COMPSCI 250: Introduction to Computation

Lecture #22: Graphs, Paths, and Trees

David Mix Barrington and Ghazaleh Parvini

25 October 2023

Graphs, Paths, and Trees

- Left Over: Speed of Euclidean Algorithm
- Graph Definitions
- Paths and the Path Predicate
- Cycles, Directed and Undirected
- Forests and Trees
- The Unique Simple Path Theorem
- Rooted Trees
- A Theorem About Trees

Speed of Euclidean Algorithm

- Consecutive Fibonacci numbers take a relatively long time, e.g., 233, 144, 89, 55, 34, 21, 13, 8, 5, 3, 2, and 1. But actually F(n) is about $(1.61)^n$, so that if x is a Fibonacci number we take about $\log_{1.61} x$ steps.
- Here we'll show that if the two initial numbers are each at most 2^n, the EA will terminate in at most $2n + 1$ steps. The base case of our induction says that if both numbers are at most $2^0 = 1$, we need $2(0) + 1 = 1$ step.

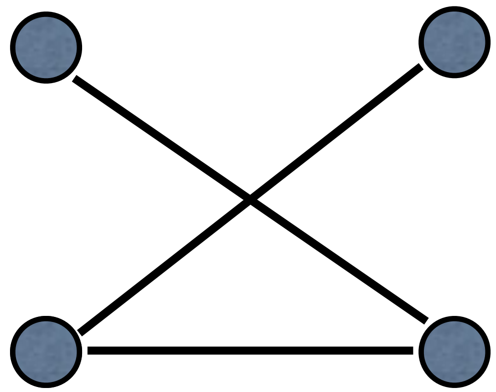
Speed of Euclidean Algorithm

- The inductive step uses the contrapositive method.
- We start with a and b, and compute $a = qb + c$ and $b = rc + d$, so $a = (qr + 1)c + qd$. If c or d is *greater than* 2^n, then a is greater than 2^{n+1}.
- So if $a \leq 2^{n+1}$, then $c \leq 2^n$. By the IH we need at most $2n + 1$ steps starting with c and d, so we need at most $2n + 3$ total steps.

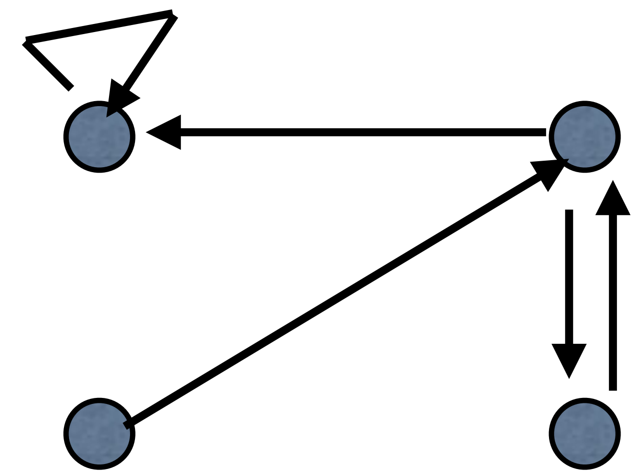
Graph Definitions

- A **graph** is a set of points called **nodes** or **vertices**, together with a set of **edges**.
- In an **undirected graph** each edge connects two different nodes.
- In a **directed graph** each edge (or **arc**) goes from some node to some node, possibly the same one.

Graph Pictures



Undirected Graph



Directed Graph

Directed graphs may have self-loops,
undirected graphs may not.

The Edge Predicate

- Two graphs are considered to be equal if their **edge predicates** are the same.
- The edge predicate E(x, y) takes two nodes x and y as arguments, and is true if there is an edge from x to y (or between x and y, in the case of an undirected graph).
- There are also **multigraphs**, which are allowed to have more than one edge with the same starting point and ending point.
- Graphs can be **labelled** by assigning some information to each node or edge.

Paths and the Path Predicate

- A **path** in a graph is a sequence of *zero or more* edges, where the endpoint of each edge is the starting point of the next edge.
- We can have **undirected paths** in an undirected graph or **directed paths** in a directed graph.
- The **path predicate** P(x, y) is true if and only if there is a path from node x to node y. We define the path predicate and the set of paths recursively.

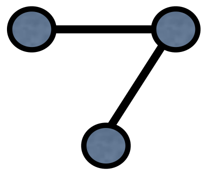
Clicker Question #1

- Which of the following statements *is true*?
- (a) The relation E is reflexive and transitive on all graphs.
- (b) If the relation P is not symmetric on some graph, that graph must be directed.
- (c) If the relation P is reflexive, the graph must have self-loops.
- (d) If the relation E is not transitive on some graph, that graph must be directed.

Not the Answer

Clicker Answer #1

- Which of the following statements *is true*?
- (a) The relation E is reflexive and transitive on all graphs. P has this property but not E
- (b) If the relation P is not symmetric on some graph, that graph must be directed. For undirected graphs, E and P are both symmetric
- (c) If the relation P is reflexive, the graph must have self-loops. P is reflexive by the trivial paths
- (d) If the relation E is not transitive on some graph, that graph must be directed. Counterexample



More About Paths

- For any node x, P(x, x) is true and the **empty path** λ is a path from x to x.
- If α is a path from x to y, and there is an edge from y to z, then P(x, z) is true and β is a path from x to z, where β consists of α followed by the edge (y, z).
- Thus if P(x, y) and E(y, z) are both true, then P(x, z) is true.

Transitivity of Paths

- It stands to reason that if there is a path α from node x to node y, and a path β from node y to node z, then there exists a path from node x to node z obtained by first taking α and then taking β.
- Proving this will take an induction on the second path β, using the recursive definition of paths.

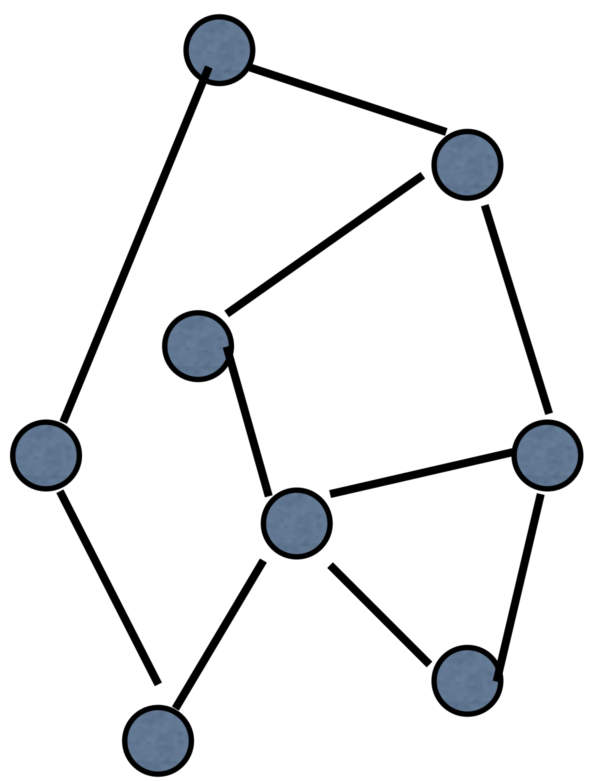
Proving Transitivity

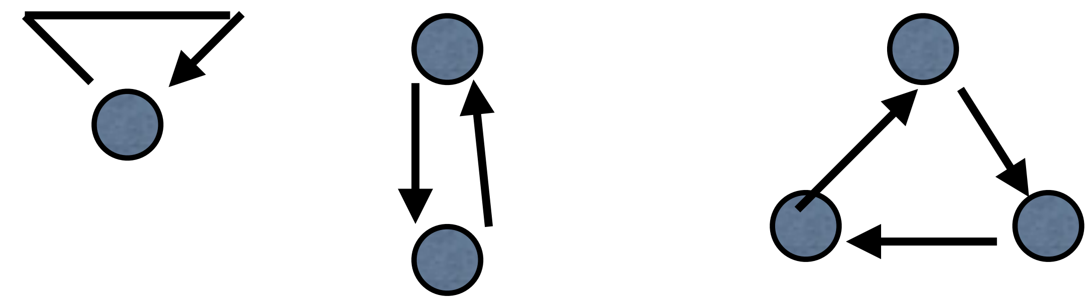
- The base case is when β is an empty path. In this case α, which is a path from x to y, is also the desired path from x to z because $y = z$.
- For the inductive case, assume that β is made by adding an edge (w, z) to some path γ from y to w, and that the IH applies to γ. So there exists a path from x to w made from α and γ. By the definition of paths, we can add the edge (w, z) to this path and get the desired path from x to z.

Cycles

- A **cycle** is a path from a node to itself that meets certain “non-triviality” conditions.
- In an undirected graph, a cycle is a **simple** nonempty path from a node to itself, which means a path that does not reuse a node or edge.
- An undirected cycle must have three or more edges.

Cycle Pictures





Undirected Cycles

(of length 3, 4, 5, 6, 7)

Directed Cycles

(of length 1, 2, 3)

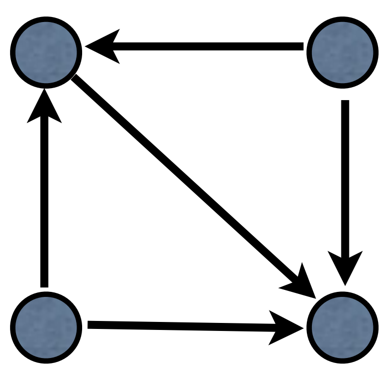
Cycle Vocabulary

- A **directed cycle** in a directed graph is any nonempty directed path from a node to itself.
- A graph is **acyclic** if it has no cycles.
- A **directed acyclic graph** or **DAG** is a directed graph with no directed cycles.
- Acyclic undirected graphs (with no undirected cycles) are called **forests**.

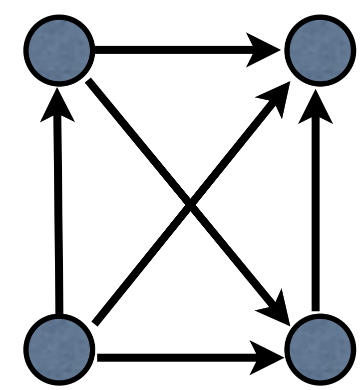
Clicker Question #2

- Which graph is *not* ayclic?

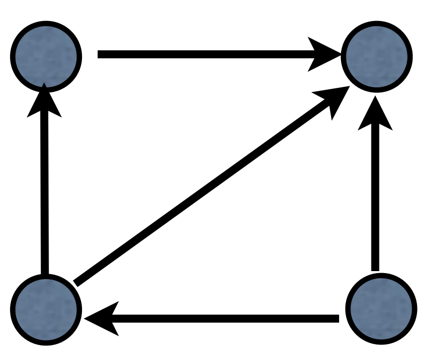
(a)



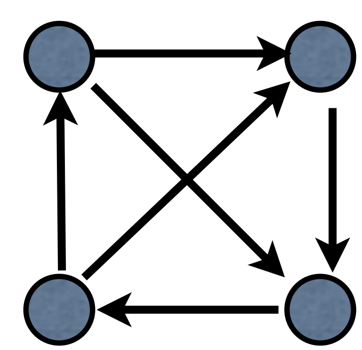
(b)



(c)



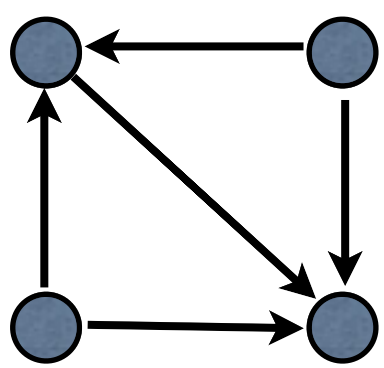
(d)

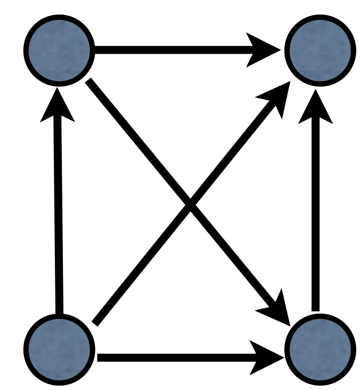


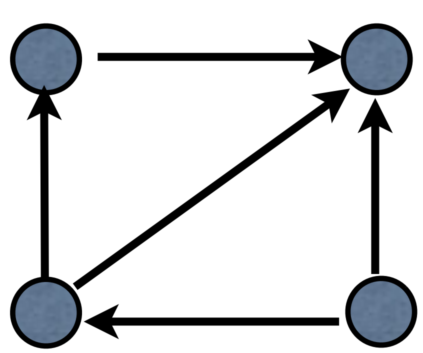
Not the Answer

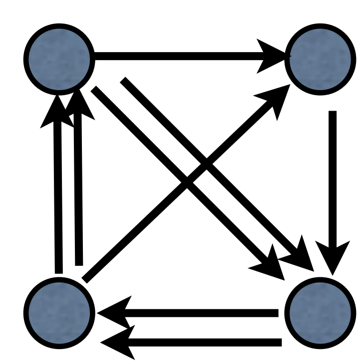
Clicker Answer #2

- Which graph is *not* ayclic?

(a) 

(b) 

(c) 

(d) 

Forests and Trees

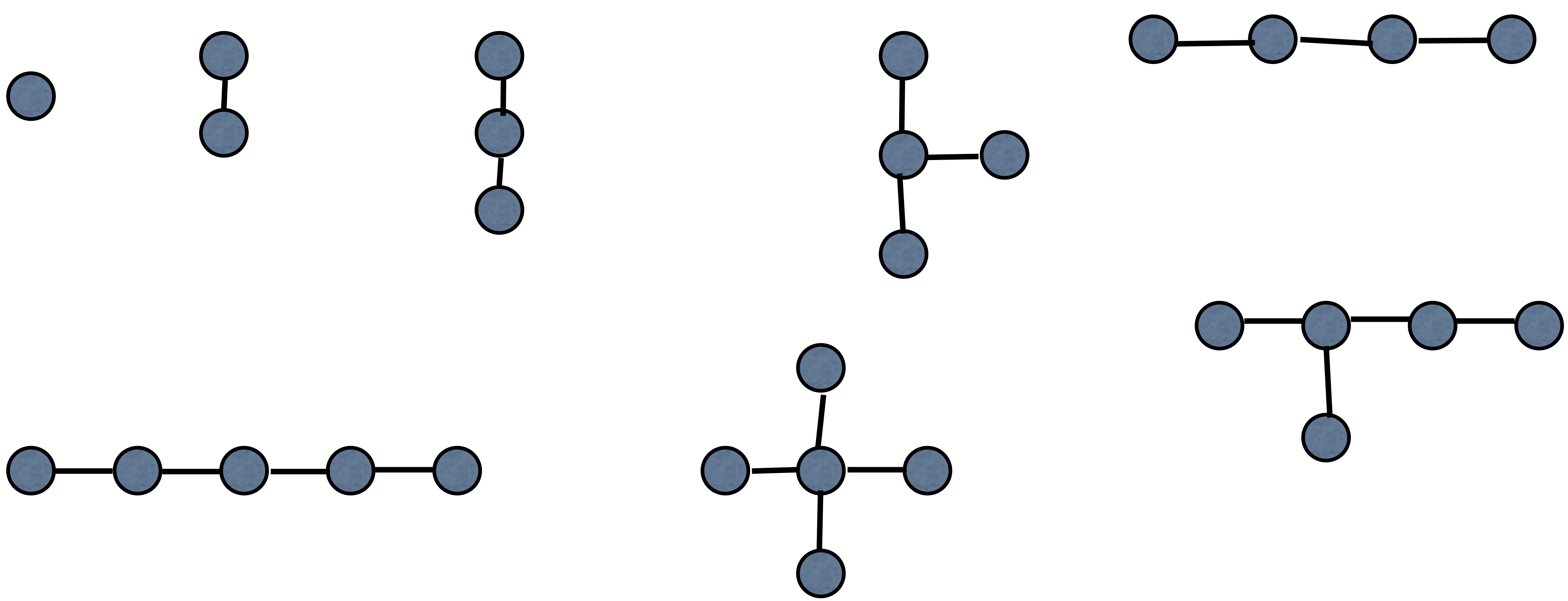
- Any undirected graph can be divided into **connected components**.
- It is easy to show that the path predicate in an *undirected graph* is an equivalence relation, and we define the connected components to be the equivalence classes of this relation.
- They are the maximal subgraphs that are connected -- a node's connected component is the subgraph formed by all the nodes to which it has a path.

Forests and Trees

- An undirected graph with no cycles is called a **forest** because it is divided into one or more connected components called **trees**.
- A tree, in graph theory, is a **connected** undirected graph with no cycles. Remember that we can draw a graph with the nodes and edges anywhere, as long as the edges connect the correct nodes. So a graph-theoretic tree may or may not look like the other trees in computer science.

Small Graph-Theoretic Trees

- Trees of one, two, or three nodes have only one shape per size.
- There are two shapes of four-node trees, and three shapes of five-node trees.



Unique Simple Path Theorem

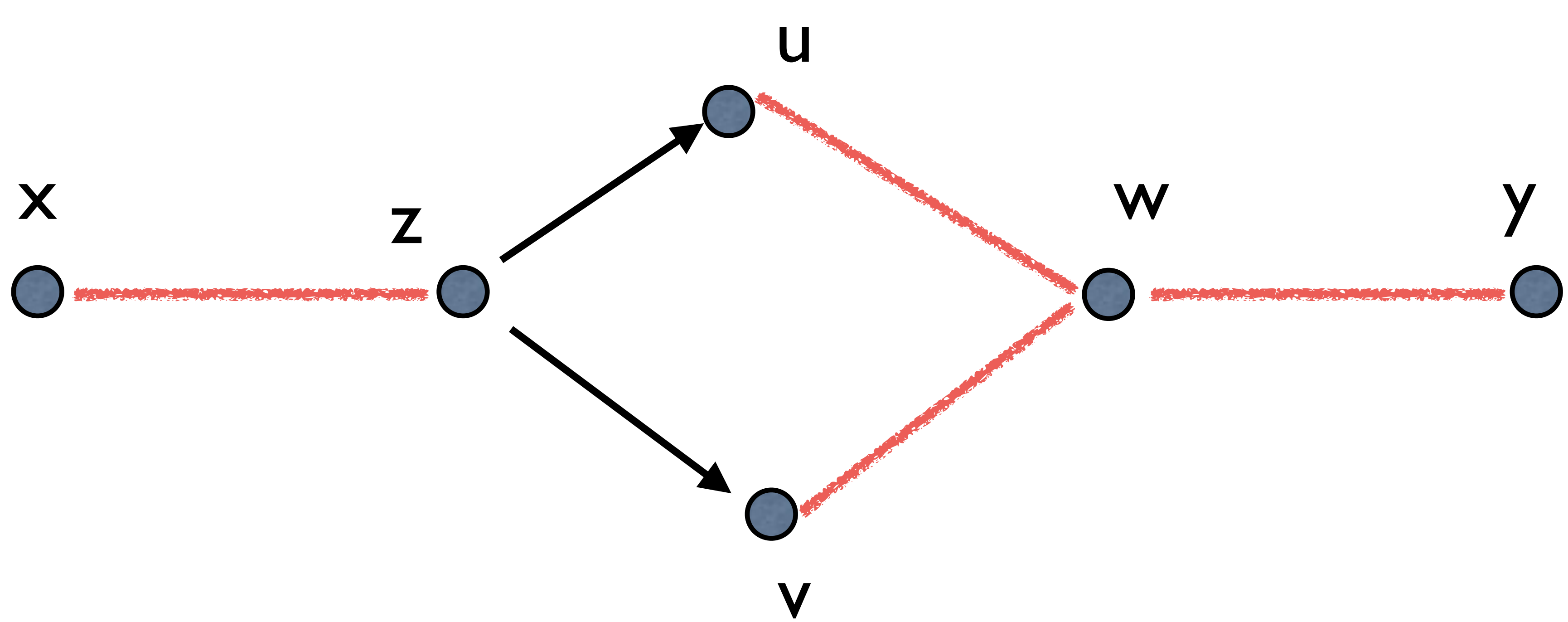
- **Theorem**: If x and y are nodes in a tree T, there is exactly one simple path in T from x to y. (Remember that a simple path is one that does not reuse a node or edge.)
- **Proof**: First, there must be at least one path because a tree is defined to be a connected graph, where every node has a path to every other node.

Unique Simple Path Theorem

- Could there be two different simple paths α and β from x to y? Suppose there were. Let z be the first node where the two paths split (z might be x). Let u be the next node after z on α, and v be the next node after z on β. Note that z, u, and v are three different nodes.

Picture for USP Theorem

Upper path = α



Lower path = β

Cycle exists from z to u to w to v to z, three or more edges as u and v can't both be equal to w.

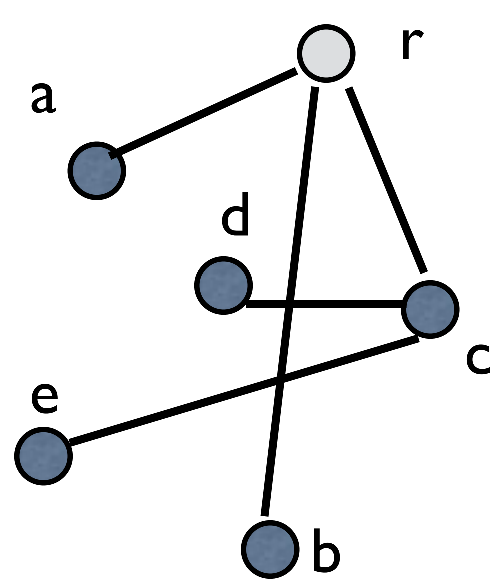
Unique Simple Path Theorem

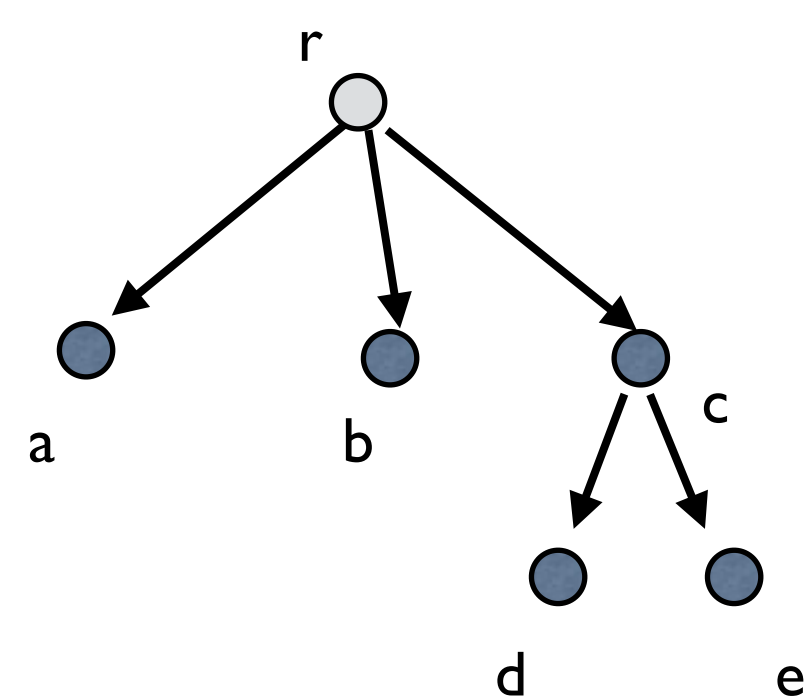
- There must be some point w, at or after u on α and at or after v on β, that is on both paths. (Certainly y is such a point, but let w be the earliest one, which might be u or v.)
- Then there is a simple path from z to u to w to v to z, and since this path has at least three edges, it is a cycle. But T is a tree, so our assumption that there were two paths has led to a contradiction.

Rooted Trees

- A **rooted tree** is a graph-theoretic tree with one of its nodes designated as the **root**. We can make a directed tree out of the undirected rooted tree by directing every edge away from the root.
- If we now draw such a tree with the root at the top, it looks like other "trees" we have seen in computer science.

Rooted Tree Pictures





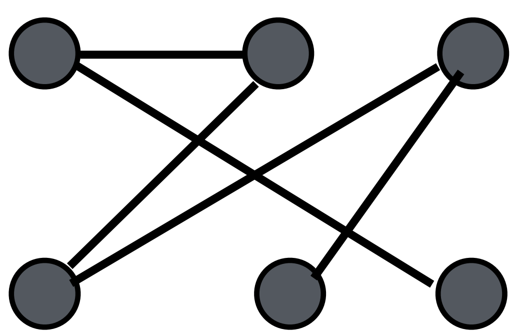
Rooted Tree Vocabulary

- If we call the root Level 0, we have its **children** at level 1, the nodes to which it now has directed edges. Level 1 nodes have children at Level 2, and so forth.
- The **depth** of a tree is its largest level number, which is the length of the longest directed path from the root.
- Nodes with no children are called **leaves**.

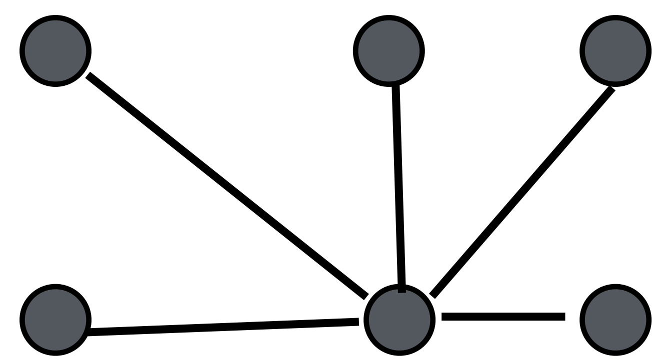
Clicker Question #3

- Here are four undirected trees. Which one cannot have exactly depth 3, no matter how we pick the root node?

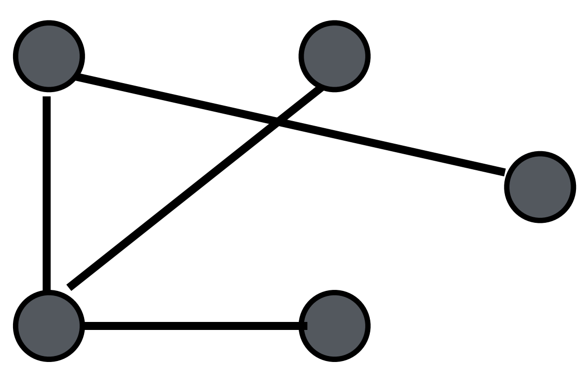
(a)



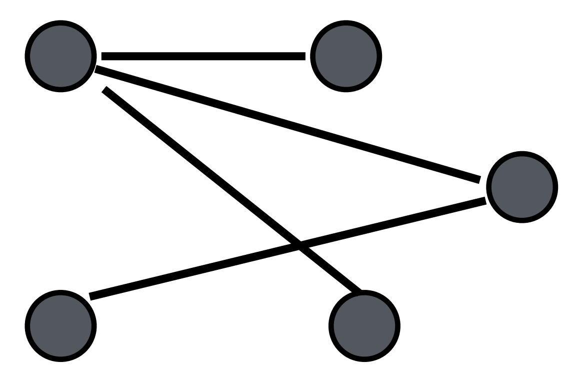
(b)



(c)



(d)

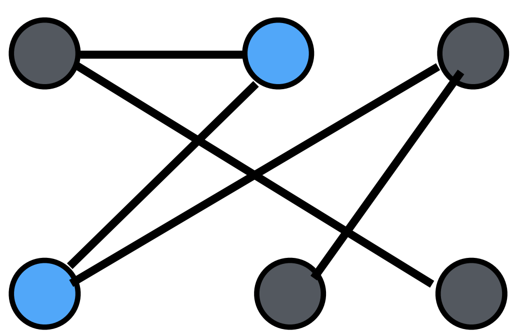


Not the Answer

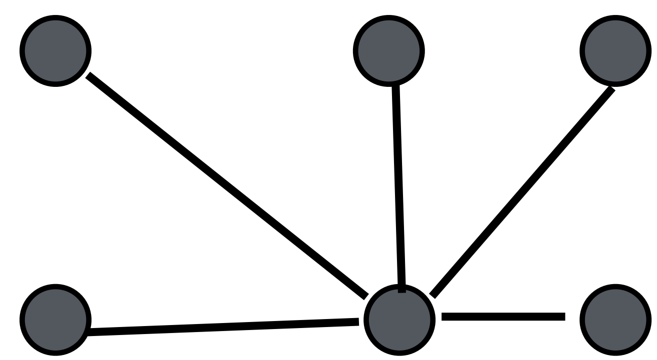
Clicker Answer #3

- Here are four undirected trees. Which one cannot have exactly 3, no matter how we pick the root node? Blue nodes have depth exactly 3.

(a)

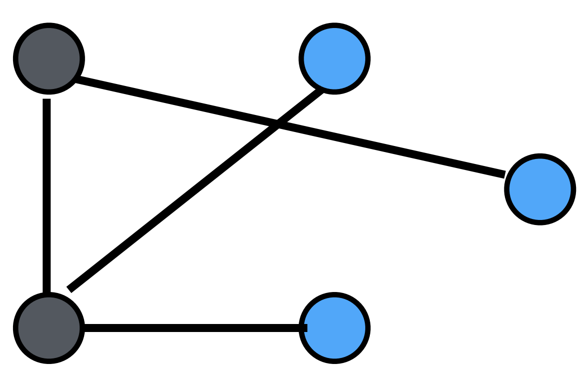


(b)

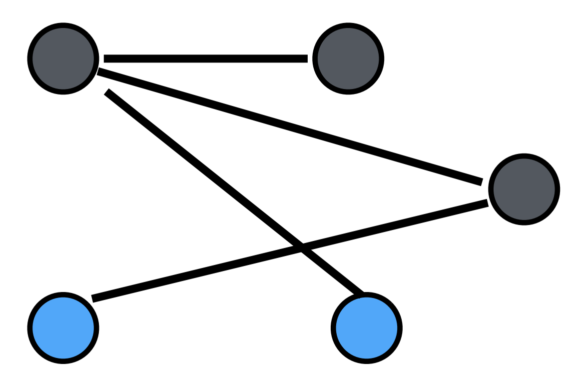


Depth 1 or 2

(c)

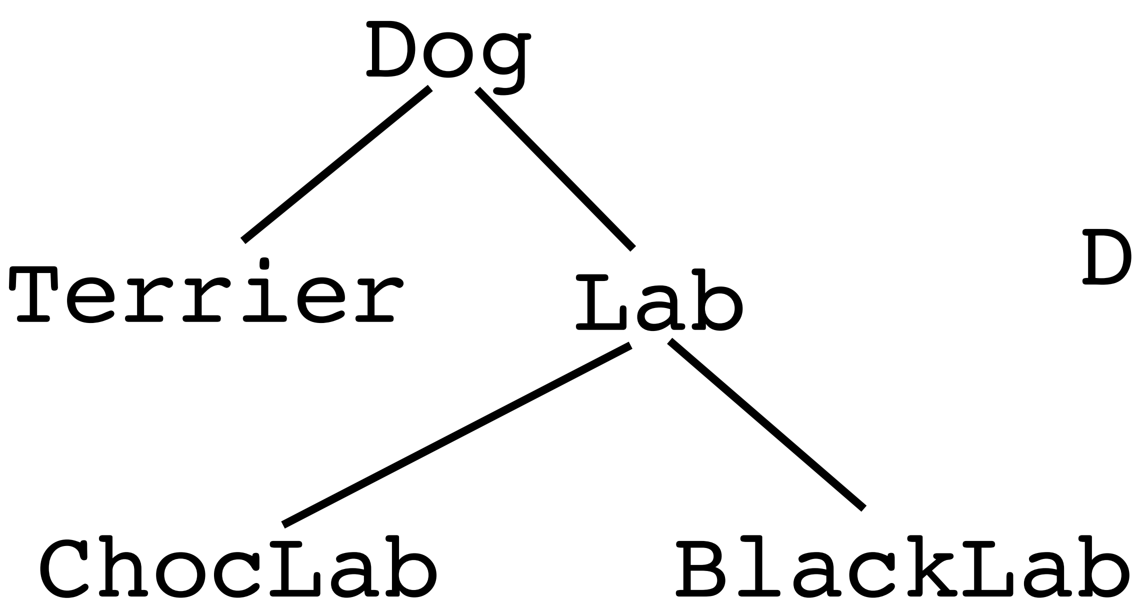


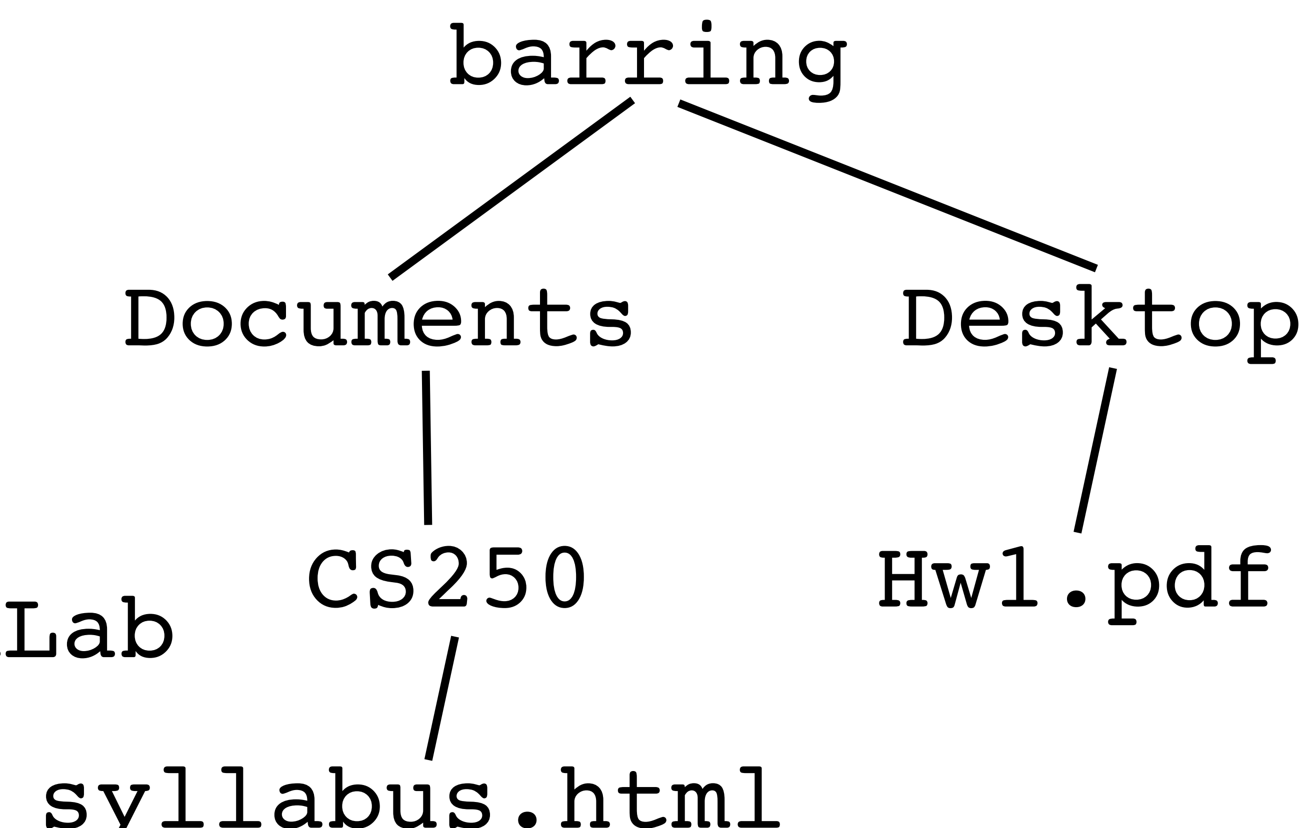
(d)



Examples of Trees

- Such trees model many kinds of **hierarchies**, such as parts of an organization, inheritance of classes in Java, or the hierarchy of directories (folders) on a computer.





A Recursive Tree Definition

- A single node, with no edges, is a rooted tree and the node is its root.
- We can make a rooted tree out of one or more existing rooted trees plus a new node x. The root of the new tree is x, and we add edges from x to the roots of each of the existing trees.
- The only possible rooted trees are those made by the two rules above.

Induction on Rooted Trees

- This is a recursive definition of rooted trees.
- As with our other recursively defined types, we now have a new Law of Mathematical Induction for rooted trees.
- If we prove $P(T)$ whenever T has only one node, and that $P(T)$ is true when T is made from subtrees U_1, U_2,..., U_k and $P(U_i)$ is true for all i, then we may conclude that $P(T)$ is true for any rooted tree T.

A Theorem About Rooted Trees

- Let's use this induction rule to prove a theorem.
- **Theorem**: If T is any rooted tree with n nodes and e edges, then $e = n - 1$.
- Base Case: If T is a one-node tree, then $e = 0$ and $n = 1$ so $e = n - 1$ is true.
- Now we have to set up the inductive step.

A Theorem About Rooted Trees

- Inductive Step: Let T be made by the second rule from U_1, U_2,..., U_k and say that each of the U_i's has n_i nodes and e_i edges, so that $e_i = n_i - 1$ by the IH.
- T has all the nodes and edges from all the subtrees, plus one new node (its root) and k new edges (one from its root to each of the existing roots).

A Theorem About Rooted Trees

- So n, the number of nodes in T, is the sum of the n_i's plus 1.
- And e, the number of edges in T, is the sum of the e_i's plus k.
- The sum S of the e_i's is the sum of the n_i's minus k, so $e = S + k$ and $n = (S + k) + 1$, and therefore $= n - 1$.
- We've completed the inductive step and thus proved our P(T) for all rooted trees T.