Simple Graph

A simple graph (or just: graph) is a tuple $\mathcal{G} = (V, E)$ where

$$V = \{A_1, \ldots, A_n\}$$

represents a finite set of **vertices** (or **nodes**) and

$$E \subseteq (V \times V) \setminus \{(A, A) \mid A \in V\}$$

denotes the set of **edges**.
It is called simple since there are no self-loops and no multiple edges.

Edge Types

Let $\mathcal{G} = (V, E)$ be a graph. An edge $e = (A, B)$ is called

directed if $(A, B) \in E \Rightarrow (B, A) \notin E$
Notation: $A \to B$

undirected if $(A, B) \in E \Rightarrow (B, A) \in E$
Notation: $A - B$ or $B - A$

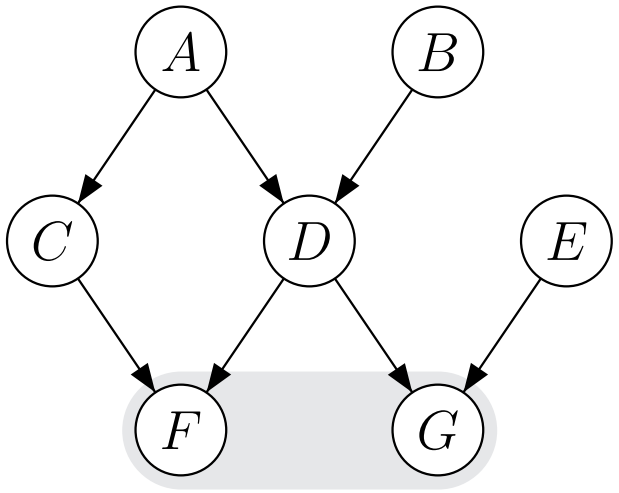
(Un)directed Graph

A graph with only (un)directed edges is called an (un)directed graph.

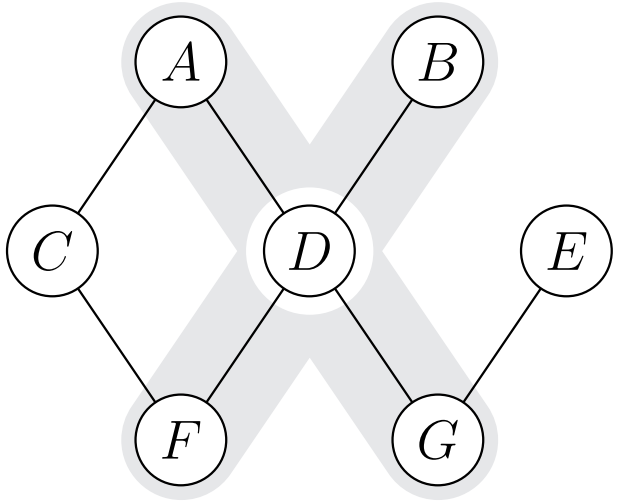
Adjacency Set

Let $\mathcal{G} = (V, E)$ be a graph. The set of nodes that is accessible via a given node $A \in V$ is called the **adjacency set** of A:

$$\text{adj}(A) = \{B \in V \mid (A, B) \in E\}$$



adj(D)



Paths

Let $\mathcal{G} = (V, E)$ be a graph. A series ρ of r pairwise different nodes

$$\rho = \left\langle A_{i_1}, \ldots, A_{i_r} \right\rangle$$

is called a **path** from A_i to A_j if

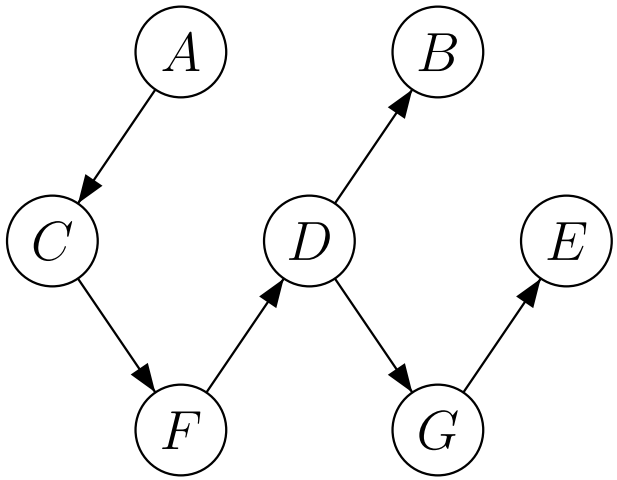
$$A_{i_1} = A_i, \quad A_{i_r} = A_j$$
$$A_{i_{k+1}} \in \mathrm{adj}(A_{i_k}), \quad 1 \leq k < r$$

A path with only undirected edges is called an **undirected path**

$$\rho = A_{i_1} - \cdots - A_{i_r}$$

whereas a path with only directed edges is referred to as a **directed path**

$$\rho = A_{i_1} \to \cdots \to A_{i_r}$$



If there is a directed path ρ from node A to node B in a directed graph $\mathcal{G}$ we write

$$A \overset{\rho}{\underset{\mathcal{G}}{\rightsquigarrow}} B.$$

If the path ρ is undirected we denote this with

$$A \overset{\rho}{\underset{\mathcal{G}}{\leftrightsquigarrow}} B.$$

Graph Types

Loop

Let $\mathcal{G} = (V, E)$ be an undirected graph. A path

$$\rho \;=\; X_1 - \cdots - X_k$$

with $X_k - X_1 \in E$ is called a loop.

Cycle

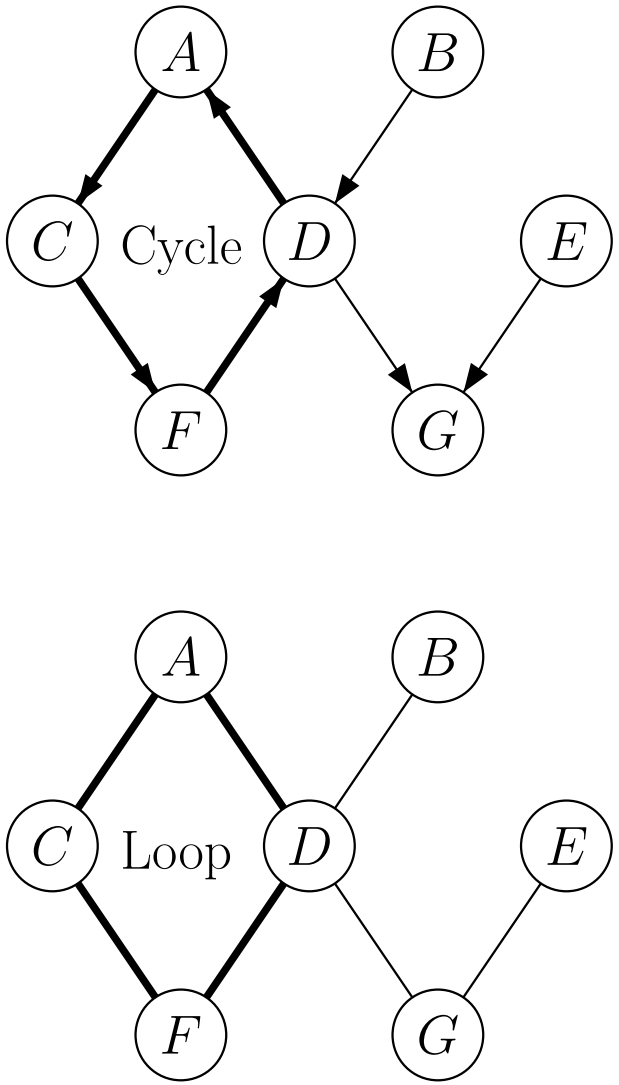
Let $\mathcal{G} = (V, E)$ be a directed graph. A path

$$\rho \;=\; X_1 \to \cdots \to X_k$$

with $X_k \to X_1 \in E$ is called a cycle.

Directed Acyclic Graph (DAG)

A directed graph $\mathcal{G} = (V, E)$ is called **acyclic** if for every path $X_1 \to \cdots \to X_k$ in $\mathcal{G}$ the condition $X_k \to X_1 \notin E$ is satisfied, i. e. it contains no cycle.



Parents, Children and Families

Let $\mathcal{G} = (V, E)$ be a directed graph. For every node $A \in V$ we define the following sets:

Parents:

$$\text{parents}_{\mathcal{G}}(A) = \{B \in V \mid B \to A \in E\}$$

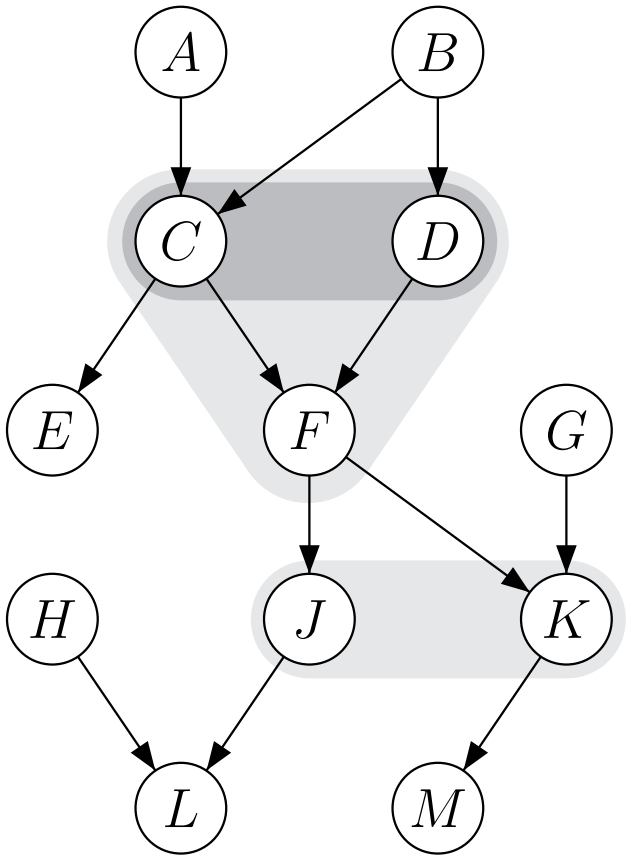
Children:

$$\text{children}_{\mathcal{G}}(A) = \{B \in V \mid A \to B \in E\}$$

Family:

$$\text{family}_{\mathcal{G}}(A) = \{A\} \cup \text{parents}_{\mathcal{G}}(A)$$

If the respective graph is clear from the context, the index $\mathcal{G}$ is omitted.



$$\begin{aligned} \text{parents}(F) &= \{C, D\} \\ \text{children}(F) &= \{J, K\} \\ \text{family}(F) &= \{C, D, F\} \end{aligned}$$

Ancestors, Descendants, Non-Descendants

Let $\mathcal{G} = (V, E)$ be a DAG. For every node $A \in V$ we define the following sets:

Ancestors:

$$\text{ancs}_{\mathcal{G}}(A) \;=\; \{B \in V \mid \exists \rho : B \overset{\rho}{\underset{\mathcal{G}}{\rightsquigarrow}} A\}$$

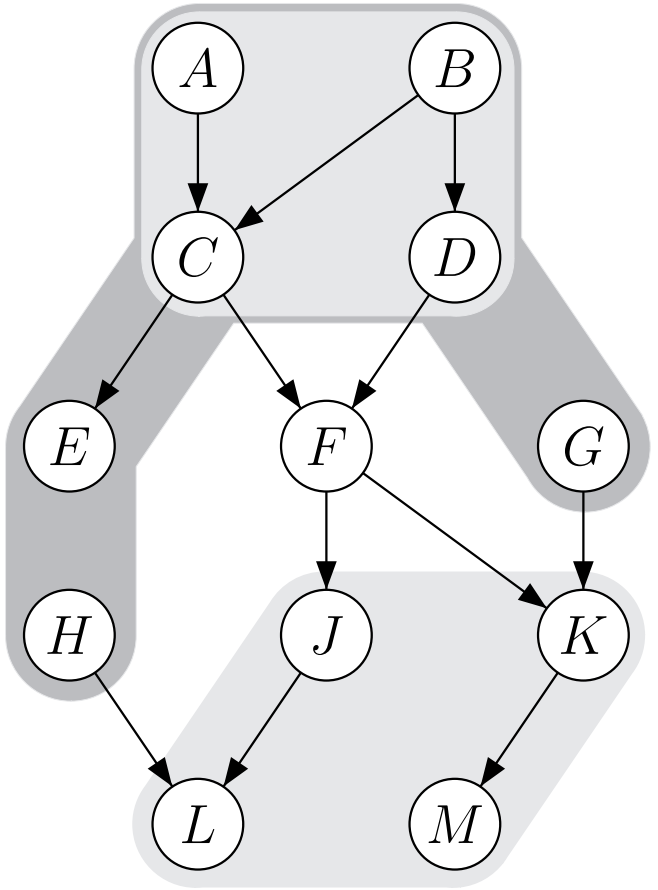
Descendants:

$$\text{descs}_{\mathcal{G}}(A) \;=\; \{B \in V \mid \exists \rho : A \overset{\rho}{\underset{\mathcal{G}}{\rightsquigarrow}} B\}$$

Non-Descendants:

$$\text{non-descs}_{\mathcal{G}}(A) \;=\; V \setminus \{A\} \setminus \text{descs}_{\mathcal{G}}(A)$$

If the respective graph is clear from the context, the index $\mathcal{G}$ is omitted.



$$\text{ancs}(F) = \{A, B, C, D\}$$
$$\text{descs}(F) = \{J, K, L, M\}$$
$$\text{non-descs}(F) = \{A, B, C, D, E, G, H\}$$

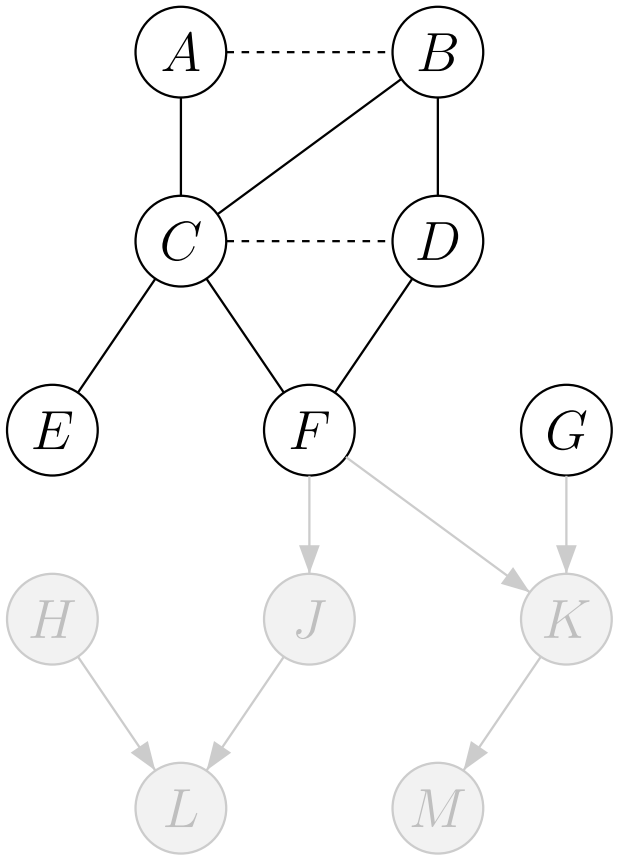
Operations on Graphs

Let $\mathcal{G} = (V, E)$ be a DAG.

The **Minimal Ancestral Subgraph** of $\mathcal{G}$ given a set $M \subseteq V$ of nodes is the smallest subgraph that contains all ancestors of all nodes in M.

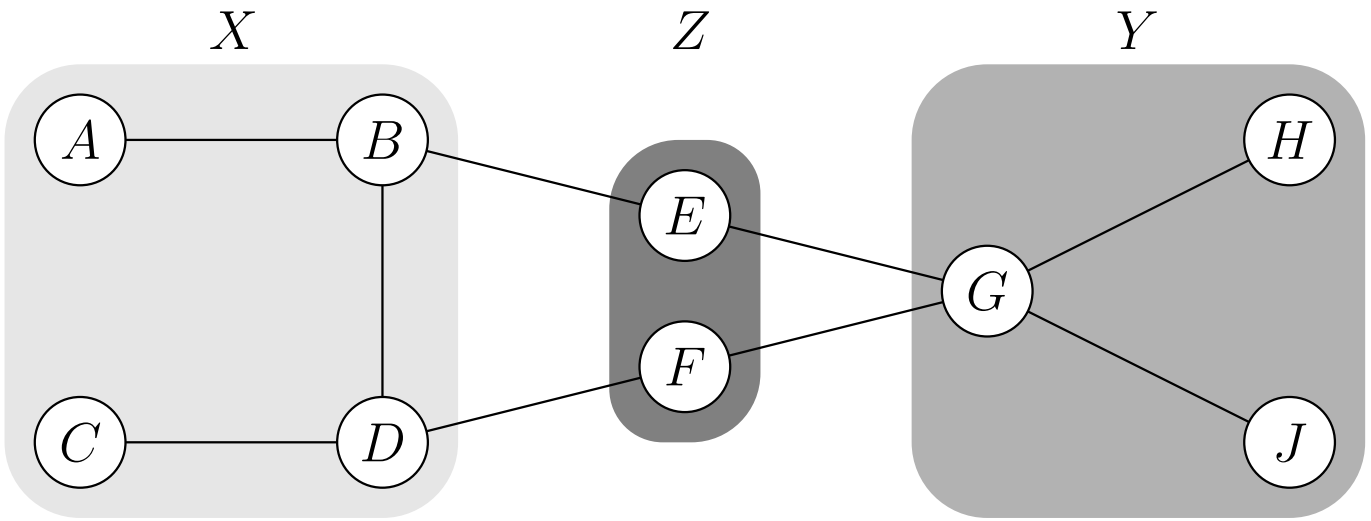
The **Moral Graph** of $\mathcal{G}$ is the undirected graph that is obtained by

1. connecting nodes that share a common child with an arbitrarily directed edge and,
2. converting all directed edges into undirected ones by dropping the arrow heads.



Moral graph of ancestral graph induced by the set $\{E, F, G\}$.

u-Separation



Let $\mathcal{G} = (V, E)$ be an undirected graph and $X, Y, Z \subseteq V$ three disjoint subsets of nodes. We agree on the following separation criteria:

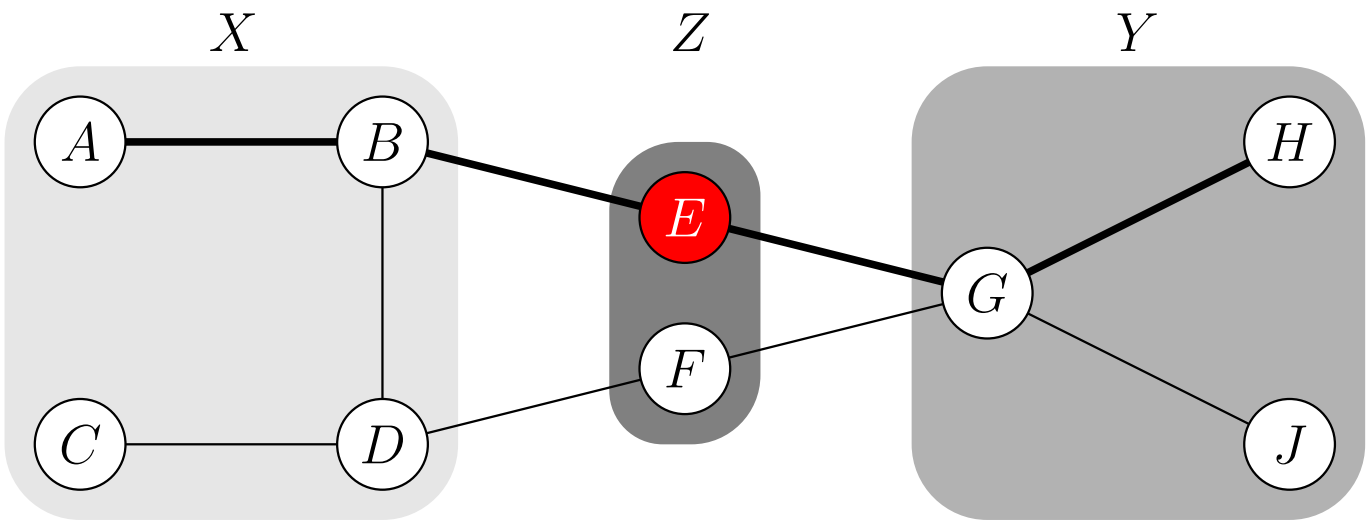
1. Z u-separates X from Y — written as

$$X \perp\!\!\!\perp_{\mathcal{G}} Y \mid Z,$$

   if every possible path from a node in X to a node in Y is blocked.

2. A path is blocked if it contains one (or more) **blocking nodes**.
3. A node is a blocking node if it lies in Z.

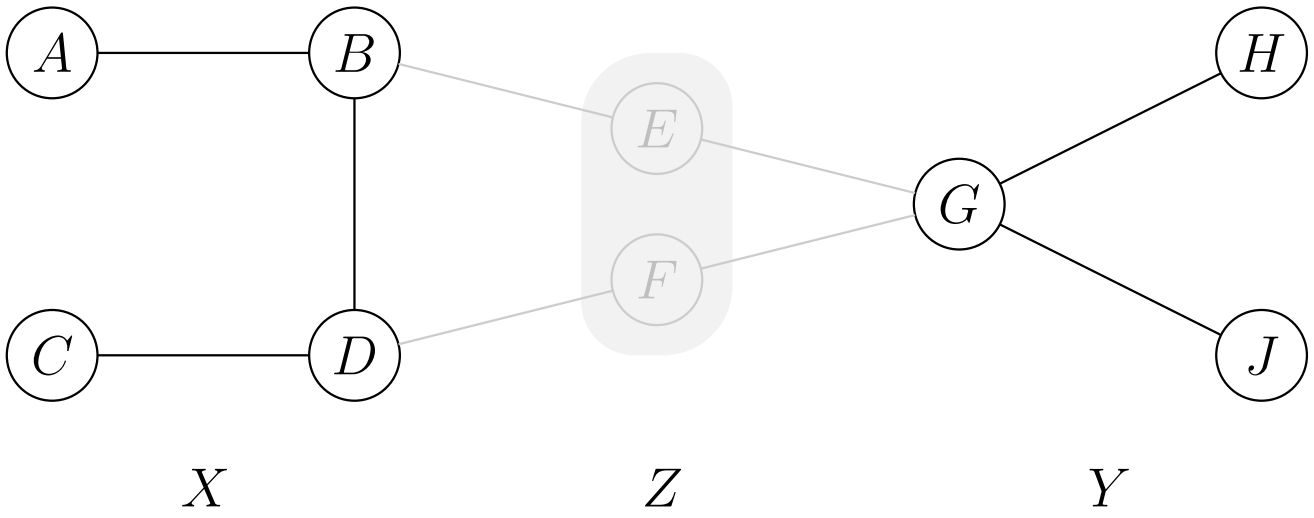
u-Separation



E. g. path $A - B - E - G - H$ is blocked by $E \in Z$. It can be easily verified, that every path from X to Y is blocked by Z. Hence we have:

$$\{A, B, C, D\} \perp\!\!\!\perp_{\mathcal{G}} \{G, H, J\} \mid \{E, F\}$$

u-Separation



Another way to check for u-separation: Remove the nodes in Z from the graph (and all the edges adjacent to these nodes). X and Y are u-separated by Z if the remaining graph is disconnected with X and Y in separate subgraphs.

Node

E seperates K and B in the directed graph

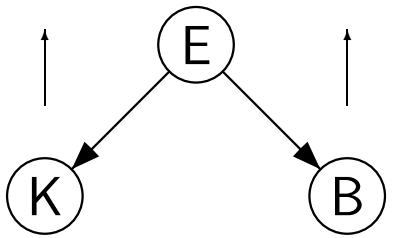
Example — Qualitative Aspects

Lecture theatre in winter: Waiting for Mr. K and Mr. B.
Not clear whether there is ice on the roads.

3 variables:

- E road condition: $\mathrm{dom}(\mathsf{E}) = \{\mathsf{ice}, \neg\mathsf{ice}\}$
- K K had an accident: $\mathrm{dom}(\mathsf{K}) = \{\mathsf{yes}, \mathsf{no}\}$
- B B had an accident: $\mathrm{dom}(\mathsf{B}) = \{\mathsf{yes}, \mathsf{no}\}$

Ignorance about these states is modelled via the observer's belief.

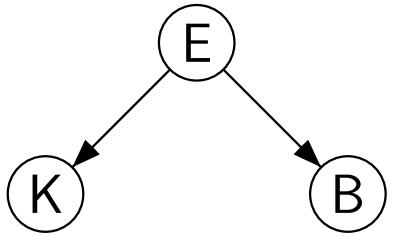


$\downarrow$ E influences K and B
(the more ice the more accidents)

$\uparrow$ Knowledge about accident increases belief in ice

Example

A priori knowledge	Evidence	Inferences
E unknown	B has accident	$\Rightarrow$ E = ice more likely
		$\Rightarrow$ K has accident more likely
E = $\neg$ice	B has accident	$\Rightarrow$ no change in belief about E
		$\Rightarrow$ no change in belief about accident of K
E unknown		K and B dependent
E known		K and B independent



Node E seperates K and B in the directed graph.

Now: Separation criterion for directed graphs.

We use the same principles as for u-separation. Two modifications are necessary:

Directed paths may lead also in reverse to the arrows.

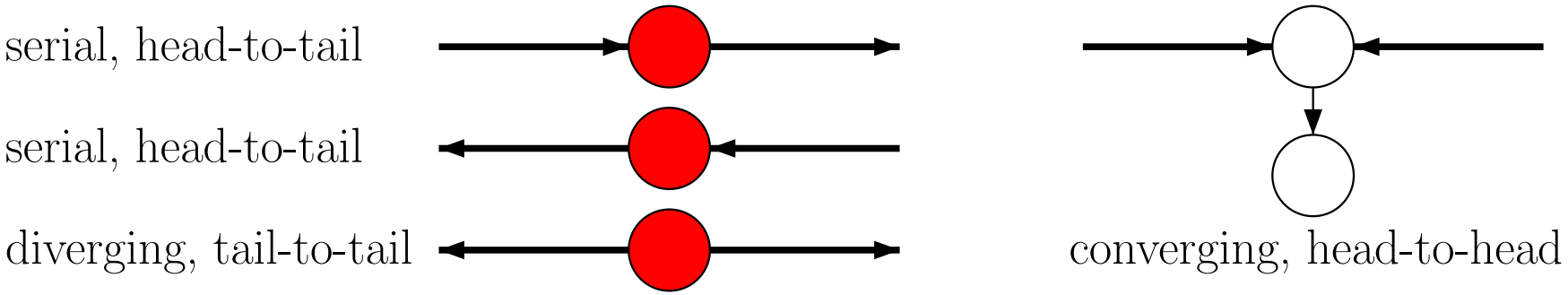
The blocking node condition is more sophisticated.

Blocking Node (in a directed path)

A node A is blocking if its edge directions **along the path**

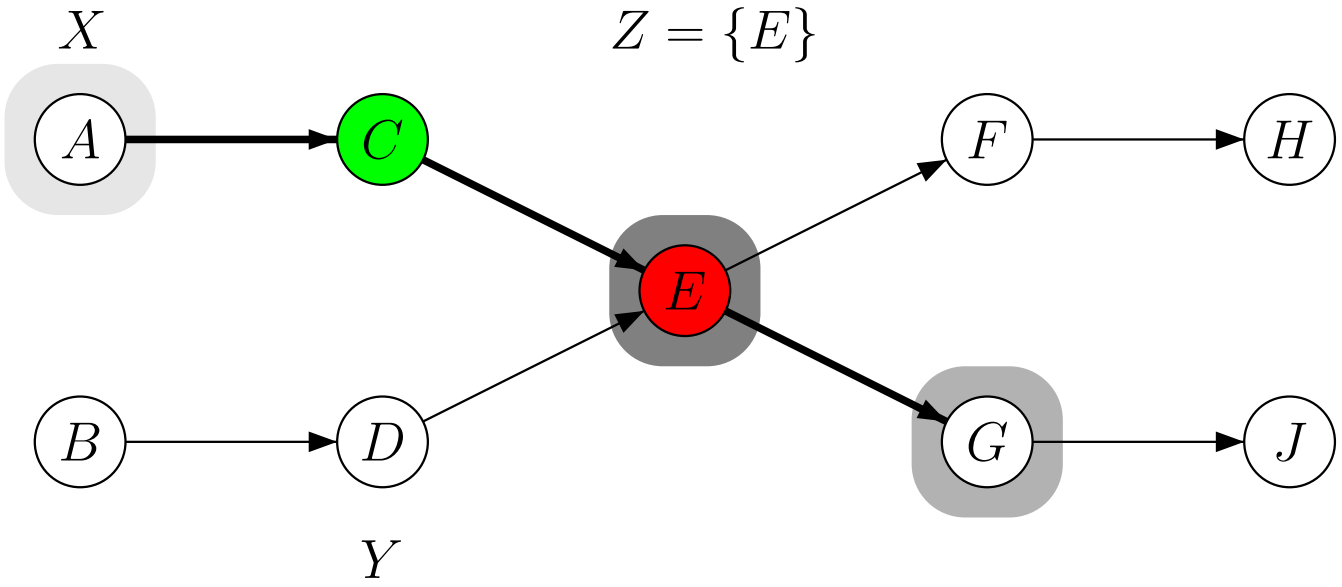
are of type 1 and $A \in Z$, or

are of type 2 and neither A nor one of its descendants is in Z.



d-Separation

Checking path $A \to C \to E \to G$



Checking path $A \to C \to E \leftarrow D$:

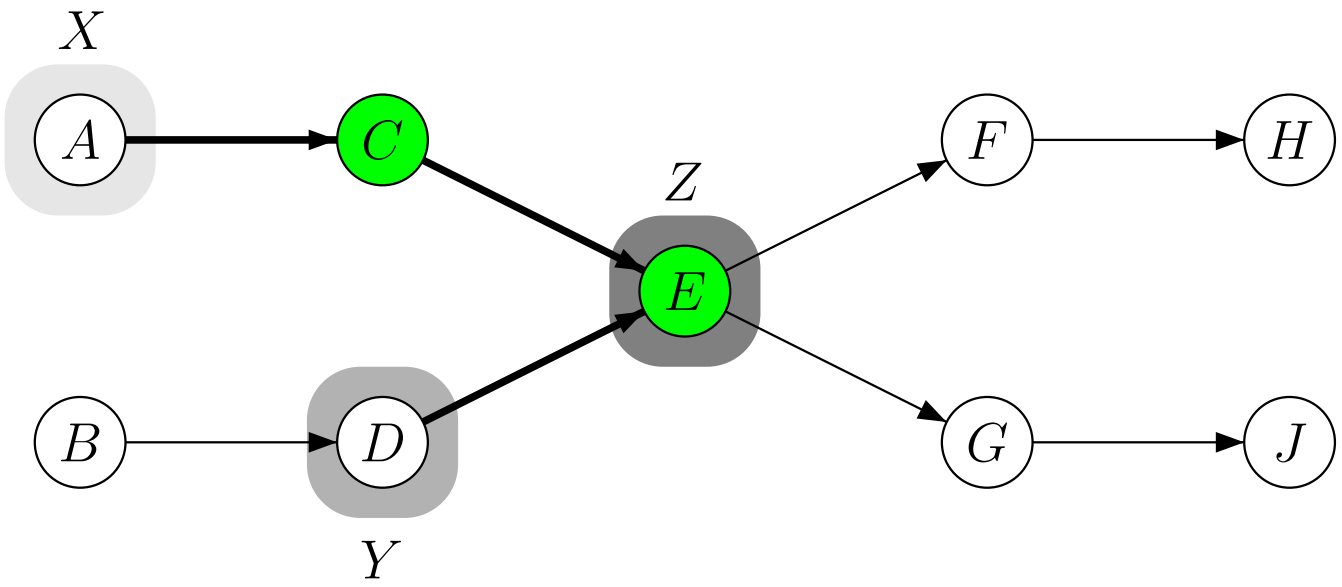
C is **serial** and not in Z: non-blocking

E is also **serial** but in Z: **blocking**

Path is blocked, no other paths between A and G are available

$$\Rightarrow A \perp\!\!\!\perp G \mid E$$

d-Separation



Checking path $A \rightarrow C \rightarrow E \leftarrow D$:

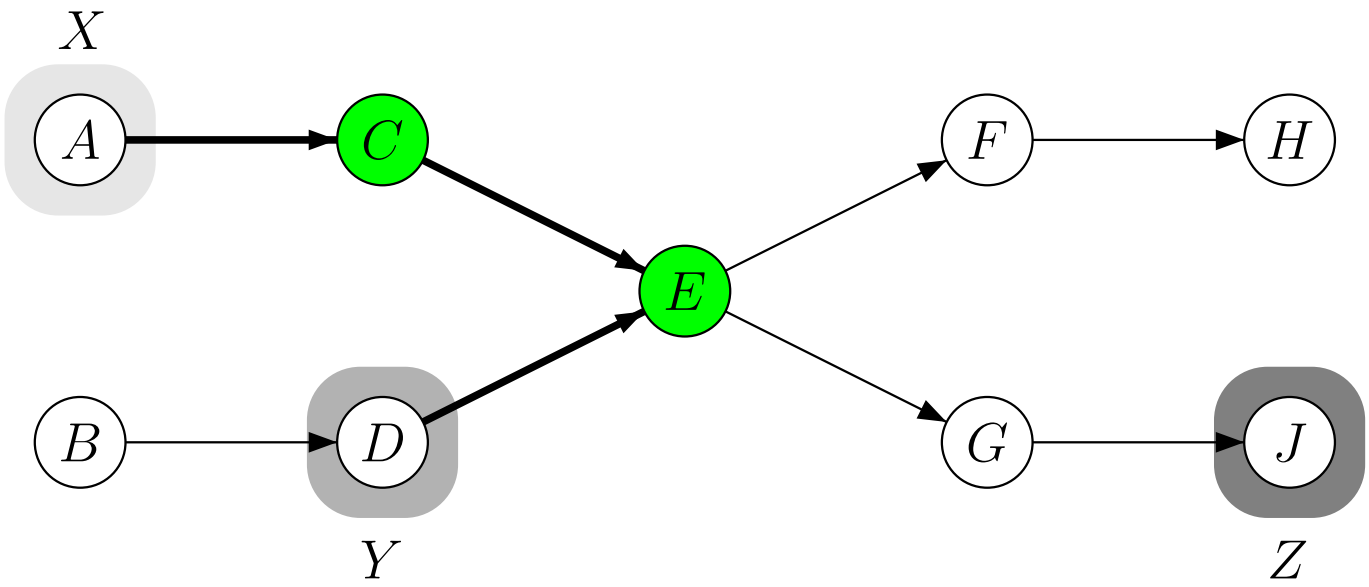
C is **serial** and not in Z: non-blocking

E is **converging** and in Z: non-blocking

$\Rightarrow$ Path is not blocked

$$A \not\perp\!\!\!\perp D \mid E$$

d-Separation



Checking path $A \to C \to E \leftarrow D$:

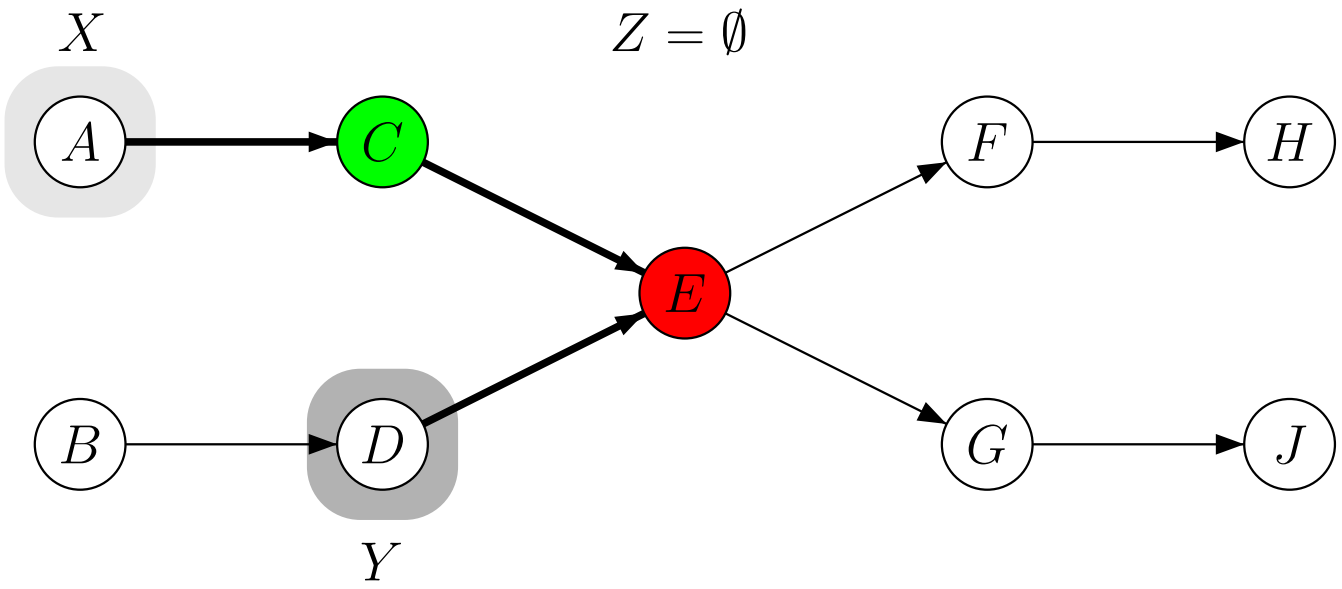
C is **serial** and not in Z: non-blocking

E is **converging** and not in Z but one of its descendants (J) is in Z: non-blocking

$\Rightarrow$ Path is not blocked

$$A \not\perp\!\!\!\perp D \mid J$$

d-Separation



Checking path $A \to C \to E \leftarrow D$:

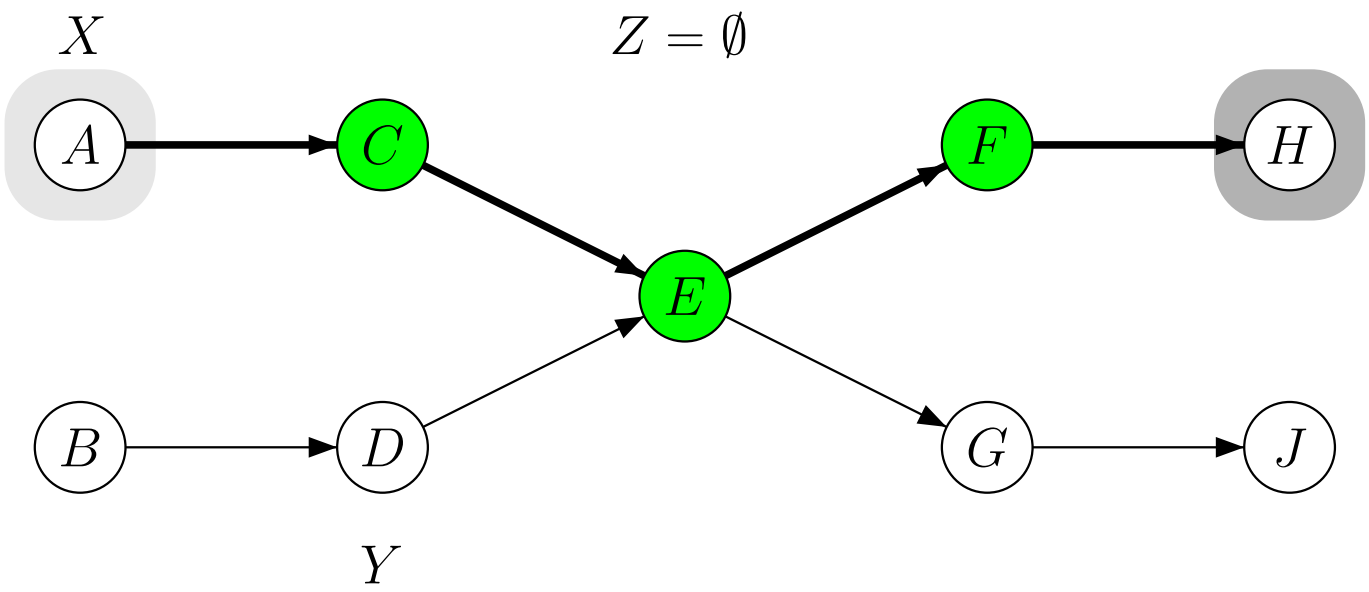
C is **serial** and not in Z: non-blocking

E is **converging** and not in Z, neither is F, G, H or J: **blocking**

$\Rightarrow$ Path is blocked

$$A \perp\!\!\!\perp D \mid \emptyset$$

d-Separation



Checking path $A \to C \to E \to F \to H$:

C is **serial** and not in Z: non-blocking

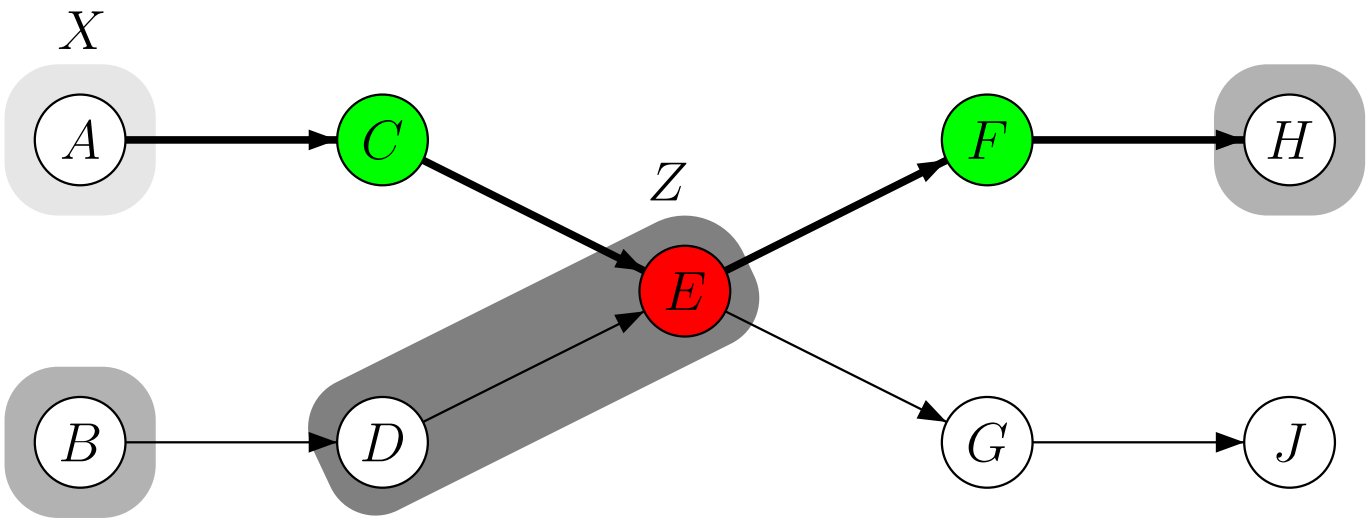
E is **serial** and not in Z: non-blocking

F is **serial** and not in Z: non-blocking

$\Rightarrow$ Path is not blocked

$$A \not\perp\!\!\!\perp H \mid \emptyset$$

d-Separation



Checking path $A \to C \to E \to F \to H$:

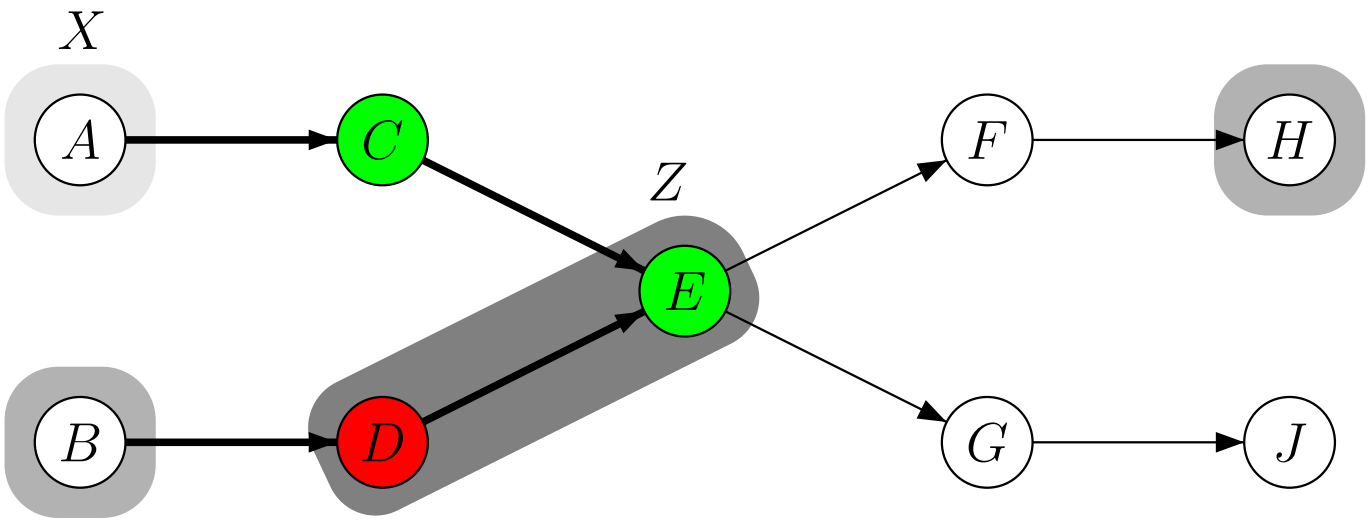
C is **serial** and not in Z: non-blocking

E is **serial** and in Z: **blocking**

F is **serial** and not in Z: non-blocking

$\Rightarrow$ Path is blocked

d-Separation



Checking path $A \to C \to E \leftarrow D \to B$:

C is **serial** and not in Z: non-blocking

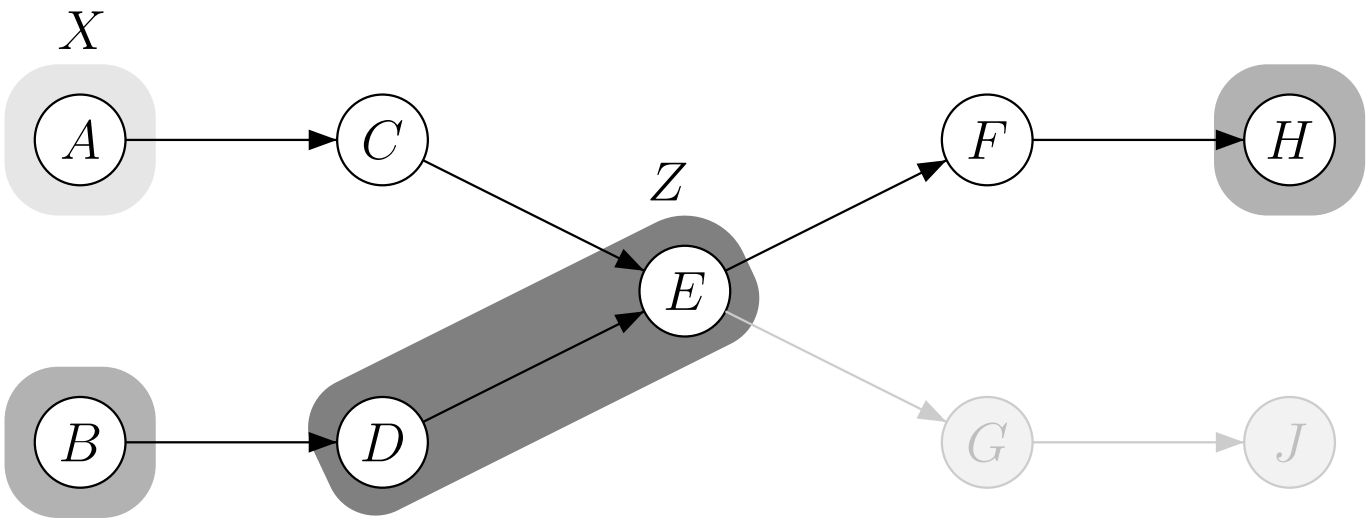
E is **converging** and in Z: non-blocking

D is **serial** and in Z: **blocking**

$\Rightarrow$ Path is blocked

$$A \perp\!\!\!\perp H, B \mid D, E$$

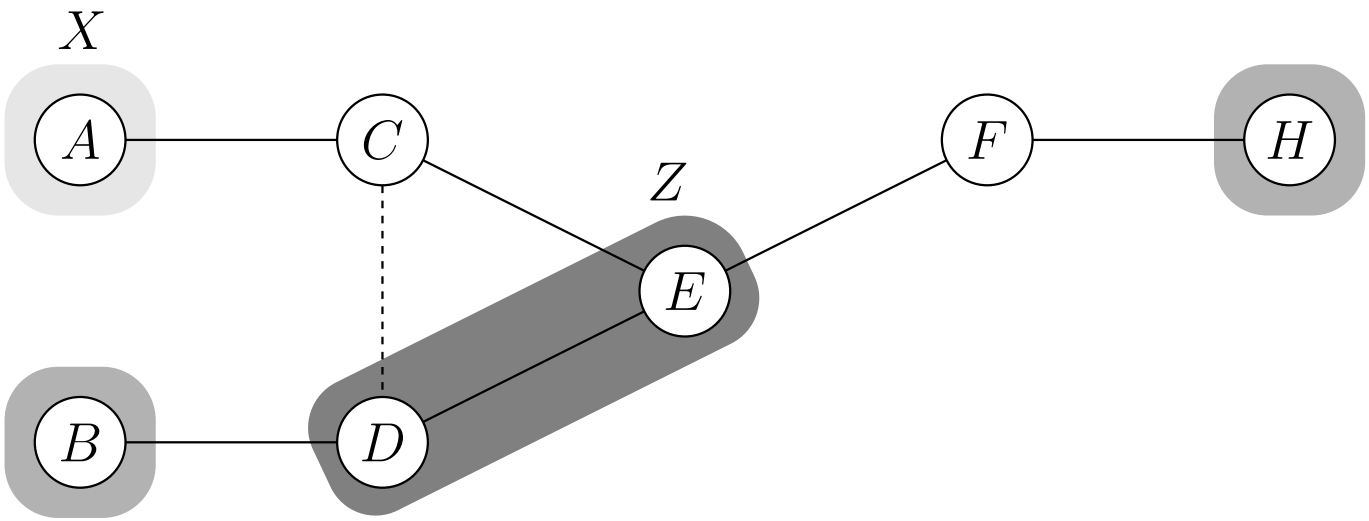
d-Separation: Alternative Way for Checking



Steps

Create the minimal ancestral subgraph induced by $X \cup Y \cup Z$.

d-Separation: Alternative Way for Checking



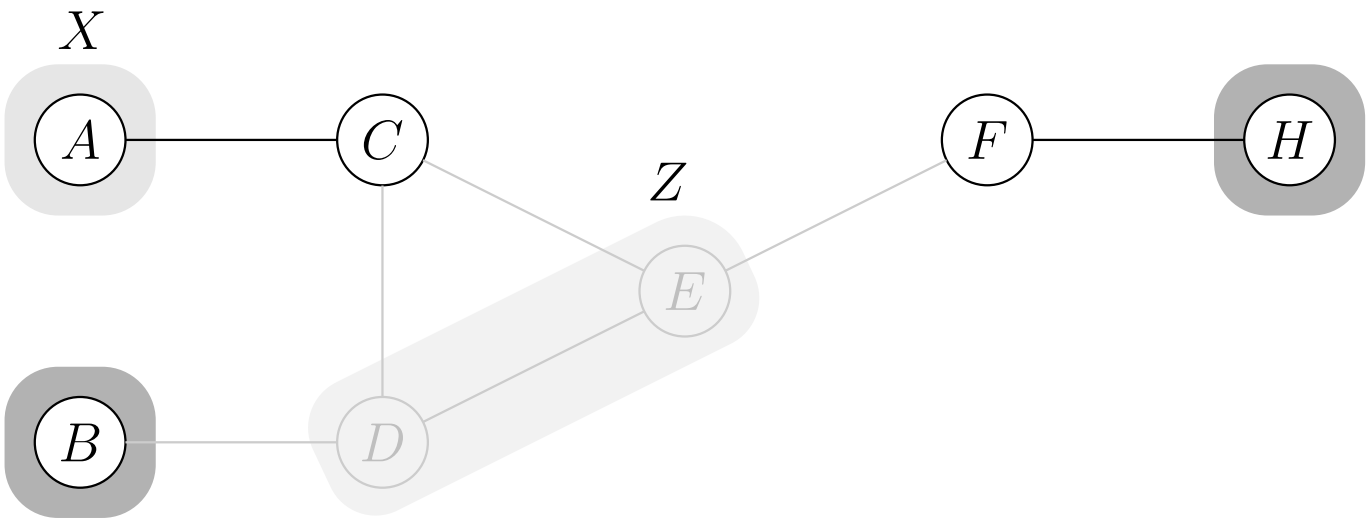
$$Y = \{B, H\}$$

Steps

Create the minimal ancestral subgraph induced by $X \cup Y \cup Z$.

Moralize that subgraph.

d-Separation: Alternative Way for Checking



$$Y = \{B, H\}$$

Steps:

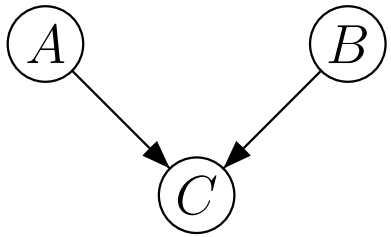
Create the minimal ancestral subgraph induced by $X \cup Y \cup Z$.

Moralize that subgraph.

Check for u-Separation in that undirected graph.

$$A \perp\!\!\!\perp H, B \mid D, E$$

Example



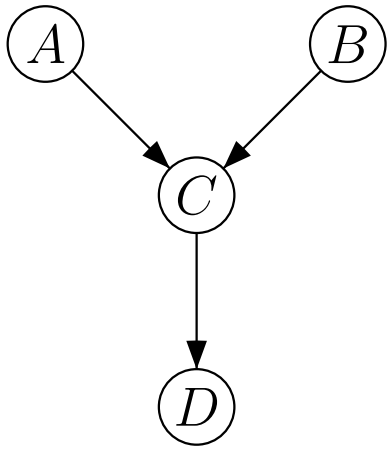
Meal quality	
A	quality of ingredients
B	cook's skill
C	meal quality

If C is not known, A and B are independent.

If C is known, then A and B become (conditionally) dependent given C.

$A \not\perp\!\!\!\perp B \mid C$

Example (cont.)



Meal quality	
A	quality of ingredients
B	cook's skill
C	meal quality
D	restaurant success

If nothing is known about the restaurant success or meal quality or both, the cook's skills and quality of the ingredients are unrelated, that is, *independent*.

However, if we observe that the restaurant has no success, we can infer that the meal quality might be bad.

If we further learn that the ingredients quality is high, we will conclude that the cook's skills must be low, thus rendering both variables *dependent*.

$A \not\perp\!\!\!\perp B \mid D$