Is a Graph Connected?

Algorithm 1: Breadth-first search - From a starting node, find closest nodes first.

Start at a:

Visit all nodes at distance 1 from a:

Visit all nodes at distance 2 from a:
BFS Tree

- If we keep only the edges traversed while doing a breadth-first search, we will have a tree.

Edges to layer 1:

Plus edges to layer 2:

Done. Discard edges not traversed.

Breadth-First Search

- **Property.** Let $T$ be a BFS tree of $G = (V, E)$, and let $(x, y)$ be an edge of $G$. Then the layer of $x$ and $y$ differ by at most 1.

Layer 0: $\{a\}$
Layer 1: $\{b, c, d\}$
Layer 2: $\{e\}$

Proof?

Shortest Path

- When we use an edge to add a node in BFS, remember the other endpoint.
- To find the shortest path, start at the end and walk backwards.

$\{(a, \emptyset), (c, a), (d, a), (b, a), (e, c)\}$

new node  edge traversed

Shortest path from a to e: $a \rightarrow c \rightarrow e$
Is a Graph Connected?

Algorithm 2: Depth-first search - When adding a node to the visited set, recursively add nodes from that node.

---

Is a Graph Connected?

Algorithm 2: Depth-first search - When adding a node to the visited set, recursively add nodes from that node.

---

Is a Graph Connected?

Algorithm 2: Depth-first search - When adding a node to the visited set, recursively add nodes from that node.
Is a Graph Connected?

Algorithm 2: Depth-first search - When adding a node to the visited set, recursively add nodes from that node.

Depth First Search

Theorem: Let $T$ be a depth-first search tree. Let $x$ and $y$ be 2 nodes in the tree. Let $(x, y)$ be an edge that is in $G$ but not in $T$. Then either $x$ is an ancestor of $y$ or $y$ is an ancestor of $x$ in the DFS tree.

Proof?