omega(\mathrm{g(x)}) \end{cases} \\ \boldsymbol{c} & \begin{cases} f(x) = \mathrm{O}(\mathrm{g(x)}) \\ f(x) = \Omega(\mathrm{g(x)}) \\ f(x) = \Theta(\mathrm{g(x)}) \end{cases} \\ \mathbf{0} & \begin{cases} f(x) = \mathrm{O}(\mathrm{g(x)}) \\ f(x) = \mathrm{o}(\mathrm{g(x)}) \end{cases} \end{cases}$$

Take these functions: $f(x) = 3n^2 + 5n + 7$
$g(x) = n^3 + 100n + 250$

Applying the above rule twice, we can get the form:

$$\lim_{n\to\infty}\left(\frac{1}{n}\right) = 0$$

This means that f(n) is both big-Oh and little-Oh of g(n).
We should prefer to say little-oh, because it is more strict in its definition.

Another Example

$$\begin{Bmatrix} f(x) = 1000n^2 \\ g(x) = 2n^2 \end{Bmatrix} f(n) = \Theta(\mathrm{g(n)})$$

B/c co-efficient don't matter. Both functions grow at the same rate.

Another Example

$$\begin{Bmatrix} f(x) = 1000n^2 \\ g(x) = 2n^3 \end{Bmatrix} f(n) = \mathrm{O}(\mathrm{g(n)}) and\ o(g(n))$$

Even though the second function grows at a faster rate as n approaches infinity, it is smaller at lower values.

Exponential Functions:
Base coefficients or the coefficients within the exponents are both important and cannot be discarded. But any other constants can be discarded the same as in the polynomial examples. (Short example: $2^{n+1} + 100c$ The (n+1) or the 2 cannot be discarded, but the 100c can be.)

Logs and Orders of Magnitudes:

$$\begin{Bmatrix} f(x) = 2^n \\ g(x) = 2^{2n} => 4^n \end{Bmatrix}$$

Generally speaking, numerically $\log_2 n > \log_3 n$
But approximately they are Θ of each other.

$\log_b n$ *and* $\log_c n$ are the same order of growth because $\log_b n = \frac{\log_c n}{\log_c b}$ *and* $\log_c b = 1/\log_b c$ which is a constant co-efficient.

On the other hand, the following function DO NOT have the same order of magnitude:

$$\begin{Bmatrix} f(x) = n^{(\log_2 4)} \\ g(x) = n^{(\log_2 3)} \end{Bmatrix}$$

Some more complex functions:

$$n!\ vs\ 2^n$$

Which is bigger? Or grows faster? It is not easy to tell just by looking sometimes.
Sometimes taking logarithm or exponents of functions helps identify orders of magnitude.

For n!, by definition of the factorial function,

$$n(n-1)(n-2)\ldots(1)$$

the highest order of magnitude is n^n. So taking log of this function, and applying the logarithm product rule gives:

$$\log_2 n! \leq n \log_2 n$$

For 2^n, taking the log gives:

$$\log_2 2^n = n \log_2 2 = n$$