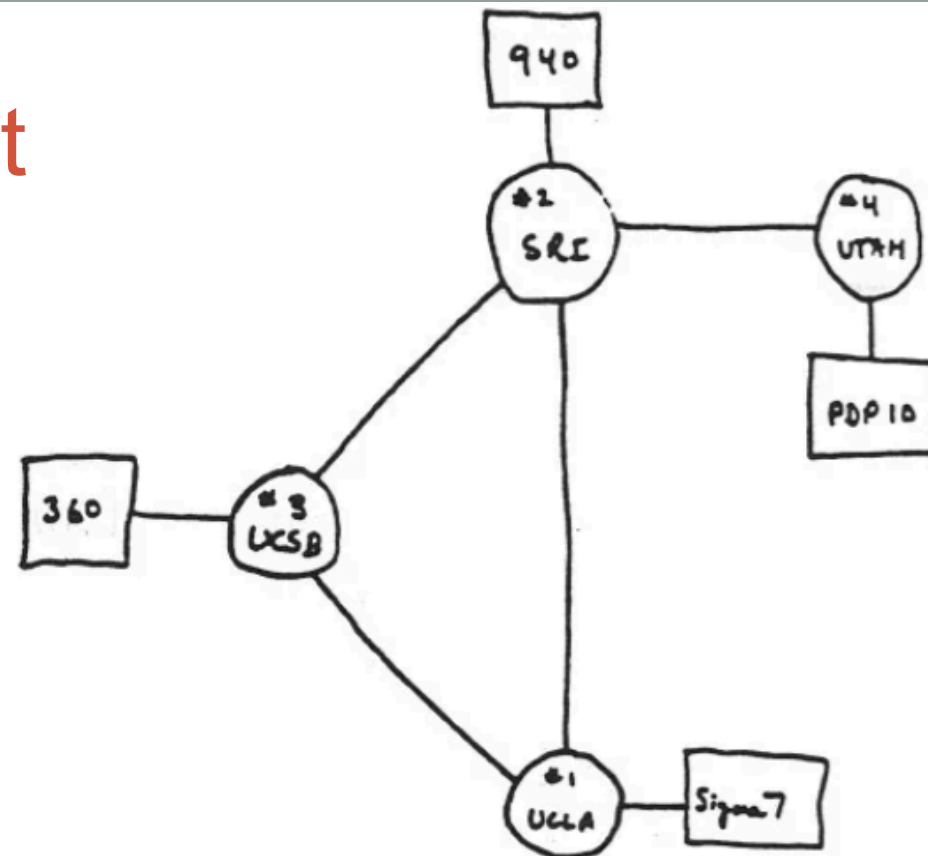


GRAPHS

,

The first four nodes of the internet



THE ARPA NETWORK

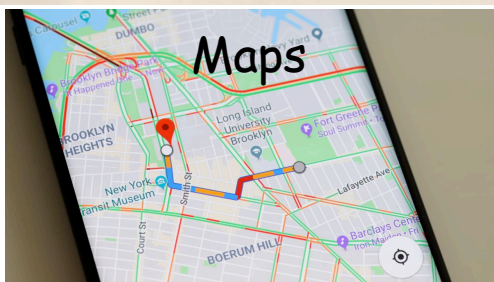
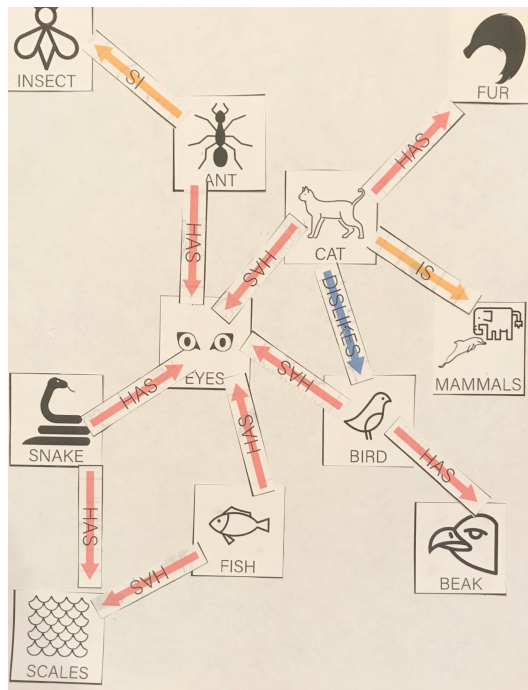
DEC 1969

4 NODES

The IBM 360, the IMP, and the workstations were all located in North Hall.
<https://jeweledplatypus.org/news/text/ucsbnet.html>

Graphs: applications

Semantic networks



Biological networks

How to Read the Circle of Life

Primordial life begins at the center and branches out in all directions, leading to the groups of species that exist today (colored rings)

Outer ring: Estimated proportion of all species*

Inner ring: Proportion of the groups named to date

Each black line represents at least 500 descendant species

Dark lines: Many species have been genetically sequenced

Light lines: Few species have been genetically sequenced

Nematodes (roundworms)

Lophotrochozoa (mollusks, segmented worms, brachiopods)

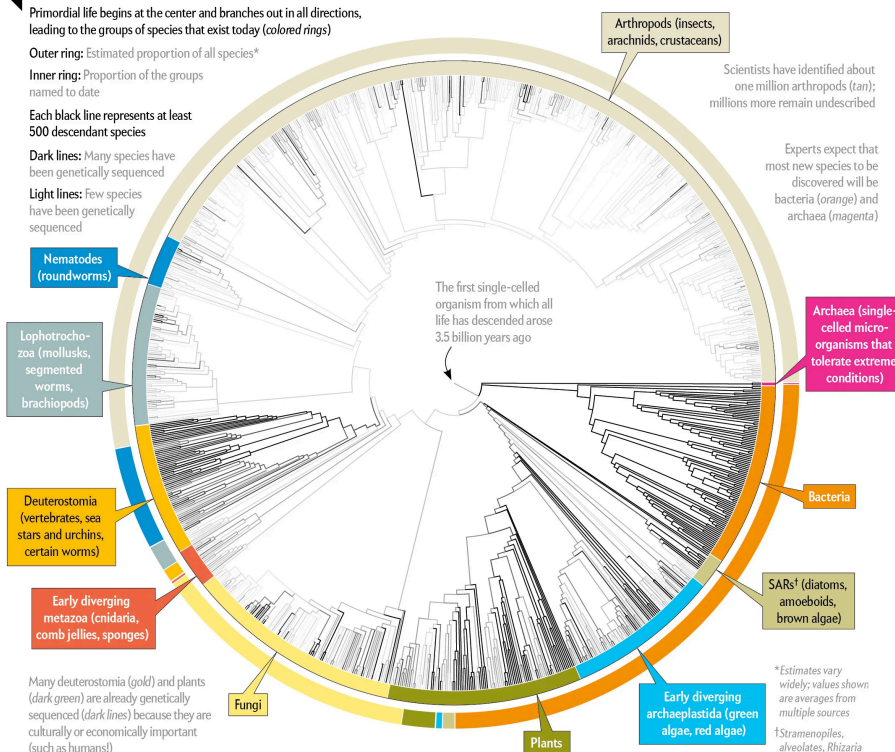
Deuterostomia (vertebrates, sea stars and urchins, certain worms)

Early diverging metazoa (cnidaria, comb jellies, sponges)

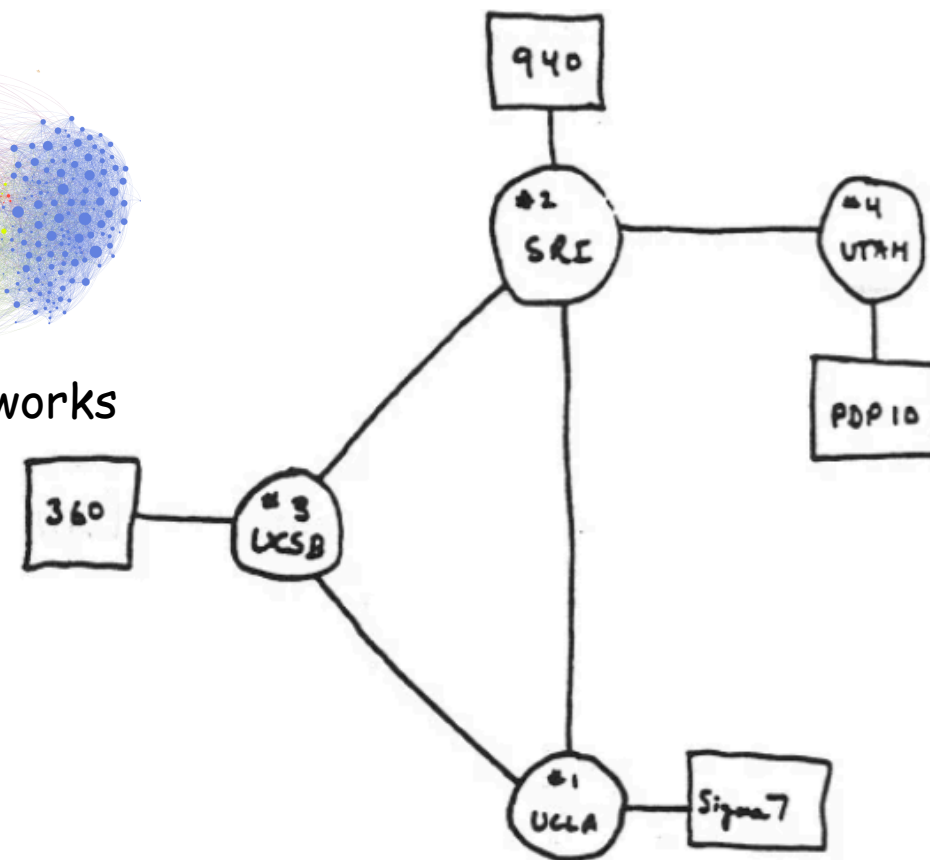
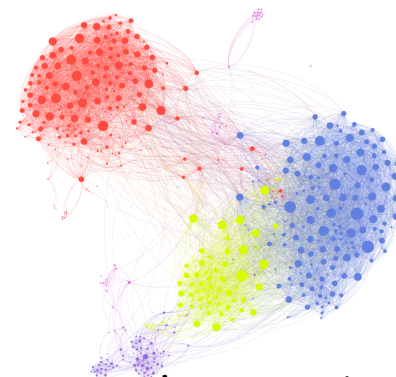
Fungi

Plants

Many deuterostomia (gold) and plants (dark green) are already genetically sequenced (dark lines) because they are culturally or economically important (such as humans!)



Social networks



THE ARPA NETWORK

DEC 1969

4 NODES

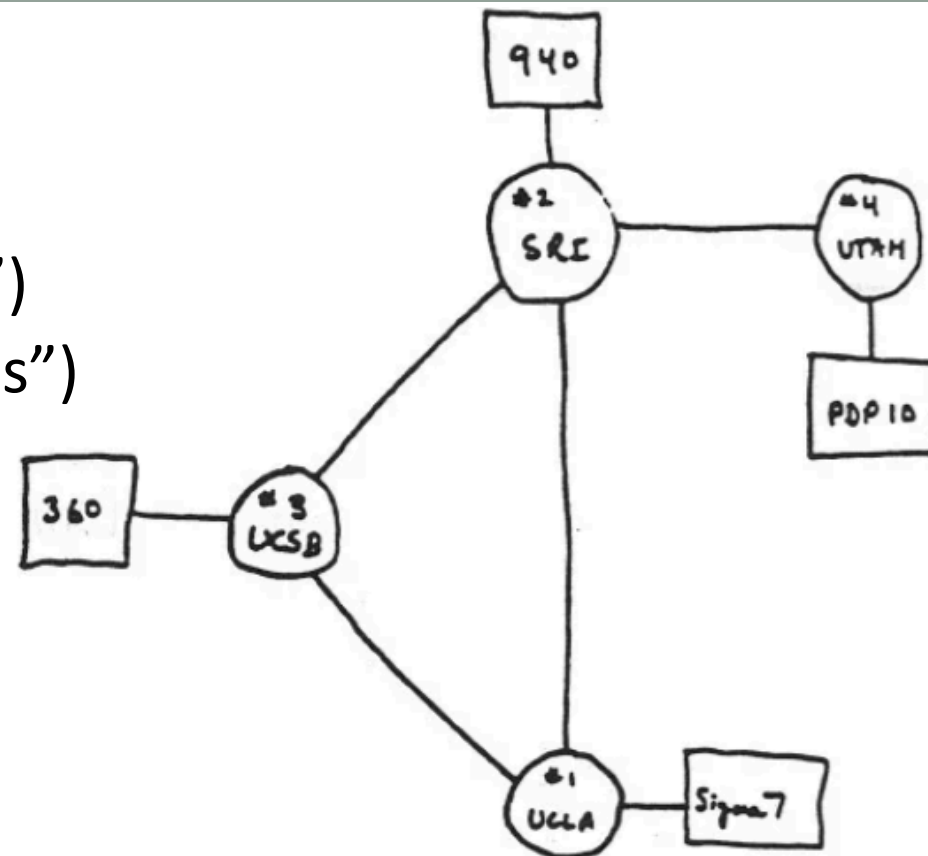
Graphs: terminology

A collection of elements (“nodes” or “vertices”)

A set of connections (“edges” or “links” or “arcs”) between pairs of nodes.

Edges may be directed or undirected

Edges may have weight associated with them



THE ARPA NETWORK

DEC 1969

4 NODES

Representing graphs

| | | | |
|--|--|--|--|
| | | | |
| | | | |
| | | | |
| | | | |

Adjacency Matrix

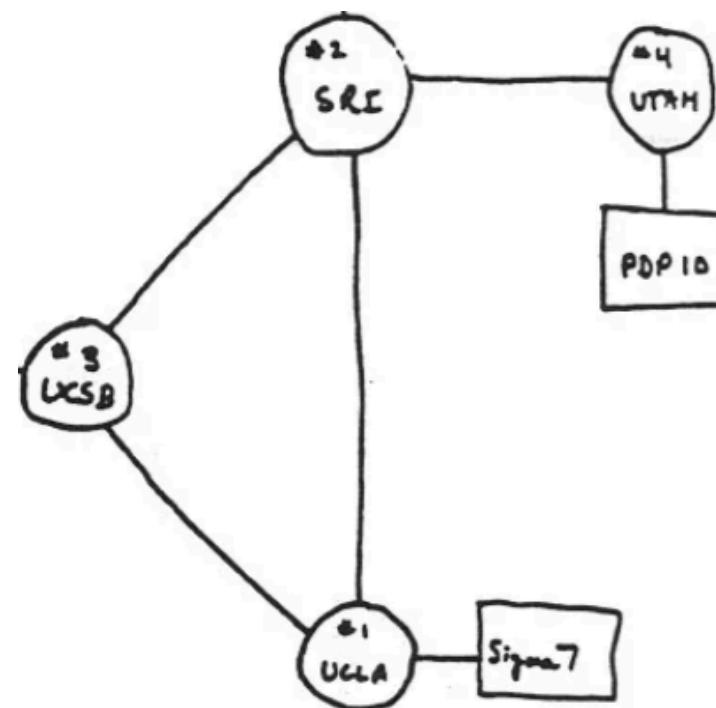
0 : 1, 2

1 : 0, 2, 3

2 : 0, 1

3 : 1

Adjacency List



THE ARPA NETWORK

DEC 1969

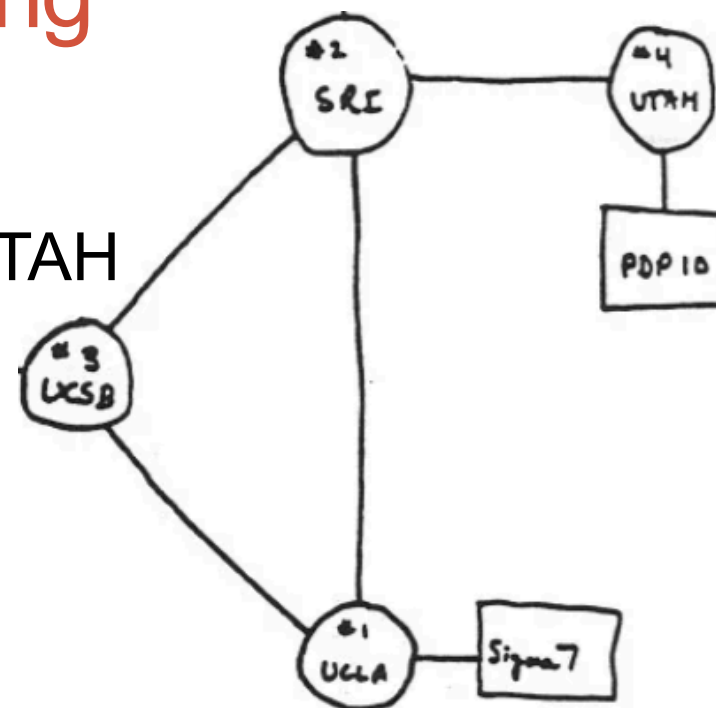
4 NODES

Assume each node is identified by a string

```
class graph{
    private:
        _____ adjlist;
};
```

UCLA : SRI, UCSB
 SRI : UCLA, UCSB, UTAH
 UCSB : UCLA, SRI
 UTAH : SRI

Adjacency List: adjlist



Choose the type for adjlist

- A. `vector<string>`
- B. `vector<list<string>>`
- C. `set<pair<string, list<string>>`
- D. `map<string, list<string>>`
- E. `priority_queue<string>`

THE ARPA NETWORK

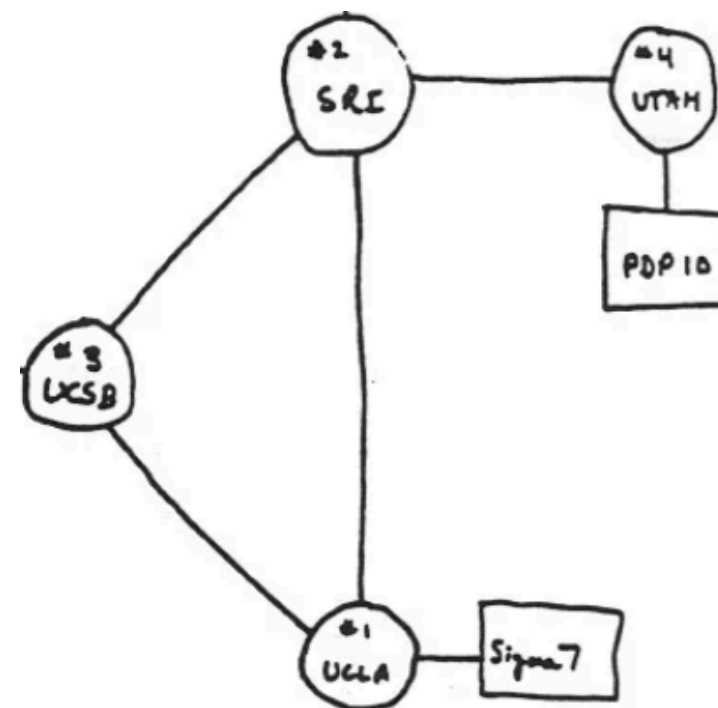
DEC 1969

4 NODES

Graph search: general approach

Starting with a source node

- find everything that can be explored
- don't explore anything twice



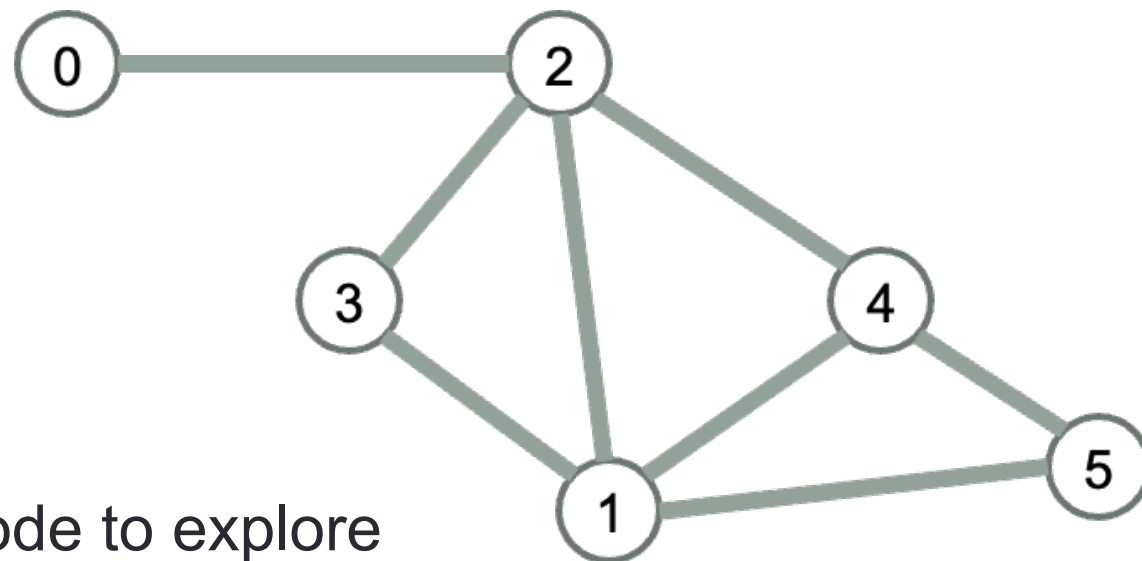
THE ARPA NETWORK

DEC 1969

4 NODES

Graph search: breadth first (BFS)

Explore all the nodes reachable from a given node before moving on to the next node to explore



Assume BFS chooses the lower number node to explore first, in what order does BFS visit the nodes in this graph

- A. 0, 1, 2, 3, 4, 5
- B. 0, 1, 3, 2, 4, 5
- C. 0, 2, 3, 1, 4, 5
- D. 0, 2, 1, 3, 4, 5
- E. Something else

BFS Traverse: Sketch of Algorithm

Start at source s ;

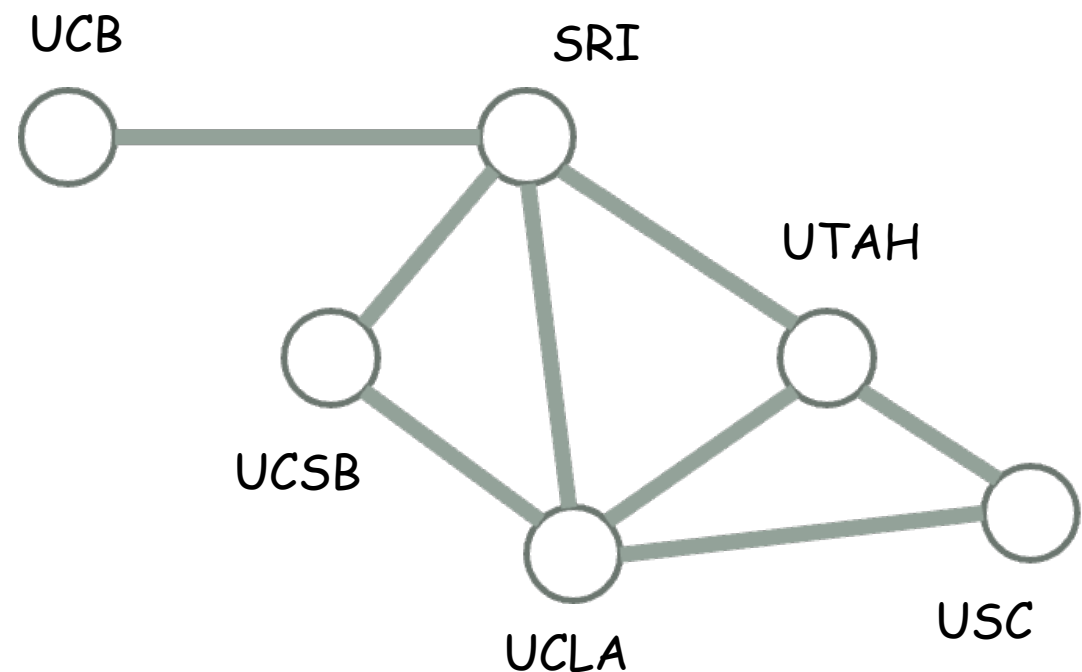
push s into a queue

while the queue is not empty:

pop the vertex u from the front of the queue

for each of u 's adjacent nodes that has not yet been visited (v):

- Push v in the queue



Questions:

- How can you tell if a node has been visited yet?
- What data do you need to keep track of for each node?

BFS Traverse: Sketch of Algorithm

Start at source s ; give s distance = 0

Mark s as visited

push s into a queue

while the queue is not empty:

pop the vertex u from the front of the queue

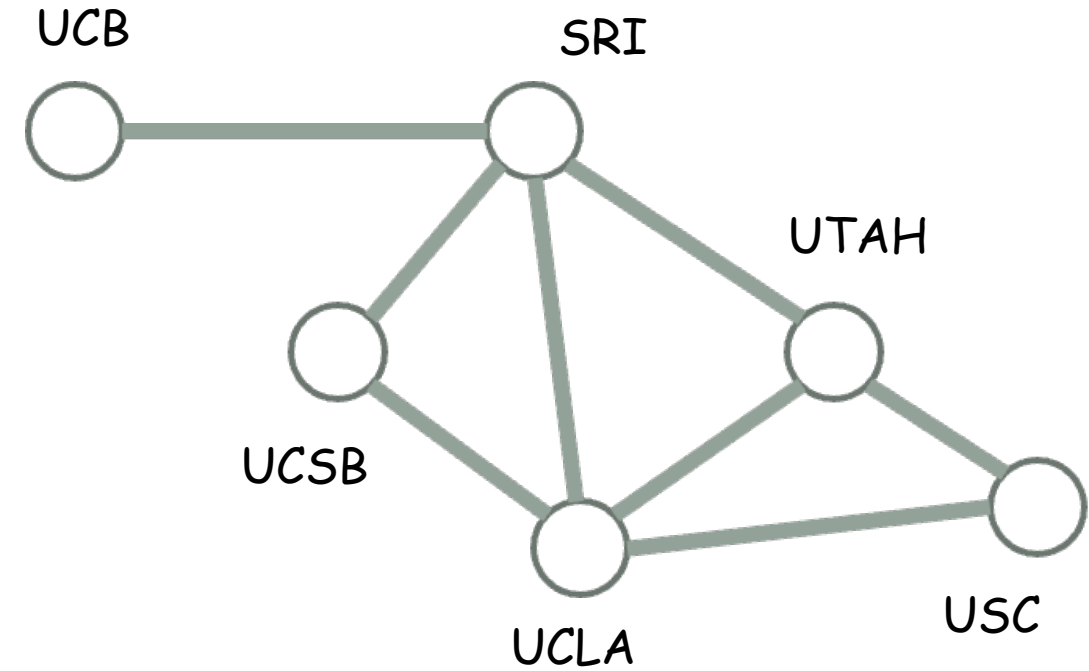
for each of u 's adjacent nodes that has not yet been visited (v):

- Mark v as visited
- Mark its distance as $1 +$ the distance to u
- Push v in the queue

Question (discuss 1 min):

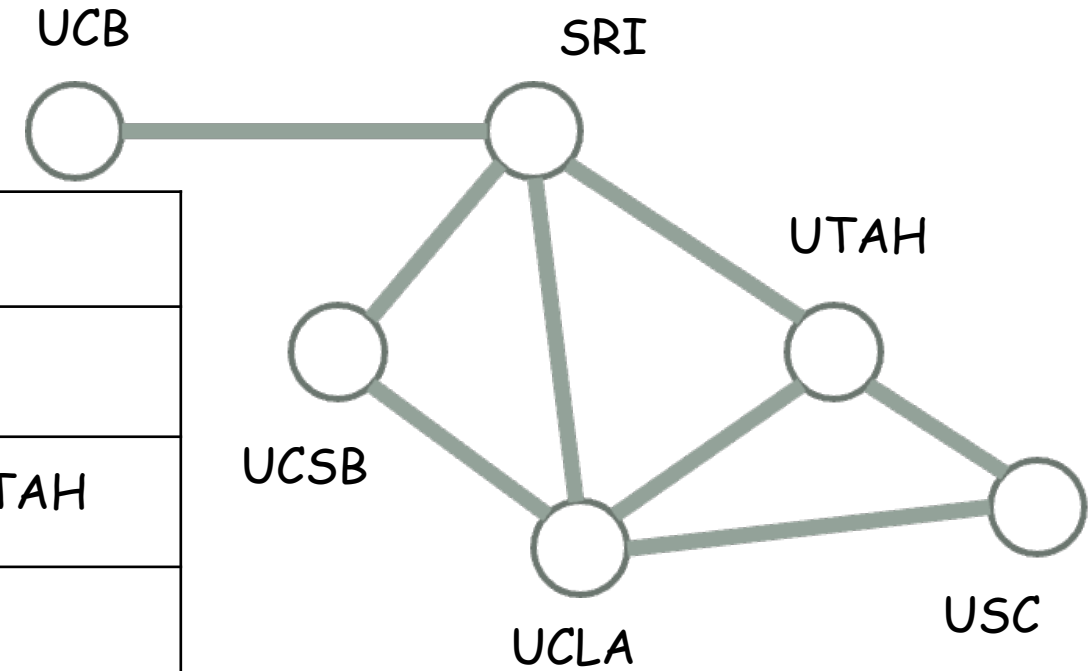
This algorithm finds the length of the shortest path from a source node to all nodes.

How can you also find the path itself?



BFS Traverse: Trace Algorithm

| Node | dist | prev | adjlist |
|------|------|------|-----------------------|
| UCB | | | SRI |
| SRI | | | UCB, UCSB, UCLA, UTAH |
| UCSB | | | SRI, UCLA |
| UCLA | | | UCSB, SRI, UTAH, USC |
| UTAH | | | UCLA, SRI, USC |
| USC | | | UTAH, UCLA |



GRAPHS

To model a graph and implement BFS we used all the data structures we have learned so far with the exception of `priority_queue` :)

