CSCI 104L Lecture 18: Search Algorithms

Breadth-First Search is a simple algorithm that finds the shortest path (in terms of number of edges) from a given node \( u \) to all other nodes. BFS explores the graph in layers.

- Layer 0 consists only of the node \( u \).
- Layer 1 consists of all nodes \( v \) with a direct edge \((u, v) \in E\).
- Layer \( k \) consists of all nodes \( w \) with a direct edge \((v, w) \in E\) from some node \( v \) in layer \( k-1 \). It does not revisit nodes from previous layers.

If we run BFS on the above graph, which nodes are in each level?

What ADT/Data Structure will be needed to implement this?

```c
//nodes are named 0 through n-1
int d[n]; //stores distances from u
int p[n]; //stores paths
void BFS(int u) { //u is the start node
    enqueue node u
    d[u] = 0;
    while the queue is not empty {
        dequeue the next node v
        for all outgoing edges \((v, w)\) from v {
            if we haven’t yet visited w {
                d[w] = d[v]+1;
                p[w] = v; //tells us which node led to w
                enqueue node w
            }
        }
    }
}
```

Question 1. : What is the runtime of BFS?
Figure 1: XKCD # 761. A breadth-first search makes a lot of sense for dating in general, actually; it suggests dating a bunch of people casually before getting serious, rather than having a series of five-year relationships one after the other.

**Question 2.** How would DFS explore the graph?

**Question 3.** What ADT/Data Structure will be needed to implement this?

**Question 4.** Does this find the shortest path?

**Question 5.** What is the runtime of DFS?