Teach A level Computing: Algorithms and Data Structures

Eliot Williams

@MrEliotWilliams

Queen Mary University of London · SUPPORTED BY MAYOR OF LONDON · COMPUTING AT SCHOOL EDUCATE · ENGAGE · ENCOURAGE · KING'S College LONDON

Course Outline

1	Representations of data structures: Arrays, tuples, Stacks, Queues,Lists
2	Recursive Algorithms
3	Searching and Sorting
4	Hashing and Dictionaries, Graphs and Trees
5	Depth and breadth first searching ; tree traversals

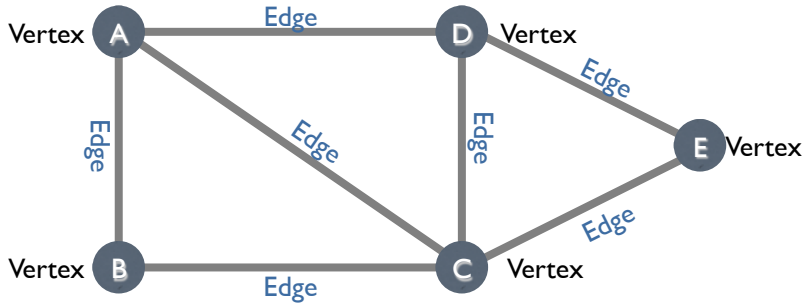
Queen Mary University of London · SUPPORTED BY MAYOR OF LONDON · COMPUTING AT SCHOOL EDUCATE · ENGAGE · ENCOURAGE · KING'S College LONDON

- http://www.aqa.org.uk/subjects/computer-science-and-it/as-and-a-level/computer-science-7516-7517/subject-content-a-level/fundamentals-of-algorithms

Graphs

Queen Mary University of London SUPPORTED BY MAYOR OF LONDON COMPUTING AT SCHOOL EDUCATE · ENGAGE · ENCOURAGE KING'S College LONDON

What is a graph



- Some resources may refer to nodes and arcs…..

Terminology

- A graph can have cycles (loops)
- A weighted graph has a value on the edge
- A directed graph has arrows to indicate direction
- A multigraph has multiple edges connecting the same vertices
- A tree is a type of graph where…
  - All vertices are connected
  - The graph is undirected
  - There are no cycles

Uses of graphs

- Route planning and map
- Web page links
- Social media connections
- Underground tube links
- …. etc

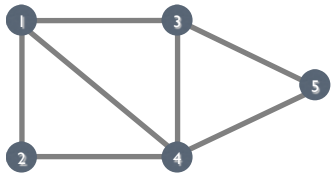
Representing graphs

- Graphs need to be processed by computer programs:
  - Adjacency matrix
  - Adjacency list

Queen Mary University of London
SUPPORTED BY MAYOR OF LONDON
COMPUTING AT SCHOOL EDUCATE · ENGAGE · ENCOURAGE
KING'S College LONDON

Adjacency Matrix

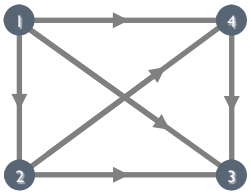
- (Undirected)
- If two vertices are connected then a 1 is place in two positions in the matrix:
  - Find the row for the first vertex, move to the cell in this row which is in the column for the second vertex
  - Find the row for the second vertex, move to the cell in this row which is in the column for the first vertex



	1	2	3	4	5
1	0	1	1	1	0
2	1	0	0	1	0
3	1	0	0	1	1
4	1	1	1	0	1
5	0	0	1	1	0

Adjacency Matrix

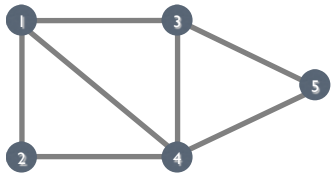
- (Directed)
- Positioning of 1s does not have to be symmetrical



	1	2	3	4
1	0	1	1	1
2	0	0	1	1
3	0	0	0	0
4	0	0	1	0

Adjacency List

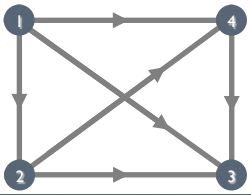
- Undirected Graph:
- Specifies the vertices that are adjacent to each vertex of the graph



Vertex	Adjacent Vertices
1	2,3,4
2	1,4
3	1,4,5
4	1,2,3,5
5	3,4

Adjacency List

- Directed Graph:
  - Must reflect direction of edges
  - Adjacent vertices are called **terminal vertices**
  - Terminal vertices are adjacent to the **initial vertex**

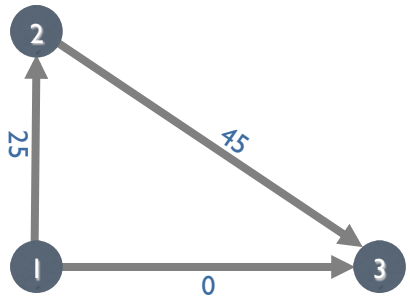


Initial Vertex	Terminal Vertices
1	2,3,4
2	3,4
3	
4	3

Labelled Graph

- Adjacency Matrix

	1	2	3
1	∞	25	0
2	∞	∞	45
3	∞	∞	∞



- Adjacency List

Initial Vertex	Terminal Vertices
1	2,25;3,0
2	3,45
3	

When to use adjacency matrices and lists

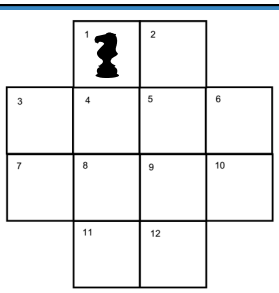
- **Matrix**
  - Use when there are many vertex pairs connected by edges
  - Indicates edge exists with one access
- **List**
  - Use if the graph is sparse (not as many edges)

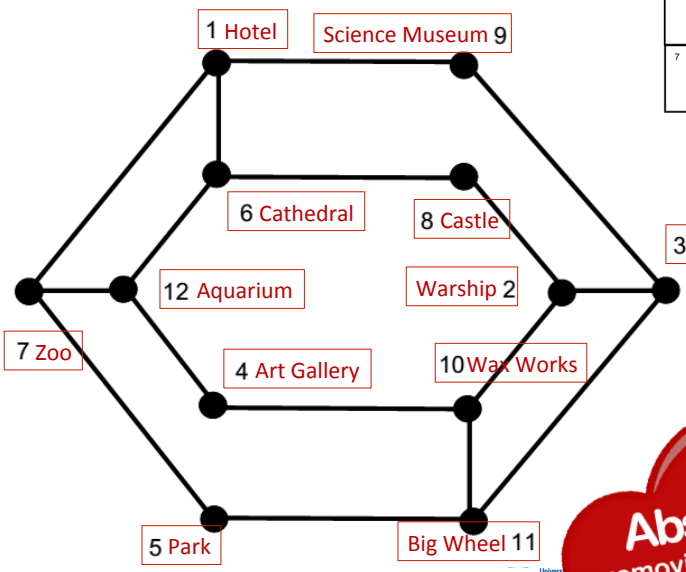
Activity 1.1 & 1.2

- Which is harder and why?

Queen Mary University of London · SUPPORTED BY MAYOR OF LONDON · COMPUTING AT SCHOOL EDUCATE · ENGAGE · ENCOURAGE · KING'S College LONDON

The Knights Tour Graph





Abstraction
removing unnecessary detail

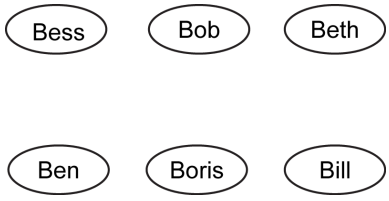
KING'S College LONDON

Exercise 2: A Puzzle

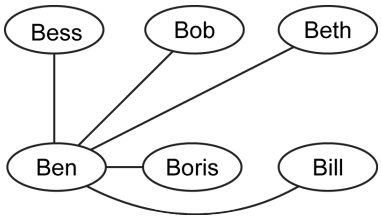
-
- Boris, Ben, Bob, Bess, Beth and Bill are waiting in a doctor's surgery. Looking around the room, each of them sees that they know some of the other people in the room. Ben knows five of the other people sitting in the room, Bob knows four of the others, Beth and Bill both know three of the others, Bess knows two of the others and Boris only knows one of the other people. Who knows who in the waiting room?

- Solve the puzzle (in pairs) by drawing a graph
- (this is a good starter exercise)

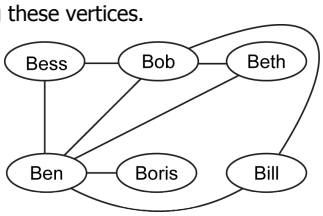
Step 1
Represent each of the people as a vertex.



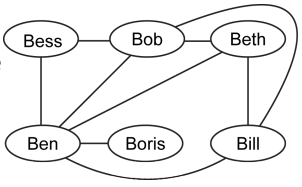
Step 2
Ben knows five of the other people in the room; this means that he knows all the others so draw edges connecting him to the other vertices.



Step 3
Now Boris only knows one other person – we can now tell that this is Ben. Bob knows four other people in the room; we know one of these is Ben and we know that Boris is not one of them so Bob must also know Beth, Bess and Bill. Draw edges representing these vertices.



Step 4
Bess only knows two other people; we can see from our graph that they are Ben and Bob. Bill and Beth both know three other people. From our graph, we can see that they both know Ben and Bob. They do not know Boris or Bess, therefore they must know each other. Draw the edge representing this relationship and the graph has now been completed and shows us who knows who in the waiting room

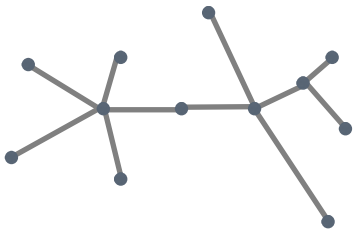


Connectivity

- Many applications of graphs involve getting from one vertex to another: e.g.
  - Finding the shortest route between two places
  - Finding the route out of a maze
- A graph is connected if there is a path between each pair of vertices
- Otherwise the graph is said to be disconnected

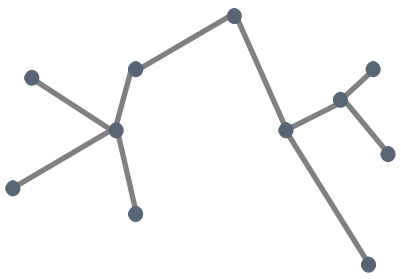
Trees

- All vertices are connected
- The graph is undirected
- There are no cycles



Rooted Tree

- Has one vertex that has been designated as the root
- Every edge is directed away from the root



Binary Tree

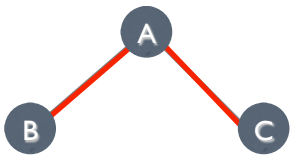
If each node in the tree has a maximum of two children then it is called a binary tree

Binary Search Tree

- Sorted Binary Tree:
  - left sub-tree contains only nodes with values less than the node's
  - right sub-tree contains only nodes with values greater than the node's
  - both left and right sub-trees are binary search trees

Binary Tree Traversal

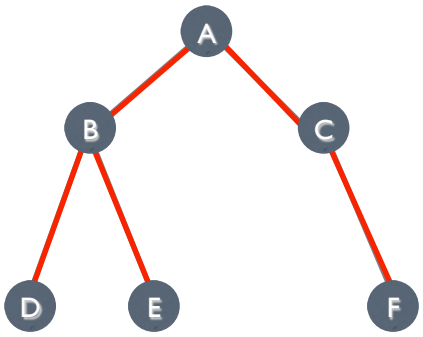
- Pre order (visit,left, right)
  [A, B, C]
- In order (left visit right)
  [B,A,C]
- Post order (left, right visit)
  - [B,C A]



Pre order Binary Tree Traversal

```
Def pre(tree,start):
        Visit node(print?)
        If node to the left of start:
                then dft left
        If node to the right of start:
                then dft right
```

A B D E C F



Exercise 3.1

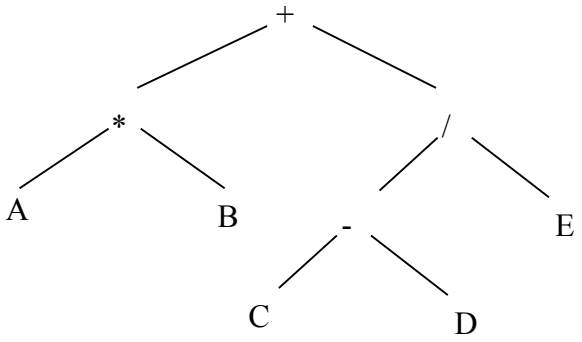
- Complete the Skelton for the pre-order Tree traversal

- 3.2 and 3.3
- Extend with a post and in -order

Queen Mary University of London
SUPPORTED BY MAYOR OF LONDON
COMPUTING AT SCHOOL EDUCATE · ENGAGE · ENCOURAGE
KING'S College LONDON

Exercise 4: More traversals

An algebraic expression is represented by the following binary tree.



- Show the output produced if the tree is traversed using:
  - ❖ pre-order traversal
  - ❖ in-order traversal
  - ❖ post-order traversal

+ * A B ? – C D E

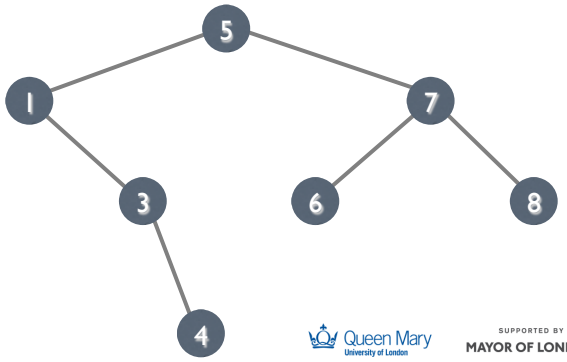
A * B + C – D / E

A B * C D - E /

Example Binary Search Tree

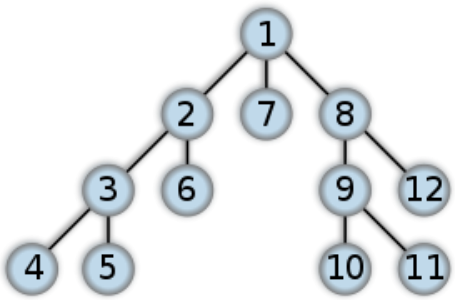
- Search algorithms for a BST are very efficient

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



Depth First Traversal

https://commons.wikimedia.org/wiki/File:Depth-first-tree.svg



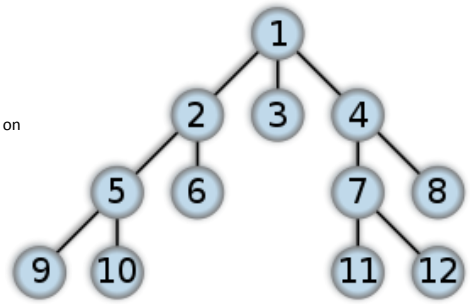
- Standard Algorithm for searching trees
- The Depth First Search (DFS), follows a chain of vertices until it cannot go further, at which point it backs up.
- This is a pre order traversal

```
def dfspreorder(tree,start):
    if start in tree:
        print(start)
        branches=tree[start]
        #print(branches)
        for i in range(0,len(branches)):
            dfspreorder(tree,branches[i])
```

This can also been done post order so would output 4,5,3,6,2 in the diagram this will involve a stack!

Queen Mary University of London | SUPPORTED BY MAYOR OF LONDON | COMPUTING AT SCHOOL EDUCATE · ENGAGE · ENCOURAGE | KING'S College LONDON

Breadth First Traversal



- Standard Algorithm for searching trees
- The Breadth First Search (BFS) visits all neighbours of each vertex before moving on to another vertex.

```
Breadth-First-Search(Graph, root):
    create empty set S
    create empty queue Q
    root.parent = NIL
    add root to S
    Q.enqueue(root)
    while Q is not empty:
            current = Q.dequeue()
            if current is the goal:
                    return current
            for each node n that is adjacent to current:
                    if n is not in S:
                        add n to S
                        n.parent = current
                        Q.enqueue(n)
```

Queen Mary University of London | SUPPORTED BY MAYOR OF LONDON | COMPUTING AT SCHOOL EDUCATE · ENGAGE · ENCOURAGE | KING'S College LONDON

Breadth vs depth

- You met Sam at a friends party and didn't get a number
- You know they must be friends with someone at the party to have been there
- To find Sam Do you depth first or breadth first search your facebook friend?

- DFS
- must see whole tree anyway
- you know d, the level of the answer
- don't care if the answer is closest to root

- BFS
  - answer is close to the root
  - you want the answer that is closest to the root
  - have multiple cores/processors
  - will always find the answer in finite time…..

Will also find you the shortest path to your destination (least number of edges)

Connectivity

- Many applications of graphs involve getting from one vertex to another: e.g.
  - Finding the shortest route between two places
  - Finding the route out of a maze

Connectivity

- A graph is **connected** if there is a path between each pair of vertices
- Else the graph is said to be **disconnected**

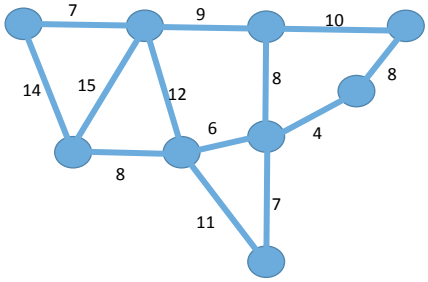
Paths

- Sequence of edge that begins at a vertex of a graph and travels from vertex to vertex along the edges of the graph
- http://oracleofbacon.org

Dijkstra's (Shortest Path)

Queen Mary University of London
SUPPORTED BY MAYOR OF LONDON
COMPUTING AT SCHOOL EDUCATE · ENGAGE · ENCOURAGE
KING'S College LONDON

Tueb failure….We need to walk from Bond street to Leicester Square to meet our date….



Step 1 Label the start vertex as 0.

Step 2 Box this number (permanent label).

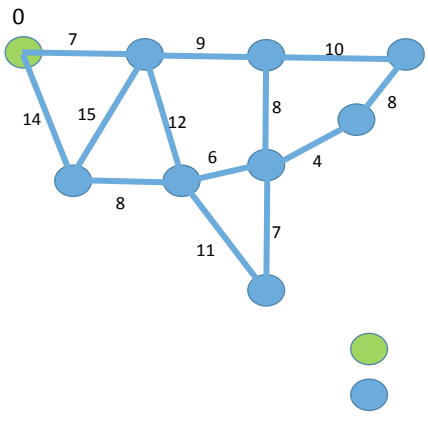
Step 3 Label each vertex that is connected to the start vertex with its distance (temporary label).

Step 4 Box the smallest number.

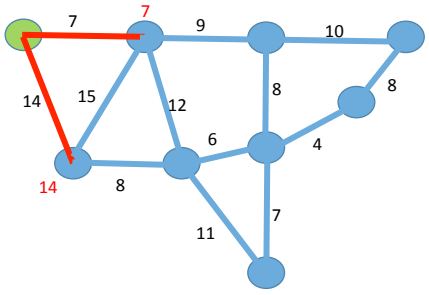
Step 5 From this vertex, consider the distance to each connected vertex.

Step 6 If a distance is less than a distance already at this vertex, cross out this distance and write in the new distance. If there was no distance at the vertex, write down the new distance.

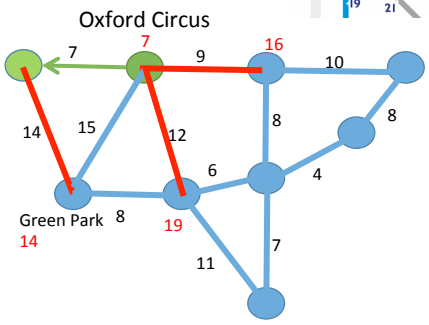
Step 7 Repeat from step 4 until the destination vertex is boxed.



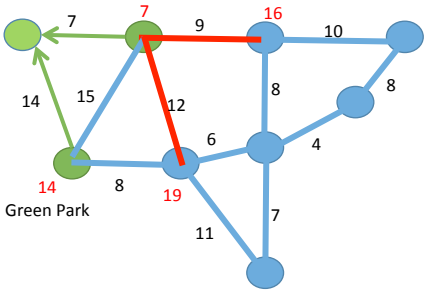
- Look at all unvisited neighbours of the current node and work out the tentative distance from the start



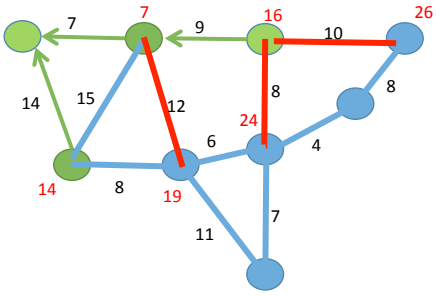
- Choose the unvisited node with the lowest tentative distance (Oxford Circus)
- Calculate tentative distance to all its neighbours and replace if lower
- Green Park is not updated as its tentative distance via Oxford circus is 22 and its current tentative distance is 14
- Mark the current node as visited
- Store the neighbouring node with the lowest distance



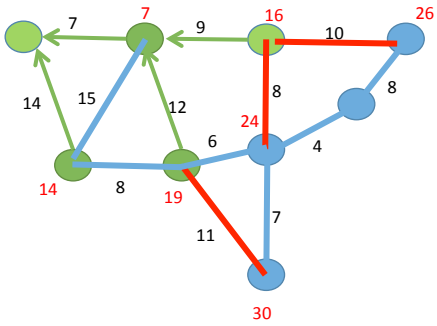
- Choose the unvisited node with the lowest tentative distance (Green Park)
- Calculate tentative distance to all its neighbours and replace if lower
- No changes….
- Mark the current node as visited
- Store the neighbouring node with the lowest distance



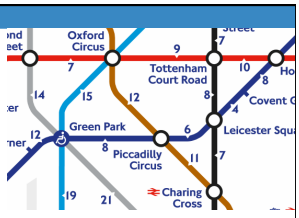
- Choose the unvisited node with the lowest tentative distance (16 – Tottenham Court Road)
- Calculate tentative distance to all its neighbours and replace if lower
- Mark the current node as visited
- Store the neighbouring node with the lowest distance

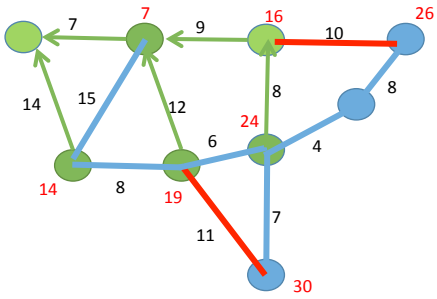


- Choose the unvisited node with the lowest tentative distance (19 – Piccadilly Circus)
- Calculate tentative distance to all its neighbours and replace if lower
- (19+6=25 so Leicester Square not updated)
- Mark the current node as visited & Store the neighbouring node with the lowest distance



- Choose the unvisited node with the lowest tentative distance (24 – Piccadilly Circus)
- In CS we are trying to find the shortest path to this point so we just need to find the neighbour with the lowest tentative distance
- We can then follow the shortest route back that we have stored…
- (mathematicians may be used to continuing to find the minimal spanning tree)





Exercise 5

- Also
- Interactive example here
- http://optlab-server.sce.carleton.ca/POAnimations2007/DijkstrasAlgo.html

Comparing Search Algorithms

- Breadth First search – Explores equally in all directions
- Dijkstra's Algorithm
  - Favours shorter(cheaper/quicker) paths.
  - calculates the distance from the start point.
- A*- modification of Dijkstra's algorithm
  - prioritises paths in the direction the goal
  - Uses the sum of distance from start and estimated distance from goal..

Taking this further….

- http://www.redblobgames.com/pathfinding/a-star/introduction.html
- http://www.redblobgames.com/pathfinding/a-star/implementation.html
- Computerphile – A star Search Algorithm
- https://youtu.be/ySN5Wnu88nE
- Maze Solving
- https://youtu.be/rop0W4QDOUI

Homework & Certificates

- No homework
- No certificates today
- Final homework deadline this time next week
  - (please email me if you are planning on completing the homework)

Queen Mary University of London SUPPORTED BY MAYOR OF LONDON COMPUTING AT SCHOOL EDUCATE · ENGAGE · ENCOURAGE KING'S College LONDON