Graph Traversal – BFS

**Big Ideas: Utility of a BFS Traversal**

**Obs. 1:** Traversals can be used to count components.
**Obs. 2:** Traversals can be used to detect cycles.
**Obs. 3:** In BFS, $d$ provides the shortest distance to every vertex.
**Obs. 4:** In BFS, the endpoints of a cross edge never differ in distance, $d$, by more than 1: $|d(u) - d(v)| = 1$

DFS Graph Traversal

**Idea:** Traverse deep into the graph quickly, visiting more distant nodes before neighbors.

Two types of edges:

- 
- 

Minimum Spanning Tree

“The Muddy City” by CS Unplugged, Creative Commons BY-NC-SA 4.0
A **Spanning Tree** on a connected graph $G$ is a subgraph, $G'$, such that:
1. Every vertex is in $G'$ and
2. $G'$ is connected with the minimum number of edges

This construction will always create a new graph that is a ________ (connected, acyclic graph) that spans $G$.

---

A **Minimum Spanning Tree** is a spanning tree with the **minimal total edge weights** among all spanning trees.

- Every edge must have a weight
  - The weights are unconstrained, except they must be additive (eg: can be negative, can be non-integers)
- Output of a MST algorithm produces $G'$:
  - $G'$ is a spanning graph of $G$
  - $G'$ is a tree

$G'$ has a minimal total weight among all spanning trees. There may be multiple minimum spanning trees, but they will have the same total weight.

---

**Pseudocode for Kruskal’s MST Algorithm**

```python
KruskalMST(G):
    DisjointSets forest
    foreach (Vertex v : G):
        forest.makeSet(v)
    PriorityQueue Q    // min edge weight
    foreach (Edge e : G):
        Q.insert(e)
    Graph T = (V, {})
    while |T.edges()| < n-1:
        Vertex (u, v) = Q.removeMin()
        if forest.find(u) == forest.find(v):
            T.addEdge(u, v)
            forest.union( forest.find(u), forest.find(v) )
    return T
```

---

**CS 225 – Things To Be Doing:**

1. **Programming Exam C** is different than usual schedule:
   - Exam: Sunday, Dec 2 – Tuesday, Dec 4
2. **MP7 Released** – Slightly different structure:
   - Hard Deadline on Monday, Dec. 3 for Part 1
3. **lab_ml** this week in lab
4. Daily POTDs are ongoing for +1 point /problem