Design and Analysis of Algorithms

Graphs 3: Depth-First Search

Daniel Shannon

April 27th, 2022

4.2.2

> Does the BFS tree always give you the shortest path from the starting node to every other node? How can we modify the BFS pseudocode to keep track of what the shortest paths actually are? Prove or give a counterexample.

Claim: The BFS tree always will give the shortest path tree.

Proof: *by induction...* Say we have node u on the queue Q with a distance of $k = 0$ from the origin, u. All connected nodes v have a distance of $k = 1$ from u. *Induction:* When we push the neighbors of v, v', onto Q, v' is $k + 1$ distance from u. So each additional node is the shortest path from $u[k]$ to $u[k + 1]$.

We can modify the pseudo code by adding a distance property to each node as we push it to the queue.

4.2.4

> How can we modify the code to keep track of what the edges on the shortest path are?

```
for all u in V:
  dist(u)=infin
  //HERE we can initialize a vector of edges,
  path(u)=[]

dist(s)=0
Q=[s] (queue containing just s)
while Q is not empty:
  u = eject(Q)
  for all edges (u,v) in E: //put all neighbors on queue
    if dist(v) == infin:
      inject(Q,v)
      dist(v)=dist(u)+1 //distance to neighbor=1 + distance to
          current node
      //HERE we can populate child node v, with the edges to u,
      path[u]=path[u]+(u,v)
```

On line 4 we can initialize a vector of edges to null for each node On line 15 we can populate the child node with its parent node.

4.3.2

> What does this even mean? Any examples of real-world graph with negative edges?

Say we're hiking and rather than only examining distance between nodes, we include elevation. Perhaps we could say positive weights are downhill or flat and negative weights are uphill. Uphill actively works against you and downhill is a lot easier.