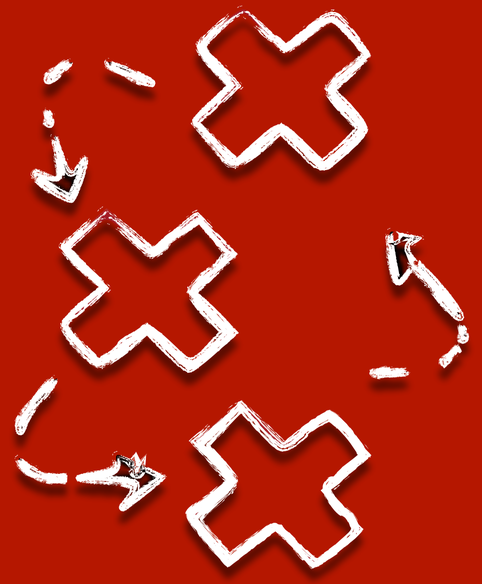


Causal Data Science

Lecture 3.1: Graphical models

Lecturer: Sara Magliacane

UvA - Spring 2024

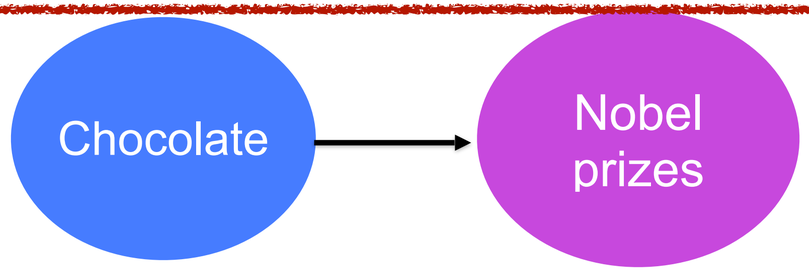
From the introduction: A working definition of causality

Informal definition: A variable X causes another variable Y, if changing (the distribution of) X, e.g. by fixing its value, changes (the distribution of) Y

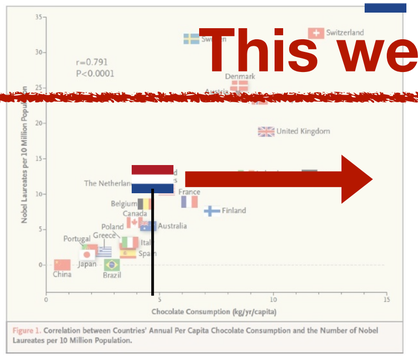
Intervention

Challenge: estimate the causal effect of an intervention, when we do not have (all possible) interventional data **(e.g. observational data)**

Representation: We can represent causal relations in **causal graphs:** nodes are random variables, edges causal relations



This week

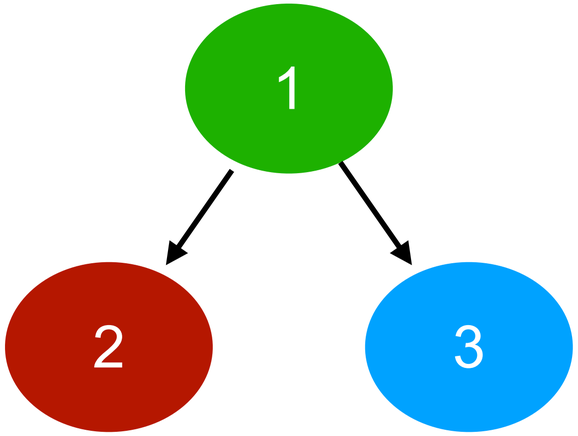


Today: graphs and random variables

- We can represent a (factorisation of) joint probability as a **graph**

- **Each node $i \in \mathbf{V}$** represents a **random variable X_i**
  - For $\mathbf{A} \subseteq \mathbf{V}$, we can define $X_{\mathbf{A}} := \{X_i : i \in \mathbf{A}\}$

- **Edges** represent relationships between variables (*it will be clearer later*)



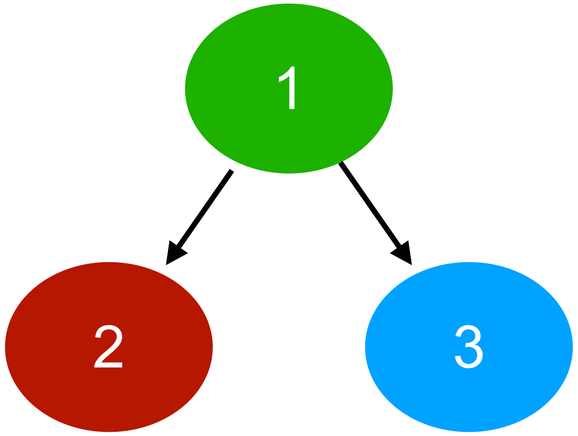
Today: graphs and random variables

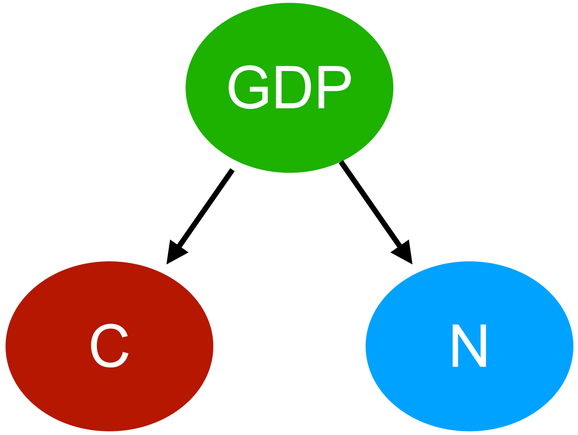
- We can represent a (factorisation of) joint probability as a **graph**

[Sometimes: slight abuse of notation we call both the same]

- **Each node $i \in \mathbf{V}$** represents a **random variable X_i**
  - For $\mathbf{A} \subseteq \mathbf{V}$, we can define $X_{\mathbf{A}} := \{X_i : i \in \mathbf{A}\}$

- **Edges** represent relationships between variables (*it will be clearer later*)





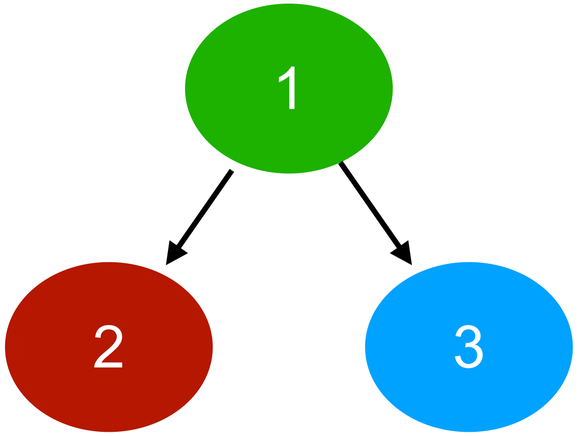
Today: graphs and random variables

- We can represent a (factorisation of) joint probability as a **graph**

[Sometimes: slight abuse of notation we call both the same]

- **Each node $i \in \mathbf{V}$** represents a **random variable X_i**
  - For $\mathbf{A} \subseteq \mathbf{V}$, we can define $X_{\mathbf{A}} := \{X_i : i \in \mathbf{A}\}$

- **Edges** represent **relationships** between variables (*it will be clearer later*)

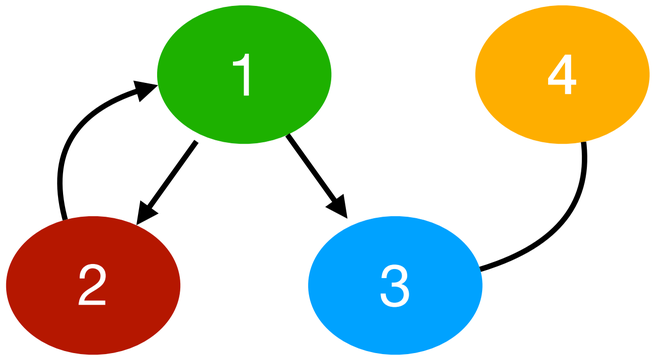


In this class (Mon) the relationship not necessarily causal, in next class (Thu) we will show when the graph is causal

Graph terminology

Graph terminology

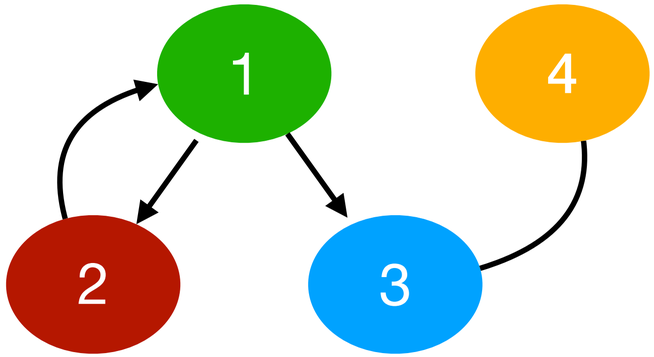
- A graph G is a tuple $G = (\mathbf{V}, \mathbf{E})$:
  - $\mathbf{V}$ is the set of **nodes** (vertices)
  - $\mathbf{E}$ is the set of **edges** between two nodes, i.e. $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V}$



Graph terminology

- A graph G is a tuple $G = (\mathbf{V}, \mathbf{E})$:
  - $\mathbf{V}$ is the set of **nodes** (vertices)
  - $\mathbf{E}$ is the set of **edges** between two nodes, i.e. $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V}$

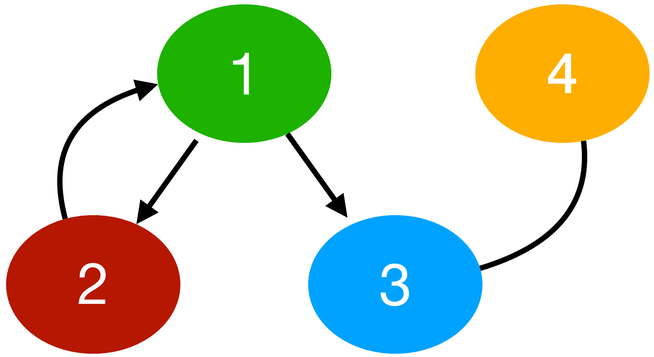
    $\Longrightarrow$ only one edge between an ordered pair of nodes



Graph terminology

- A graph G is a tuple $G = (\mathbf{V}, \mathbf{E})$:
  - $\mathbf{V}$ is the set of **nodes** (vertices)
  - $\mathbf{E}$ is the set of **edges** between two nodes, i.e. $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V}$

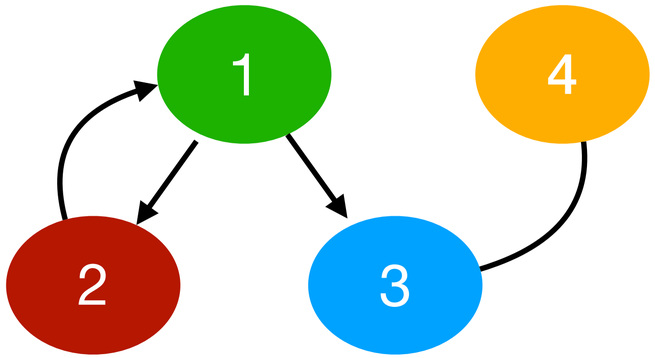
    $\Longrightarrow$ only one edge between an ordered pair of nodes

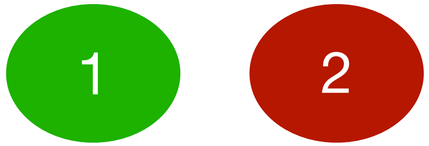


  - Two nodes connected by an edge are **adjacent**

Graph terminology: paths

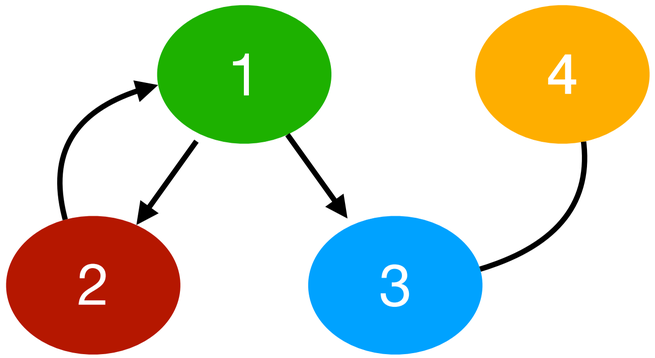
- A **path** between **node i and node j** is a sequence of **distinct nodes** $(i, \ldots, j)$ such that each two **consecutive nodes** are **adjacent**

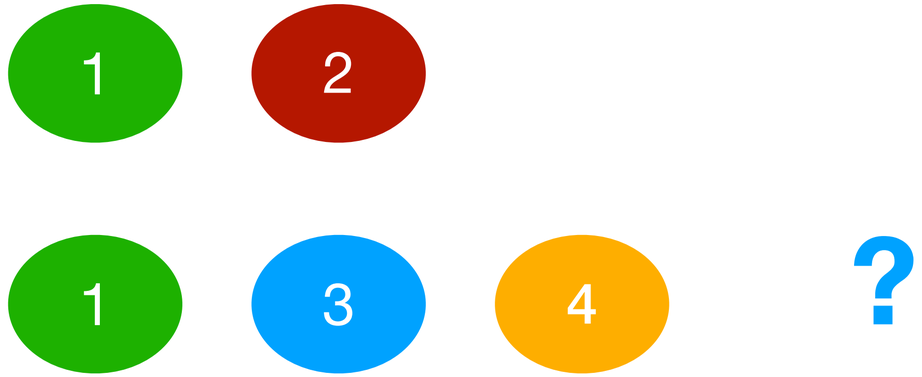




Graph terminology: paths

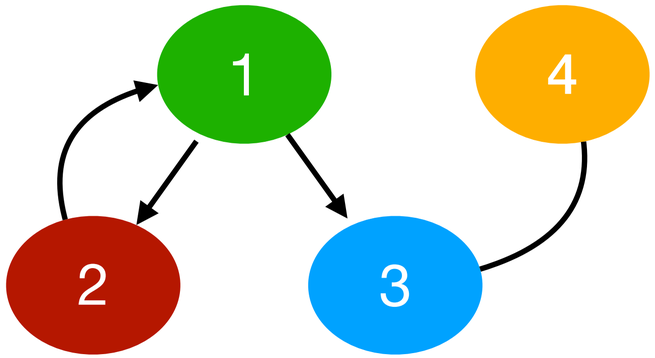
- A **path** between **node i and node j** is a sequence of **distinct nodes** $(i, \ldots, j)$ such that each two **consecutive nodes** are **adjacent**

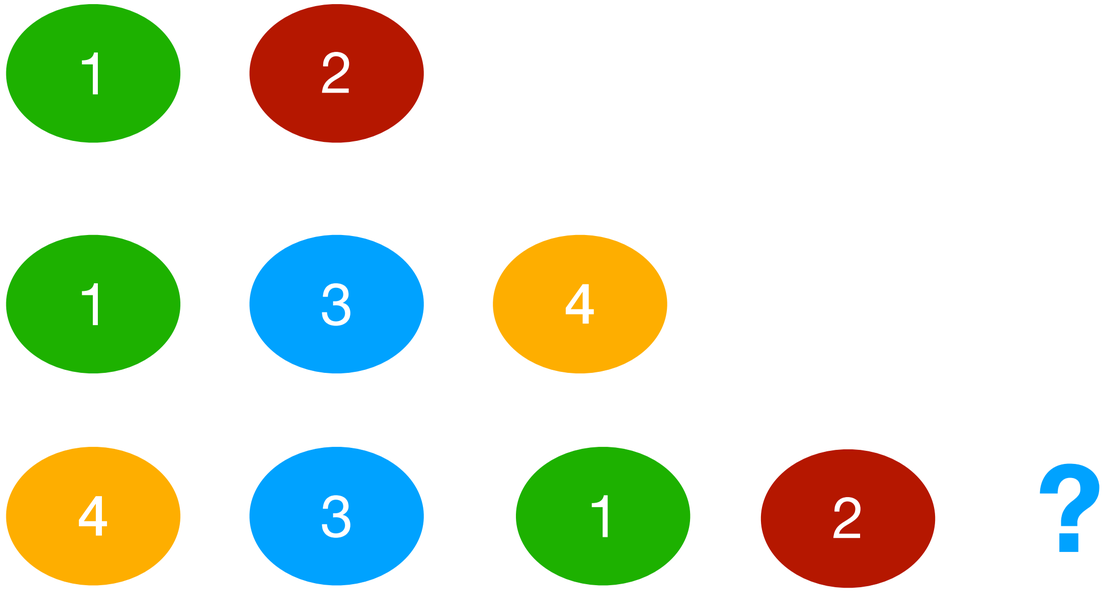




Graph terminology: paths

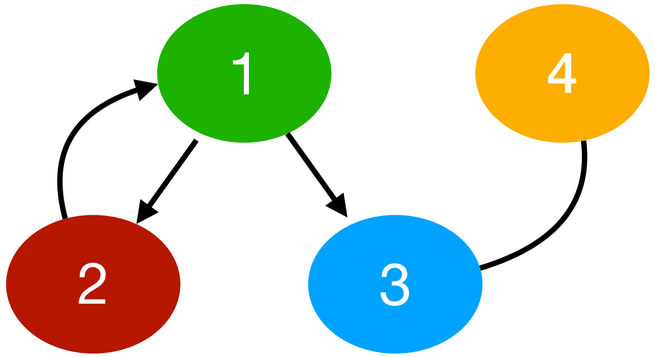
- A **path** between **node i and node j** is a sequence of **distinct nodes** $(i, \ldots, j)$ such that each two **consecutive nodes** are **adjacent**





Graph terminology: paths

- A **path** between **node i and node j** is a sequence of **distinct nodes** $(i, \ldots, j)$ such that each two **consecutive nodes** are **adjacent**

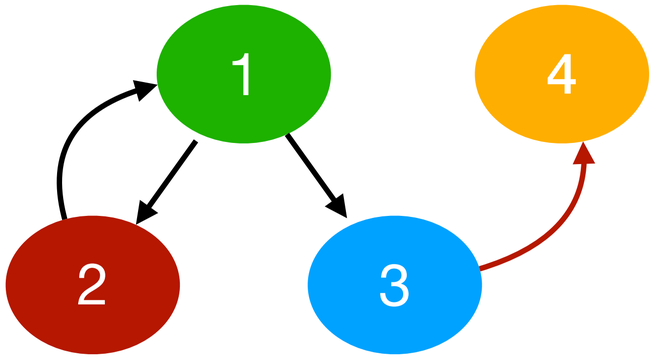


Directed graphs vs mixed graphs

- A graph G is a tuple $G = (\mathbf{V}, \mathbf{E})$:
  - $\mathbf{V}$ is the set of **nodes** (vertices)
  - $\mathbf{E}$ is the set of **edges** between two nodes, i.e. $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V}$
  - If all edges are **directed** $\rightarrow$ then the graph is **directed**

Directed graphs vs mixed graphs

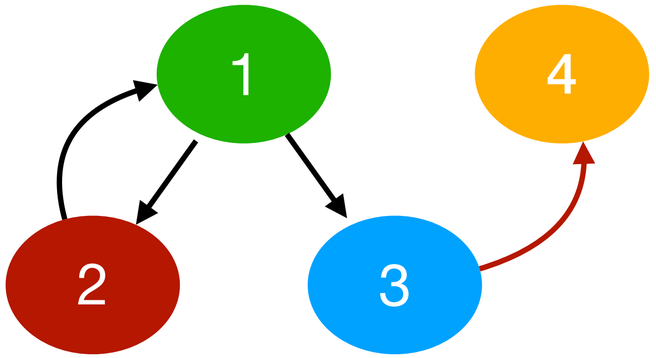
- A graph G is a tuple $G = (\mathbf{V}, \mathbf{E})$:
  - $\mathbf{V}$ is the set of **nodes** (vertices)
  - $\mathbf{E}$ is the set of **edges** between two nodes, i.e. $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V}$
  - If all edges are **directed** $\rightarrow$ then the graph is **directed**



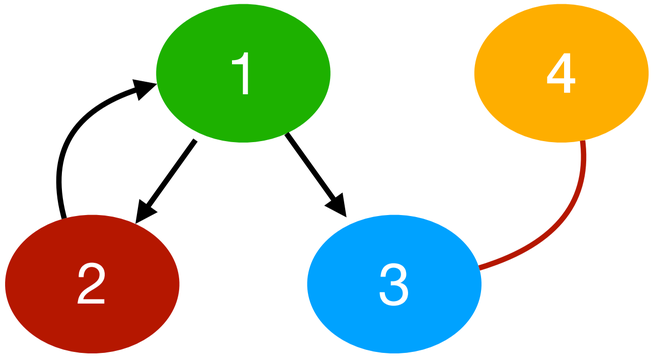
Directed graph

Directed graphs vs mixed graphs

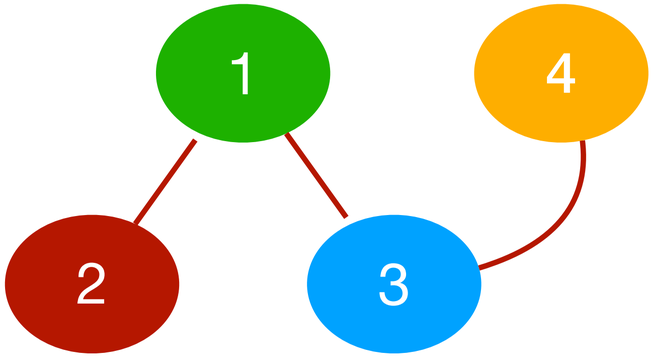
- A graph G is a tuple $G = (\mathbf{V}, \mathbf{E})$:
  - $\mathbf{V}$ is the set of **nodes** (vertices)
  - $\mathbf{E}$ is the set of **edges** between two nodes, i.e. $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V}$
  - If all edges are **directed** $\rightarrow$ then the graph is **directed**



Directed graph



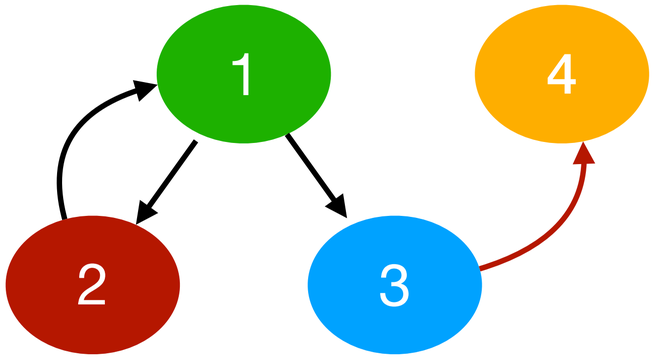
Mixed graph



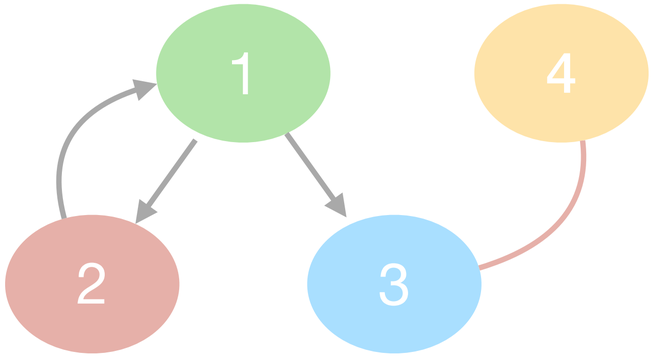
Undirected graph

Directed graphs vs mixed graphs

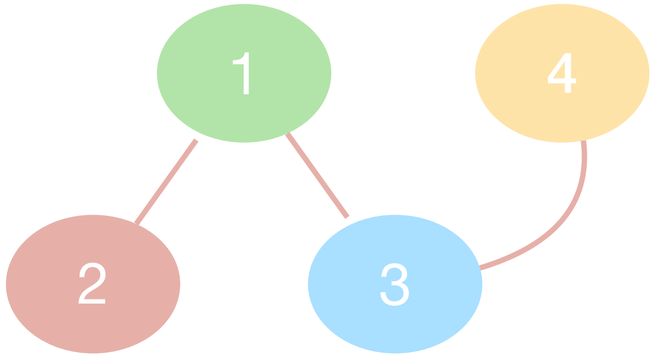
- A graph G is a tuple $G = (\mathbf{V}, \mathbf{E})$:
  - $\mathbf{V}$ is the set of **nodes** (vertices)
  - $\mathbf{E}$ is the set of **edges** between two nodes, i.e. $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V}$
  - If all edges are **directed** $\rightarrow$ then the graph is **directed**



Directed graph



Mixed graph



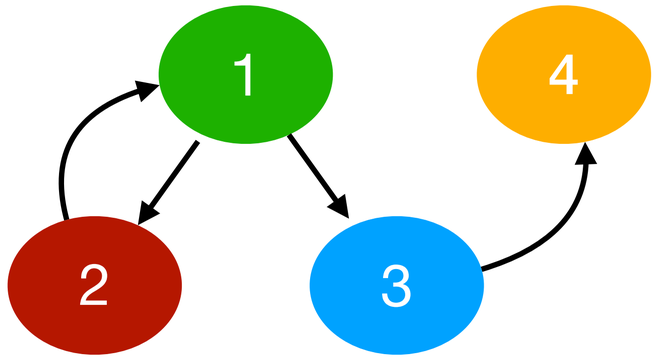
Undirected graph

Directed graphs: paths vs directed paths

- A **path** between **node i and node j** is a sequence of **distinct nodes** $(i, \ldots, j)$ such that each two **consecutive nodes** are **adjacent**
- A **directed path** between **node i and node j** is a path where **all edges point towards j**, i.e. $i \rightarrow \ldots \rightarrow j$

Directed graphs: paths vs directed paths

- A **path** between **node i and node j** is a sequence of **distinct nodes** $(i, \ldots, j)$ such that each two **consecutive nodes** are **adjacent**
- A **directed path** between **node i and node j** is a path where **all edges point towards j**, i.e. $i \rightarrow \ldots \rightarrow j$



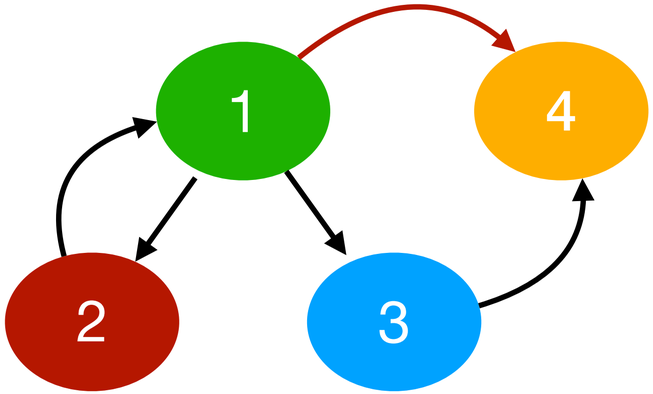
Directed path from 1 to 4

A path from 4 to 1, but not a directed path

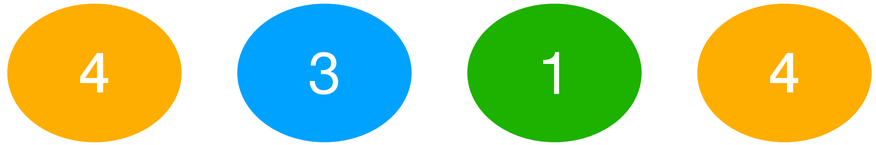
Cycles and directed cycles

- A **cycle** is a path $(i, \ldots, i)$ (could also be a **self-cycle**)*
- A **directed cycle** is a directed path $(i, \ldots, i)$ *

*** (with the exception of the endpoints)**



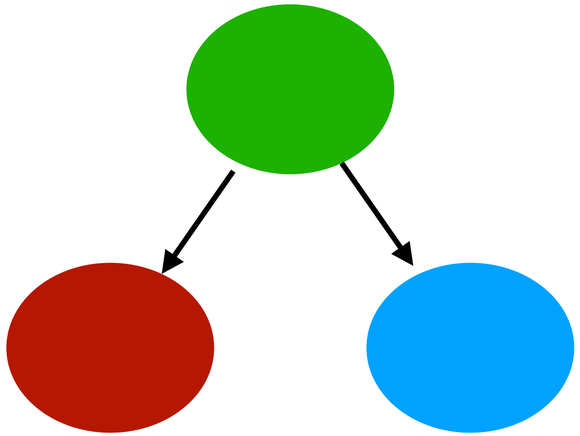
Directed cycle

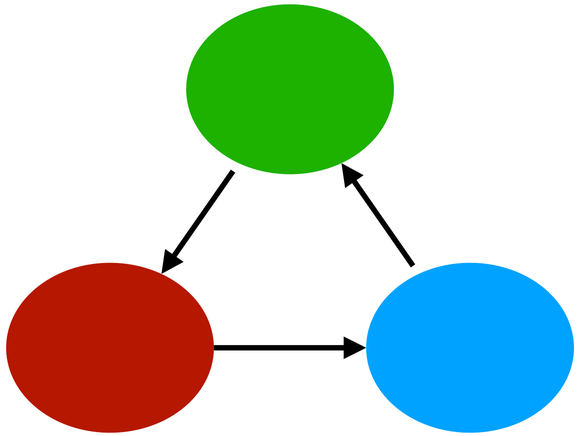


Cycle, but not directed cycle

Directed Acyclic Graphs (DAGs)

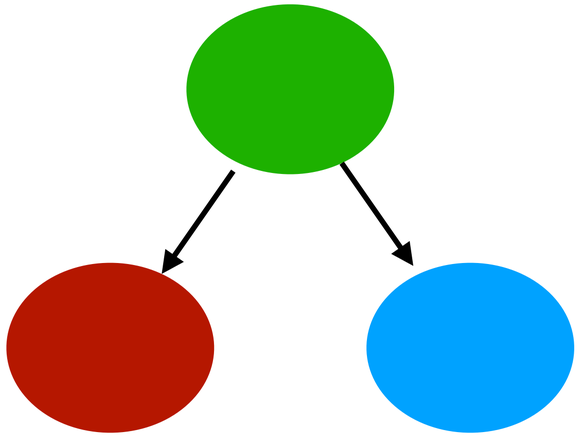
- A DAG is a directed graph $G = (V, E)$:
  - V is the set of **nodes** (vertices)
  - E is the set of **directed edges** between the nodes
  - There are **no directed cycles**

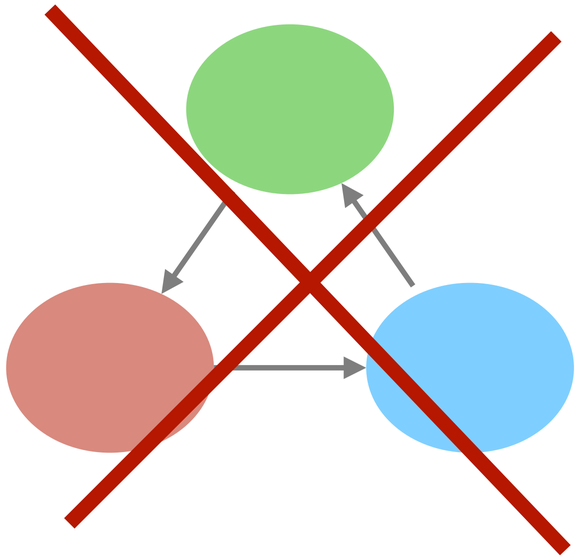




Directed Acyclic Graphs (DAGs)

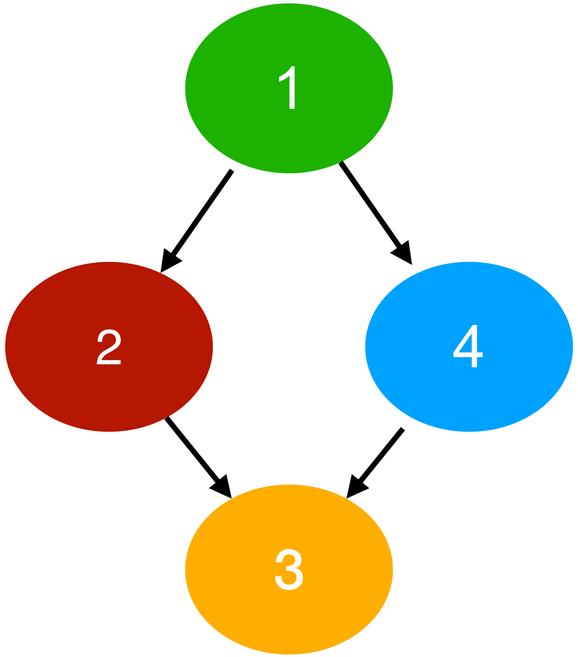
- A DAG is a directed graph $G = (V, E)$:
  - V is the set of **nodes** (vertices)
  - E is the set of **directed edges** between the nodes
  - There are **no directed cycles**





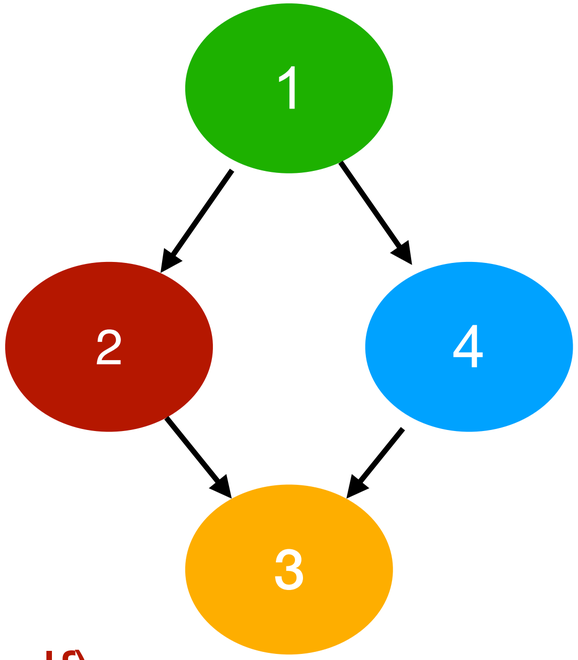
Relationships between nodes in a DAG

- **Parents** of a node $\mathrm{Pa}_G(V)$
  - Nodes that have an edge pointing to V
- **Children** of a node $\mathrm{Ch}_G(V)$
  - Nodes that have an edge pointing from V



Relationships between nodes in a DAG

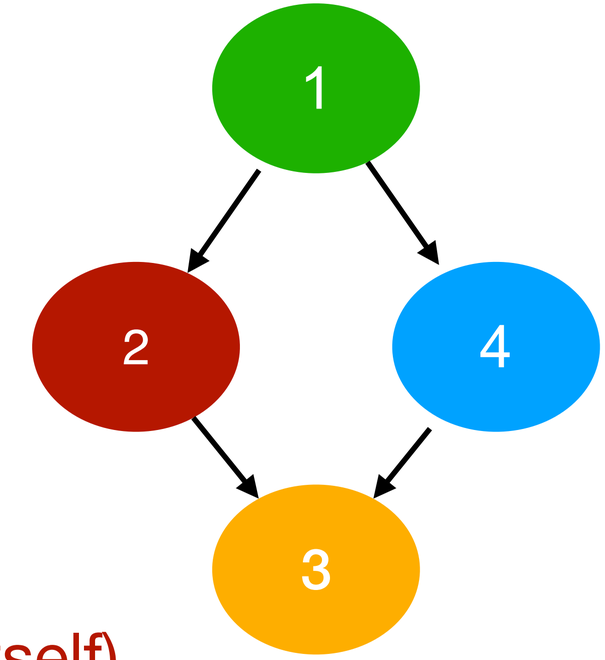
- **Parents** of a node $\mathrm{Pa}_G(V)$
  - Nodes that have an edge pointing to V
- **Children** of a node $\mathrm{Ch}_G(V)$
  - Nodes that have an edge pointing from V
- **Ancestors** of a node $\mathrm{An}_G(V)$
  - Nodes that have a **directed path** to V (including V itself)
- **Descendants** of a node $\mathrm{Desc}_G(V)$
  - Nodes that are reached from V via **directed paths** (including V itself)



Relationships between nodes in a DAG

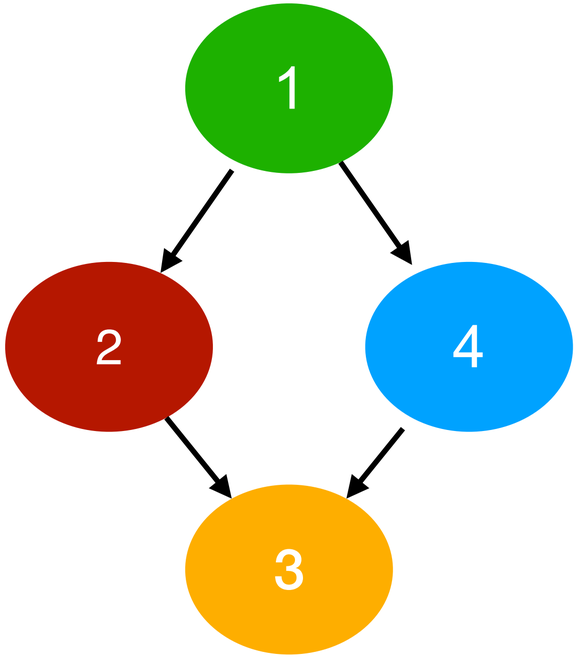
[We omit G, when it is clear from the context]

- **Parents** of a node $\mathrm{Pa}(V)$
  - Nodes that have an edge pointing to V
- **Children** of a node $\mathrm{Ch}(V)$
  - Nodes that have an edge pointing from V
- **Ancestors** of a node $\mathrm{An}(V)$
  - Nodes that have a **directed path** to V (including V itself)
- **Descendants** of a node $\mathrm{Desc}(V)$
  - Nodes that are reached from V via **directed paths** (including V itself)



Kinship relationships for sets of nodes

- We will use **bold** for sets (including sets of nodes)



- **Parents** of a set of nodes $\mathbf{A} \subseteq \mathbf{V}$:

$$\mathrm{Pa}(\mathbf{A}) := \bigcup_{V \in \mathbf{A}} \mathrm{Pa}(V)$$

$$\mathrm{Pa}(\{2,3\}) = \mathrm{Pa}(2) \cup \mathrm{Pa}(3) = \{1\} \cup \{2,4\} = \{1,2,4\}$$

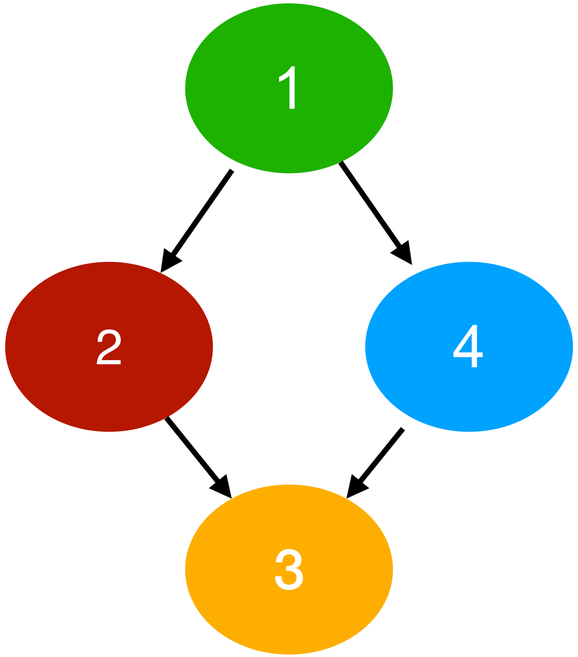
- Similarly for children, ancestors and descendants

Descendants and non-descendants

- **Descendants** of a set of nodes $\mathbf{A} \subseteq \mathbf{V}$:

$$\text{Desc}(\mathbf{A}) := \bigcup_{V \in \mathbf{A}} \text{Desc}(V)$$

$$\text{Desc}(\{2,3\}) = \text{Desc}(2) \cup \text{Desc}(3) = \{2,3\} \cup \{3\} = \{2,3\}$$



Descendants and non-descendants

- **Descendants** of a set of nodes $\mathbf{A} \subseteq \mathbf{V}$:

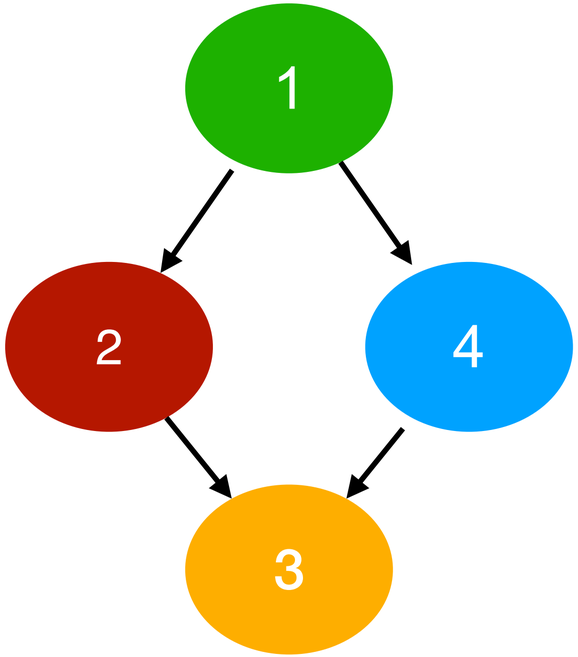
$$\text{Desc}(\mathbf{A}) := \bigcup_{V \in \mathbf{A}} \text{Desc}(V)$$

$$\text{Desc}(\{2,3\}) = \text{Desc}(2) \cup \text{Desc}(3) = \{2,3\} \cup \{3\} = \{2,3\}$$

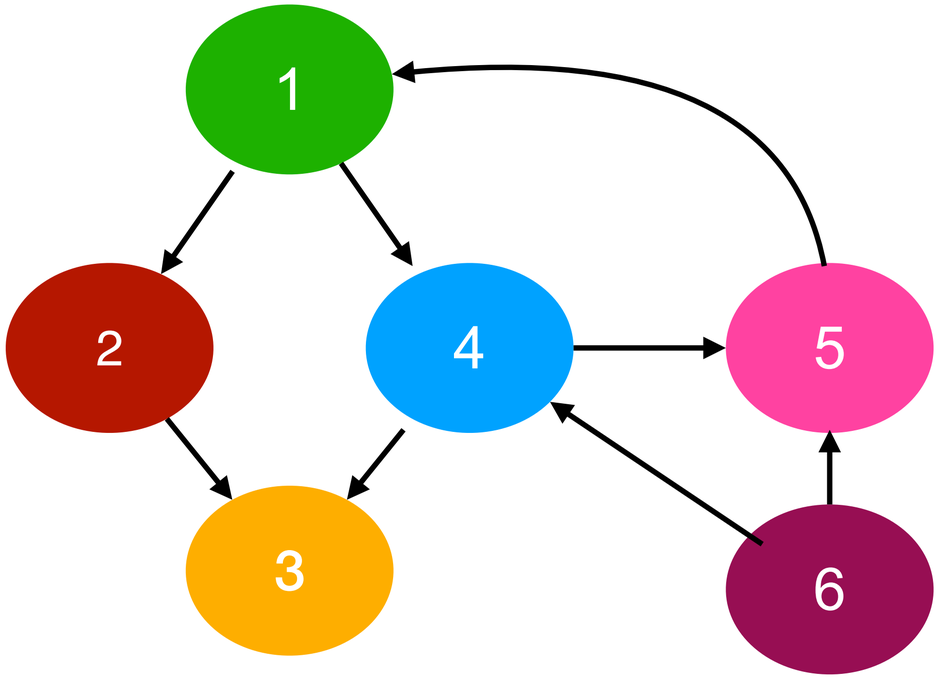
- **Non-descendants** of a set of nodes $\mathbf{A} \subseteq \mathbf{V}$:

$$\text{Nondesc}(\mathbf{A}) := \mathbf{V} \backslash \text{Desc}(\mathbf{A})$$

$$\text{Nondesc}(\{2,3\}) = \mathbf{V} \backslash \{2,3\} = \{1,4\}$$

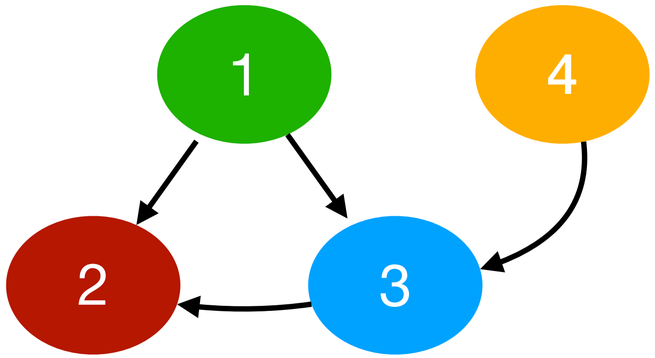


Exercise in Canvas: graph terminology



Graph terminology: collider on a path

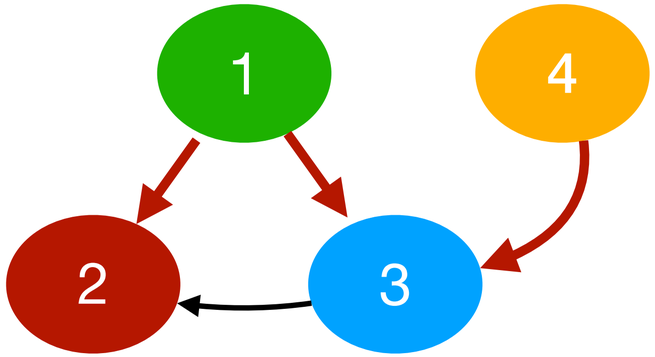
- A **path** between **node i and node j** is a sequence of **distinct nodes** $(i, \ldots, j)$ such that each two **consecutive nodes** are **adjacent**

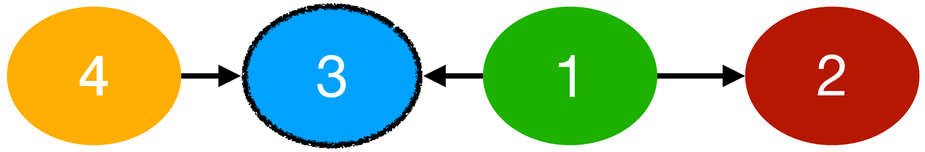


- A **collider** k on a **path** $\pi = (i, \ldots, j)$ is a non-endpoint node $(k \neq i, j)$ s.t. the path π contains $\rightarrow k \leftarrow$, nodes without this pattern are **non-colliders**

Graph terminology: collider on a path

- A **path** between **node i and node j** is a sequence of **distinct nodes** $(i, \ldots, j)$ such that each two **consecutive nodes** are **adjacent**



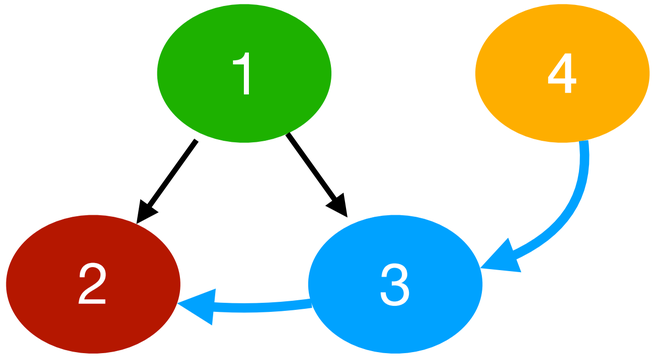


collider

- A **collider** k on a **path** $\pi = (i, \ldots, j)$ is a non-endpoint node $(k \neq i, j)$ s.t. the path π contains $\rightarrow k \leftarrow$, nodes without this pattern are **non-colliders**

Graph terminology: collider on a path

- A **path** between **node i and node j** is a sequence of **distinct nodes** $(i, \ldots, j)$ such that each two **consecutive nodes** are **adjacent**

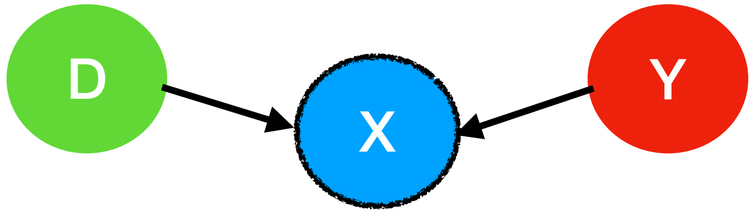


non-collider (chain)

- A **collider** k on a **path** $\pi = (i, \ldots, j)$ is a non-endpoint node $(k \neq i, j)$ s.t. the path π contains $\rightarrow k \leftarrow$, nodes without this pattern are **non-colliders**

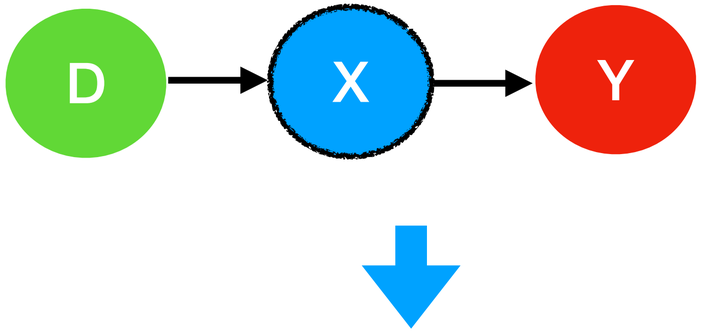
From the introduction: d-separation for causal discovery

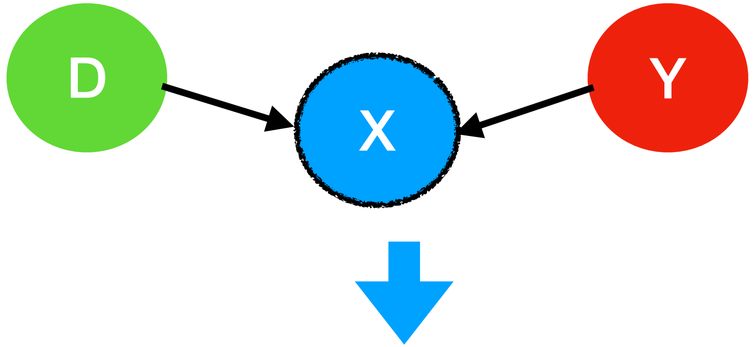
- Given a causal graph, **d-separation** is a graphical criterion that (under standard conditions) allows us to read **conditional independences**



From the introduction: d-separation for causal discovery

- Given a causal graph, **d-separation** is a graphical criterion that (under standard conditions) allows us to read **conditional independences**

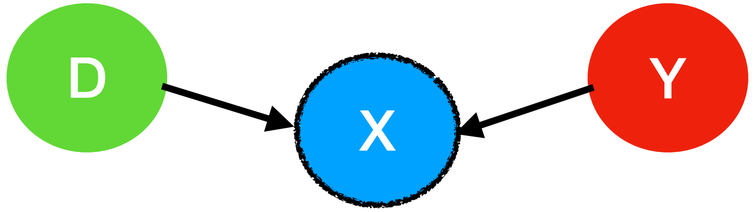




Graph: $Y \not\perp_d D$ $Y \perp_d D \mid X$ $Y \perp_d D$ $Y \not\perp_d D \mid X$

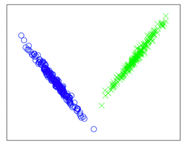
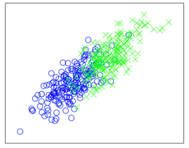
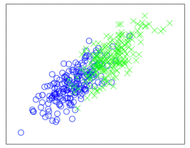
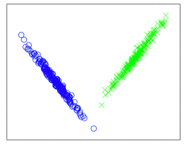
From the introduction: d-separation for causal discovery

- Given a causal graph, **d-separation** is a graphical criterion that (under standard conditions) allows us to read **conditional independences**



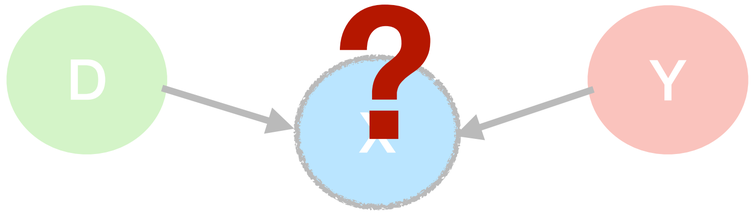
Graph: $Y \not\perp\!\!\!\perp_d D$ $Y \perp_d D \mid X$ $Y \perp_d D$ $Y \not\perp\!\!\!\perp_d D \mid X$

Data: $Y \not\perp\!\!\!\perp D$ $Y \perp\!\!\!\perp D \mid X$ $Y \perp\!\!\!\perp D$ $Y \not\perp\!\!\!\perp D \mid X$

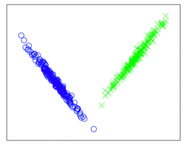
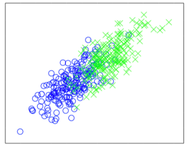
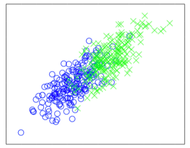
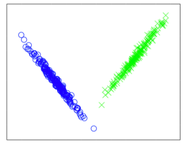
From the introduction: d-separation for causal discovery

- Given a causal graph, **d-separation** is a graphical criterion that (under standard conditions) allows us to read **conditional independences**



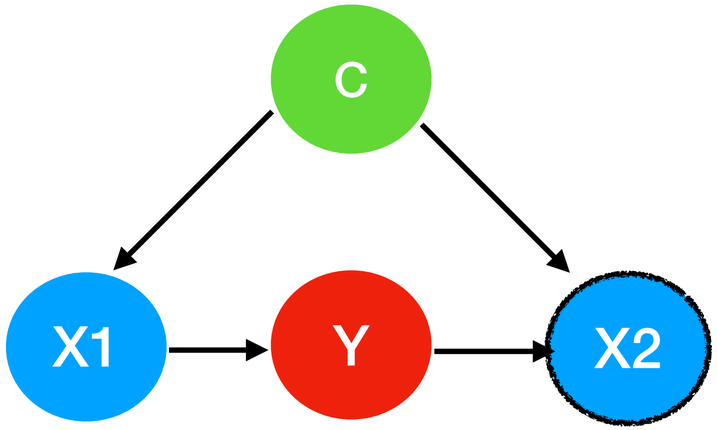
Graph: $Y \not\perp\!\!\!\perp_d D$ $Y \perp_d D \mid X$ $Y \perp_d D$ $Y \not\perp\!\!\!\perp_d D \mid X$

Data: $Y \not\perp\!\!\!\perp D$ $Y \perp\!\!\!\perp D \mid X$ $Y \perp\!\!\!\perp D$ $Y \not\perp\!\!\!\perp D \mid X$

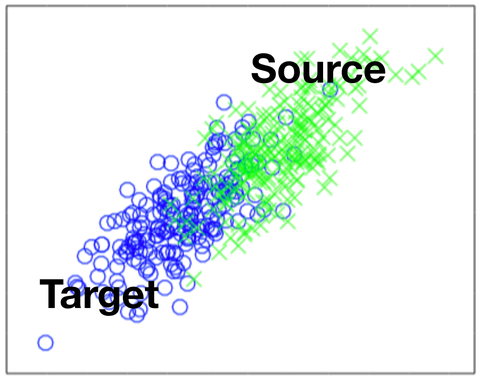





Teaser for last week: d-separation and distribution shift

- Causal graphs and d-separation [Pearl 2009] are a principled way to reason about **invariances and distribution shift**



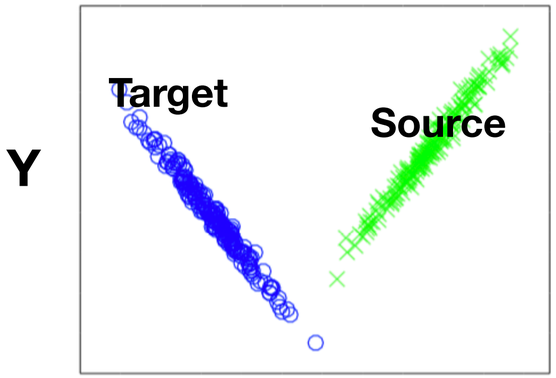
$$Y \not\perp\!\!\!\perp_d C \mid X_2$$



X1

$$Y \perp\!\!\!\perp C \mid X_1 \equiv$$

$$P(Y \mid X_1, C = 0) = P(Y \mid X_1, C = 1)$$



X2

$$Y \not\perp\!\!\!\perp C \mid X_2 \equiv$$

$$P(Y \mid X_2, C = 0) \neq P(Y \mid X_2, C = 1)$$

d-separation

- Nodes **i and j** are **d-separated by** $\mathbf{A} \subseteq \mathbf{V} \backslash \{i, j\}$ if all paths between i, j are **blocked**

d-separation: blocked paths

- A **path** between **i and j** is **blocked by** $\mathbf{A} \subseteq \mathbf{V} \backslash \{i, j\}$ **if** at least one condition holds:
  - There is a **non-collider** on the path that is in $\mathbf{A}$**, or**
  - There is a **collider** k on the path, but $k \notin \mathbf{A}$ and $\mathrm{Desc}(k) \cap \mathbf{A} = \varnothing$

d-separation: blocked paths

- A **path** between **i and j** is **blocked by** $\mathbf{A} \subseteq \mathbf{V} \backslash \{i, j\}$ **if** at least one condition holds, otherwise it is **active**
  - There is a **non-collider** on the path that is in $\mathbf{A}$**, or**
  - There is a **collider** k on the path, but $k \notin \mathbf{A}$ and $\mathrm{Desc}(k) \cap \mathbf{A} = \varnothing$

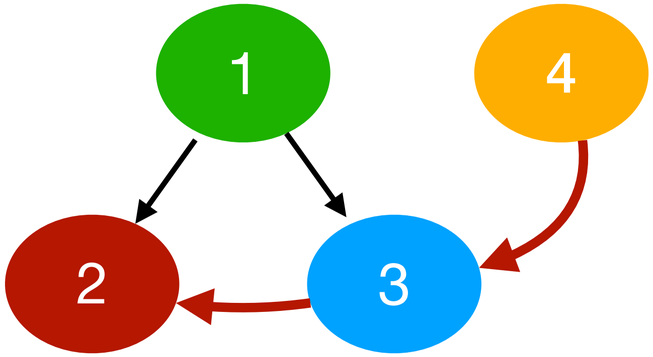
d-separation: blocked paths

- A **path** between **i and j** is **blocked by** $\mathbf{A} \subseteq \mathbf{V} \backslash \{i, j\}$ **if** at least one condition holds, otherwise it is **active**
  - There is a **non-collider** on the path that is in $\mathbf{A}$**, or**
  - There is a **collider** k on the path, but $k \notin \mathbf{A}$ and $\mathrm{Desc}(k) \cap \mathbf{A} = \varnothing$

Note: descendants w.r.t. the **whole graph**

d-separation: blocked paths - example 1

- A **path** between **i and j** is **blocked by** $\mathbf{A} \subseteq \mathbf{V} \backslash \{i, j\}$ **if** at least one condition holds, otherwise it is **active**
  - There is a **non-collider** on the path that is in $\mathbf{A}$**, or**
  - ~~There is a collider k on the path, but $k \notin \mathbf{A}$ and $\text{Desc}(k) \cap \mathbf{A} = \emptyset$~~

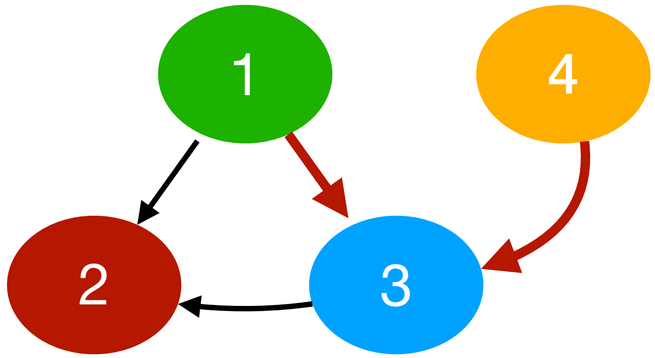


non-collider

If $3 \in \mathbf{A}$, the path is **blocked,** otherwise it is **active**

d-separation: blocked paths - example 2

- A **path** between **i and j** is **blocked by** $\mathbf{A} \subseteq \mathbf{V} \backslash \{i, j\}$ **if** at least one condition holds, otherwise it is **active**
  - ~~There is a non-collider on the path that is in $\mathbf{A}$, or~~
  - There is a **collider** k on the path, but $k \notin \mathbf{A}$ and $\text{Desc}(k) \cap \mathbf{A} = \emptyset$

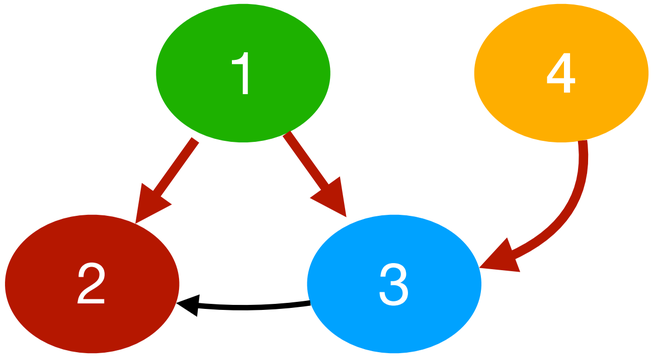


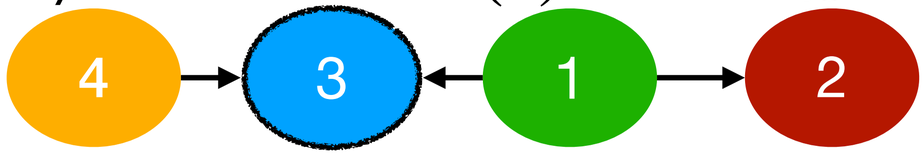
collider

If $3 \notin \mathbf{A}$ and $2 \notin \mathbf{A}$, the path is **blocked,** otherwise it is **active**

d-separation: blocked paths - example 3

- A **path** between **i and j** is **blocked by** $\mathbf{A} \subseteq \mathbf{V} \backslash \{i, j\}$ **if** at least one condition holds, otherwise it is **active**
  - There is a **non-collider** on the path that is in $\mathbf{A}$**, or**
  - There is a **collider** k on the path, but $k \notin \mathbf{A}$ and $\mathrm{Desc}(k) \cap \mathbf{A} = \emptyset$





collider non-collider

If $1 \in \mathbf{A}$, the path is **blocked**

OR

If $3 \notin \mathbf{A}$, the path is **blocked**

d-separation

- Nodes **i and j** are **d-separated by** $\mathbf{A} \subseteq \mathbf{V} \backslash \{i, j\}$ if all paths between i, j are **blocked**
  - We denote d-separation as $i \perp j \,|\, \mathbf{A}$

Note: d-separation is symmetric

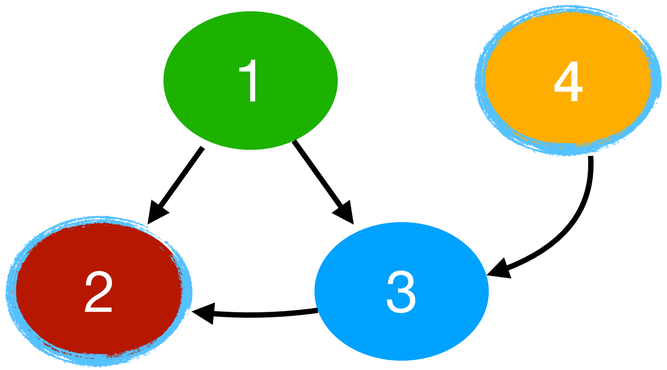
- Otherwise we say they are **d-connected**
  - We denote d-connection as $i \not\perp j \,|\, \mathbf{A}$

Notation: Sometimes d-separation is also denoted as $\perp_d$ and sometimes as $\perp\!\!\!\perp_d$.

In both cases the symbol is similar to conditional independence $\perp\!\!\!\perp$ (on purpose)

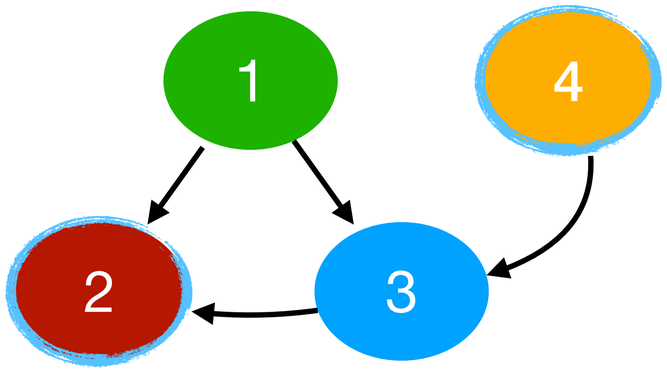
d-separation - complete example

- Nodes **i and j** are **d-separated by** $\mathbf{A} \subseteq \mathbf{V} \backslash \{i, j\}$ if all paths between i, j are **blocked**



d-separation - complete example

- Nodes **i and j** are **d-separated by** $\mathbf{A} \subseteq \mathbf{V} \backslash \{i, j\}$ if all paths between i, j are **blocked**

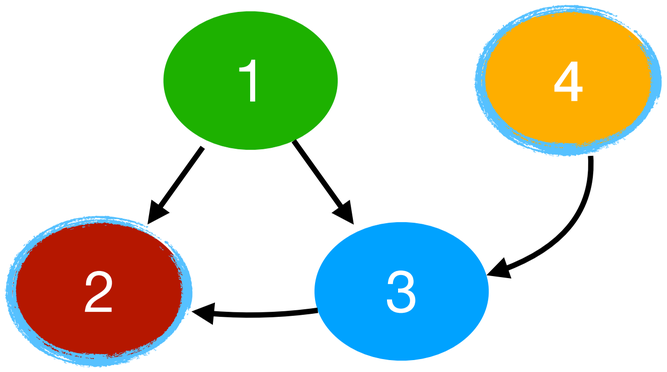


non-collider

If $3 \in \mathbf{A}$ the path is **blocked,** otherwise it is **active**

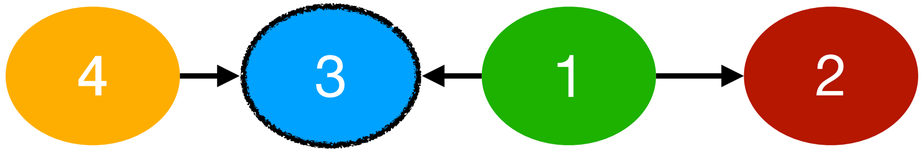
d-separation - complete example

- Nodes **i and j** are **d-separated by** $\mathbf{A} \subseteq \mathbf{V} \backslash \{i, j\}$ if all paths between i, j are **blocked**



non-collider

If $3 \in \mathbf{A}$ the path is **blocked,** otherwise it is **active**

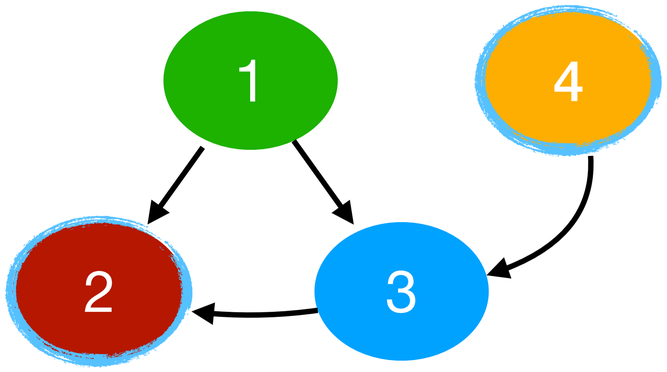


collider non-collider

If $1 \in \mathbf{A}$ **OR** if $3 \notin \mathbf{A}$, the path is **blocked,** otherwise it is **active**

d-separation - complete example

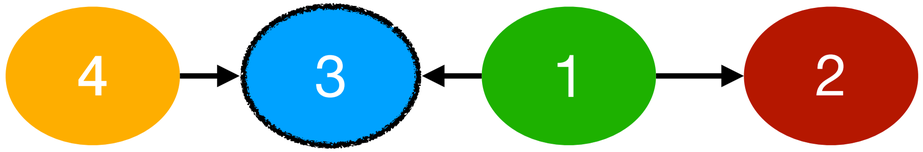
- Nodes **i and j** are **d-separated by** $\mathbf{A} \subseteq \mathbf{V} \backslash \{i, j\}$ if all paths between i, j are **blocked**



How can we keep both paths blocked?

non-collider

If $3 \in \mathbf{A}$ the path is **blocked,** otherwise it is **active**

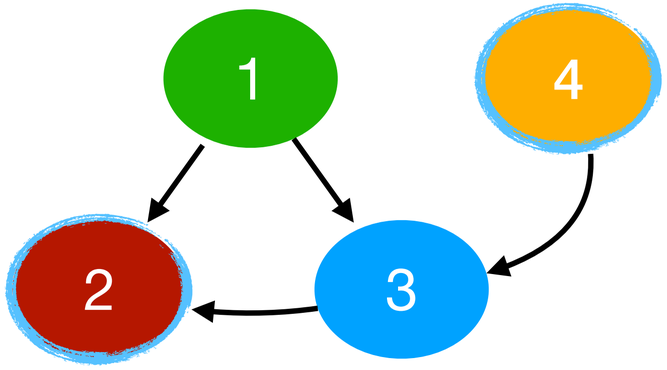


collider non-collider

If $1 \in \mathbf{A}$ **OR** if $3 \notin \mathbf{A}$, the path is **blocked,** otherwise it is **active**

d-separation - complete example

- Nodes **i and j** is **d-separated by A** if all paths between them are **blocked**

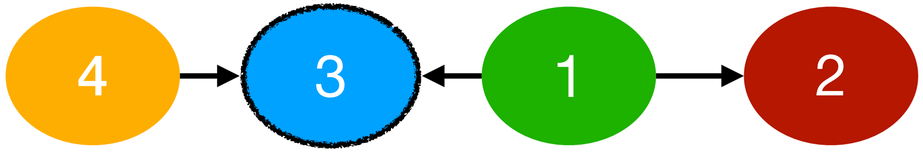


How can we keep both paths blocked?

- $3 \in \mathbf{A}$

non-collider

If $3 \in \mathbf{A}$ the path is blocked, otherwise it is active



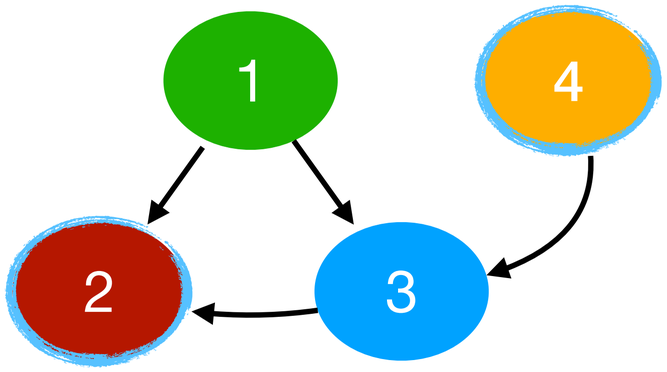
collider non-collider

If $1 \in \mathbf{A}$ **OR** if ~~$3 \notin \mathbf{A}$~~, the path is **blocked,**

otherwise it is **active**

d-separation - complete example

- Nodes **i and j** are **d-separated by** $\mathbf{A} \subseteq \mathbf{V} \backslash \{i, j\}$ if all paths between i, j are **blocked**

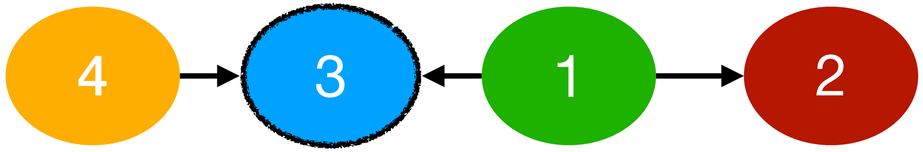


How can we keep both paths blocked?

- $3 \in \mathbf{A}$
- $1 \in \mathbf{A}$

non-collider

✅ If $3 \in \mathbf{A}$ the path is blocked, otherwise it is active

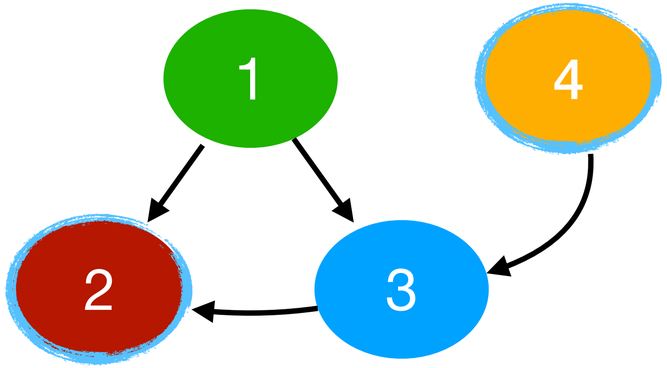


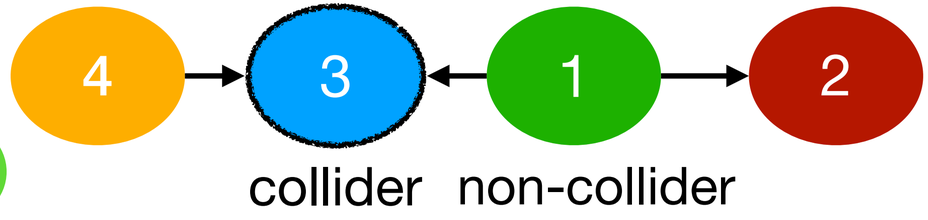
collider non-collider

✅ If $1 \in \mathbf{A}$ **OR** if ~~$3 \notin \mathbf{A}$~~, the path is **blocked**, otherwise it is **active**

d-separation - complete example

- Nodes **i and j** are **d-separated by** $\mathbf{A} \subseteq \mathbf{V} \backslash \{i, j\}$ if all paths between i, j are **blocked**





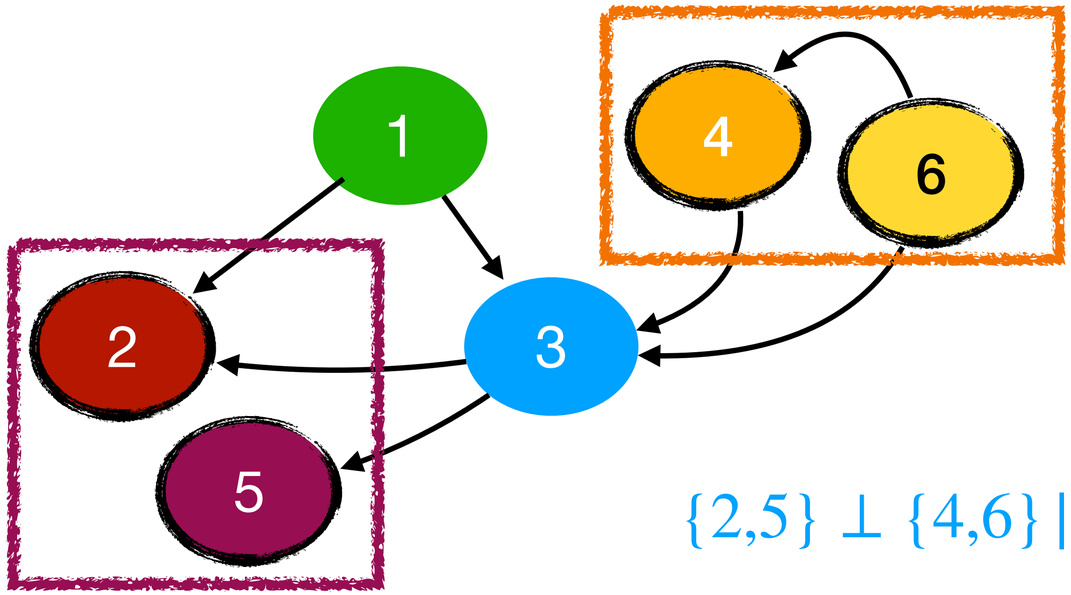
How can we keep both paths blocked?

- $3 \in \mathbf{A}$
- $1 \in \mathbf{A}$

$$2 \perp 4 \,|\, \mathbf{A} = \{1,3\}$$

d-separation for sets of nodes

- Sets of nodes $\mathbf{A}$ **and** $\mathbf{B}$ are **d-separated by** $\mathbf{C} \subseteq \mathbf{V} \backslash \mathbf{A} \cup \mathbf{B}$**,** if all paths between $i \in \mathbf{A}$ and $j \in \mathbf{B}$ are **blocked**
  - We denote d-separation as $\mathbf{A} \perp \mathbf{B} \mid \mathbf{C}$



$\{2,5\} \perp \{4,6\} \mid \{1,3\}$

Exercise in Canvas: d-separation