EE360C: Algorithms

Graphs

Summer 2019

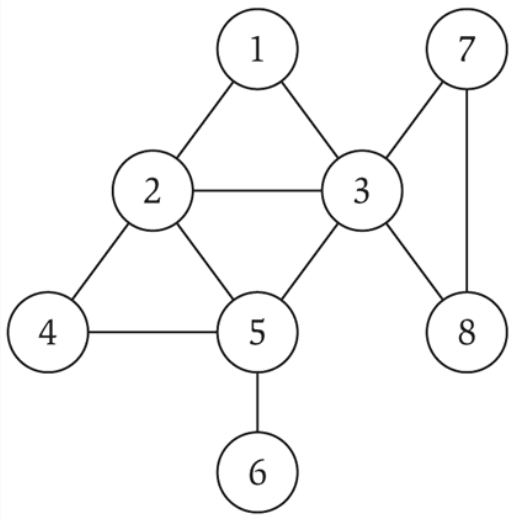
Department of Electrical and Computer Engineering
University of Texas at Austin

Definitions and Applications

Undirected Graphs

Undirected graph: $G = (V, E)$

- V: nodes
- E: edges between pairs of nodes
- captures pairwise relationships between objects
- graph size parameters: $n = |V|$, $m = |E|$

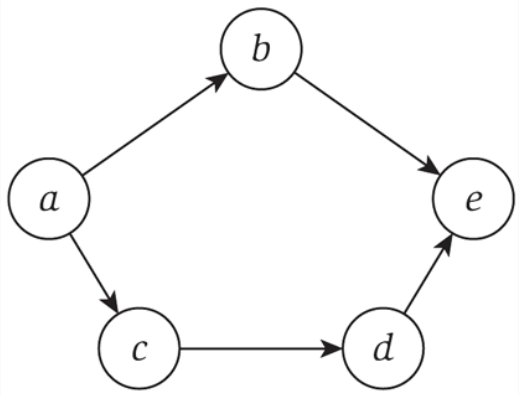


- $V = \{1, 2, 3, 4, 5, 6, 7, 8\}$
- $E = \{\{1,2\}, \{1,3\}, \{2,3\}, \{2,4\}, \{2,5\}, \{3,5\}, \{3,7\}, \{3,8\}, \{4,5\}, \{5,6\}\}$
- $n = 8$
- $m = 11$

Directed Graphs

Directed graph: $G = (V, E)$

- V: nodes
- E: edges between pairs of nodes
- captures one-way relationships between objects
- graph size parameters: $n = |V|$, $m = |E|$

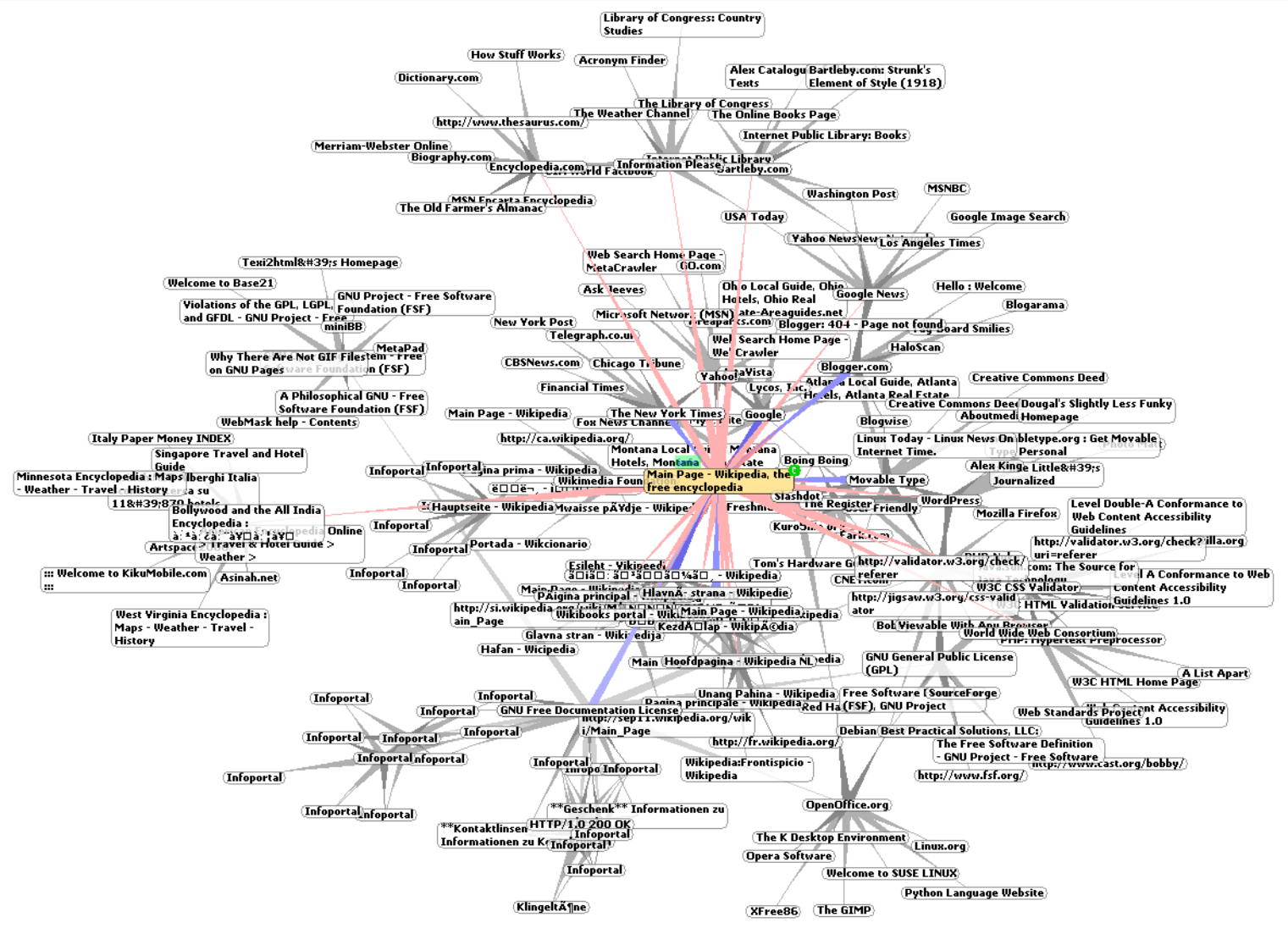


- $V = \{a, b, c, d, e\}$
- $E = \{\{a, b\}, \{a, c\}, \{b, e\}, \{c, d\}, \{d, e\}\}$
- $n = 5$
- $m = 5$

Example Graph Applications

Graph	Nodes	Edges	Directed
transportation	intersections	highways	no
communication	computers	fiber optic cables	no
World Wide Web	web pages	hyperlinks	yes
social	people	relationships	maybe
food web	species	predator/prey	yes
software systems	functions	function calls	yes
scheduling	tasks	precendences	yes

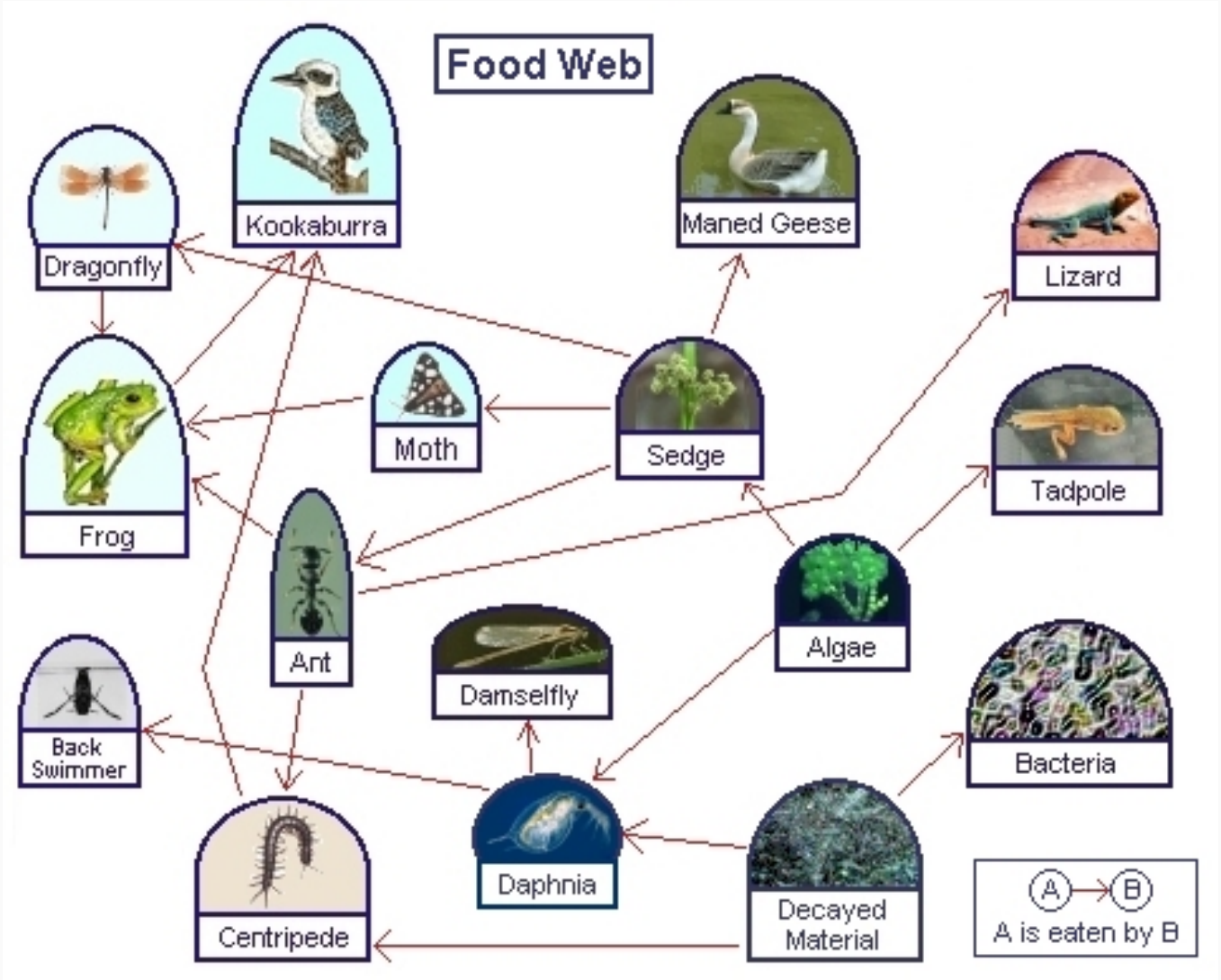
World Wide Web



Chris 73/Wikimedia Commons

http://en.wikipedia.org/wiki/File:WorldWideWebAroundWikipedia.png

Ecological Food Web



http://cbc.amnh.org/crisis/foodweb.html

Graph Representation

Graph Representation

There are two common ways of representing a graph $G = (V, E)$:

- a collection of *adjacency lists*
- *adjacency matrix*

Recall, $m = |E|$ and $n = |V|$.
Generally, we prefer the adjacency list because it uses considerably less memory for the more common *sparse graphs* (i.e., when $m \ll n^2$). We prefer the adjacency matrix when either:

- the graph is *dense*, i.e., m is close to n^2
- we need to quickly check if a particular edge exists

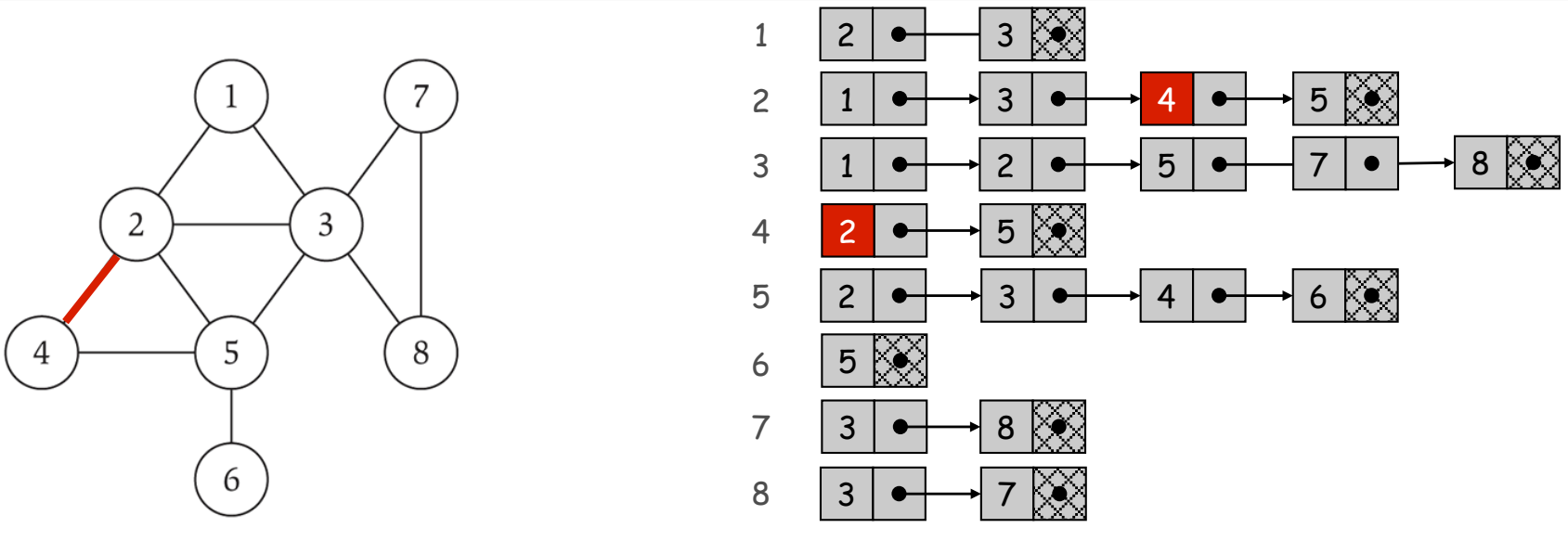
Adjacency List Representation

Adjacency List

Given a graph $G = (V, E)$, we represent it by an array Adj of n lists, one for each vertex in V.

- for each $u \in V$, Adj[u] is a list of vertices v such that there is an edge from u to v
- the order of the list is arbitrary
- for a directed graph, $\sum_{u \in V} |\text{Adj}[u]| = m$
- for an undirected graph, $\sum_{u \in V} |\text{Adj}[u]| = 2m$
- in either case, the total memory required to represent the graph is $\Theta(n + m)$

Adjacency Lists Visualized



Adjacency Lists and Weights

We often want to store weighted graphs; the adjacency list representation lends itself well to this:

- in a weighted graph, each edge has an associated weight, i.e., $w : E \to \mathbf{R}$
- we can easily store $w(u, v)$ in an adjacency list representation

Adjacency Matrix Representation

One problem with adjacency lists is that to determine if there's an edge from u to v, we have to search u's entire list. This can be expensive if there are a lot of edges.

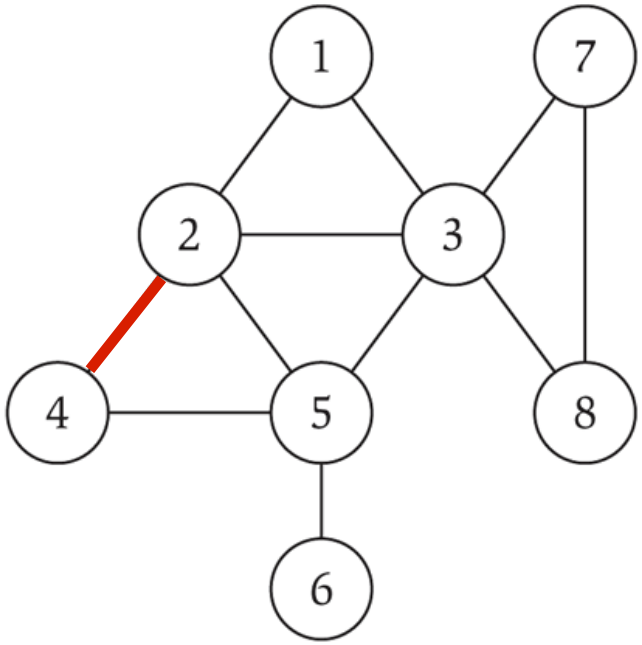
Adjacency Matrix

An adjacency matrix represents a graph $G = (V, E)$ using a $n \times n$ matrix $A = (a_{ij})$, such that $a_{ij} = 1$ if $(i, j) \in E$ and 0 otherwise.

- this requires $\Theta(n^2)$ memory, regardless of the number of edges.
- given that A^T is the transpose of A, i.e., $a_{ij}^T = a_{ji}$, for an undirected graph, $a_{ij} = a_{ji}$, and we can use half of the storage space

For a weighted graph, instead of storing 1 in a_{ij}, we store the weight of the edge (i, j).

Adjacency Matrix Visualized



	1	2	3	4	5	6	7	8
1	0	1	1	0	0	0	0	0
2	1	0	1	1	1	0	0	0
3	1	1	0	0	1	0	1	1
4	0	1	0	1	1	0	0	0
5	0	1	1	1	0	1	0	0
6	0	0	0	0	1	0	0	0
7	0	0	1	0	0	0	0	1
8	0	0	1	0	0	0	1	0

Questions

Question #1

Given an adjacency matrix representation of a graph, what is the time complexity for checking if the edge (u, v) exists in the graph?

Question #2

Given an adjacency list representation of a graph, what is the time complexity for checking if the edge (u, v) exists in the graph?

Paths and Connectivity

Definition: Path

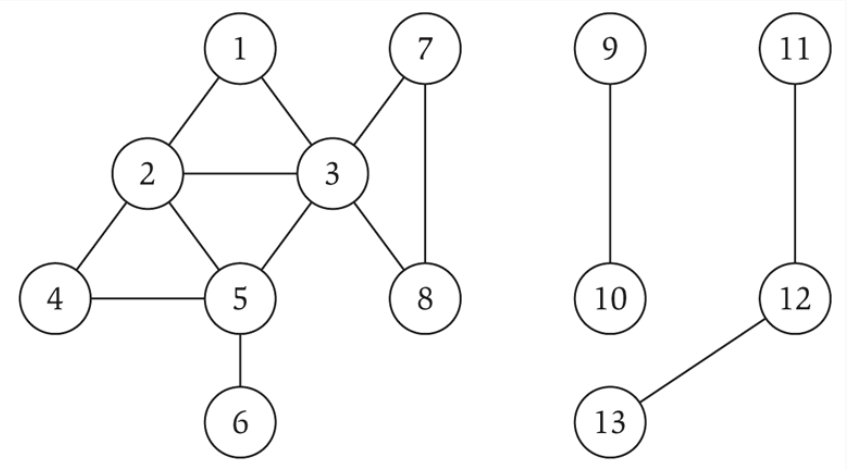
A path in an undirected graph $G = (V, E)$ is a sequence P of nodes $v_1, v_2, \ldots, v_{k-1}, v_k$ with the property that each consecutive pair $v_i, v_i + 1$ is joined by an edge in E.

Definition: Simple Path

A path P is simple if all nodes in P are distinct.

Definition: A Connected Undirected Graph

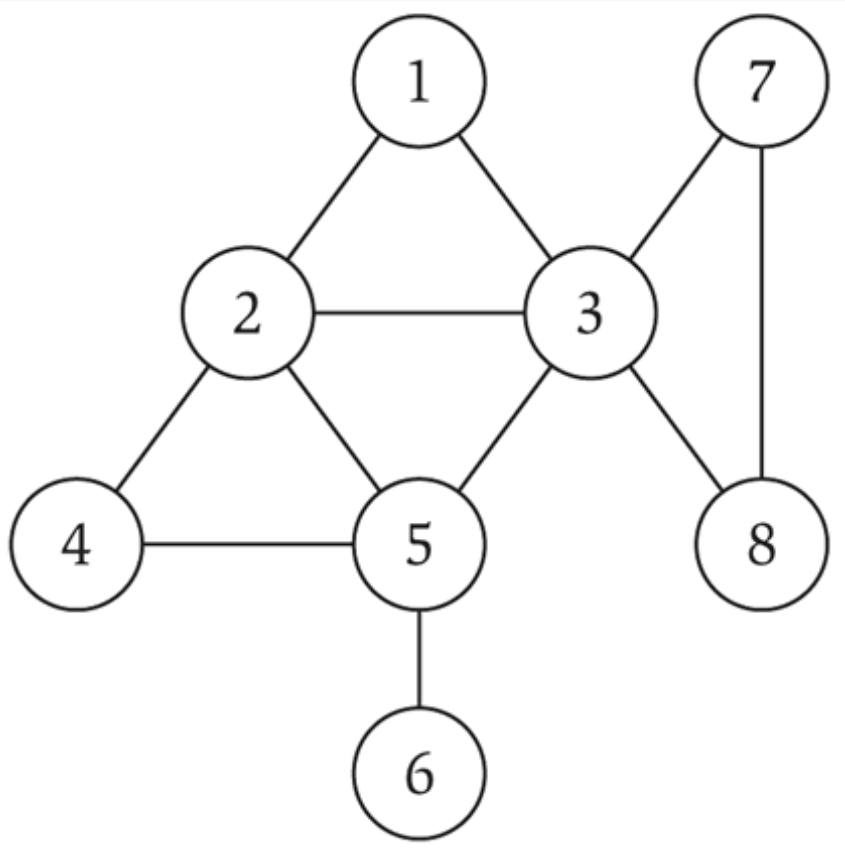
An undirected graph is connected if for every pair of nodes u and v, there is a path between u and v.



Cycles

Definition: Cycle

A cycle is a path $v_1, v_2, \ldots v_{k-1}, v_k$ in which $v_1 = v_k$, $k > 2$, and the first $k - 1$ nodes are all distinct.



cycle $= 1 - 2 - 4 - 5 - 3 - 1$

Trees

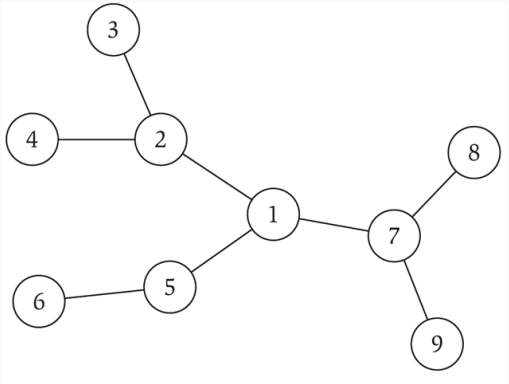
Definition: Tree

An undirected graph is a tree if it is connected and does not contain a cycle.

Theorem

Let G be an undirected graph on n nodes. Any two of the following statements imply the third:

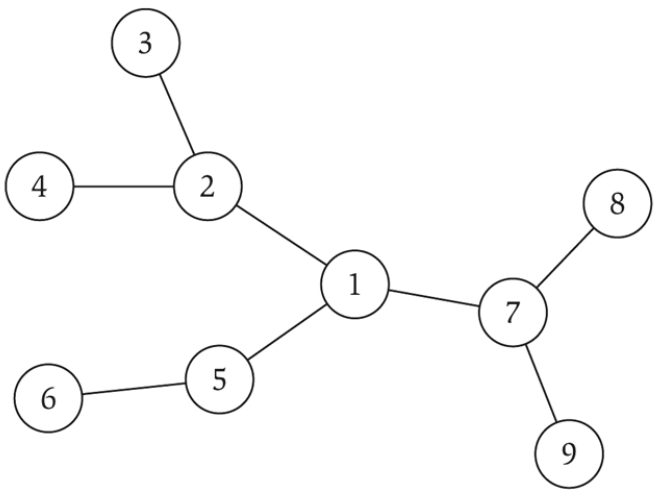
- G is connected
- G does not contain a cycle
- G has $n - 1$ edges



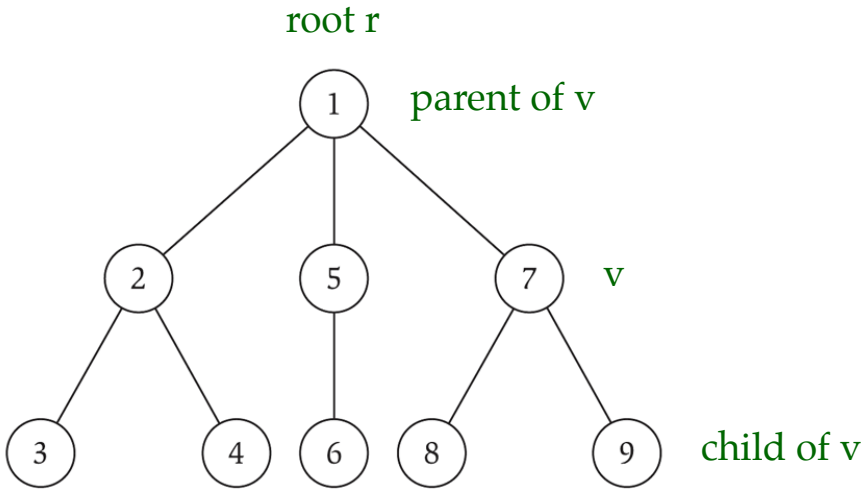
Rooted Trees

Definition: Rooted Tree

Given a tree T, choose a root node r and orient each edge away from r. This enables one to model hierarchical structure.

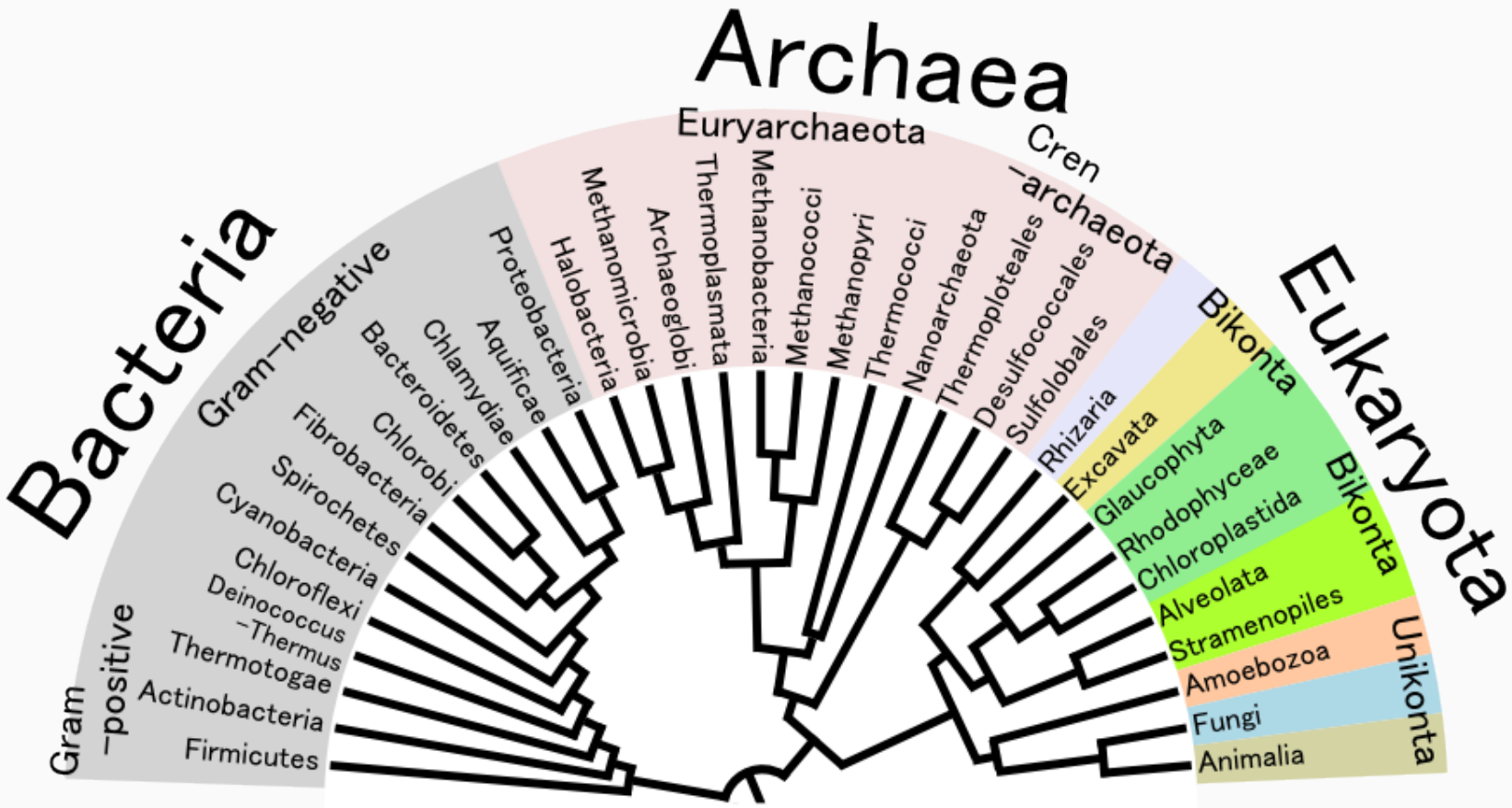


a tree



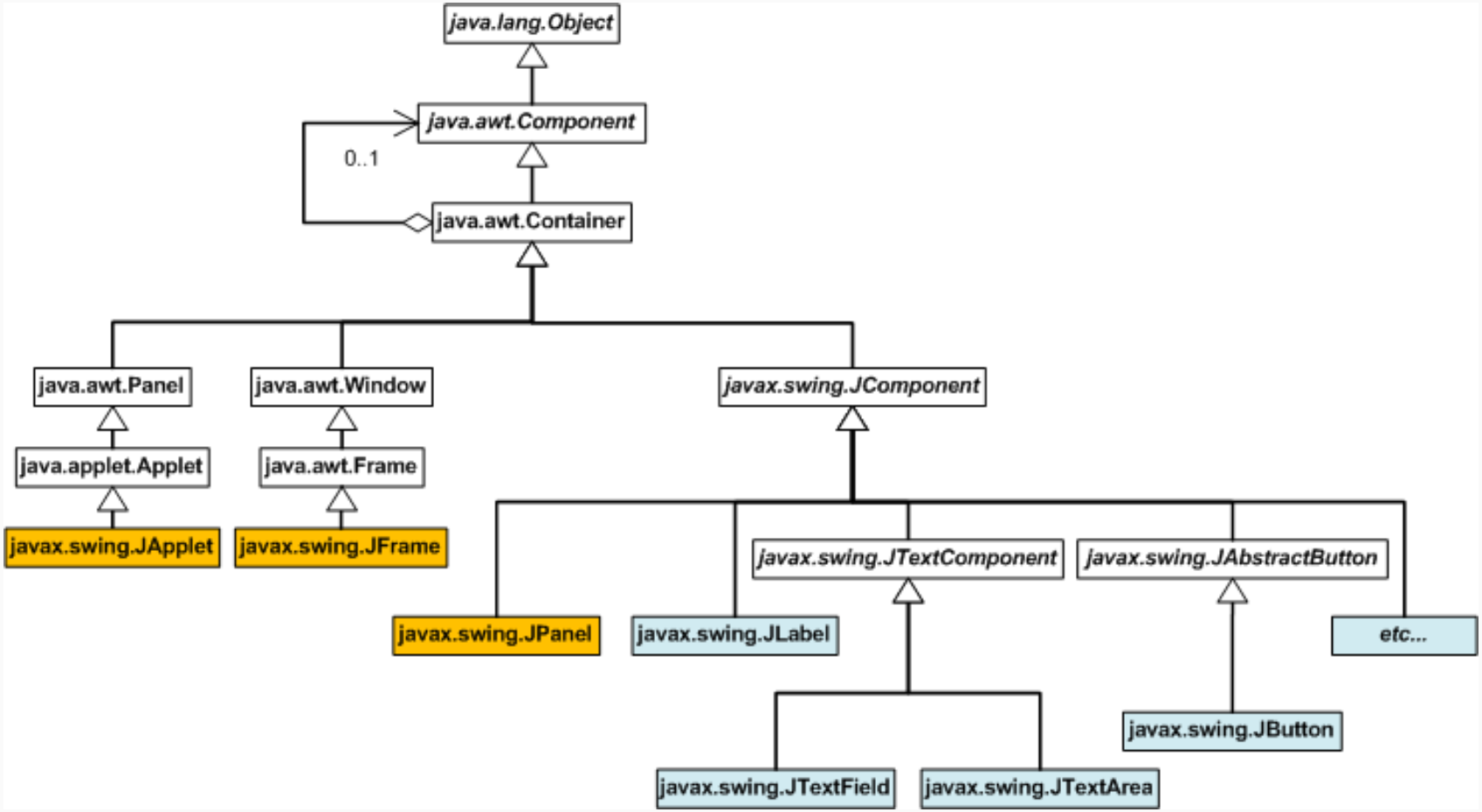
the same tree, rooted at 1

An Example Tree: Phylogeny Tree



http://en.wikipedia.org/wiki/File:Phylogenetic_Tree_of_Life.png

Another Example Tree: Object Oriented Class Architecture



http://www.clear.rice.edu/comp310/JavaResources/GUI/

Questions