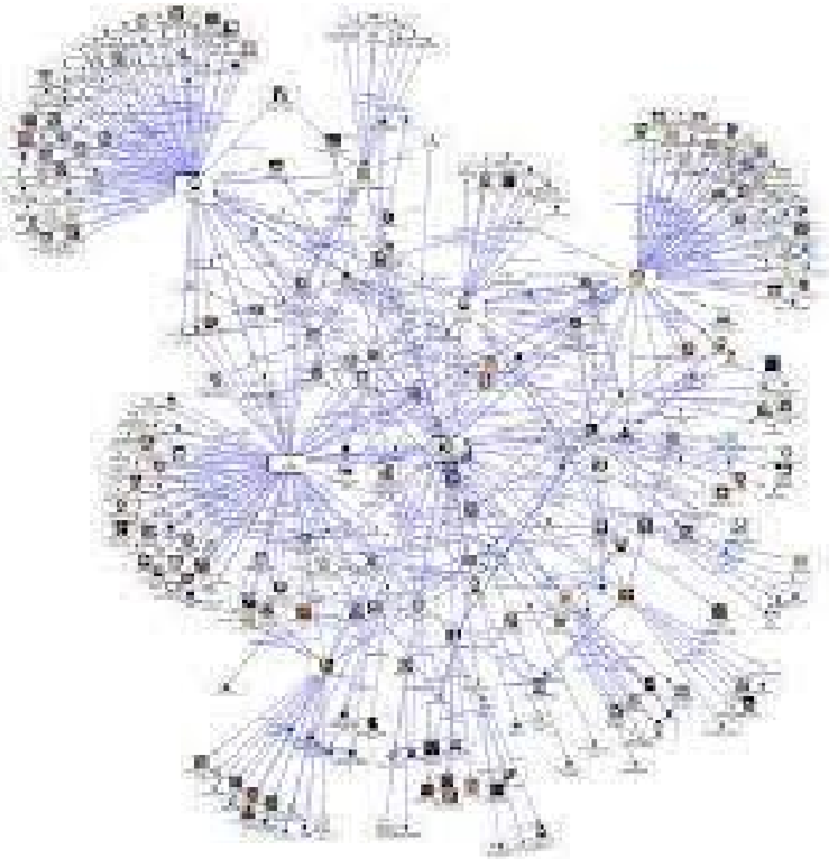
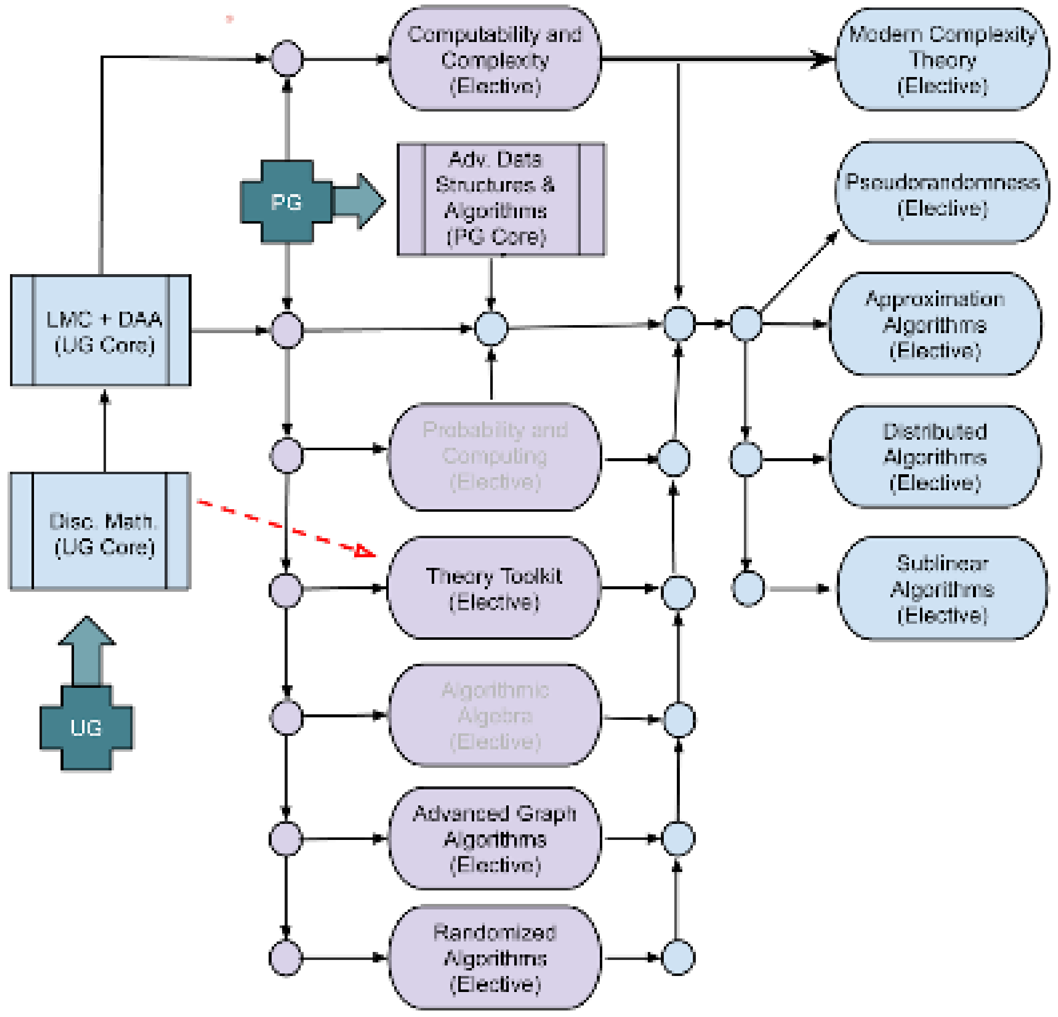
CS2700 : Programming and Data Structures.

GRAPH ADT.

- Graphs in real life.
- How to represent graphs?
- some basic (and very useful) algorithms on graphs.

Graphs are around us everywhere,





Computability and Complexity (Elective)

Modern Complexity Theory (Elective)

PG

Adv. Data Structures & Algorithms (PG Core)

Pseudorandomness (Elective)

LMC + DAA (UG Core)

Approximation Algorithms (Elective)

Probability and Computing (Elective)

Distributed Algorithms (Elective)

Disc. Math. (UG Core)

Theory Toolkit (Elective)

Sublinear Algorithms (Elective)

UG

Algorithmic Algebra (Elective)

Advanced Graph Algorithms (Elective)

Randomized Algorithms (Elective)

What is a graph

- a structure to represent relations amongst entities

- Social n/w graph :

- Road transport graph :

- Course dependancy graph :

Graphs : Basic definitions

A graph $G = (V, E)$

set of edges.

set of vertices

entities,

people, junctions,

courses

is a friend of,

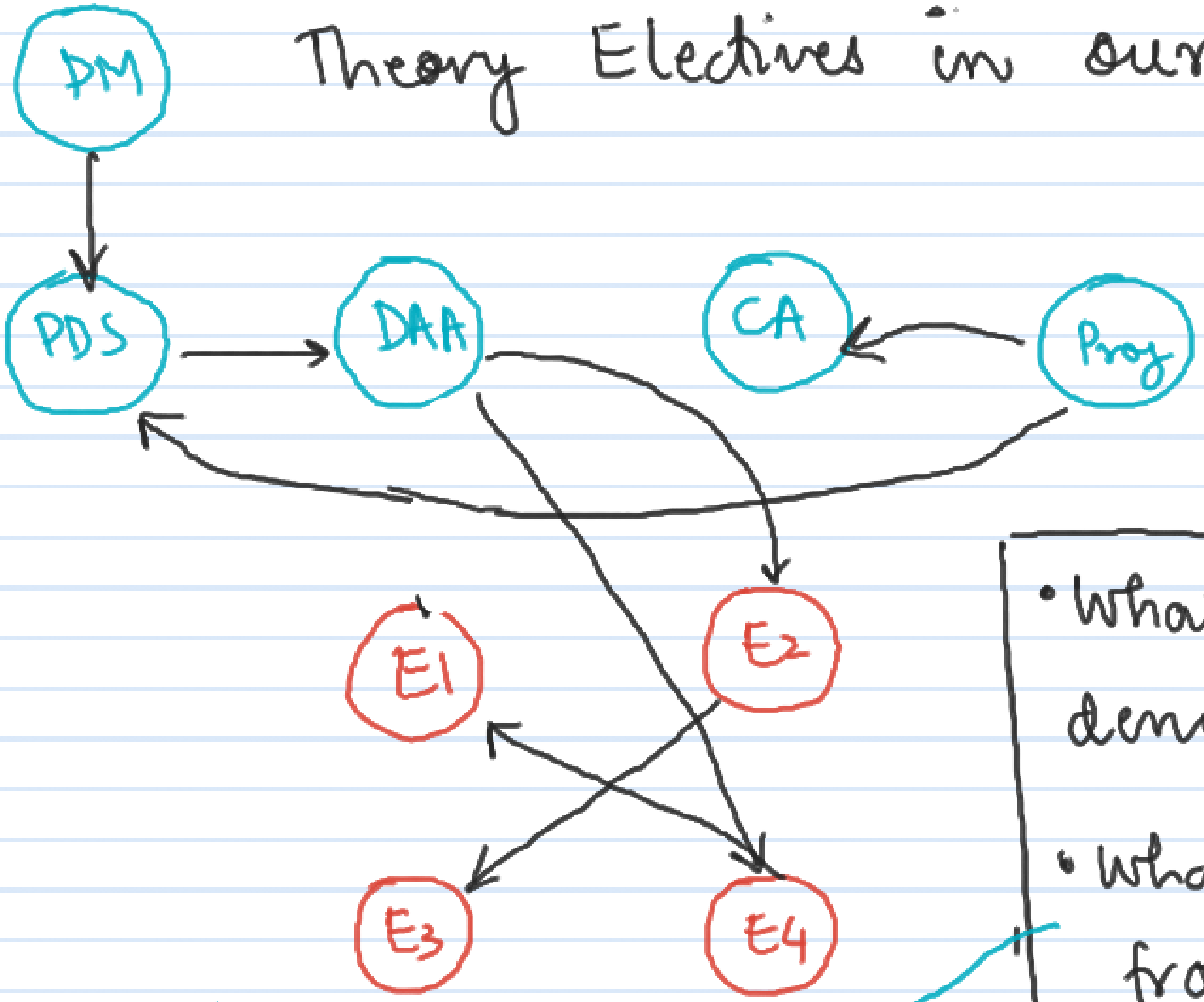
is connected to

is a prerq for :

Different kinds of graphs ?

GRAPHS IN PRACTICE : EXAMPLE 1.

Theory Electives in our dept.



1. Randomized algos. E1

2. Graph algo E2

3. Advanced graph algo. E3

4 Probability and Linear alg. E4

- What does an edge denote? (prerequisite)
- what does a path from u to v denote?
- which electives can be credited in 5th sem?

a transitive prereq

E4, E2 only

GRAPHS IN PRACTICE : EXAMPLE 2.

(SEAT)

- Students want to credit electives
- Elective courses want students enrolled.
- Are there any constraints?

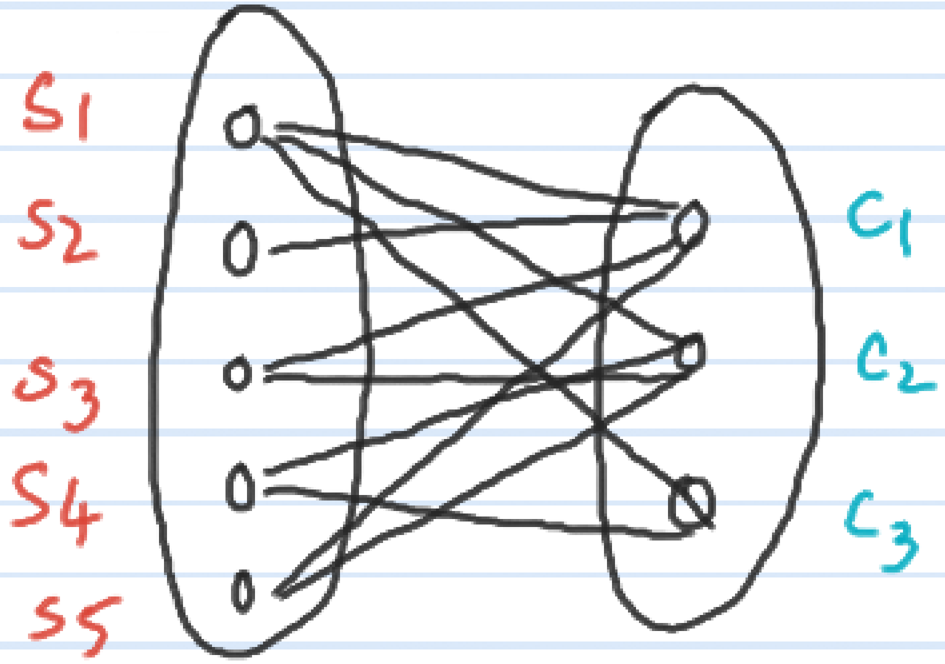
- How do we formulate it as a graph problem?

GRAPHS IN PRACTICE : EXAMPLE 2.

(SEAT for HS courses)

- Students want to credit HS electives [lets simplify matters for now]
- Elective courses want students enrolled.

ASSUME {
- A student wants at most one HS elective in a semester
- An HS course can accomodate at most 60 students in a class.



- What does an edge denote?
- What does a non-edge denote?
- How do we represent the solu^n?

GRAPHS IN PRACTICE : Learnings from the examples.

- Problems that affect us on a day to day basis can be modelled as graphs.

- While both problems deal with electives, the graphs are entirely different

  ex1 : directed acyclic graph

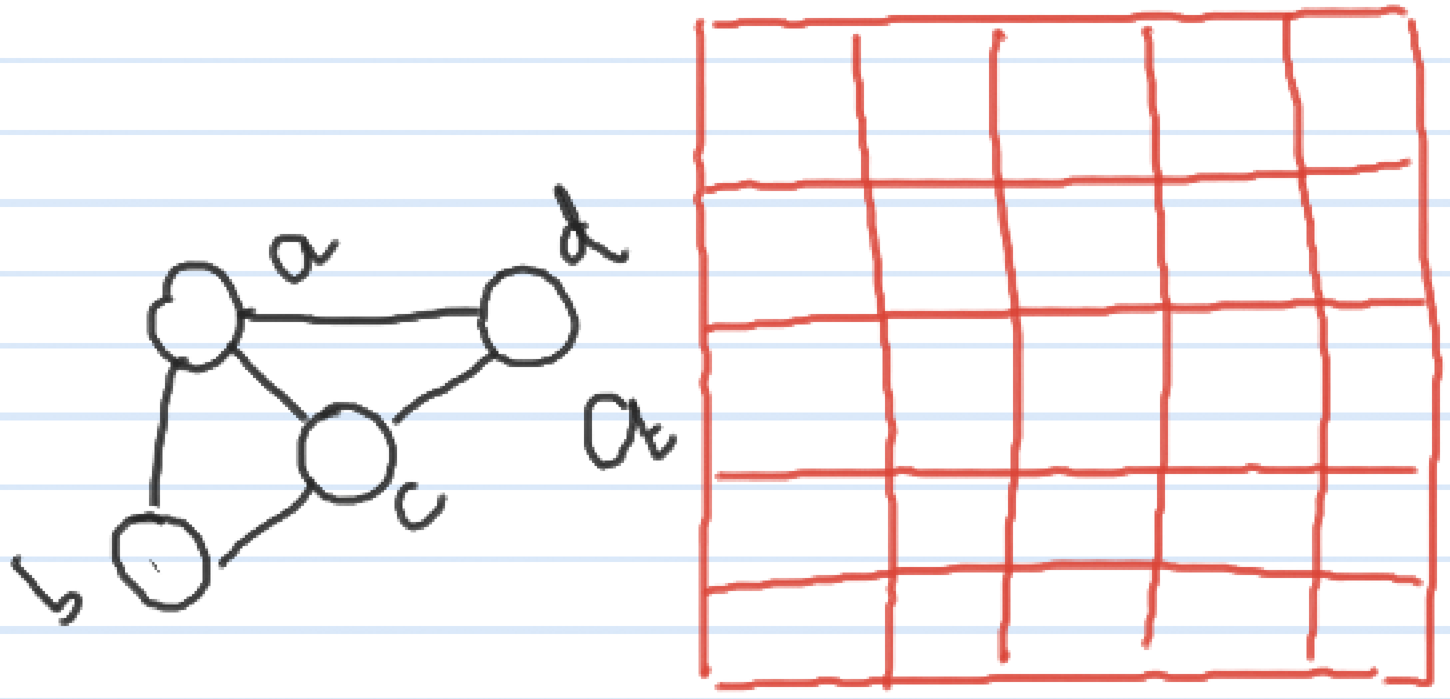
  ex2 : undirected bipartite graph.

- Modeling the real world problem graph problem is an interesting challenge.

GRAPH REPRESENTATIONS

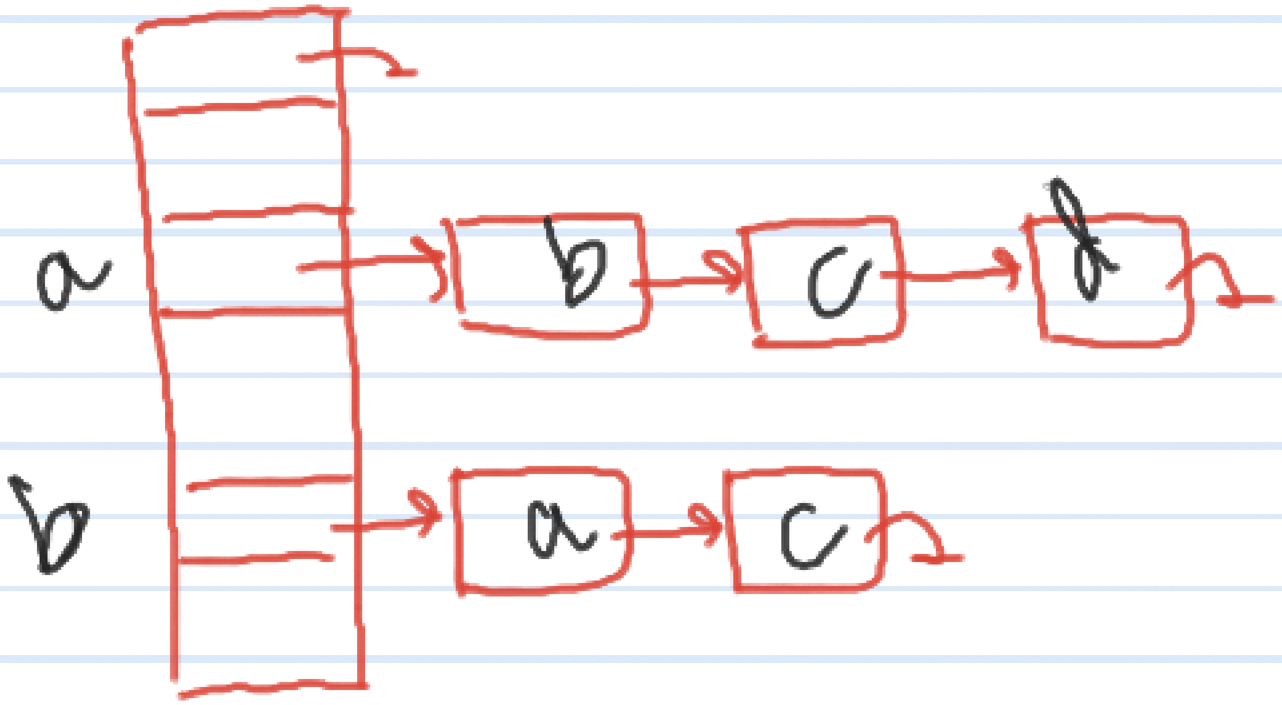
- Adjacency matrix

$G = (V, E)$



- size of matrix?
- entry(i, j)?
- properties of matrix
- what abt graphs with weights?

- Adjacency list

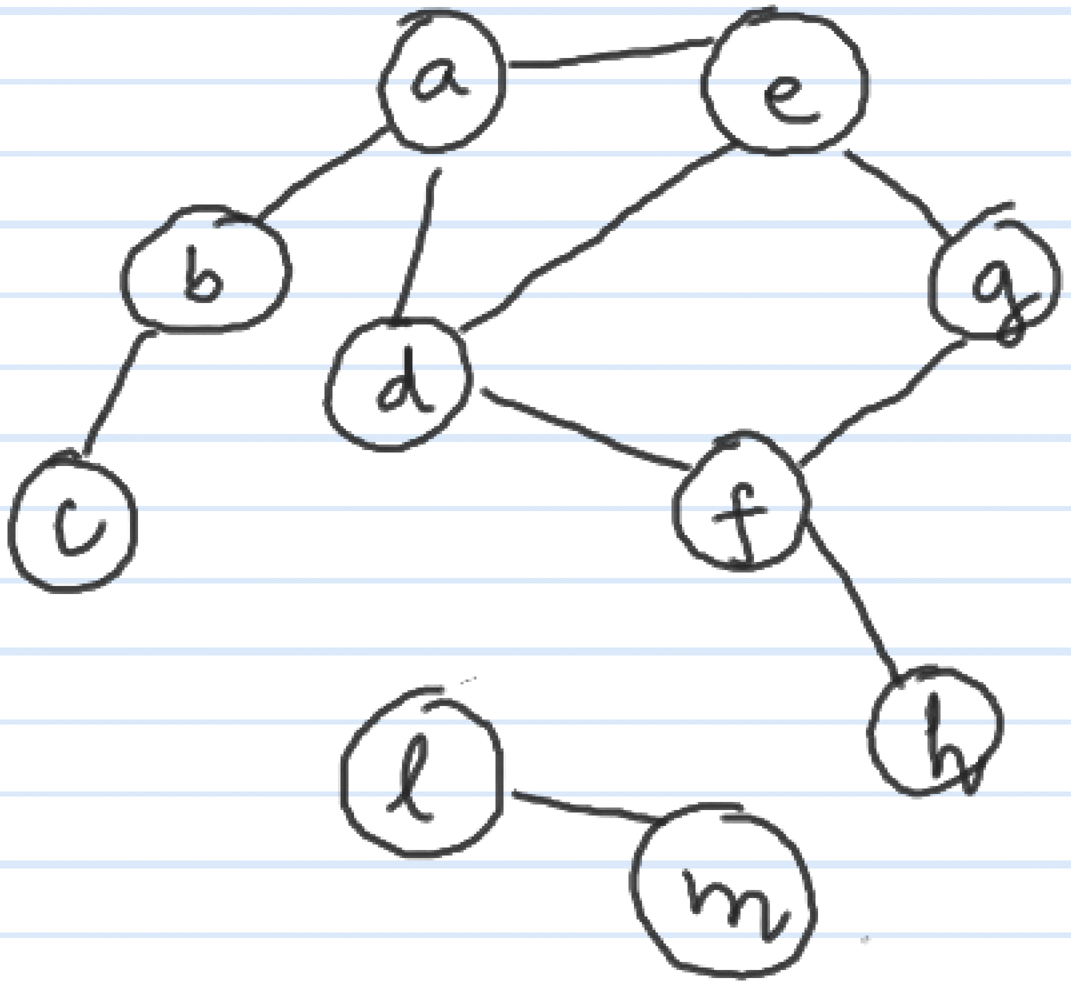


- length of each list?
- how do you find if i is adjacent to j?

SOME BASIC QUESTIONS ON GRAPHS.

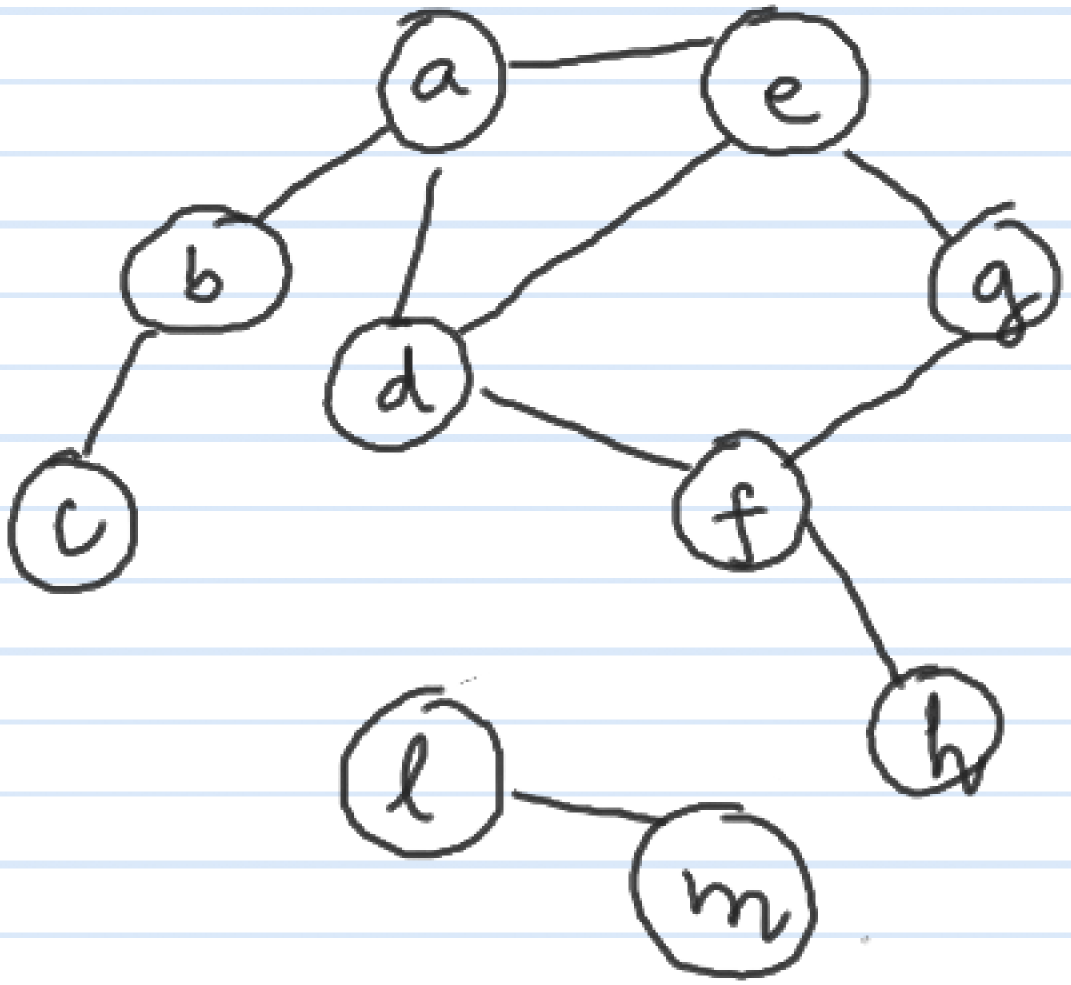
- is G connected? (undirected)
- is there a path from u to v? (directed)
- is G a tree?
- is G a bipartite graph?
- is G an acyclic directed graph? (DAG)
- how many Δs does G contain!

GRAPH TRAVERSALS ; BREADTH OR DEPTH FIRST?



- How does one "explore" a graph?
- where do you start?
- At vertex "a" do you go to b or d or e?
- After visiting b, do you go "deeper" or do you go "broader"?

GRAPH TRAVERSALS : BREADTH FIRST.

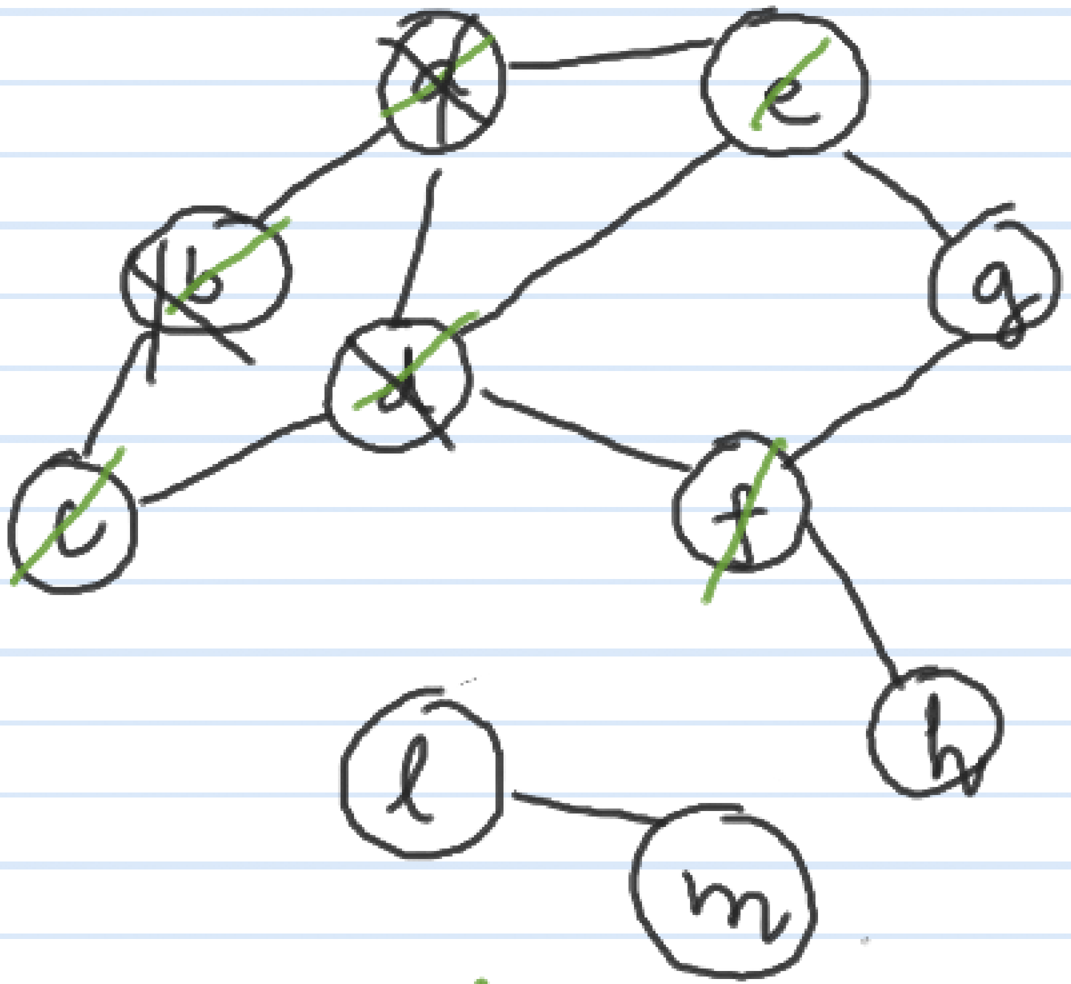


- Also called level wise traversal.
- Computes a BFS tree starting at a source node.
- Uses a Queue to keep track of the vertices

- Can be used to compute shortest paths in special graphs
- Can be used to detect bipartiteness

GRAPH TRAVERSALS : BREADTH FIRST.

~~a~~	~~b~~	~~d~~	e	c	f



let source (starting point) be "a"

```
- s = a
- all vertices are marked white
    clr(s) = gray
- add s to empty Q.
- while Q not empty do
        u = Q.dequeue();
        for every v ∈ Nbr(u) do
              if (clr(v) == white) {
                  ** add v to Queue Q
                     clr(v) = gray;
              }
        clr(u) = black }
```

Uses 3 colors

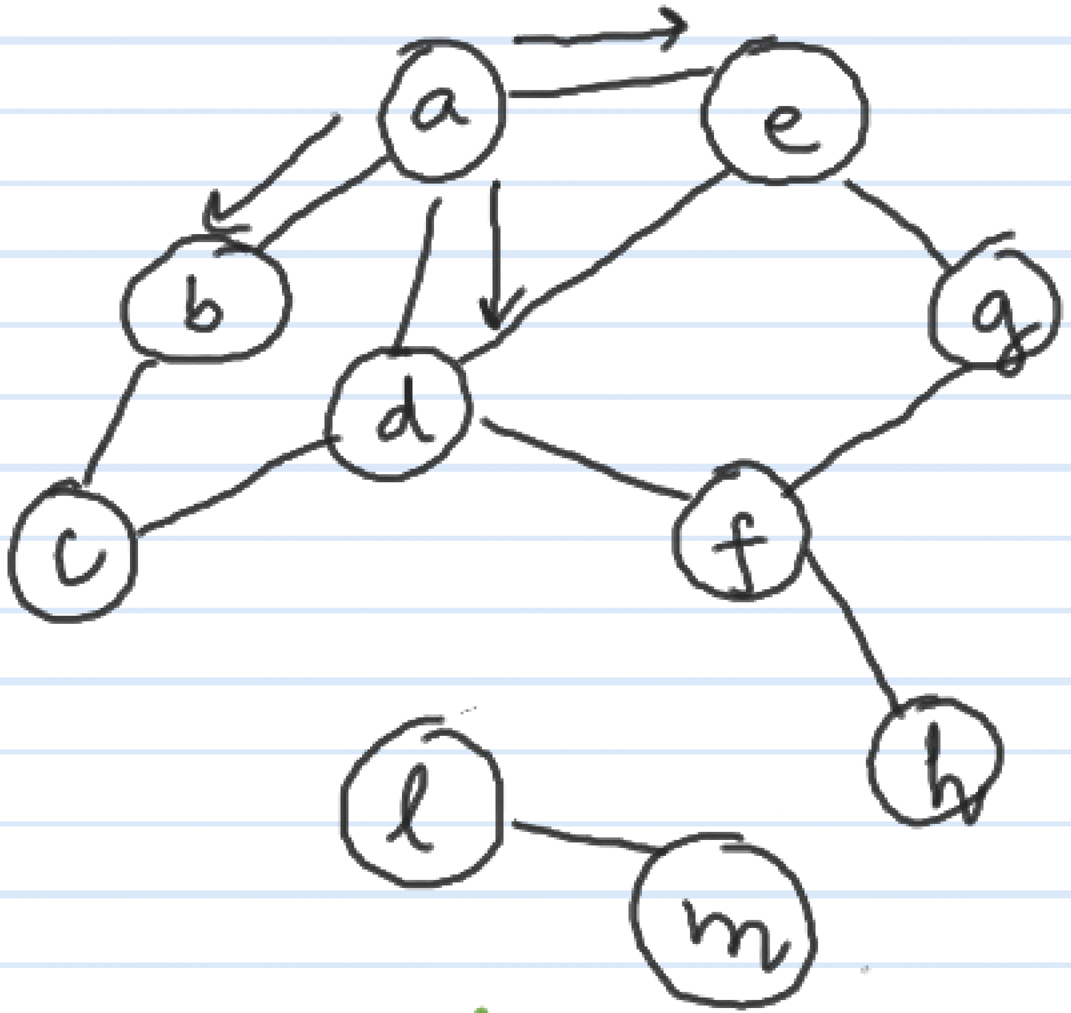
white, black, gray

white ↓ unexplored

black ↓ finished

gray ↳ explored (not finished)

GRAPH TRAVERSALS : BREADTH FIRST.



Uses 3 colors

white, black, gray

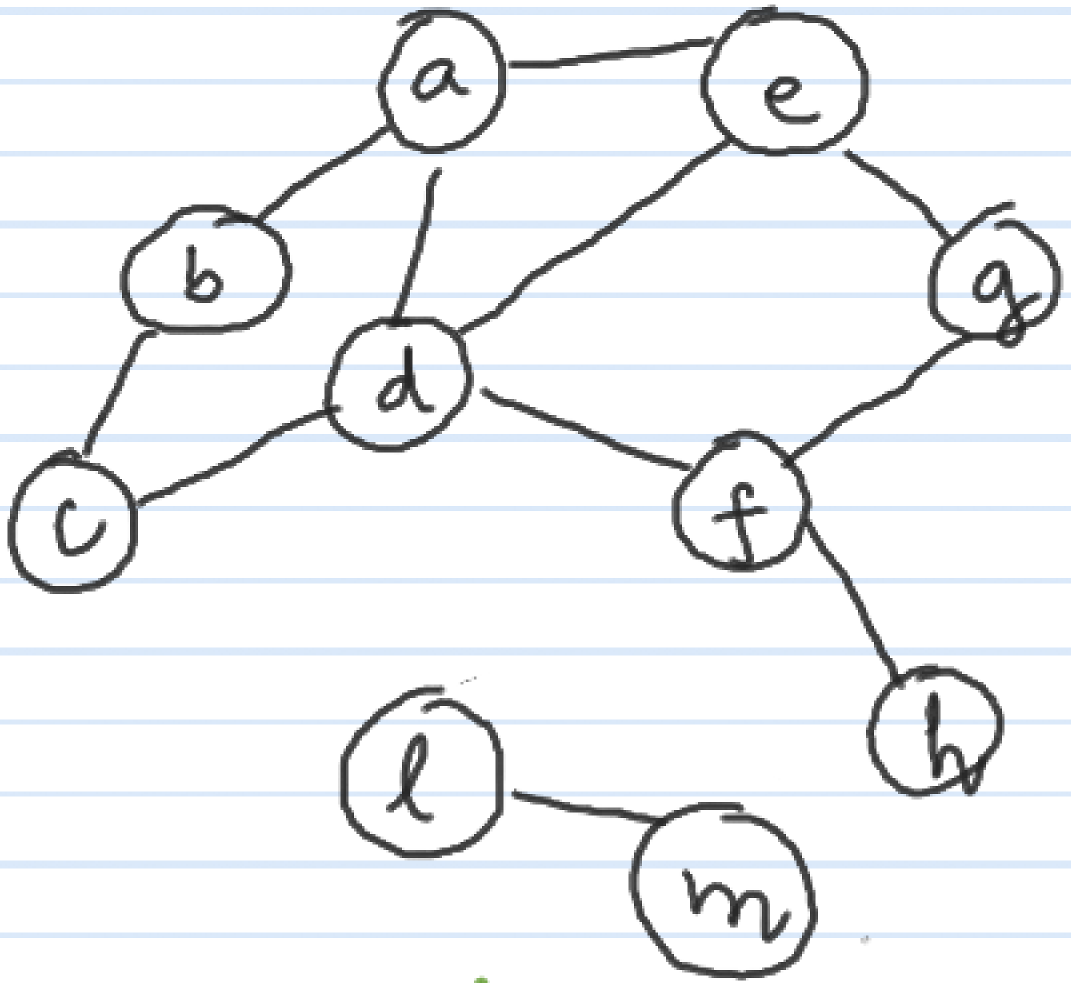
unexplored finished explored (not finished)

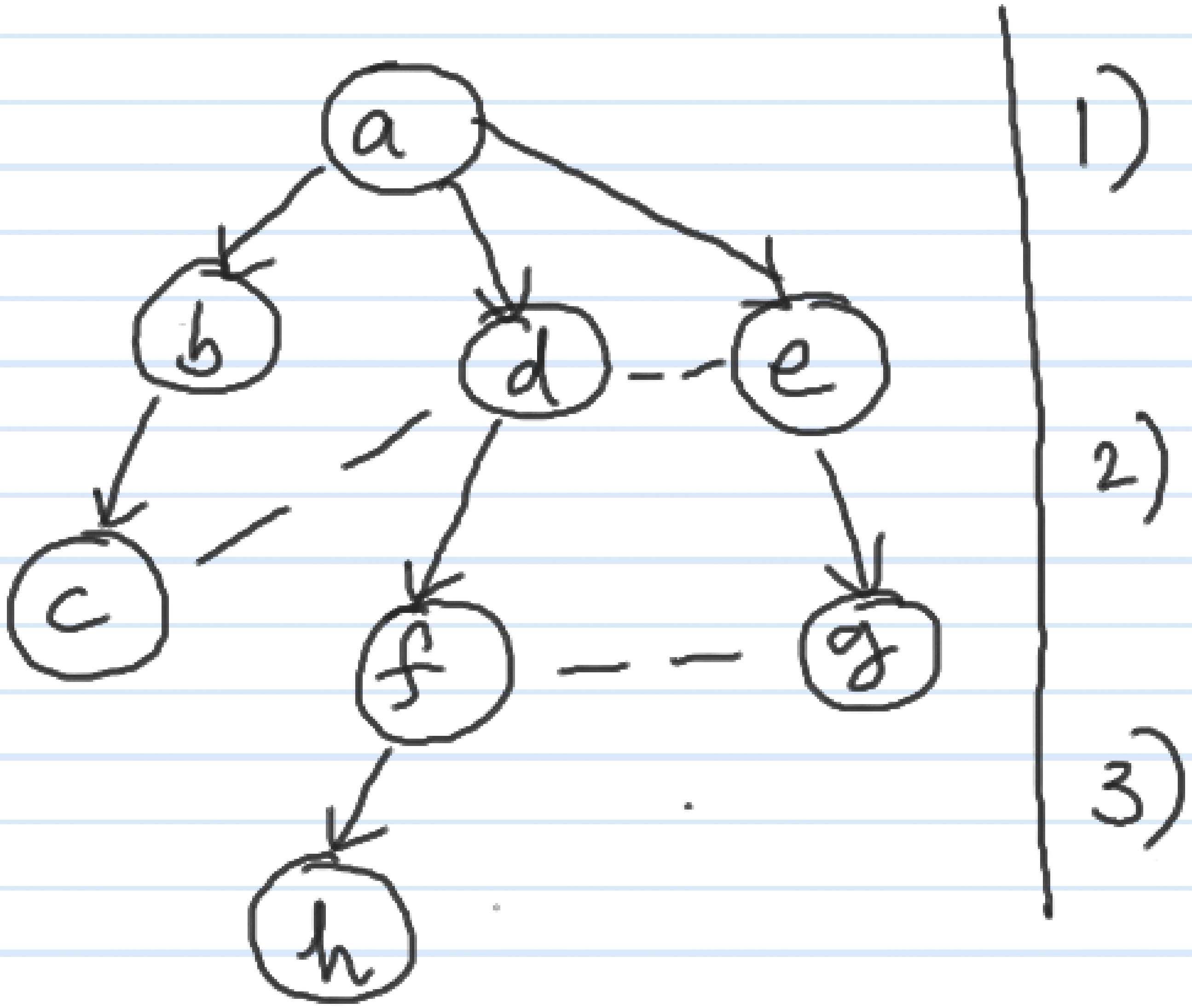
Let us execute the BFS algo on the graph starting at vertex a.

** add edge $(u \rightarrow v)$ to Tree T.

need to initialize an empty tree.

GRAPH TRAVERSALS : BREADTH FIRST.





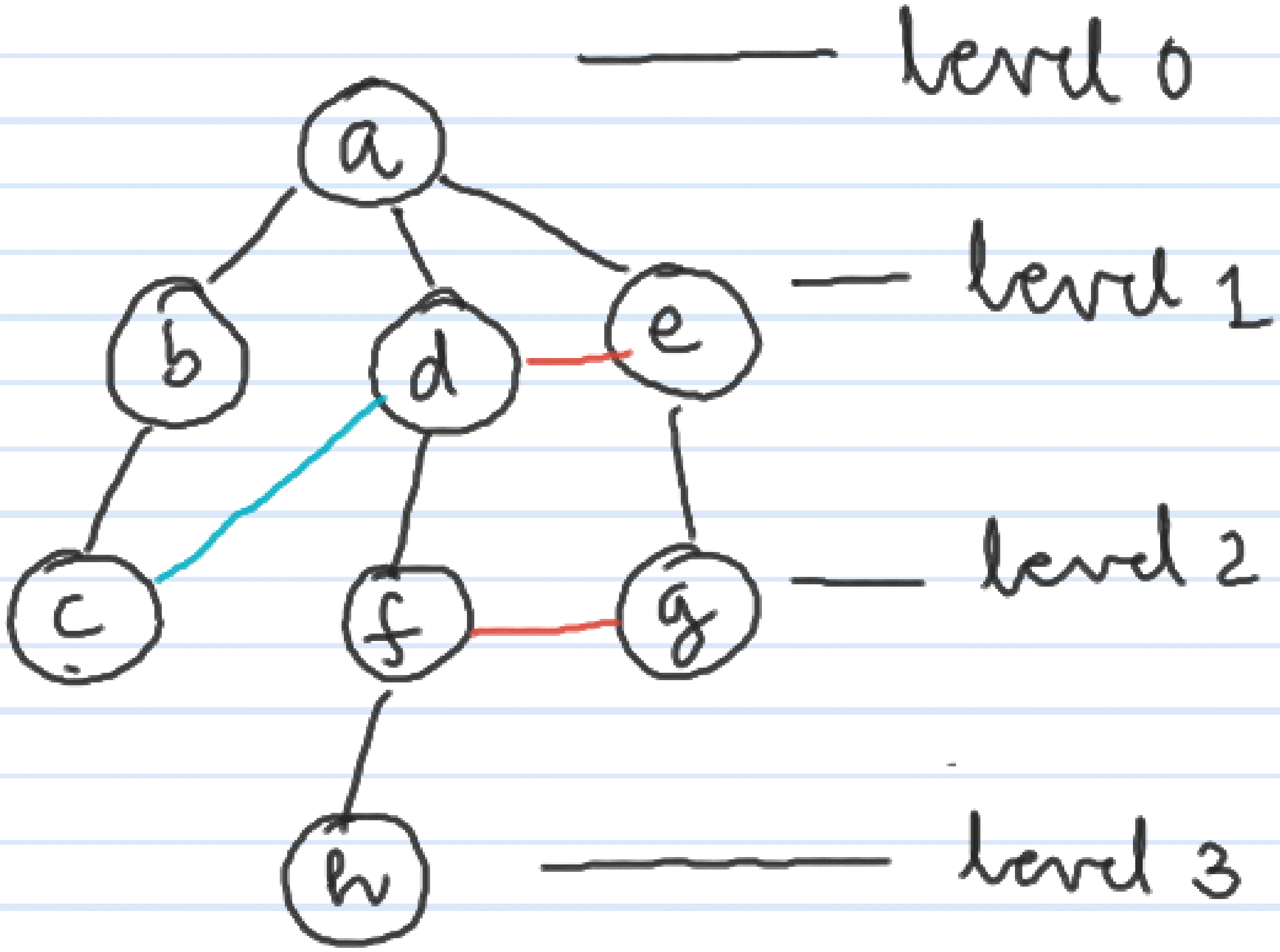
Uses 3 colors

white, black, gray

unexplored finished explored (not finished)

What are the properties of non tree edges of G?

GRAPH TRAVERSALS : BREADTH FIRST SEARCH TREE (BFS Tree)



- Running time of BFS

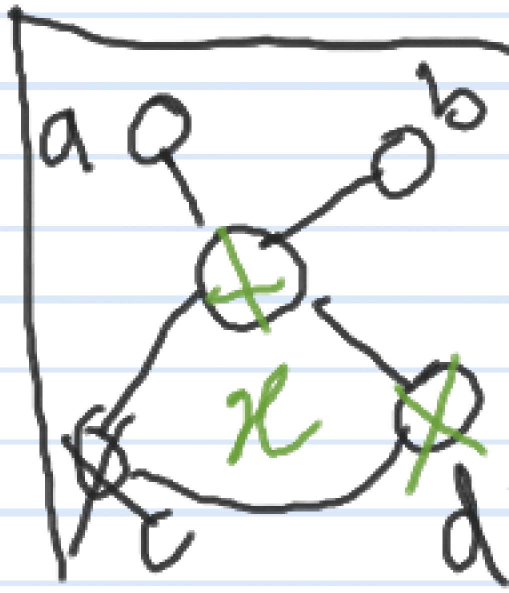
- is BFS tree unique?

- what do level numbers indicate?

- what do red edges imply?

- what does absence of any red edge indicate?

- is the graph connected?



BREADTH FIRST SEARCH

- Testing bipartiteness :

  A connected graph G is bipartite iff BFS starting at any vertex does not discover any red edge.