Mathematics and Problem Solving

Lecture 9

Graph Theory

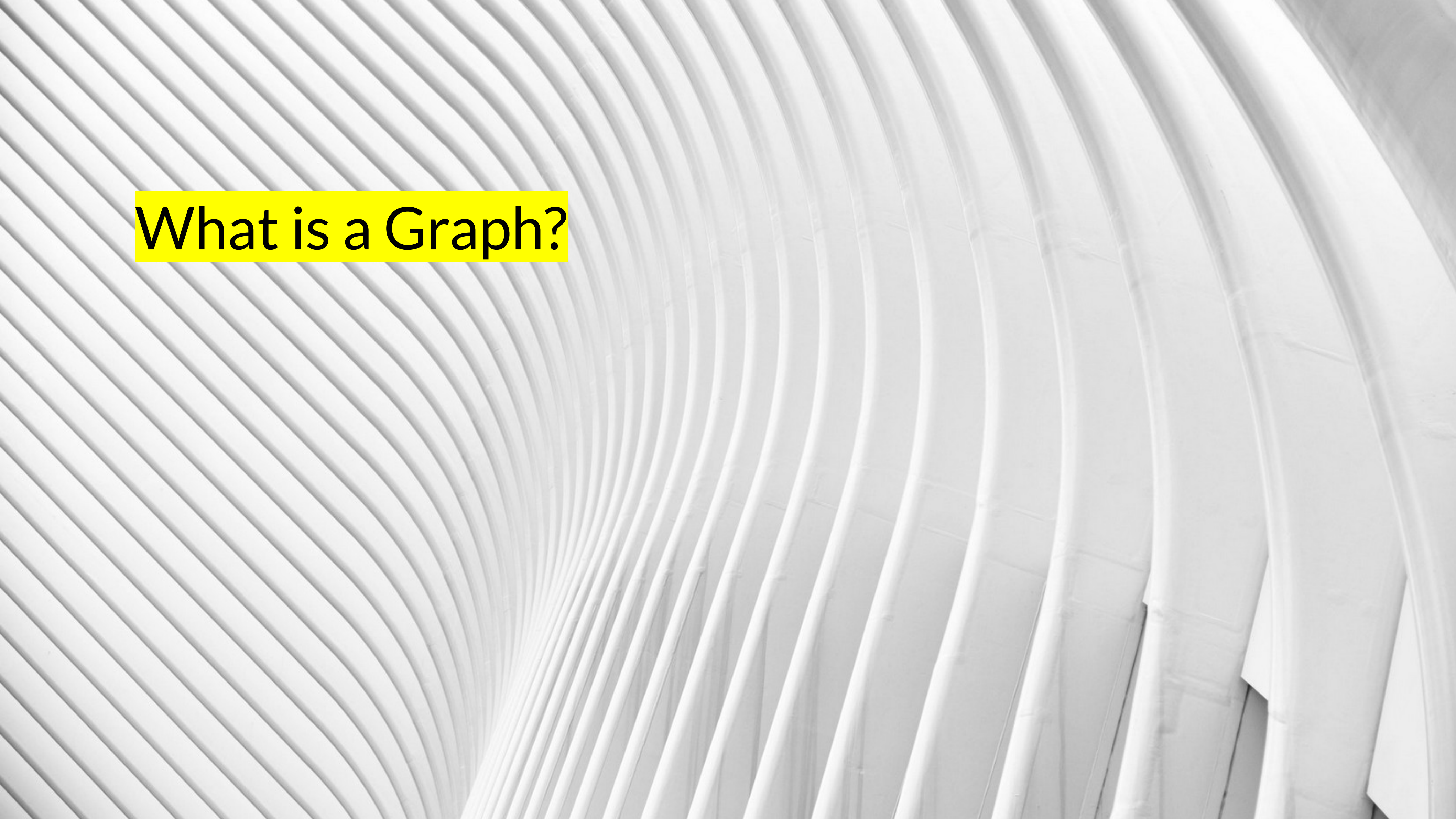
Overview

- What is a Graph
- Terminology
- Representing Graphs

Why do we study graph theory ?

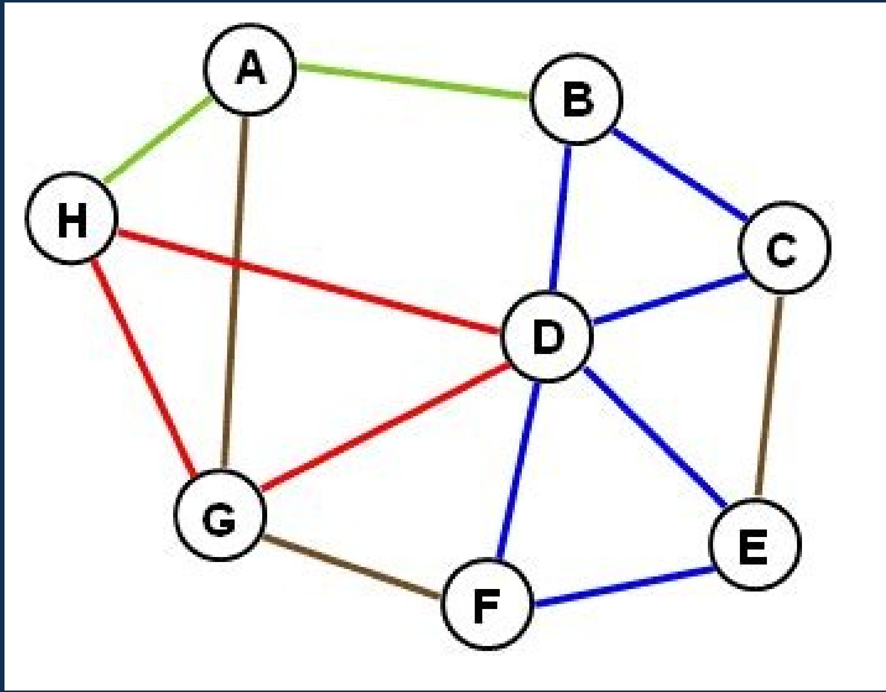
- Graph gives us the tool to formally study structures in graphical representations
  - Databases
  - Task planning
  - Pattern recognition
  - Tasks optimization
  - Scheduling
  - Data mining – Clustering
  - Solve shortest path problems

What is a Graph?



What is a graph?

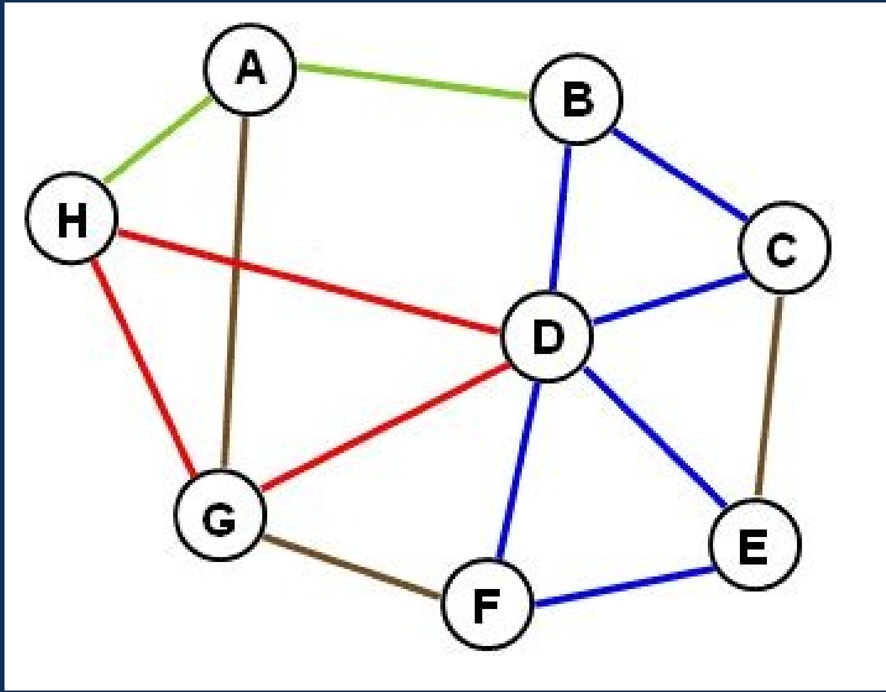
- A **graph** is a set of **vertices** connected by **edges**
- In other words, it's
  - Circles and lines
  - Dots and arcs
  - Boxes and arrows
  - Junctions and pipes
  - Cities and roads
  - Rooms and doors
  - ...etc.



Exercise 1:

How might you use a graph to solve the following problems? What are the nodes? What are the edges?

1. Planning a delivery route
2. Playing Tic-Tac-Toe
3. Creating levels for a video game
4. Machine translation



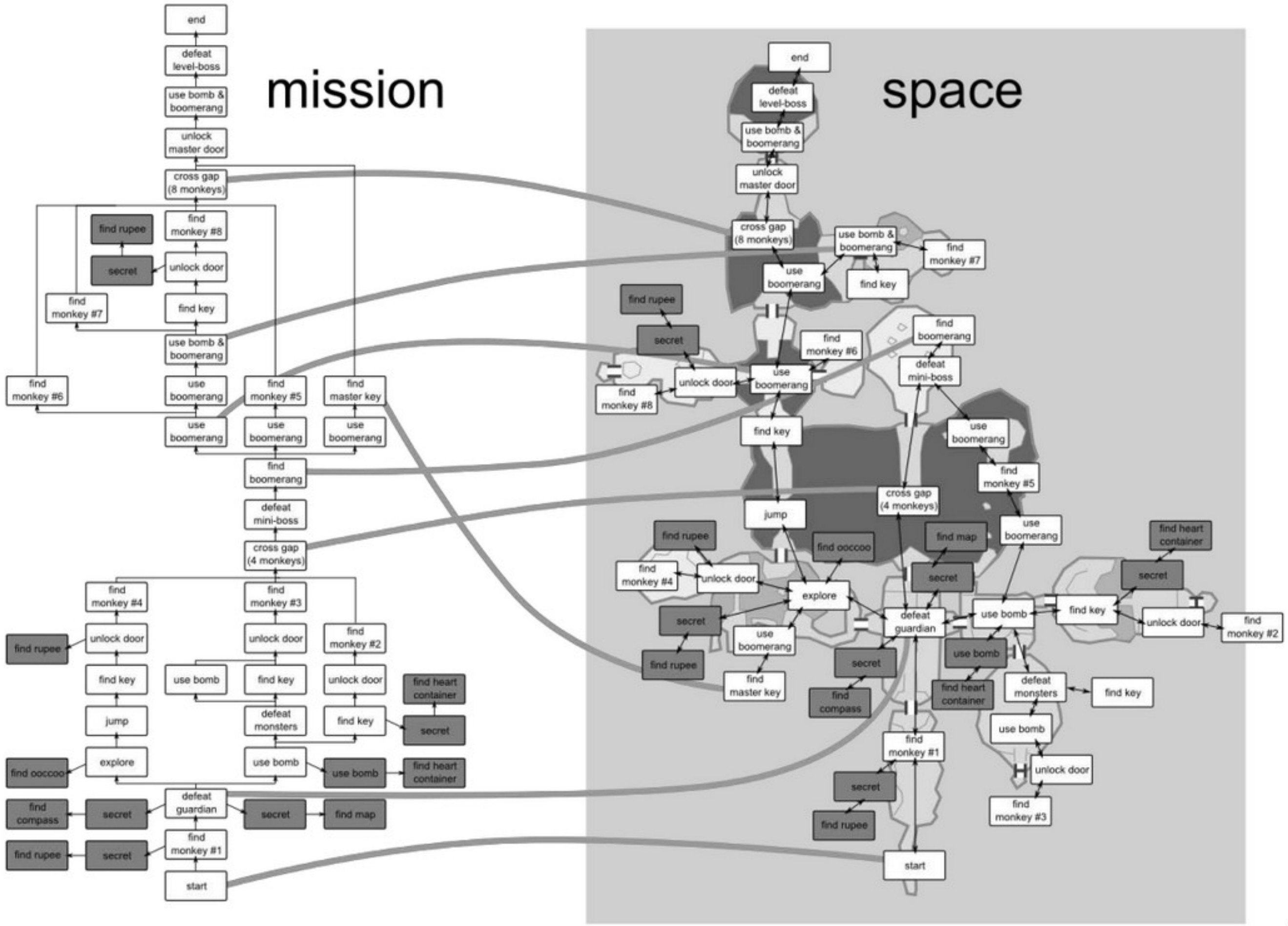
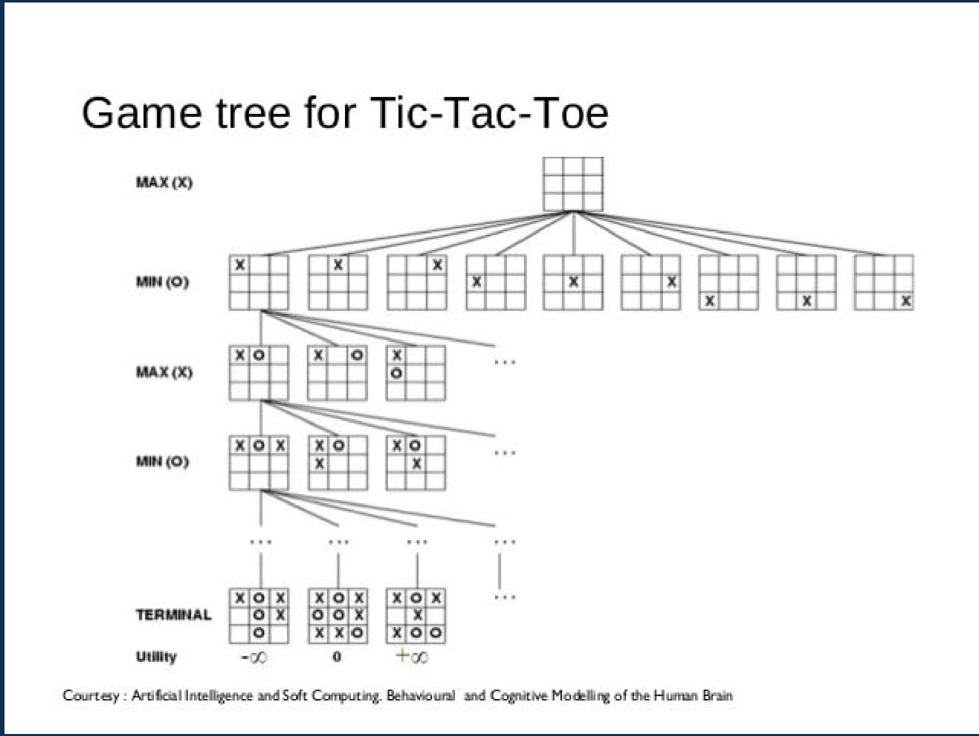


Figure 1. Mission and space in the Forest Temple level of *The Legend of Zelda: The Twilight Princess*

Dormans, Joris. "Adventures in level design: generating missions and spaces for action adventure Games." Proceedings of the 2010 workshop on procedural content generation in games. 2010.

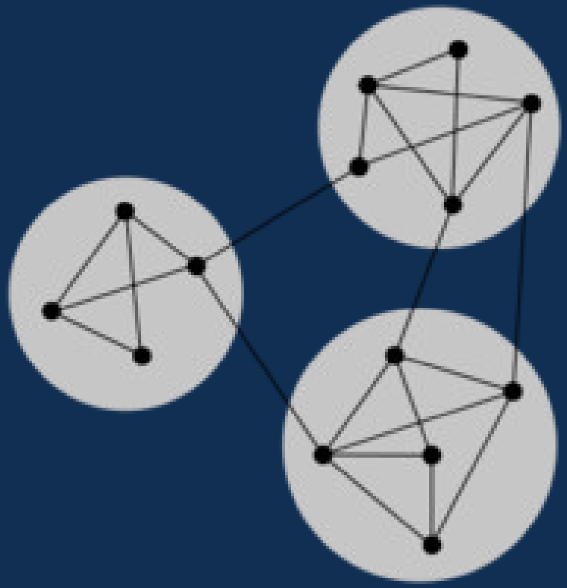
- Game AI with MINIMAX search
- Construct a game state tree
  - Terminal states have +/- utility
  - You take moves to maximise utility
  - Assume oponent takes moves to minimise utility



Terminology

Connected

- If a graph is connected there is a path from every point to every other point
- Nodes are immediately connected if they share an edge



Connected

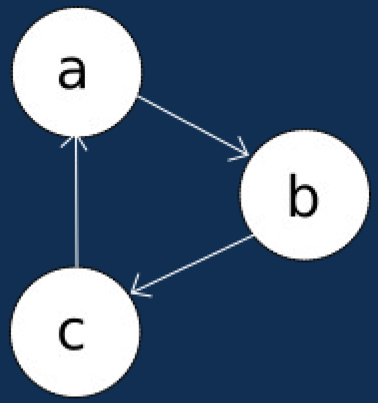
Exercise 2:

Which of the following are connected graphs?

1. A graph of possible derivations in a formal system
2. The road network in the UK
3. A social network
4. The Internet

Directed Graph (Digraph)

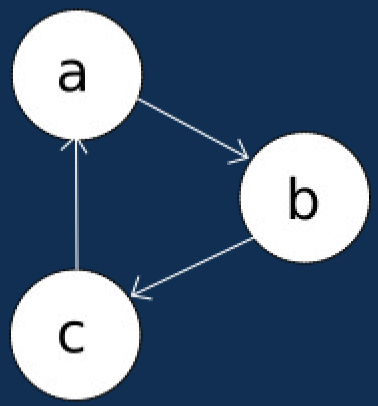
- A graph is **directed** if the edges have a **direction**
  - Represented with arrow heads
- Otherwise it is **undirected**
- Directed graphs might represent
  - The flow of water through pipes
  - A one-way relationship such as derivation: $A \Rightarrow_R B$



Exercise 3:

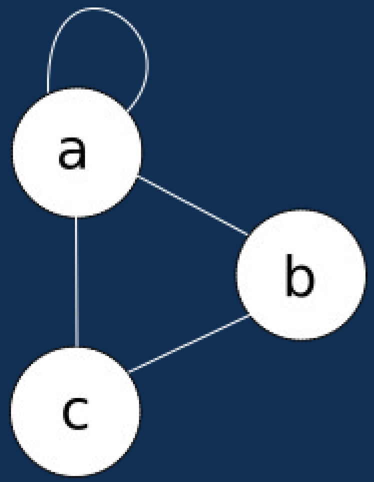
Which of the following are digraphs?

1. Sat-Nav directions
2. A graph of game states in a game of chess
3. A computer network
4. A snakes and ladders game board



Degree of a Node

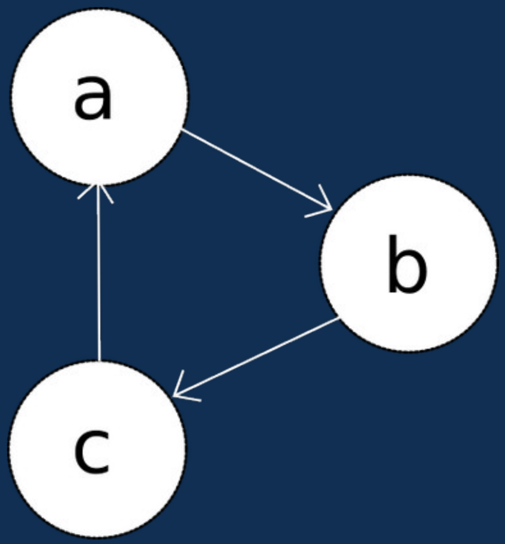
- The **degree** of a node is the number of edge connections it has
  - The numbers of edges incident to it
  - Loops count for 2
- For example
  - deg(a) = 4
  - deg(b) = 2
  - deg(c) = 2



Indegree and Outdegree

- For a node in a directed graph
  - **indegree** is the number of edges pointing to a node
  - **outdegree** is the number of edges pointing away from it

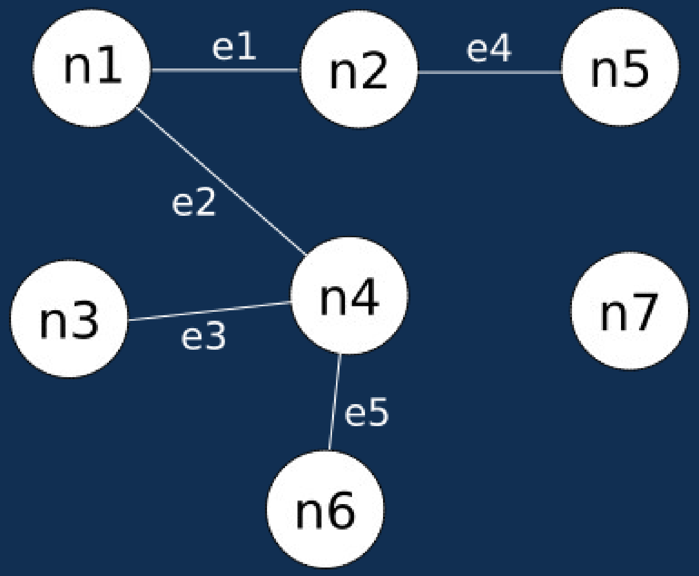
degree = indegree + outdegree



Exercise 4:

What is the degree of the following nodes:

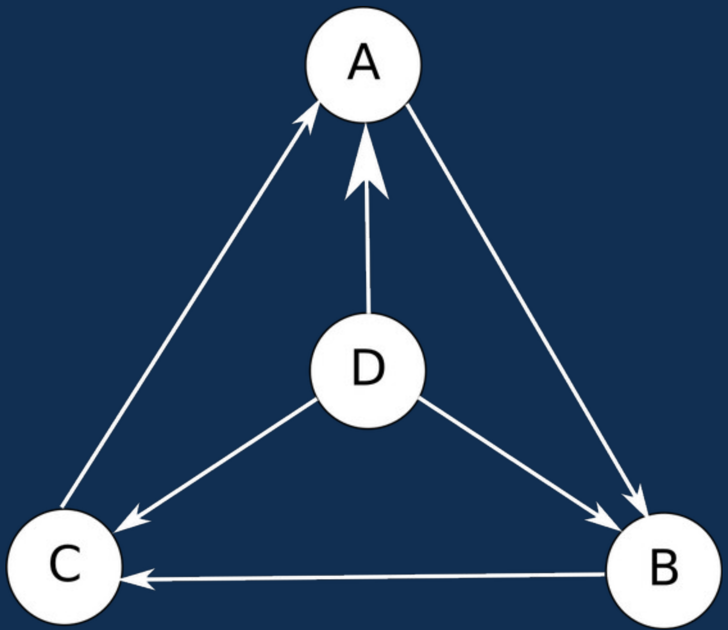
1. n1
2. n3
3. n4
4. n7



Exercise 5:

What is the in- and out-degree of the following nodes?

1. A
2. B
3. C
4. D



Some “types” of node for directed graph

- Source node is a node with

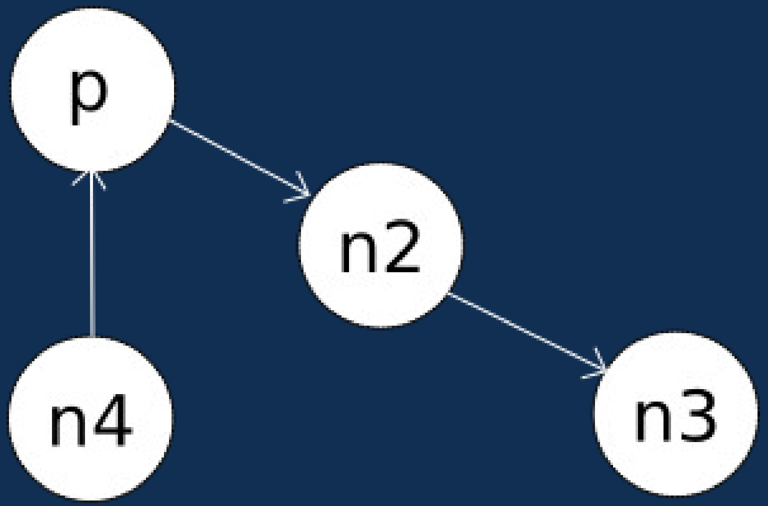
  indegree = 0.

- Sink node is a node with

  outdegree = 0.

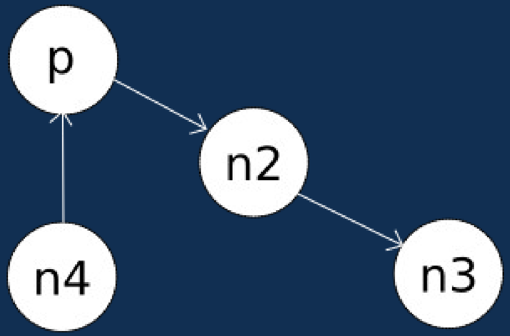
- Transfer node is node with

  indegree ≠ 0 and outdegree ≠ 0.



Exercise 6:

Which of the nodes for the graph below are sources, sinks, and transfer nodes?



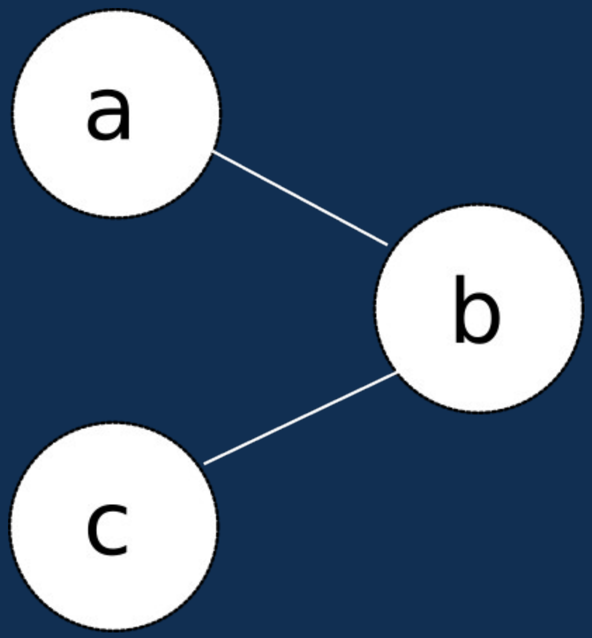
- Hint:
  - Source node is a node with indegree = 0.
  - Sink node is a node with outdegree = 0.
  - Transfer node is node with indegree ≠ 0 and outdegree ≠ 0.

Terminology

Paths

Immediately Connected

- Two nodes are **immediately connected** if there is an edge between the two nodes.
- In the example:
  - *a* and *b*
  - *b* and *c*



Path

- A **path** is a sequence of immediately connected vertices:

  $N_i, \ldots N_j$

  N_i is immediately connected to N_{i+1}

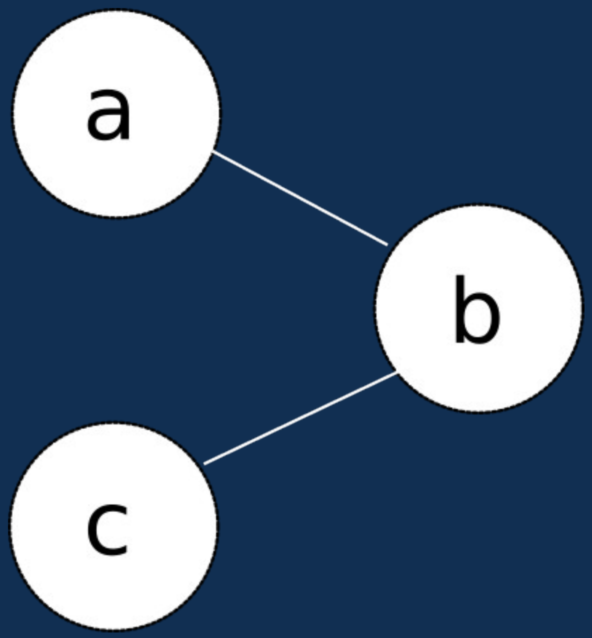
  N_{i+1} is immediately connected to N_{i+2}

  ...

  N_{j-1} is immediately connected to N_j

- In the example there is a path

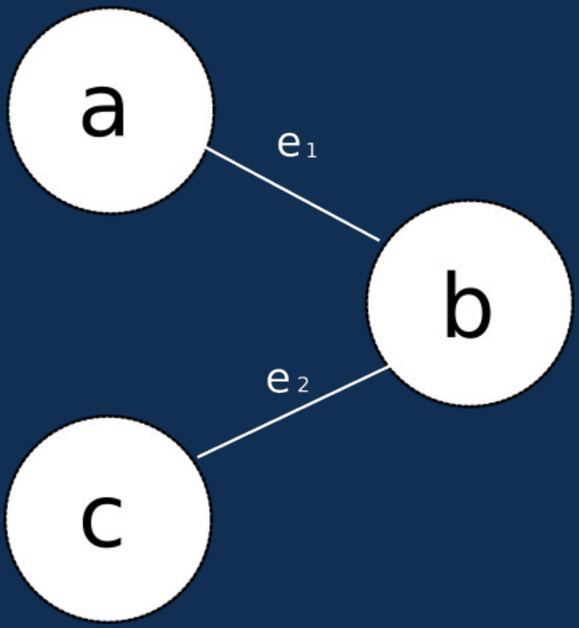
  a, b, c



Representing Paths with Edges

- A path may also be defined as a sequence of edges, such that
  - for any pair of edges, e_i and e_{i+1}
  - e_i and e_{i+1} share a common node.
- In the example, there is a path (represented by edges):

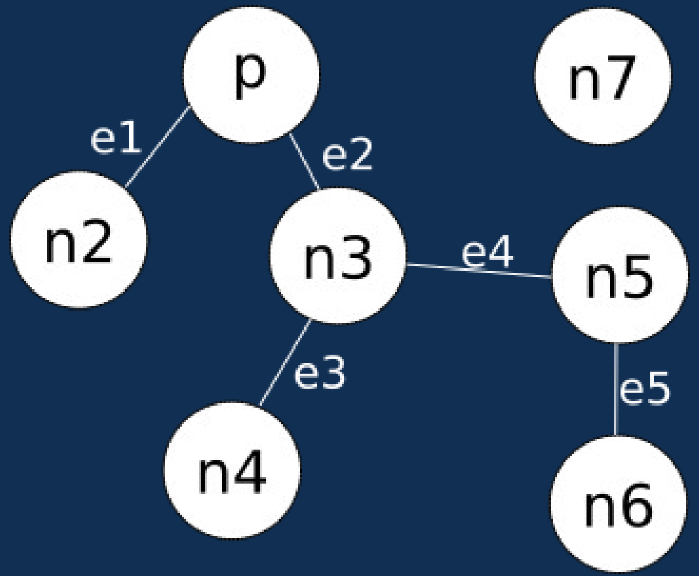
e_1, e_2



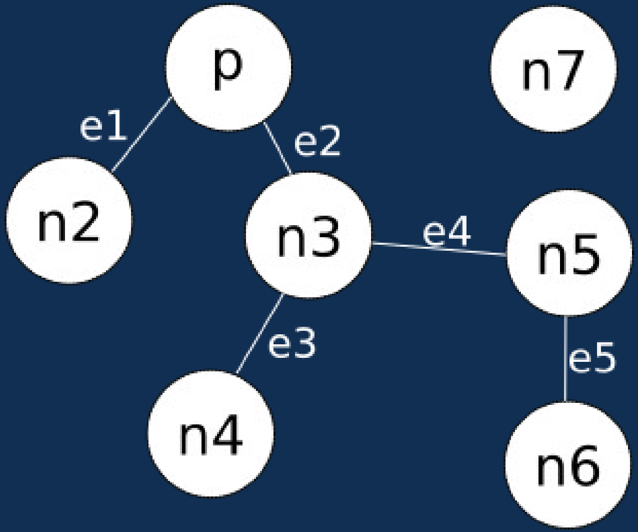
Exercise 6:

Identify a path in the graph shown. Express it as both a

1. sequence of vertices
2. sequence of edges



Example Paths



There are several paths in the example graph.

- Path 1: n2, p, n3, n5
- Path 2: p, n3, n4

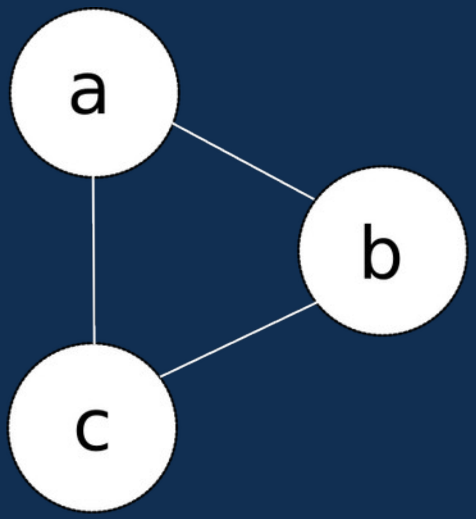
- Or represented with edges:
  - Path 1: e1, e2, e3
  - Path 2: e2, e3

Cycle

- A cycle is a path whose first and last vertices are the same
- The path

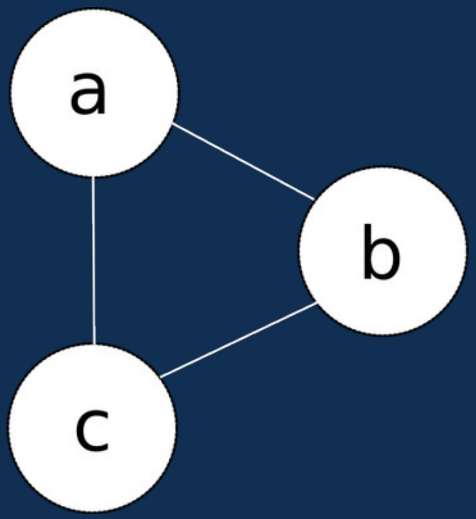
  a, b, c, a

  is a cycle
- A graph that contains one or more cycles is called **cyclic**
  - Otherwise it's called **acyclic**



Simple Paths and Cycles

- A path (or cycle) is **simple** if it has no repeated vertices
  - Except for first and last in the case of a cycle
- A simple cycle:
  - a, b, c, a
- A cycle that is not simple:
  - a, b, c, a



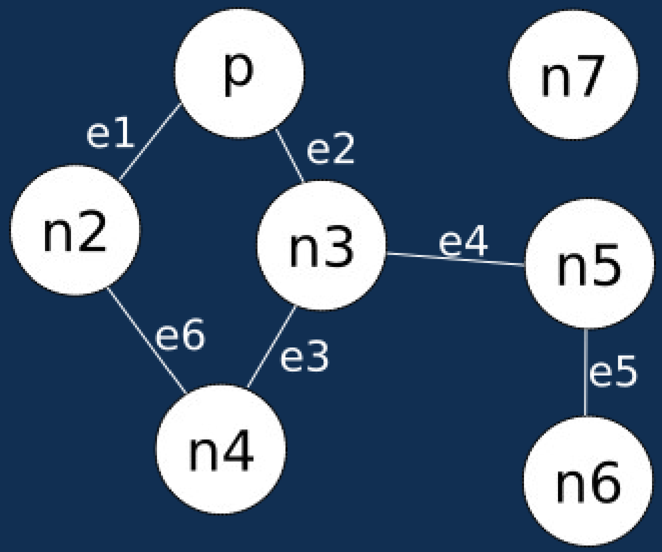
Path Length

- The **length** of a path (or cycle) is the number of edges

Path Length

Exercise 7:

1. Identify a simple path of length 6
2. Identify a simple cycle
3. Is it possible to connect every node with one path?

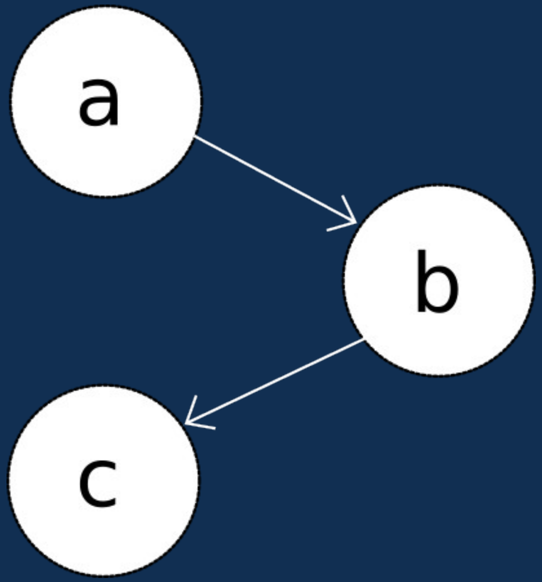


Directed Paths

- A sequence of vertices connected by directed edges is a **directed path**
  - A directed path:

    a b c
  - A path that is **not** directed

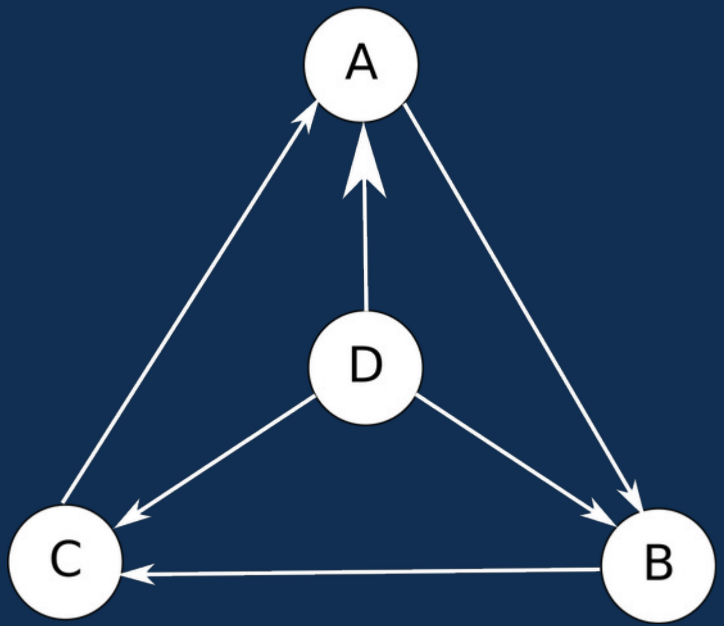
    c b a



Exercise 8:

List all the directed simple paths in this graph

Identify which ones are cycles.

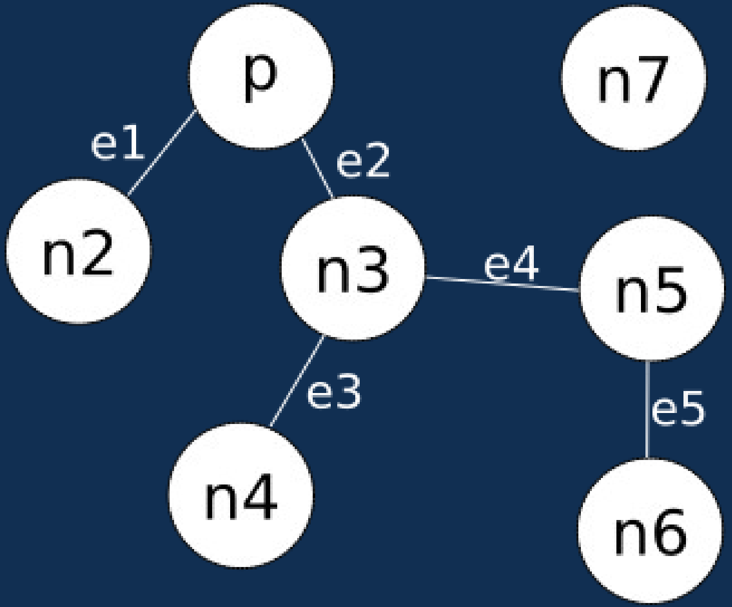


Connectedness

- Node n_i is **connected** to node n_j if there is a path from n_i to n_j
  - An undirected graph is connected if there is a path from every node to every other node
- Node n_i is **strongly connected** to node n_j if there is a directed path from n_i to n_j
  - A digraph is strongly connected if there is a directed path from every node to every other node

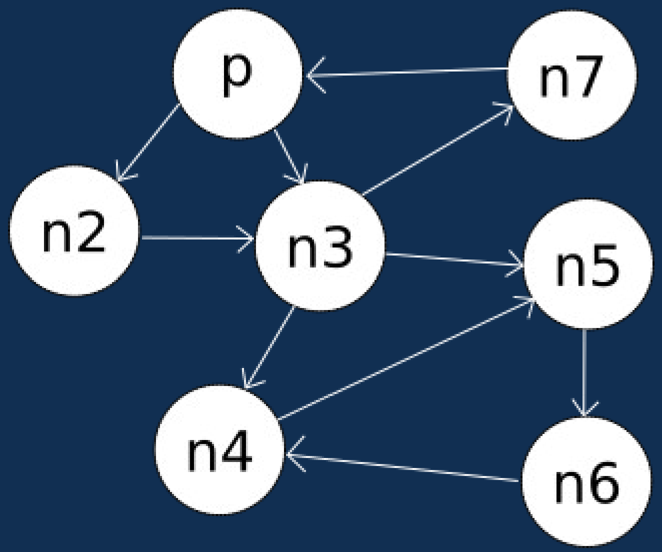
Component

- A component of a graph is defined as the maximal set of connected nodes.
  - In the example graph:
    - {p, n2, n3, n4, n5, n6} form a component
    - {n7} also forms a component



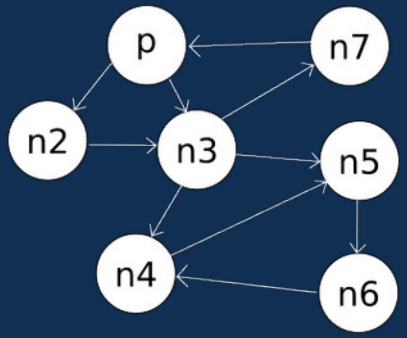
Strongly Connected Component

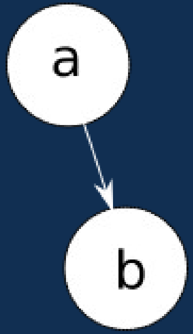
- In a digraph, a strongly connected component is maximal set of strongly connected nodes.
  - In the example graph:
    - {p, n2, n3, n7} form a strongly connected component
    - {n4, n5, n6} forms a strongly connectedcomponent



Condensation graph

- A condensation graph is formed by replacing each component in the original graph by a “condensed” single node. There are no edges.
  - Condensation graph of a graph gives us a view of the “complexity” through the number of components.
- We can consense a digraph as well, to see the relationships between its strongly connected components

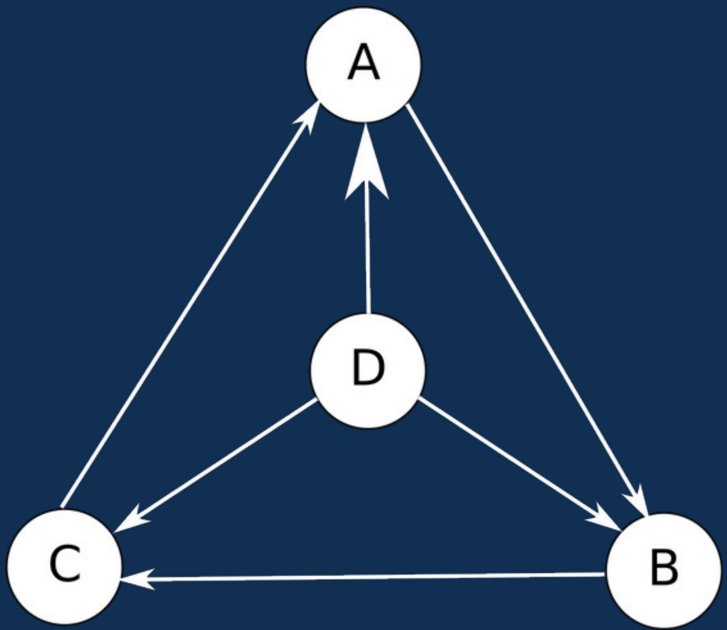




Exercise 9:

Look at the graph shown

1. Identify the strongly connected components
2. Draw its condensation graph



Terminology

- There is a lots more terminology about graphs
  - And some different ways these terms can be defined
- For this course, use the terminology and definitions given

- For example
  - Walks
  - Trails
  - Paths
  - Circuits
  - Multigraph
  - Pseudograph
  - Bouquet
  - …

Definitions – Graph Type

Type	Edges	Multiple Edges Allowed ?	Loops Allowed ?
Simple Graph	undirected	No	No
Multigraph	undirected	Yes	No
Pseudograph	undirected	Yes	Yes
Directed Graph	directed	No	Yes
Directed Multigraph	directed	Yes	Yes

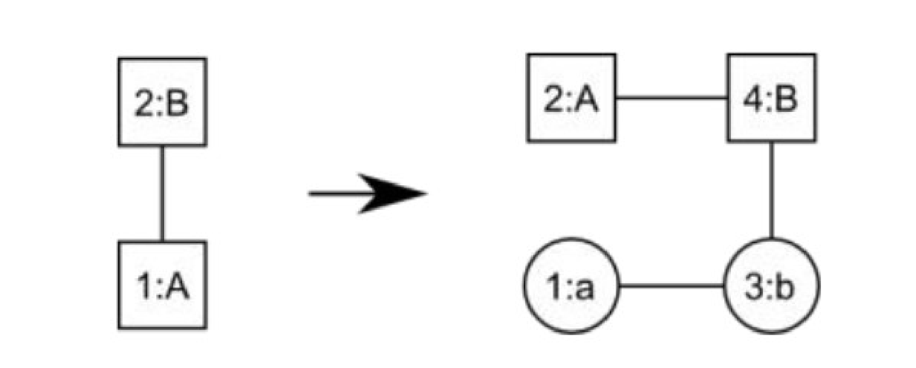
Graph Grammar

Graph Grammar

- We’ve seen formal string rewriting systems
  - The rules that defined a system were a grammar
  - These worked on strings
- We can also define grammars for graphs

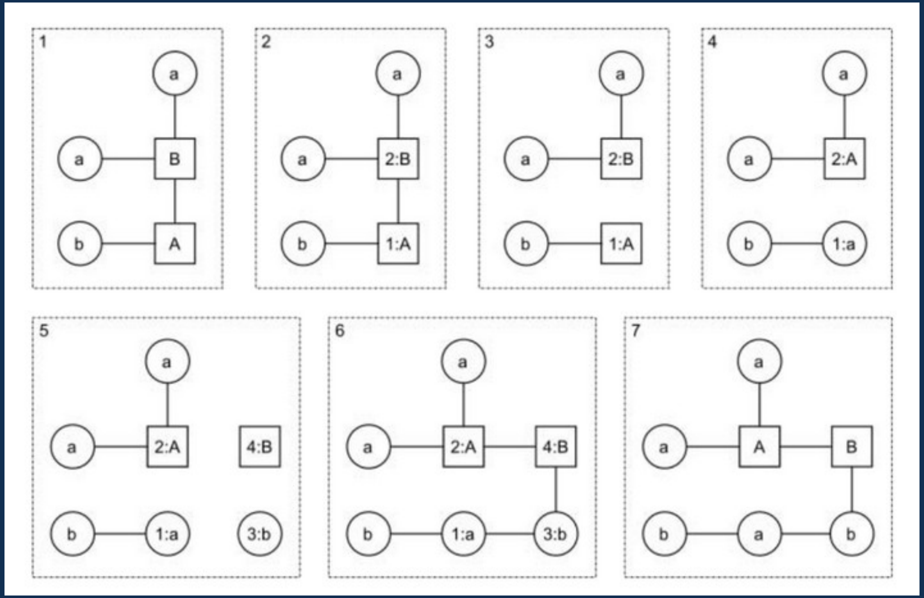
Graph Grammar Rules

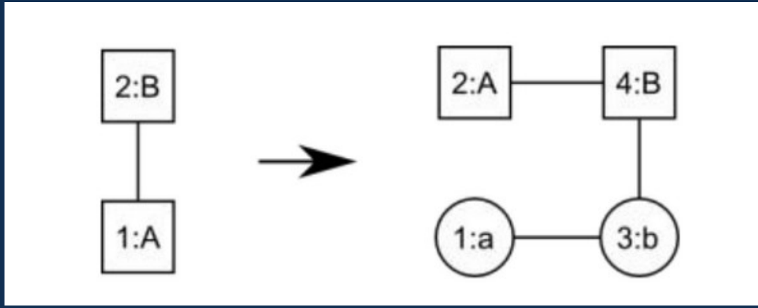
- A rule in a graph grammar transforms one graph fragment into another
  - Just like a string rewriting system transforms a symbol into another
- Here numbers identify nodes, letters are labels of node contents



Dormans, Joris. "Adventures in level design: generating missions and spaces for action adventure Games." Proceedings of the 2010 workshop on procedural content generation in games. 2010.

Applying the rule

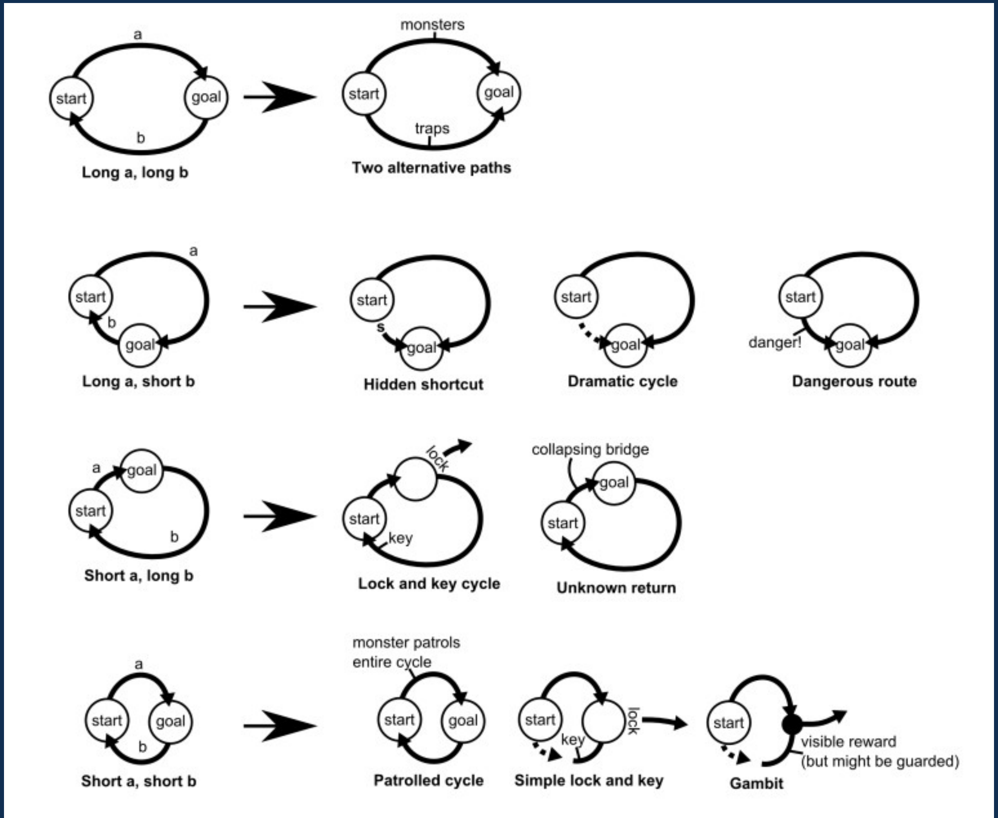




Dormans, Joris. "Adventures in level design: generating missions and spaces for action adventure Games." Proceedings of the 2010 workshop on procedural content generation in games. 2010.

Design Patterns in Unexplored 2

- Game level design patterns using cyclic graphs
  - Here each edge is marked either “short” or “long”
- Used in *Unexplored 2* by Ludomotion



https://ctrl500.com/tech/handcrafted-feel-dungeon-generation-unexplored-explores-cyclic-dungeon-generation/

Exercise 10:

Devise a graph grammar to construct simple levels for a game like the Legend of Zelda

Start by including some of the design patterns shown opposite

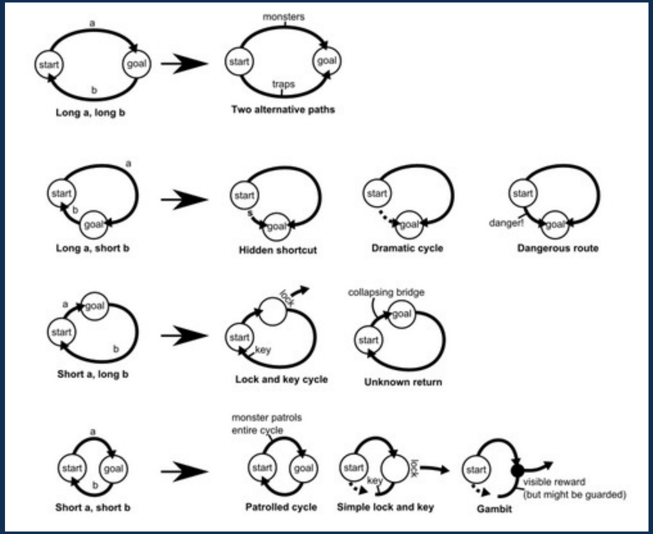
Try out your rules as you go to see what they make

Dormans, Joris. "Adventures in level design: Generating missions and spaces for action Adventure Games." Proceedings of the 2010 workshop on procedural content generation in games. 2010.

- Hint:

  Search for the title of the paper below on Google Scholar

  - Look at section 4 in detail



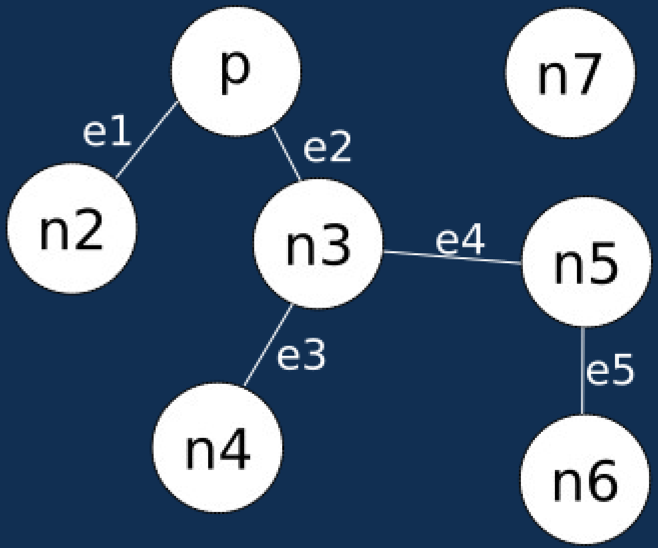
Representation

Representing a Graph

- When working with a graph mathematically, we want a formal representation
  - Allows us to express properties about our graph using the tools of Set Theory, Logic, etc.
  - Allows generalisation over possible graphs
  - Allows us to describe algorithms on graphs
- Different representations can express different sorts of graph

Set-theoretic representation

- A simple graph can be represented as an ordered pair G = (V, E) composed of :
  - a set of vertices, V, and
  - a set of edges, E, which connects the nodes.
- For the example
  - V = { p, n2, n3, n4, n5, n6, n7 }
  - E = {e1, e2, e3, e4, e5 }



Representing Vertices

- Vertices are usually represented as labels if their content doesn't matter
  - V = { a, b, c }
- We could also represent the data of each node, for example with numbers:
  - V = {2, 54, 25}
  - Or define the labels above, e.g. a = 2

Question:

The following set of vertices is not allowed, why? How could we fix it?

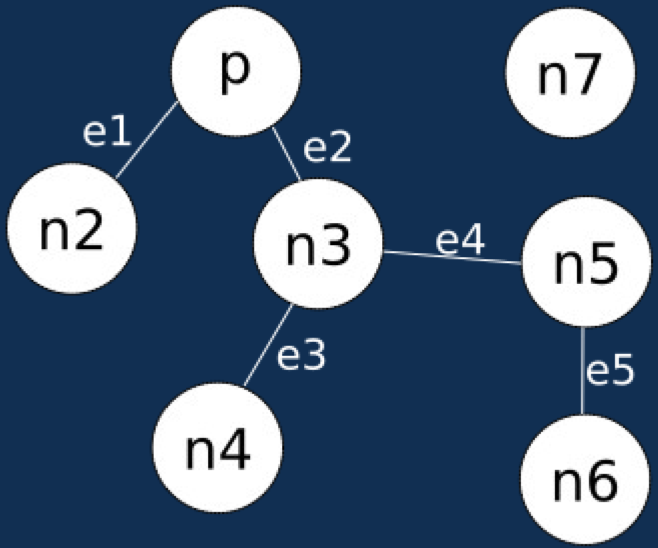
- V = { 12, 42, 42 }

Representing Edges

- There are different ways of representing edges
- Edges are commonly represented as sets or tuples
  - E1 = {n2, p}
  - E1 = (n2, p)

Question:

What difference does this make?



Exercise 11:

1. Draw the **undirected graph** $G_1 = (V_1, E_1)$ where

   $V_1 = \{ a, b, c, d \}$

   $E_1 = \{\{a, b\}, \{a, c\}, \{a, d\} \}$

2. Draw the **directed graph** $G_2 = (V_2, E_2)$ where

   $V_2 = \{ a, b, c, d \}$

   $E_2 = \{(a, b), (a, c), (a, d) \}$

Defining Edges

- So far we've seen an extensional definition of the set of edges
  - We can also describe it as follows

$$E \subseteq \{ \{x, y\} \mid x, y \in V \;\; \wedge \;\; x \neq y \}$$

- Or if you prefer, we could express the set comprehension with a term

$$E \subseteq \{ x\colon V;\, y\colon V \mid x \neq y \bullet \{x, y\} \}$$

- And if we want directed edges, we just change the term to a tuple

$$E \subseteq \{ x\colon V;\, y\colon V \mid x \neq y \bullet (x, y) \}$$

Representation

Multigraphs

Exercise 12:

Try to give an extensional definition for the graph shown. What problem do you encounter?

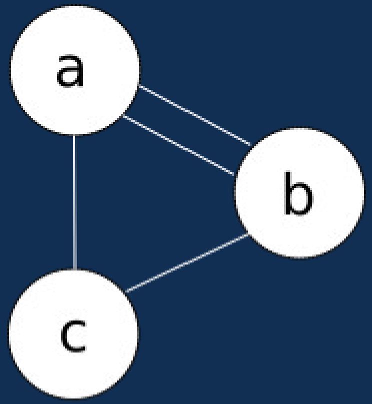
Hint:

- An extensional definition is listing all the elements of a set

$$G = (V, E)$$

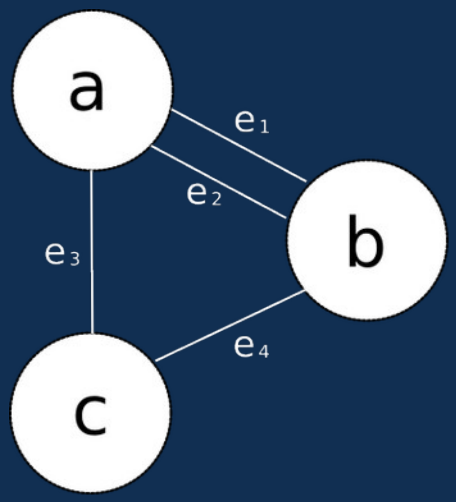
$$V = \{ \dots \}$$

$$E = \{ \dots \}$$



Multiple Edges (Multigraph)

- With a graph (V, E) there is no way to represent multiple edges
  - We need to change our representation
- Let's invent a function, ϕ, that tells us, for each edge, what vertices it joins
  - $G = (V, E, \phi)$
- For example,
  - $\phi(e_1) = \{a, b\}$
  - $\phi(e_2) = \{a, b\}$



Multigraphs

- We can define the *type* of the function ϕ.
  - Functions map inputs in a domain (edges) to a range (2-sets of vertices)

$$\phi : \mathrm{E} \rightarrow \mathrm{P}$$

$$\textit{where}\ \mathrm{P} = \{\, \{\, x, y \,\} \mid x, y \in \mathrm{V} \ \wedge\ x \neq y \,\}$$

- Here P is the set of sets {x, y}, where x and y are
  - vertices in our graph ($\in$ V)
  - not equal to one another

Functions in Set Theory

- Function, like everything else in Set Theory, are sets
- A function is a set of pairs (A, B), such that
  - A is an element of the domain
  - B is an element of the range
- For example the successor function, $S : \mathbb{N} \rightarrow \mathbb{N}$, is the set of pairs
  - $S = \{(1, 2), (2, 3), (3, 4), \ldots, (n, n+1), \ldots\}$

Functions in Set Theory

Exercise 13:

Let *f* be a function of type $\mathbb{N}_4 \rightarrow \mathbb{N}_4$

$$f(x) = x^2 \bmod 4$$

1. Give an extensional definition of *f*

Hint:

- $\mathbb{N}_4 = \{0, 1, 2, 3\}$
- Functions in set theory are sets of pairs
  - $\{ (\text{input}_1, \text{output}_1), (\text{input}_2, \text{output}_2) \ldots \}$
- An extensional definition lists all the elements

Multigraph

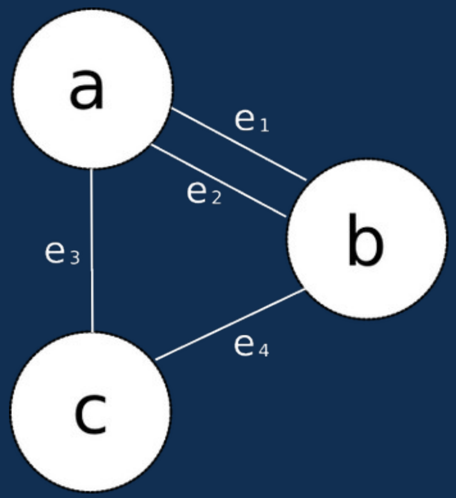
- We now have the tools to give an extensional definition of our multigraph

$$G = (V, E, \phi)$$

$$V = \{a, b, c\}$$

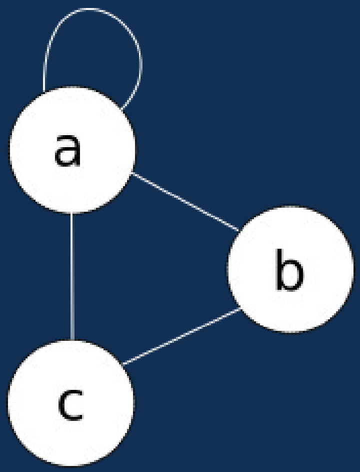
$$E = \{e_1, e_2, e_3, e_4\}$$

$$\phi = \{(e_1, \{a, b\}), (e_2, \{a, b\}), (e_3, \{a, c\}), (e_4, \{b, c\})$$



Exercise 14:

Try to give a formal representation for the graph shown. What problem do you encounter?



Hint:

- It's a simple graph (not a multigraph) so we don't need ϕ here

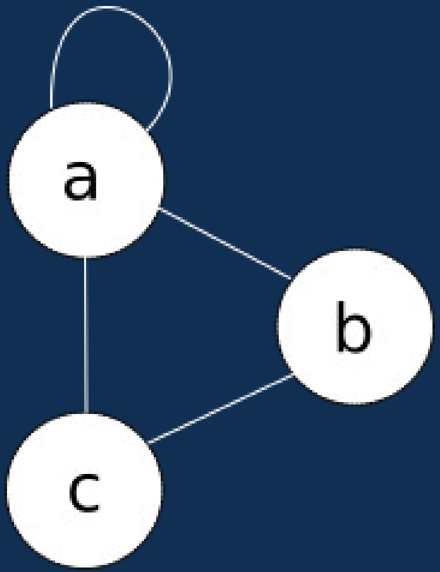
$$G = (V, E)$$

$$V = \{ \dots \}$$

$$E = \{ \dots \}$$

Loops

- Representing looping edges
  - Can't use a 2-set, as all elements in a set are unique
    - { a, a }
  - Don't want to use a tuple, as it's unordered
    - (a, a)
- We have to change our representations to allow edges to be singletons { a } representing a loop



Edges that Permit Loops

- To permit loops we need to drop the requirement that $x \neq y$

$$E \subseteq \{ \{x, y\} \mid x, y \in V \}$$

- Or using a term...

$$E \subseteq \{ x: V; y: V \bullet \{x, y\} \}$$

Exercise 15:

Allowing loops in a *multigraph* means we need to change our definition of ϕ. What does it need to become?

Hint:

- An undirected multigraph has a function ϕ of type

$$\phi : E \rightarrow \{ \{ x, y \} \mid x, y \in V \;\wedge\; x \neq y \}$$

Directed Multigraph Permitting Loops

- If we want a *directed* graph that
  - allow loops
  - allows multiple edges between nodes
- We can adapt our previous definition to that shown on the right

$$G = (V, E, \phi)$$

$$V = \{ \dots \}$$

$$E = \{ \dots \}$$

$$\phi : E \rightarrow \{ (x, y) \mid x, y \in V \}$$

Exercise 16:

Give an extensional definition of the graph opposite

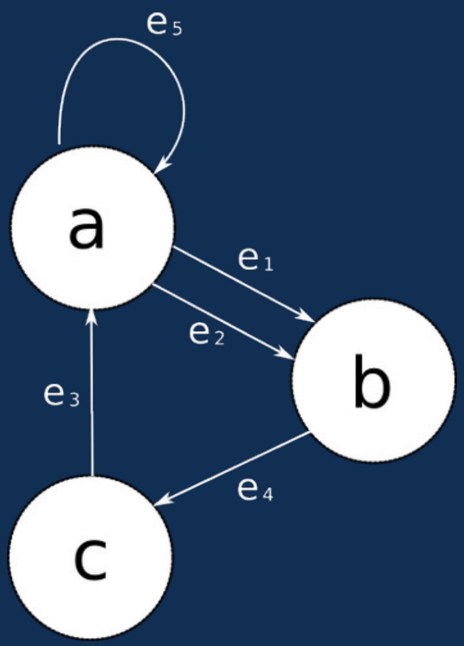
Hint:

$$G = (V, E, \phi)$$

$$V = \{ a, \ldots \}$$

$$E = \{ e_1, \ldots \}$$

$$\phi = \{ (e_1, (a, b)), \ldots \}$$



Graphs and Set Comprehensions

Exercise 17:

1. Draw a graph of the relationships between users, orders, and items
2. Intensionally define the set of edges connecting Items to Items (which have a offer price associated with them)
3. Intensionally define the set of edges connecting Users to Orders

User ID	Name	Address	Orders
5	John	1 A St	{13}
6	Mary	2 B St	{14,15}

Order ID	User	Shipping Cost	Items Cost	Items
13	5	4.00	22.50	{3, 4}
14	6	4.00	22.50	{3, 4}
15	6	2.55	4.50	{4}

Item ID	Name	Price	Offer Items
3	Shoe	20.00	{ (4, 2.50) }
4	Laces	4.50	{ }

Summary

Summary

- What is a Graph
- Terminology
  - Node/vertex, edge
  - (Strongly) Connected, Directed (digraph), (In/Out)degree,
  - (Simple) Path, Cycle,
  - Multigraph, Loops
- Representing Graphs
  - Set theoretic definitions