Algorithms: Applications of BFS

Suppose we have a graph $G = (V, E)$. A given graph could have few edges, or lots of edges, or anything in between. Let's think about the range of possible relationships between V and E.

1 How big can $|E|$ be, relative to $|V|$?

(a) The smallest possible value of $|E|$ is ________________________.

(b) $|E|$ is $O\Big(\qquad\qquad\Big)$ because ________________________________.

(c) When G is a tree, $|E|$ is $\Theta\Big(\qquad\qquad\Big)$ because ________________.

Now, recall from last class that we showed breadth-first search (BFS) can be implemented to run in $\Theta(|E|)$ time.

2 In terms of Θ, how fast does BFS run, as a function of $|V|$, when G is a tree?

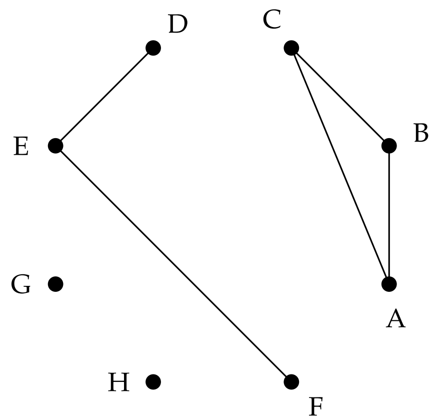
3 How fast does BFS run, as a function of $|V|$, when G is very dense, *i.e.* it contains some constant fraction (say, half) of all possible edges?

A first application of BFS

4 Describe an algorithm to find the connected components of a graph G.

Input: a graph $G = (V, E)$
Output: a set of sets of vertices, Set<Set<Vertex>>, where each set contains the vertices in some (maximal) connected component. That is, all the vertices within each set should be connected; no vertex should be connected to vertices in any other sets; and every vertex in V should be contained in exactly one of the sets.

For example, given the graph below, the algorithm should return $\{\{D, E, F\}, \{C, B, A\}, \{G\}, \{H\}\}$.



Describe your algorithm (using informal prose or pseudocode) and analyze its asymptotic running time.

A second application of BFS

Model 1: Directed graphs

See the board for examples of *directed* graphs.

5 What is the difference between directed graphs and the (undirected) graphs we saw on a previous activity?

6 The previous activity defined graphs as consisting of a set V of vertices and a set E of edges, where each edge is a set of two vertices. How would you modify this definition to allow for directed graphs?

7 For each of the following graph terms/concepts, say whether you think its definition needs to be modified for directed graphs; if so, say what the new definition should be.

1 *vertex*

2 *degree*

3 *path*

4 *cycle*

8 What (if anything) about our implementation of BFS needs to be modified for BFS to work sensibly on directed graphs?

Definition 1. A directed graph $G = (V, E)$ is *strongly connected* if for any two vertices $u, v \in V$ there is a (directed) path from u to v, *and also* from v to u.

9 Describe a brute force algorithm for determining whether a given directed graph G is strongly connected.

10 Analyze the running time of your algorithm in terms of Θ.

Let's see if we can do better!

Theorem 2. *A directed graph $G = (V, E)$ is strongly connected if and only if for any vertex $s \in V$, every other vertex in G is mutually reachable with s (that is, for each $v \in V$ there is a directed path from s to v and another directed path from v to s).*

11 In order to prove this "if and only if" statement, we must prove

both ____

and ____.

Hint: draw a picture!

Proof.

□

Definition 3. Given a directed graph G, its *reverse graph* G^{rev} is the graph with the same vertices and all edges reversed.

Theorem 4. *A directed graph $G = (V, E)$ is strongly connected if and only if given any $s \in V$, all vertices are reachable from s in G, and all vertices are reachable from s in G^{rev}.*

Hint: what is the relationship between this theorem and Theorem 2?

Proof.

□

12 Based on the above theorem, describe an algorithm to determine whether a given directed graph $G = (V, E)$ is strongly connected, and analyze its running time.