Graphs and Trees

Class 5

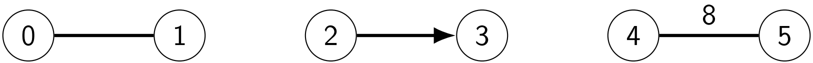
1.4 Graphs and Trees

- this section introduces the mathematical concept of a graph
- a graph is a structure that consists of vertices (aka nodes), e.g., v or w
- and edges $e = (v, w)$
- we take note of the number of vertices $n = |V|$
- and the number of edges $m = |E|$

- graphs are represented graphically (duh)
- vertices are drawn as circles, usually with labels inside
- edges as lines

Graph Variations

graphs may be



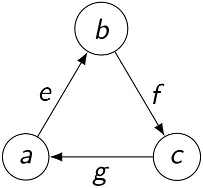
undirected directed weighted

Graph Terminology

- a graph with directed edges is a digraph
- a path is a sequence of vertices and edges that begins and ends with some vertex
- a cycle is a path which begins and ends on the same vertex, has at least one additional vertex, and has no repeated edges
- an acyclic graph has no cycles
- a digraph without cycles is termed a DAG
- a vertex in an undirected graph has degree: the number of edges touching it
- a vertex in a digraph has indegree and outdegree

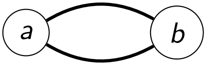
Path Terminology

- you can have a vertex without an edge
- but an edge cannot exist without two endpoint vertices
- some graphs have labeled edges, most do not
- your author defines a path as a series of vertices and edges, e.g., a, e, b, f, c
- but it can also be written as a series of edges in tuple form, e.g., $(a, b), (b, c)$
- or as a series of vertices, e.g., a, b, c

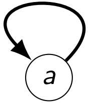


Unusual Edges

- in an undirected graph, two edges may exist between the same pair of vertices — these are parallel edges
- in a digraph, two edges are parallel if they go from the same vertex to the same vertex
- a loop is an edge, directed or not, whose two endpoints are the same vertex



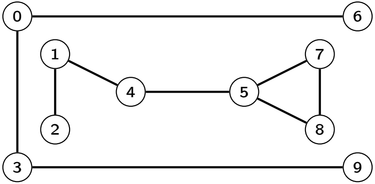
parallel



loop

Connectivity

- an undirected graph is either connected or not
- the connected components are perfect islands

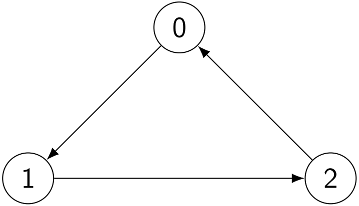


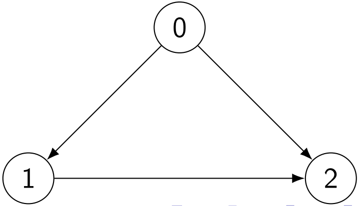
- for digraphs, connectivity is more complicated

Digraphs

- a digraph is strongly connected if for every pair v, w of vertices there is a path from v to w
- a digraph is weakly connected if it is not strongly connected, and for every pair v, w of vertices, there is either a path from v to w or a path from w to v

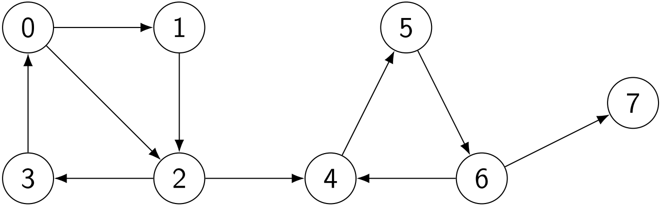
- another way of defining weak connectivity is to pretend that the graph is undirected — if it is connected when considered as an undirected graph, but not strongly connected, then it is weakly connected





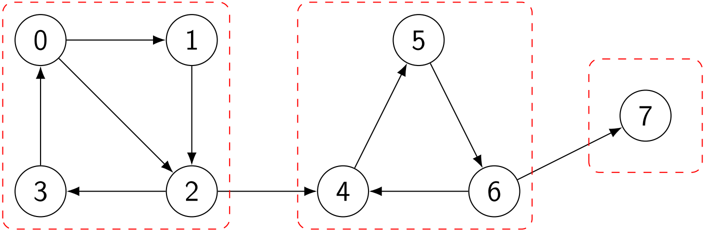
Strongly Connected Components

- a weakly connected digraph has strongly connected components
- these are subgraphs that by themselves are strongly connected
- the vertices of a digraph can be partitioned into disjoint maximal sets of vertices reachable via a directed path
- each set is a strongly connected component



Strongly Connected Components

- a weakly connected digraph has strongly connected components
- these are subgraphs that by themselves are strongly connected
- the vertices of a digraph can be partitioned into disjoint maximal sets of vertices reachable via a directed path
- each set is a strongly connected component



Trees

- a tree is a special graph

Tree

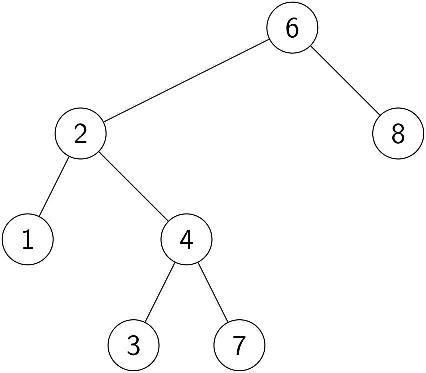
An empty graph (0 vertices and 0 edges) is a tree.
A non-empty graph is a tree if it:

$$\left.\begin{array}{l}\text{has } n \text{ vertices and } n-1 \text{ edges}\\ \text{is acyclic}\\ \text{is connected}\end{array}\right\}\ \text{any 2 are sufficient}$$

- a tree may be unrooted with no distinguished vertices
- or rooted with a distinguished root vertex
- if a digraph is a rooted tree, the root is the one vertex with indegree 0

Rooted Trees

- most of our trees will be rooted
- usually, we draw trees "upside down" with the root at the top and the leaves at the bottom
- the mental picture is a family tree



- children
- parents
- ancestors
- descendants
- root
- interior vertices
- leaves
- n-ary

Binary Search Tree: BST

- a tree every vertex of which is empty or has exactly two children (either of which may be empty) is a binary tree
- if we impose a couple of additional conditions, we get the binary search tree
    1. every vertex contains data (assume for simplicity that there are no duplicate values in the tree)
    2. the data in a vertex is greater than any value in its left subtree
    3. the data in a vertex is less than any value in its right subtree