



COMP90038 Algorithms and Complexity

Tutorial 1 Introduction to algorithms and data structures

Tutor: Chuang(Frank) Wang





Contact

Email: chuangw1@unimelb.edu.au

LinkedIn:



Tutorial Slides can be accessed at https://github.com/chuangw46/COMP90038_Algorithms



Prerequisite

- Chapter 1. Levitin, A. (2012). *Introduction to the design & analysis of algorithms.* (3rd Ed.)

- Week 1 Lectures



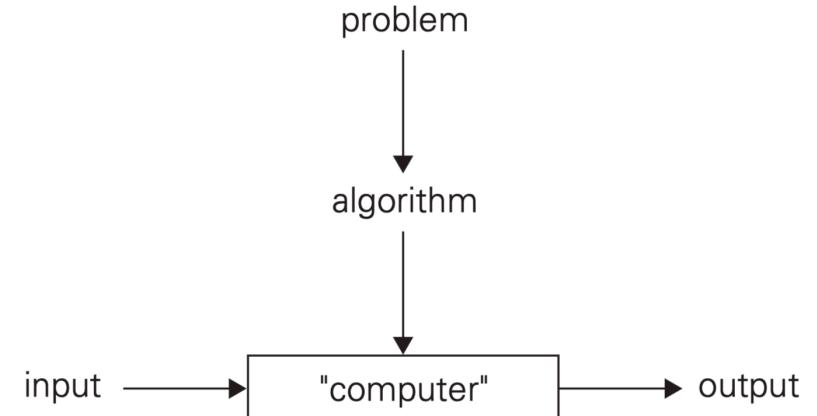
THE UNIVERSITY OF
MELBOURNE

1. Review

Algorithms & Data Structures

Algorithm:

- A sequence of **well-defined, computer-implementable instructions for solving a problem**
- Algorithms **operate on data.**



Sourced from Levitin's book

Data Structure:

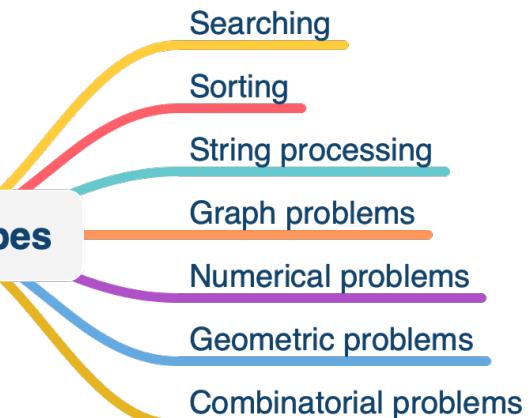
- **Storing data**
- **Organizing data so that it can be used efficiently.**



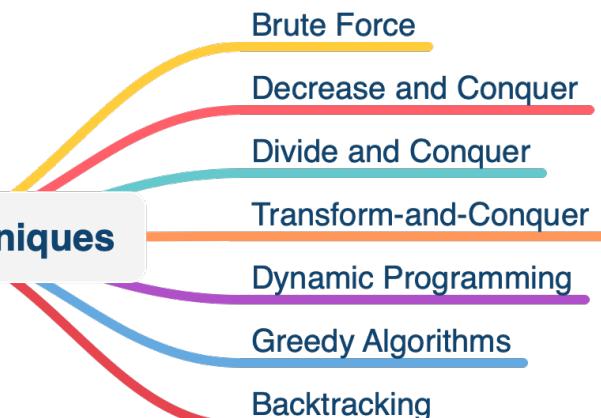
Think of a data structure as a bucket storing something

Bird's eye view of what we will study

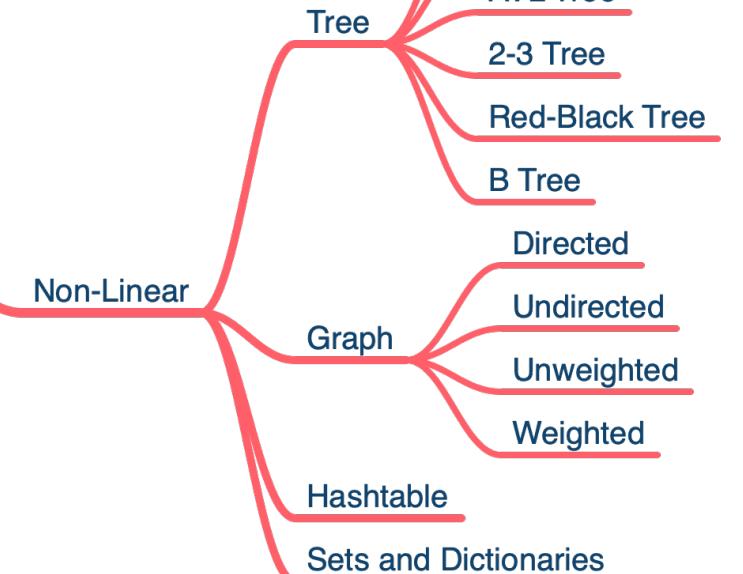
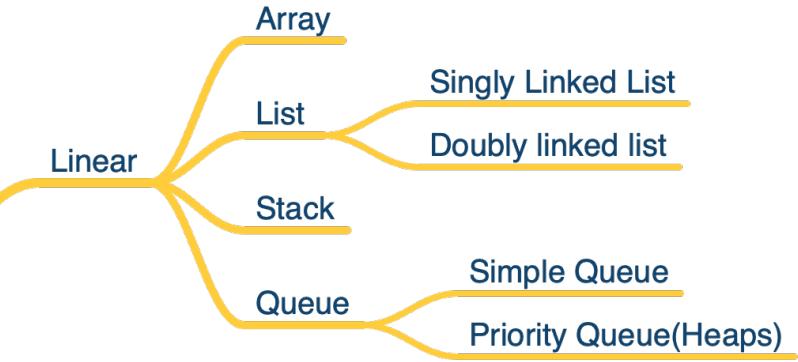
Problem Types



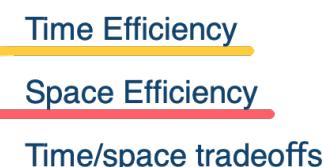
Algorithms Techniques



Data Structures



Algorithm Efficiency



It's okay! We will get through these together!!

Array

Array

Example

6	9	2	3	7	5	8
0	1	2	3	4	5	6

`data = [6, 9, 2, 3, 7, 5, 8]`

`data[4] = 7`

Memory representation

42148	6
42150	9
42152	2
42154	3
42156	7
42158	5
42160	8

Tradeoffs

Pros

Locating, storing(append), or retrieving a cell -> very fast (constant time)

Cons

insert or delete an element -> slow as we have to shift subsequent elements

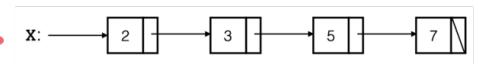
Maintaining a contiguous bank of cells with information can be difficult and time-consuming.

(Singly) Linked List

Linked List

What

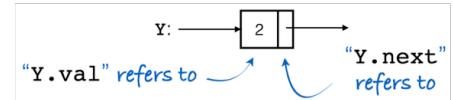
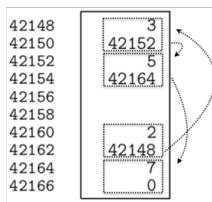
1. A data structure in which the elements are **not stored at contiguous/adjacent memory locations**.
2. To accommodate this feature, each node(cell) has two things:
Value(Data) + **Pointer(also called reference in Java)** that indicates where the next data is in the memory.



- x is (a pointer to) the head node of the linked list.
- x represents the linked list.

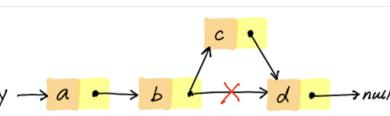
Example

Memory representation

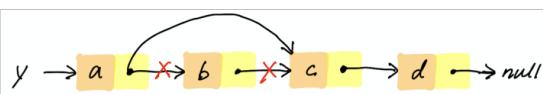


Operations

Insert



Delete



Traverse

Use loop(while loop or for loop)

Pros & Cons

Pros

Inserting and deleting data -> fast as it does not require moving and shifting subsequent data elements

Cons

insert or delete an element -> slow as we have to shift subsequent elements

access a specific element -> slow as we need to traverse the list from the head node to find it
Note: Generally, we can not randomly access any element as we do in array by index.(Depends on programming language)

Require more memory storage -> as each node contains a pointer and it requires extra memory for itself.



THE UNIVERSITY OF
MELBOURNE

2. Tutorial Questions



Question 1

1. Consider the usual (unsigned) binary representation of integers. For example, 10110010 represents 178, and 000011 represents 3.
 - (a) If we call the bits in an n -bit word $x_{n-1}, x_{n-2}, \dots, x_2, x_1, x_0$ (so x_0 is the *least significant* bit), which natural number is denoted by $x_{n-1}x_{n-2}\cdots x_2x_1x_0$?
 - (b) Describe, in English, an algorithm for converting from binary to decimal notation.
 - (c) Write the algorithm in (pseudo-) code.
 - (d) Describe, in English, how to convert the decimal representation to binary.

Answers:

$$10110010: 1 * 2^7 + 0 * 2^6 + 1 * 2^5 + 1 * 2^4 + 0 * 2^3 + 0 * 2^2 + 1 * 2^1 + 0 * 2^0 = 178$$

(a) $x_{n-1}x_{n-2}\dots x_2x_1x_0$ represents the integers from **0 to $2^n - 1$ (inclusive)** because

$$\text{it denotes } x_{n-1} \times 2^{n-1} + x_{n-2} \times 2^{n-2} + \dots + x_2 \times 2^2 + x_1 \times 2^1 + x_0 \times 2^0 = \sum_{i=0}^{n-1} x_i \times 2^i$$



Question 1

1. Consider the usual (unsigned) binary representation of integers. For example, 10110010 represents 178, and 000011 represents 3.
 - (a) If we call the bits in an n -bit word $x_{n-1}, x_{n-2}, \dots, x_2, x_1, x_0$ (so x_0 is the *least significant* bit), which natural number is denoted by $x_{n-1}x_{n-2}\cdots x_2x_1x_0$?
 - (b) Describe, in English, an algorithm for converting from binary to decimal notation.
 - (c) Write the algorithm in (pseudo-) code.
 - (d) Describe, in English, how to convert the decimal representation to binary.

Answers:

$$\begin{aligned}1011: \quad & 1 * 2^3 + 0 * 2^2 + 1 * 2^1 + 1 * 2^0 \\& = 2 * (2 * (2 * (x_3) + x_2) + x_1) + x_0 \\& = 2 * (2 * (2 * (1) + 0) + 1) + 1\end{aligned}$$

(b) To convert from binary to decimal notation,

1. visit the binary digits from left to right →
2. construct the result in an “accumulator”, the accumulator begins with 0 →
3. as long as there is a next bit to process, →
double the value of the accumulator and add the value of that next bit. →
4. get the result →

(c)

```
function BINTODEC( $x_{n-1}x_{n-2}\dots x_2x_1x_0$ )
    a ← 0
    for i ← 0 to n - 1 do
        a ← 2a +  $x_{n-1-i}$ 
    return a
```



Question 1

1. Consider the usual (unsigned) binary representation of integers. For example, 10110010 represents 178, and 000011 represents 3.
 - (a) If we call the bits in an n -bit word $x_{n-1}, x_{n-2}, \dots, x_2, x_1, x_0$ (so x_0 is the *least significant* bit), which natural number is denoted by $x_{n-1}x_{n-2}\cdots x_2x_1x_0$?
 - (b) Describe, in English, an algorithm for converting from binary to decimal notation.
 - (c) Write the algorithm in (pseudo-) code.
 - (d) Describe, in English, how to convert the decimal representation to binary.

Answers:

(d) $1001011 \quad 1 * 2^6 + 0 * 2^5 + 0 * 2^4 + 1 * 2^3 + 0 * 2^2 + 1 * 2^1 + 1 * 2^0 = 75$

Successive division:

$$75 \div 2 = 37 \text{ } R \text{ } 1 \quad \text{-- least significant bit}$$

$$37 \div 2 = 18 \text{ } R \text{ } 1$$

$$18 \div 2 = 9 \text{ } R \text{ } 0$$

$$9 \div 2 = 4 \text{ } R \text{ } 1$$

$$4 \div 2 = 2 \text{ } R \text{ } 0$$

$$2 \div 2 = 1 \text{ } R \text{ } 0$$

$$1 \div 2 = 0 \text{ } R \text{ } 1 \quad \text{-- most significant bit}$$

function DECTOBIN(d)

$n \leftarrow 0$

while $d \neq 0$ **do**

$x_n \leftarrow d \bmod 2$

$d \leftarrow \lfloor d / 2 \rfloor$

$n \leftarrow n + 1$

return $x_{n-1}x_{n-2}\dots x_2x_1x_0$

Question 2

2. Below are three (correct) formulas for calculating the area of a triangle with sides A , B , and C , of length a , b , and c , respectively.

- (a) $S = \sqrt{p(p - a)(p - b)(p - c)}$, where $p = (a + b + c)/2$
- (b) $S = \frac{1}{2}ab \sin \theta$, where θ is the angle between sides A and B
- (c) $S = \frac{1}{2}ah_A$, where h_A is the height to base A

Which of these would you accept as an *algorithm*?

Answers:

(a) Heron's formula is a fine algorithm as long as the square root operation is a primitive operation available on our computing device, or we know how to calculate square roots. For example, we can implement it as follows:

$$\text{Based on } \sum_{i=1}^k (2i - 1) = k^2$$

```
function SQRT(n)
    root ← 1
    while root * root ≤ n do
        root ← root + 1
    return xn-1xn-2 ... x2x1x0
```

```
function SQRT(n)
    square ← 1
    i ← 1
    while square ≤ n do
        square ← square + 2i + 1
        i ← i + 1
    return i - 1
```



Question 2

2. Below are three (correct) formulas for calculating the area of a triangle with sides A , B , and C , of length a , b , and c , respectively.

- (a) $S = \sqrt{p(p - a)(p - b)(p - c)}$, where $p = (a + b + c)/2$
- (b) $S = \frac{1}{2}ab \sin \theta$, where θ is the angle between sides A and B
- (c) $S = \frac{1}{2}ah_A$, where h_A is the height to base A

Which of these would you accept as an *algorithm*?

Answers:

- (b) This is a problematic formulation, because, even if the sine function is considered a primitive, there is no indication of how to compute the angle.
- (c) Again, the formula says, “the height to base A”, without any indication of how to find that. So we can’t really call that an algorithm.



Question 3

3. Consider the following problem: You are to design an algorithm to determine the best route for a subway passenger to take from one station to another in a city such as Kolkata or Tokyo.
- Discuss ways of making the problem statement less vague. In particular, what is “best” supposed to mean?
 - How would you model this problem by a graph?

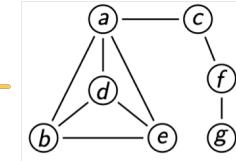
Answers:

- “best” can mean many things. We might want to minimize
 - The travel time
 - The number of train stops
 - The number of train changes
 - Some combination of these

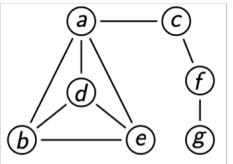
Graphs

What

1. A data structure that consists of a finite set of **nodes** (or vertices) and a set of **edges** connecting them.
2. A **pair** (x,y) is referred to as an **edge**, which communicates that the x vertex **connects** to the y vertex.
3. Mathematic notation: $G = \langle V, E \rangle$
V : Set of nodes or vertices
E: Set of edges (a binary relation on V)

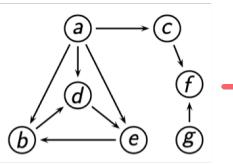


Undirected



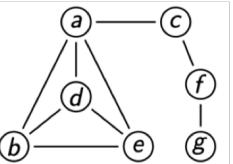
$V = \{ a, b, c, d, e, f, g \}$
 $E = \text{symmetric closure of } \{ (a, b), (a, c), (a, d), (a, e), (b, d), (b, e), (c, f), (d, e), (f, g) \}$

Directed

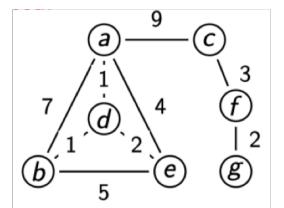


$V = \{ a, b, c, d, e, f, g \}$
 $E = \{ (a, b), (a, c), (a, d), (a, e), (b, d), (c, f), (d, e), (e, b), (g, f) \}$

Unweighted



Weighted



Common Graph Representation

Adjacency Matrix

	0	1	2	3
0	1			1
1			1	
2		1		
3	1			1

An Adjacency Matrix is a 2D array of size $V \times V$ where V is the number of nodes in a graph.

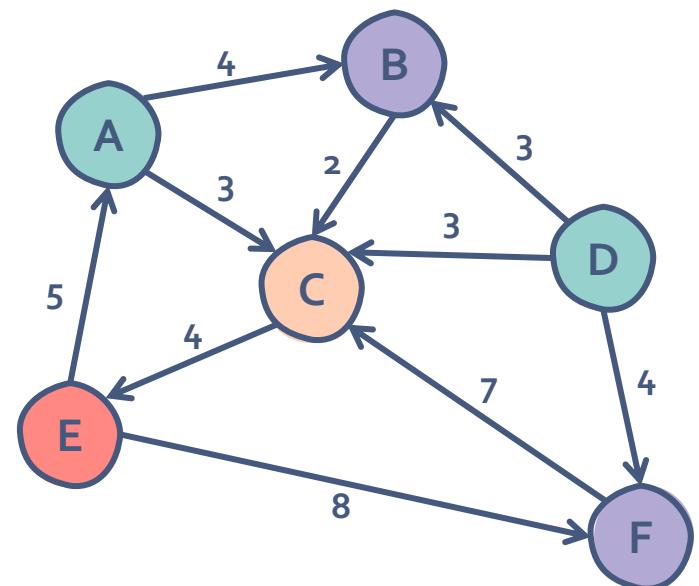
A slot $\text{matrix}[i][j] = 1$ indicates that there is an edge from node i to node j .

Question 3

3. Consider the following problem: You are to design an algorithm to determine the best route for a subway passenger to take from one station to another in a city such as Kolkata or Tokyo.
- Discuss ways of making the problem statement less vague. In particular, what is “best” supposed to mean?
 - How would you model this problem by a graph?

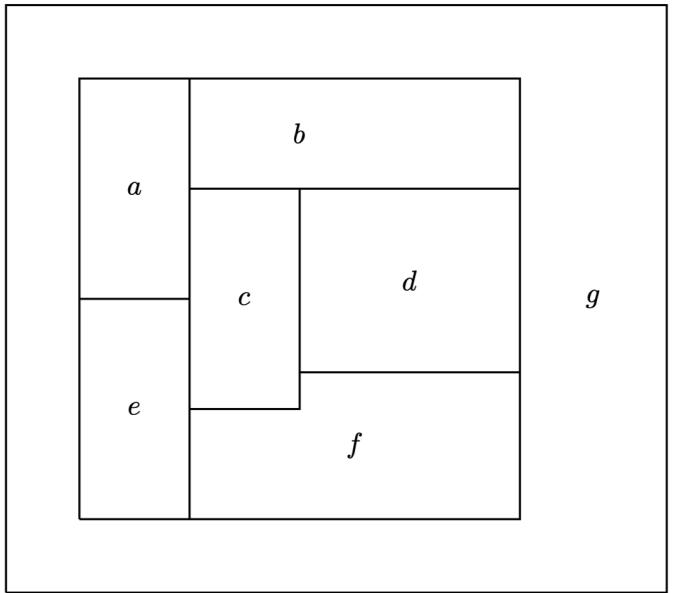
(b)

- The nodes A,B,C,D,E,F represent 5 stations in Tokyo.
- There is an edge between two nodes iff the stations that correspond to the incident nodes are directly connected by a train line.
- The direction(e.g. A->B) indicates that the route only goes in one direction.
- The weight between two stations denotes the travel time.

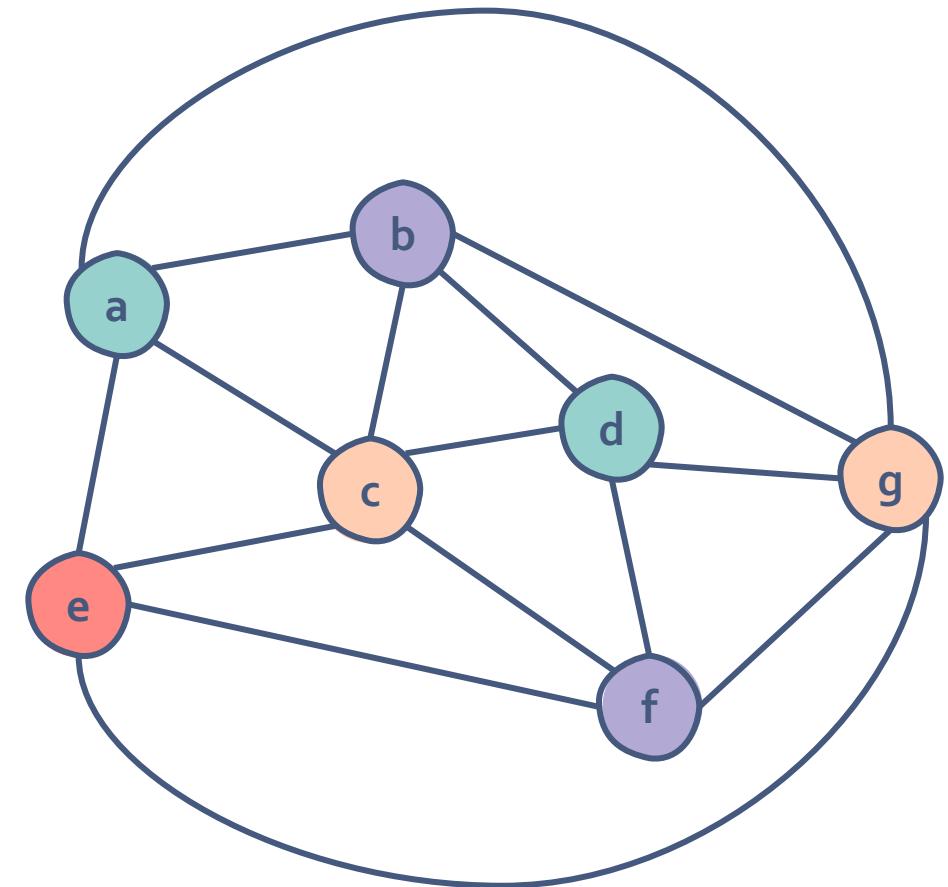


Question 4

4. Consider the following map:



- A cartographer wants to colour the map so that no two neighbouring countries have the same colour. How few colours can she get away with?
- Show how to reduce the problem to a graph-colouring problem.



Algorithm:

- Color first vertex with first color.
- Do following for remaining $V-1$ vertices.
 - Consider the currently picked vertex and color it with the lowest numbered color that has not been used on any previously colored vertices adjacent to it.
 - If all previously used colors appear on vertices adjacent to v, assign a new color to it.



Question 5

5. You have to search for a given number n in a *sorted* list of numbers.
- (a) How can you take advantage of knowing that the list is represented as a linked (and sorted) list?
 - (b) How can you take advantage of knowing the list is represented as an array?

Answers:

- (a) We can stop searching as soon as we find (n or) a number greater than n .

In a linked list, we have to traverse the whole list from the head node until we find the target because we cannot access a specific item.

- (b) With an array we can use **binary search** (we will cover this algorithm later in this course).

In an array, we can access a specific element by its index, which makes binary search possible.

Question 6

6. In the first lecture we discussed different ways of calculating the greatest common divisor of two positive integers. A mathematician might simply write

$$gcd(x, y) = \max\{z \mid \exists u, v : x = uz \wedge y = vz\}$$

and suggest we develop a functional programming language that allows us to write this, leaving it to the language implementation to translate this definition to an efficient algorithm. Do you imagine a time when we may be able to do this? If we restrict our attention to functions like gcd which takes a pair of integers and returns an integer, do you think we may some day be able to automatically turn any function definition into a working algorithm?

Answers:

The point of asking this question is to call attention to that fact that there are functions that are **not computable**.

Recall the first slide in this tutorial:

An algorithm is a sequence of well-defined, computer-implementable instructions for solving a problem



THE UNIVERSITY OF
MELBOURNE

Next Week:

Algorithm Analysis

- how to reason about an algorithm's resource consumption(time and space).

Thank you !





COMMONWEALTH OF AUSTRALIA

Copyright Regulations 1969

Warning

This material has been reproduced and communicated to you by or on behalf of the University of Melbourne pursuant to Part VB of the *Copyright Act 1968* (*the Act*).

The material in this communication may be subject to copyright under the Act. Any further copying or communication of this material by you may be the subject of copyright protection under the Act.

Do not remove this notice