Graph: Part 6 - Connectivity

Adila A. Krisnadhi

Faculty of Computer Science, Universitas Indonesia

References and acknowledgements

- Materials of these slides are taken from:
  - Kenneth H. Rosen. *Discrete Mathematics and Its Applications*, 8ed. McGraw-Hill, 2019. Section 10.3.
  - Jean Gallier. *Discrete Mathematics Second Edition in Progress*, 2017 [Draft]. Section 4.2, 4.4.
- Figures taken from the above books belong to their respective authors. I do not claim any rights whatsoever.

Path

- Informally, path is a way to travel from a node u to a node v by following the edges "correctly".
- Implicitly, every path in a graph is directed, regardless whether the graph is directed or not.
- Examples of use:
  - to decide if a message can be sent between two computers;
  - to compute the most efficient route for garbage pickup.

Path for directed graphs (1)

Definition

Given a digraph $G = (V, E, s, t)$, a **path from** a node $u \in V$ to a node $v \in V$ is a sequence $\pi = \langle u_0, e_1, u_1, e_2, u_2, \ldots, e_n, u_n \rangle$ where

- $n \geqslant 0$, $u_0 = u$, $u_n = v$,
- $u_0, \ldots, u_n \in V$,
- $e_1, \ldots, e_n \in E$
- $s(e_1) = u_0$, $t(e_n) = u_n$, and $u_i = t(e_i) = s(e_{i+1})$ for $1 \leqslant i \leqslant n - 1$.

Path for directed graphs (2)

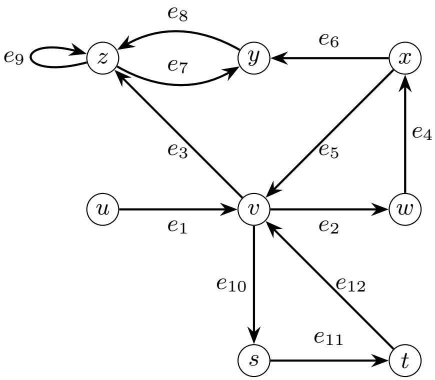
Let $\pi = \langle u_0, e_1, u_1, e_2, u_2, \ldots, e_n, u_n \rangle$ be a path.

- u_0 is called the **initial/source node** of π and u_n is the **terminal/sink node** of π.
- The path π is uniquely determined by its constituting edges, hence we sometimes represent π with its **edge sequence** $\langle e_1, \ldots, e_n \rangle$.
- The path π also induces a **node sequence** $\langle u_0, u_1, \ldots, u_n \rangle$.
- The notation $|\pi|$ denotes the **length** of π and is defined as $|\pi| = n$.
- When $|\pi| = 0$, π is called the **null path**. Its edge sequence is empty (denoted by ε) and its node sequence is $\langle u_0 \rangle$ containing just a single node u_0, which acts as both the initial and terminal node.
- If $u_0 = u_n$, π is called a **closed path**, otherwise π is an **open path**.
- A closed path of nonzero length is called a **circuit**.
- A digraph that contains no circuit is called a **directed acyclic** graph (**DAG**).

Path for directed graphs (3)

- A path is **simple** if its edge sequence does not contain duplicate edges.
- A path is **node-simple** if its node sequence does not contain duplicate nodes, except possibly for its initial node if the path is closed.
- Every node-simple path must be a simple path because in a node-simple path, every node occurs once (except possibly the initial node u if the path is closed), which means that every edge in its edge sequence must occur exactly once.

Example



- $\langle v, e_5, x, e_6, y \rangle$ and $\langle z, e_8, y \rangle$ are not a path
- $\langle u, e_1, v, e_2, w \rangle$ is a node-simple path (thus a simple path) with length 2, edge sequence $\langle e_1, e_2 \rangle$, and node sequence $\langle u, v, w \rangle$
- $\langle u, e_1, v, e_2, w, e_4, x, e_5, v \rangle$ is a simple path, but not a node-simple path
- $\langle z, e_7, y, e_8, z \rangle$ is a node-simple circuit (thus a simple circuit)
- $\langle v, e_2, w, e_4, x, e_5, v, e_{10}, s, e_{11}, t, e_{12}, v \rangle$ is a simple circuit, but not a node-simple circuit

Path for undirected graphs

Definition

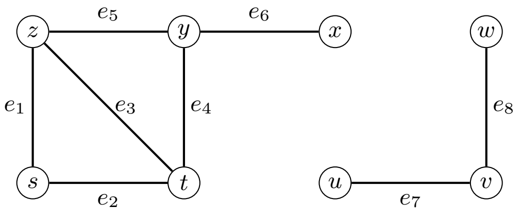
Given a undirected graph $G = (V, E, st)$, a **path from** a node $u \in V$ to a node $v \in V$ is a sequence $\pi = \langle u_0, e_1, u_1, e_2, u_2, \ldots, e_n, u_n \rangle$ where

- $n \geqslant 0$, $u_0 = u$, $u_n = v$,
- $u_0, \ldots, u_n \in V$,
- $e_1, \ldots, e_n \in E$
- $st(e_i) = \{u_{i-1}, u_i\}$ for $1 \leqslant i \leqslant n$.

- The notions of initial node, terminal node, path length, null path, closed path, open path, circuit, node-simpleness, and simpleness are the same as for digraphs.

Example

The following is a single graph with 8 nodes.



Are these paths? Simple paths? Node-simple paths? Circuits?

- $\langle s, e_2, t, e_4, u \rangle$
- $\langle u, e_7, v, e_8, w \rangle$
- $\langle s, e_2, t, e_4, y, e_5, z, e_1, s \rangle$
- $\langle x, e_6, y, e_5, z, e_3, t, e_4, y \rangle$

Connectedness

Definition

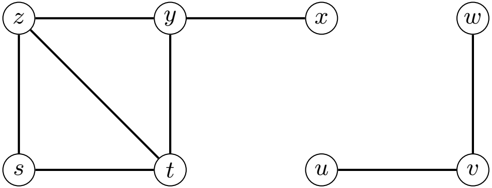
Let G be an undirected graph. G is **connected** iff there is a path (including null path) between every pair of nodes in G. Otherwise, G is **disconnected**.

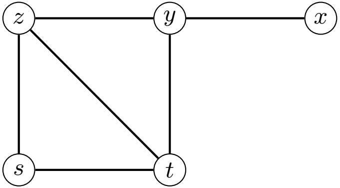
Definition

Let G be a directed graph. G is **strongly connected** iff for every two nodes a, b in G, there is a path (including null path) from a to b and a path from b to a. Furthermore, G is **weakly connected** iff the underlying undirected graph of G is connected.

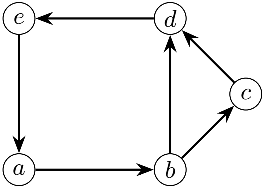
- Every node is connected to itself by a null path.

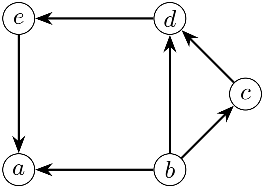
Example: Which graphs are connected?

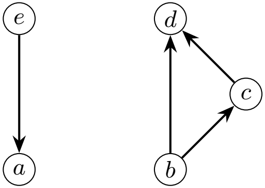




Which graphs are strongly connected? Weakly connected?







Connected components

Definition

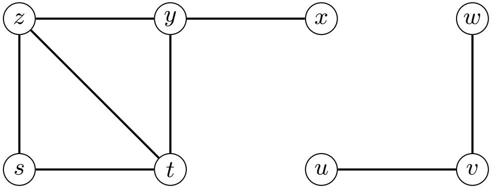
Let G be an (possibly disconnected) undirected graph. A **connected component** of G is a subgraph H of G such that H is connected and H is maximal, i.e., H is not a proper subgraph of another connected subgraph of G.

Definition

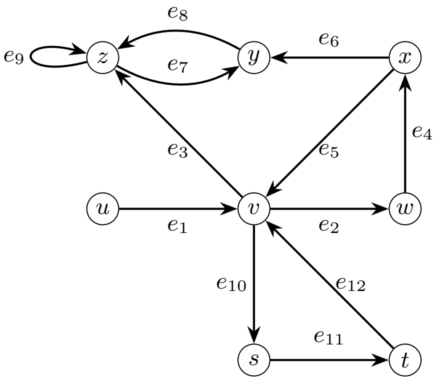
Let G be a (possibly disconnected) digraph. A subgraph H of G is a **a strongly connected component (scc)** of G iff H is strongly connected and H is maximal, i.e., H is not a proper subgraph of another strongly connected subgraph of G.

- A graph with $n > 0$ nodes can have at least 1 and at most n connected components.
- A graph is connected iff it has just a single connected component.

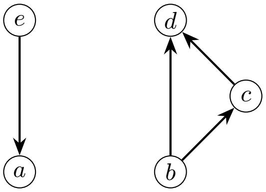
Example: Determine the connected components of this graph



Example: Determine the strongly connected components of this graph



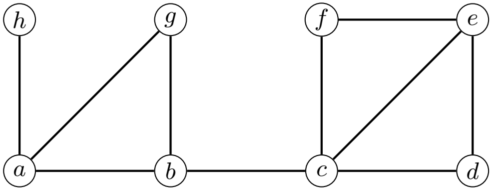
Example: Determine the strongly connected components of this graph



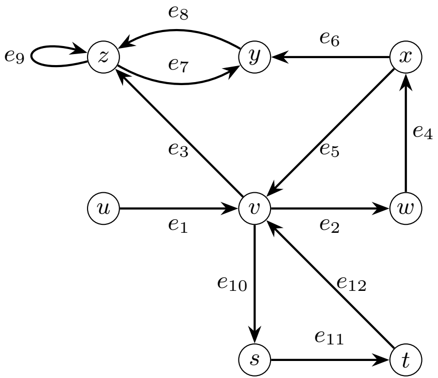
Cut vertices and cut edges

- A **cut vertex** (**cut node** or **articulation point**) v in a graph G is a node in G whose removal increases the number of (strongly) connected components of G.
- A **cut edge** (**bridge**) e in a graph G is an edge in G whose removal increases the number of (strongly) connected components of G.
- Removal of a cut vertex or a cut edge from a connected graph yields a subgraph that is disconnected.
- Practical example: in a graph representing a computer network,
  - cut vertex: essential router that cannot fail for all computers to be able to communicate
  - cut edge: essential link that cannot fail for all computers to be able to communicate

Example: What are the cut nodes and bridges?



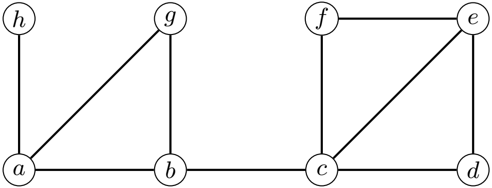
Example: What are the cut nodes and bridges?

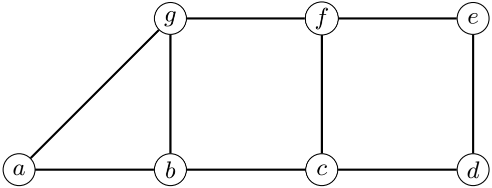


Vertex connectivity

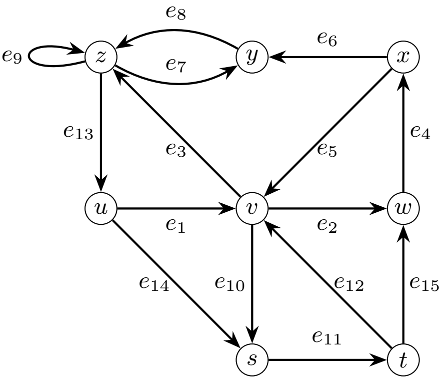
- Not all graphs have a cut node, e.g., the complete graphs K_n, $n \geqslant 3$. So graphs have different degree of connectedness.
- Let $G = (V, E)$ be a connected graph. A **vertex cut** (**separating set**) V' is a subset $V' \subseteq V$ such that $G - V'$ is disconnected.
- A graph G may have more than one vertex cut. **Vertex connectivity** $\kappa(G)$ is the size of the smallest vertex cut in G (if G is a complete graph K_n, we define $\kappa(G) = n - 1$).
- If G is disconnected, $\kappa(G) = 0$.
- If G has a cut vertex, then $\kappa(G) = 1$.
- If $\kappa(G) = k$, then G is said to be ***j*-connected** for all $0 \leqslant j \leqslant k$.

Example: Find the vertex connectivity of these graphs





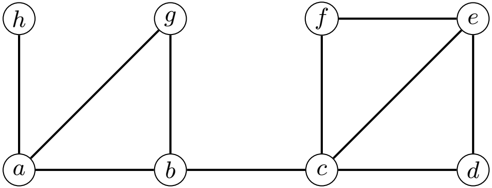
Example: Find the vertex connectivity of this graph

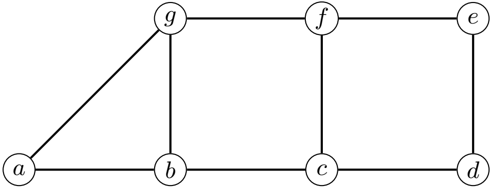


Edge connectivity

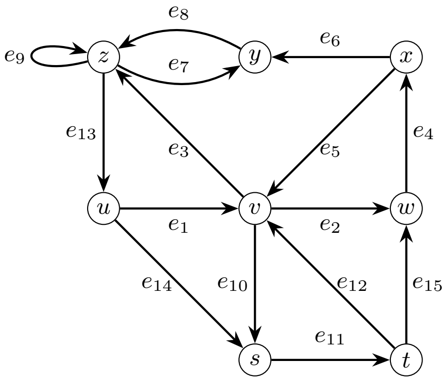
- Alternative way to define degree of connectedness of a graph is by considering the edge removal, instead of node removal.
- Let $G = (V, E)$ be a connected graph. An **edge cut** E' is a subset $E' \subseteq E$ such that $G - E'$ is disconnected.
- **Edge connectivity** $\lambda(G)$ of G is the size of the smallest edge cut in G where we define $\lambda(G) = 0$ if G is disconnected or has only one node.

Example: Find the edge connectivity of these graphs





Example: Find the edge connectivity of this graph



FACULTY OF COMPUTER SCIENCE

Lemma

Let $G = (V, E, st)$ *be an undirected graph. Then,*

$$\kappa(G) \leqslant \lambda(G) \leqslant \min_{v \in V} \deg(v)$$

Proof (Exercise).

- Show that $\kappa(G) \leqslant \min_{v \in V} \deg(v)$ and $\lambda(G) \leqslant \min_{v \in V} \deg(v)$.
- Show that $\kappa(G) \leqslant \lambda(G)$.

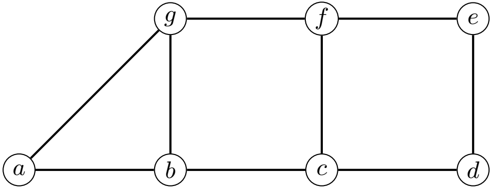
Counting paths between nodes

Theorem

Let $G = (V, E)$ be a (undirected/directed) graph with adjacency matrix $\mathbf{A}$ with respect to the ordering $v_1, \ldots, v_n$ of nodes of G. Then, the number of different paths of length r from v_i to v_j with $r > 0$ is equal to the (i, j)th entry of $\mathbf{A}^r$

See Rosen for proof.

Example: How many paths of length 3 from b to e



Example: How many paths of length 3 from v to y