



# An Introduction to **Generative** **Flow Networks**

**Juan Ramirez**

[juan.ramirez@mila.quebec](mailto:juan.ramirez@mila.quebec)

# Outline

- Motivation
- Applications
- GFlowNets
  - Flow Networks
  - Generative Flow Networks
  - Training GFlowNets
- Comparison to methods seen in class

# Important References

- *GFlowNet Foundations*. Yoshua Bengio, et al. JMLR 2023.
- [[NeurIPS 2021 GFlowNet paper](#)] *Flow Network based Generative Models for Non-Iterative Diverse Candidate Generation*. Emmanuel Bengio, et al. NeurIPS 2021.
- [[Structure Learning with GFlowNets](#)] *Bayesian Structure Learning with Generative Flow Networks*. Tristan Deleu, et al. UAI 2022.

# Setting

We want to do **inference** over an *intractable* distribution:

- Sampling:  $x \sim \mathbf{p}(x)$ .
- Computing expectations:  $\mathbb{E}_{x \sim \mathbf{p}(x)} [\mathbf{f}(x)]$ .

Sampling is intractable, but:

- Samples are discrete\* and can be built **compositionally**.
- Can not evaluate  $\mathbf{p}(x)$ , but can evaluate a reward function “ $\mathbf{R}$ ” such that  $\mathbf{p}(x) \propto \mathbf{R}(x)$ .

\*For continuous GFlowNets, see *A Theory of Continuous Generative Flow Networks* (S. Lahlou et al., 2023)

# Simplified Taxonomy for Sampling

- **Easy:** Know  $\mathbf{p}(x)$ , can efficiently sample from  $\mathbf{p}(x)$ 
  - Uniform
  - Mixture of Gaussians
  - HMM (with Gaussian emission probabilities)
- **Approximate:** sampling is expensive or impossible, but can evaluate  $\mathbf{p}(x)$ 
  - Rejection sampling: for using an easy distribution  $\mathbf{q}(x)$
  - Importance sampling
  - MCMC: exploit structure and low dimensionality of  $\mathbf{p}(x_i | x_{\sim i})$
  - Variational Inference: sample from easy  $\mathbf{q} \in \text{argmin KL}(\mathbf{q} || \mathbf{p})$
  - **GFlowNets** (can evaluate  $\mathbf{R}(x) \propto \mathbf{p}(x)$ ; next slide)

# GFlowNets Main Idea

Given a reward function  $R(x)$ ,

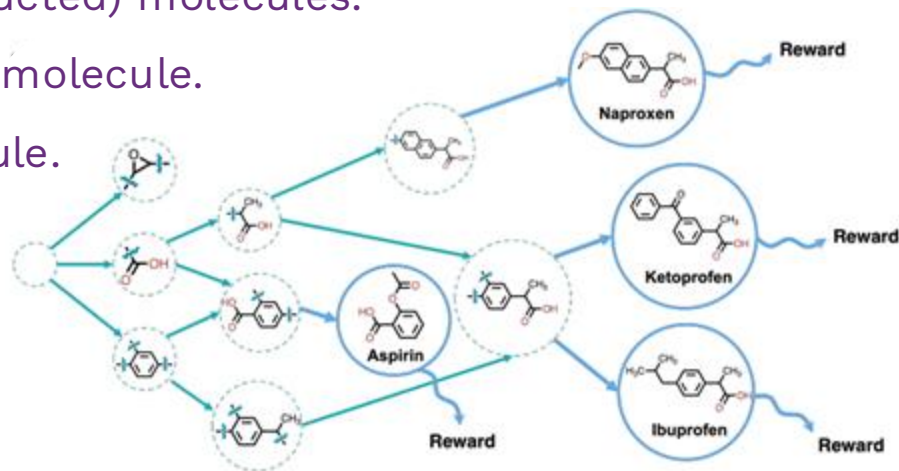
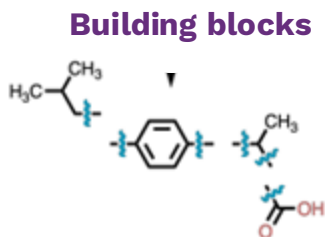
1. Construct a DAG  $G$  over all possible samples
  - a. **States:** possible samples  $x$  (e.g. all possible chemical molecules)
  - b. **Transitions:** composition of parent to produce child (e.g. adding a basic molecule to produce a more complex one)
2. Learn a probability distribution  $p \in L(G)$  such that  $p(x) \propto R(x)$ 
  - a. In practice, GFlowNets model unnormalized “flows” such that  $F(x) = R(x)$
  - b. Also, *approximate*  $F(x)$  variationally (i.e. use a neural network to output  $F(x)$ )
3. Sample over  $G$ .
  - a. Ancestral sampling, beam search, ...

# Applications

# Scientific Discovery

For instance, constructing **molecules**.

- **States:** sets of (partially constructed) molecules.
- **Transitions:** addition of a basic molecule.
- **Reward:** property of the molecule.



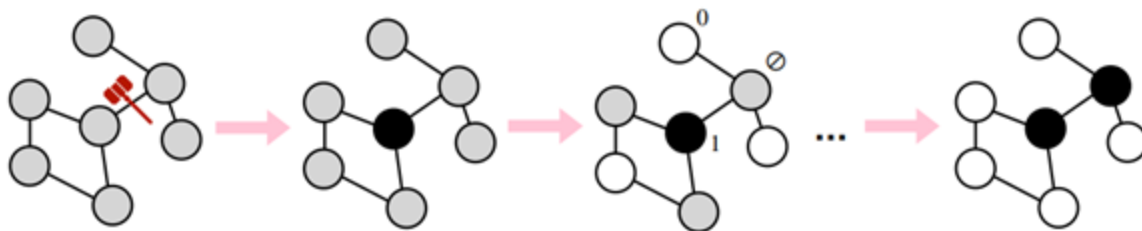
*GFlowNets for AI-Driven Scientific Discovery.* Moksh Jain, et al. Digital Discovery 2023.



# Combinatorial Optimization Problems

For instance, finding the **largest clique** in a UGM.

- **States:** sets of fully connected nodes.
- **Transitions:** addition of a basic molecule.
- **Reward:** size of the set.

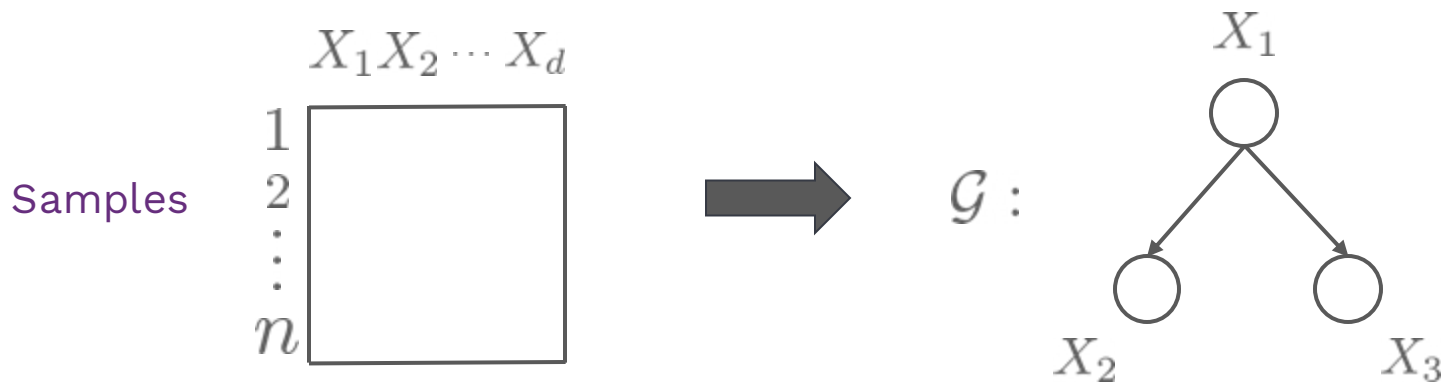


*Let the Flows Tell: Solving Graph Combinatorial Optimization Problems with GFlowNets.* Dinghuai Zhang, et al. NeurIPS 2023.

# Structure Learning

Let  $X_1, X_2, \dots, X_d$  be rvs. We want to construct a “minimal” graph  $\mathcal{G}$  such that:

$$p(x_1, x_2, \dots, x_d) \in \mathcal{L}(\mathcal{G})$$

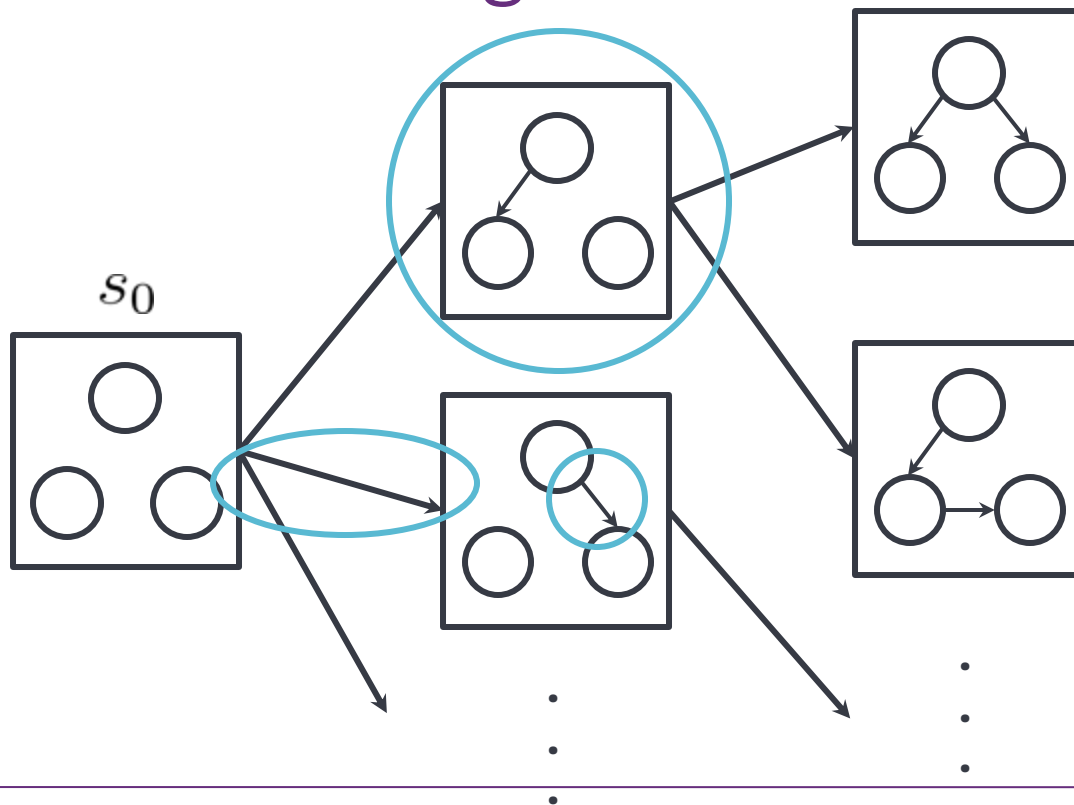


# Structure Learning

- **States:** DAGs.
- **Transitions:** addition of a directed edge.
- **Reward:** likelihood of data given the graph.

The transitions exploit the compositional aspect. Instead of looking at all possible DAGs without structure, the GFlowNet depth equals the **number of edges** in the DAG.

# Structure Learning



# Flow Networks

# Flow Networks

- DAG  $G = (\mathcal{S}, \mathbb{A})$  with source and sink states,  $s_0, s_f \in \mathcal{S}$
- Trajectories  $\tau = (s_1, \dots, s_n)$ , where  $s_t \rightarrow s_{t+1} \in \mathbb{A}$
- Let  $\mathcal{T}$  be the set of all trajectories.
- Forward transition probabilities  $\hat{P}_F(s' | s)$

$\hat{P}_F(s' | s)$  is a probability

Chain rule

$$\sum_{s' \in \text{Child}(s)} \hat{P}_F(s' | s) = 1$$

$$\hat{P}_F(\tau) := \prod_{t=1}^{n-1} \hat{P}_F(s_{t+1} | s_t)$$

# Flow Networks

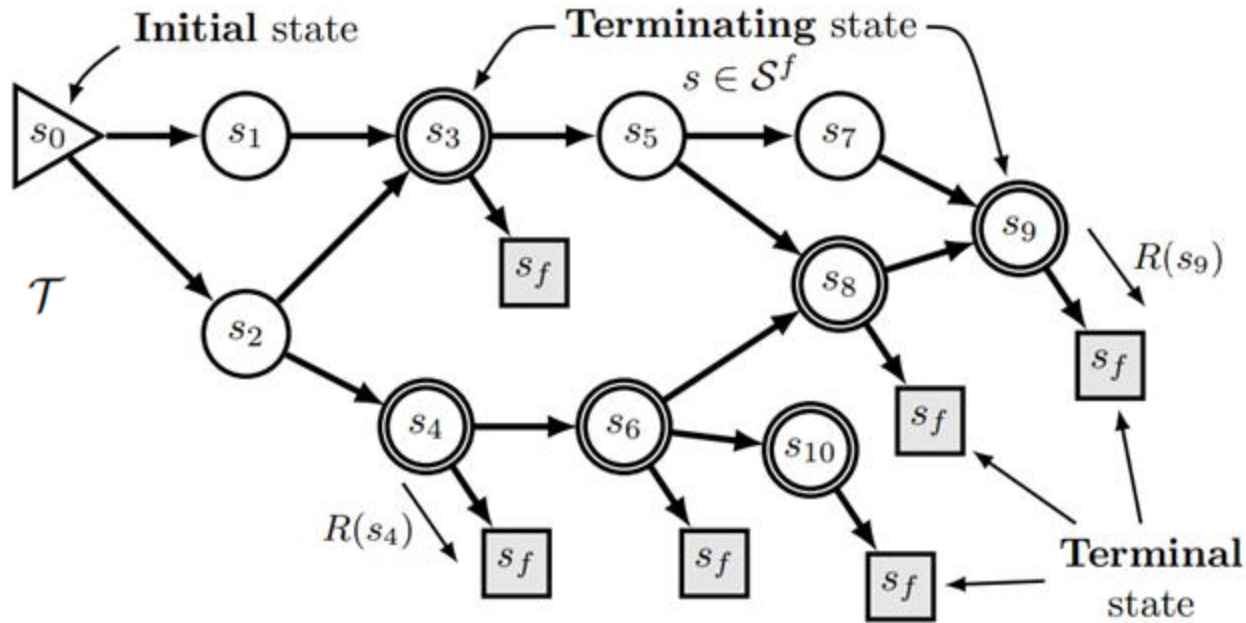
Instead of modeling probabilities, consider (unnormalized) flows:

$$F : \mathcal{T} \mapsto \mathbb{R}^+$$

$$F(s \rightarrow s') := F(\{\tau \in \mathcal{T} : s \rightarrow s' \in \tau\}) = \sum_{\tau \in \mathcal{T} : s \rightarrow s' \in \tau} F(\tau)$$

$$F(s) := F(\{\tau \in \mathcal{T} : s \in \tau\}) = \sum_{\tau \in \mathcal{T} : s \in \tau} F(\tau)$$

# Flow Networks



\*Fig 2 in GFlowNet Foundations



# Flow Networks

The flow induces a distribution over the **terminating states**.

$$F(s_0) = \sum_{\tau \in \mathcal{T}} F(\tau) = Z,$$

$$F(s_f) = \sum_{\tau \in \mathcal{T}} F(\tau) = Z.$$

$$P(s) := \frac{F(s)}{Z}$$

$$P(s \rightarrow s' \mid s) = \frac{F(s \rightarrow s')}{F(s)}$$

# Flow Networks

The flow induces a distribution over the **terminating states**.

$$P_T(s) := P(s \rightarrow s_f) = \frac{F(s \rightarrow s_f)}{Z} \quad \sum_{s \in S^f} P_T(s) = 1.$$

# Flow Networks

If we learn a flow  $\mathbf{F}$  that matches a reward function on terminating states:

- A probability distribution over terminating states follows.
- The probabilities are proportional to the reward.
- And the graph  $\mathbf{G}$  can be used to efficiently **sample**.

But parameterizing a flow is expensive! -> One value per trajectory in  $\mathbf{G}$ .

# Markovian Flow Networks

Instead, consider **Markovian** flow networks:

$$P(s \rightarrow s' \mid \tau) = P(s \rightarrow s' \mid s)$$

- Cheaper to model.
- Induces a unique forward transition probability  $P(s'|s)$ .
- **Proposition 23** in **GFlowNet Foundations**: the set of markovian flows is expressive enough to represent **all flow functions** over trajectories.

# Markovian Flow Networks

- Even Markovian flows are expensive (need flow over every node).
- Approximate  $\mathbf{F}$  with a neural network respecting flow conservation:

$$F(s) = \sum_{s' \in Child(s)} F(s \rightarrow s')$$

$$F(s') = \sum_{s \in Par(s')} F(s \rightarrow s')$$

# Generative Flow Networks - GFlowNets

A **GFlowNet** is a (Markovian) flow network where:

$$\forall s \in \mathcal{S}^f \quad F(s \rightarrow s_f) = R(s)$$

- **$R$**  is a given reward function.
  - If  **$s$**  is not a valid terminating state, set a reward of 0.
- **$F$**  is parameterized with (say) a NN.

# Training GFlowNets

# Training GFlowNets

The NeurIPS 2021 GFlowNet paper enforces *flow matching*:

$$\sum_{s \in \text{Pa}(s')} F_{\theta}(s \rightarrow s') - \sum_{s'' \in \text{Ch}(s')} F_{\theta}(s' \rightarrow s'') = R(s')$$

0 for invalid states

$F(s \rightarrow s_f)$

Which leads to the following objective:

$$\mathcal{L}(\phi) = \mathbb{E}_{\pi} \left[ \left[ \log \frac{\sum_s F_{\phi}(s \rightarrow s')}{R(s') + \sum_{s''} F_{\phi}(s' \rightarrow s'')} \right]^2 \right]$$



# Training GFlowNets

The expectation is over all trajectories -> **intractable**.

$$\mathcal{L}(\phi) = \mathbb{E}_{\pi} \left[ \left[ \log \frac{\sum_s F_{\phi}(s \rightarrow s')}{R(s') + \sum_{s''} F_{\phi}(s' \rightarrow s'')} \right]^2 \right]$$

*trajectories*


Sample trajectories instead

- If there is structure, the GFlowNet could generalize across trajectories.
- Trade-off between sampling likely trajectories and exploration.

# Training GFlowNets

The expectation is over all trajectories -> **intractable**.

$$\mathcal{L}(\phi) = \mathbb{E}_{\pi} \left[ \left[ \log \frac{\sum_s F_{\phi}(s \rightarrow s')}{R(s') + \sum_{s''} F_{\phi}(s' \rightarrow s'')} \right]^2 \right]$$

 *trajectories*

Note: this is different from supervised learning and reinforcement learning

- **SL**: the distribution over trajectories is non-stationary
- **RL**: **F** is not trained to *maximize* a reward, just to match it

# Alternative Loss Functions

Detailed balance (Bengio et al., 2021):

$$F_{\theta}(s)P_F(s' | s) = F_{\theta}(s')P_B(s | s')$$

Trajectory balance (Malkin et al., 2022):

$$Z \prod_{t=1}^n P_F(s_t | s_{t-1}) = R(s_n) \prod_{t=1}^n P_B(s_{t-1} | s_t)$$

*Trajectory Balance: Improved Credit Assignment in GFlowNets.* Nikolay Malkin et al., NeurIPS 2022.

# GFlowNets in Context

# Markov Chain Monte Carlo

## MCMC

- No “setup” cost.
- Samples are not independent
- Sampling is costly
  - Mixing time can scale poorly
  - Mode mixing

## GFlowNets

- Needs to be trained.
- Samples are independent.
- Sampling is efficient: do ancestral sampling

# Generative Modeling

## Generative Modeling

- Trained on *data*, to maximize its likelihood.
- Prone to overfitting.
- Improves with more data.

## GFlowNets

- Trained to match a reward *function*.
- Prone to underfitting.
- Improves with more trajectories.

# Reinforcement Learning

**Reinforcement Learning**

**GFlowNets**

See section 7.2 in *GFlowNet Foundations*

Thanks!



# References

- *GFlowNets for AI-Driven Scientific Discovery*. Moksh Jain, et al. Digital Discovery 2023.
- *Let the Flows Tell: Solving Graph Combinatorial Optimization Problems with GFlowNets*. Dinghuai Zhang, et al. NeurIPS 2023.
- *Trajectory Balance: Improved Credit Assignment in GFlowNets*. Nikolay Malkin et al., NeurIPS 2022.
- *GFlowNet Foundations*. Yoshua Bengio, et al. JMLR 2023.
- *Flow Network based Generative Models for Non-Iterative Diverse Candidate Generation*. Emmanuel Bengio, et al. NeurIPS 2021.
- *Bayesian Structure Learning with Generative Flow Networks*. Tristan Deleu, et al. UAI 2022.



# Generative Flow Networks and Bayesian Structure Learning

# Structure Learning

Credit to Tristan Deleu

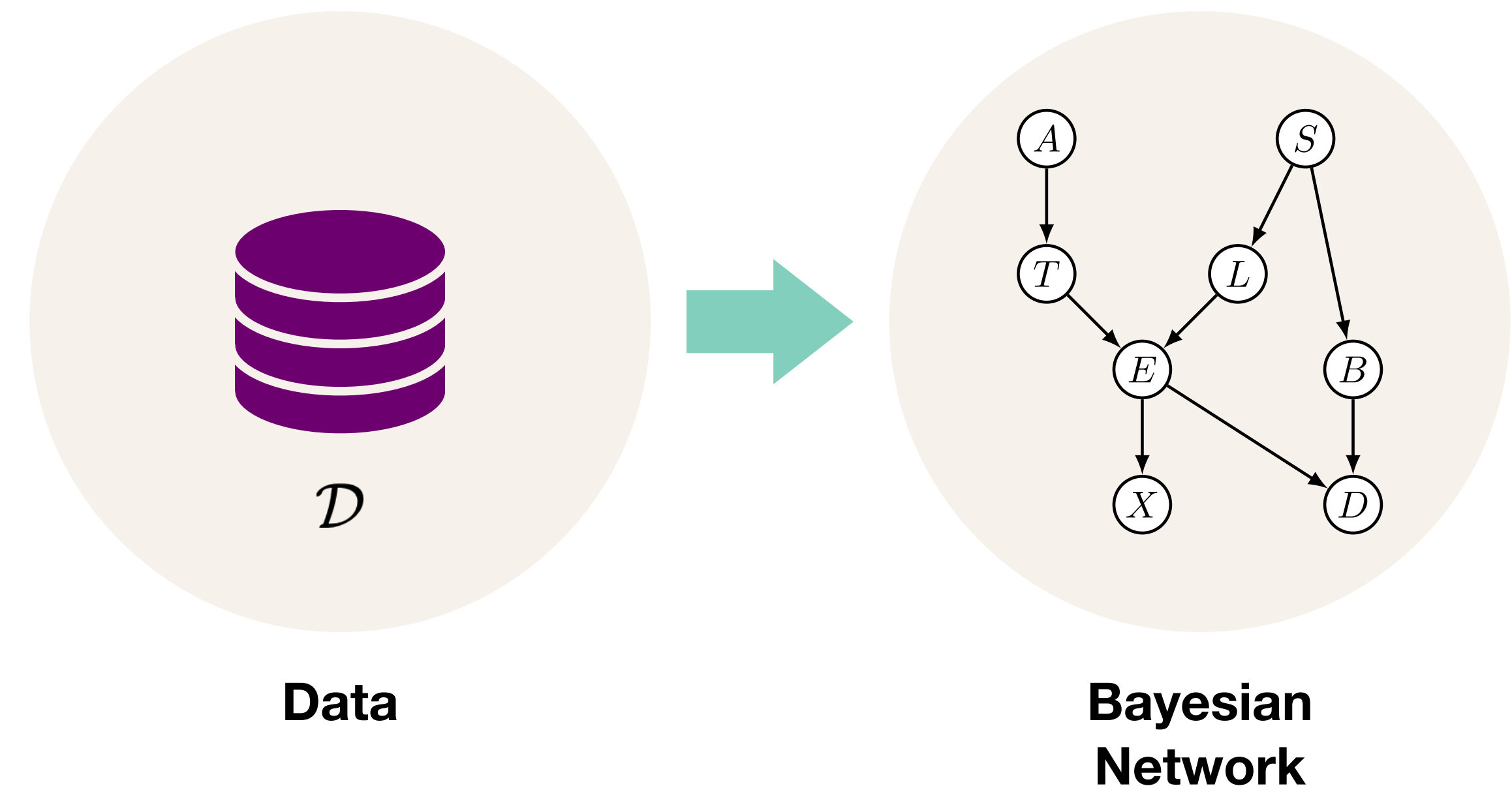
# Structure Learning

- Graphical representation of the **conditional independences** in a distribution, represented as a **Directed Acyclic Graph (DAG)**.

- The **joint distribution** is decomposed as:

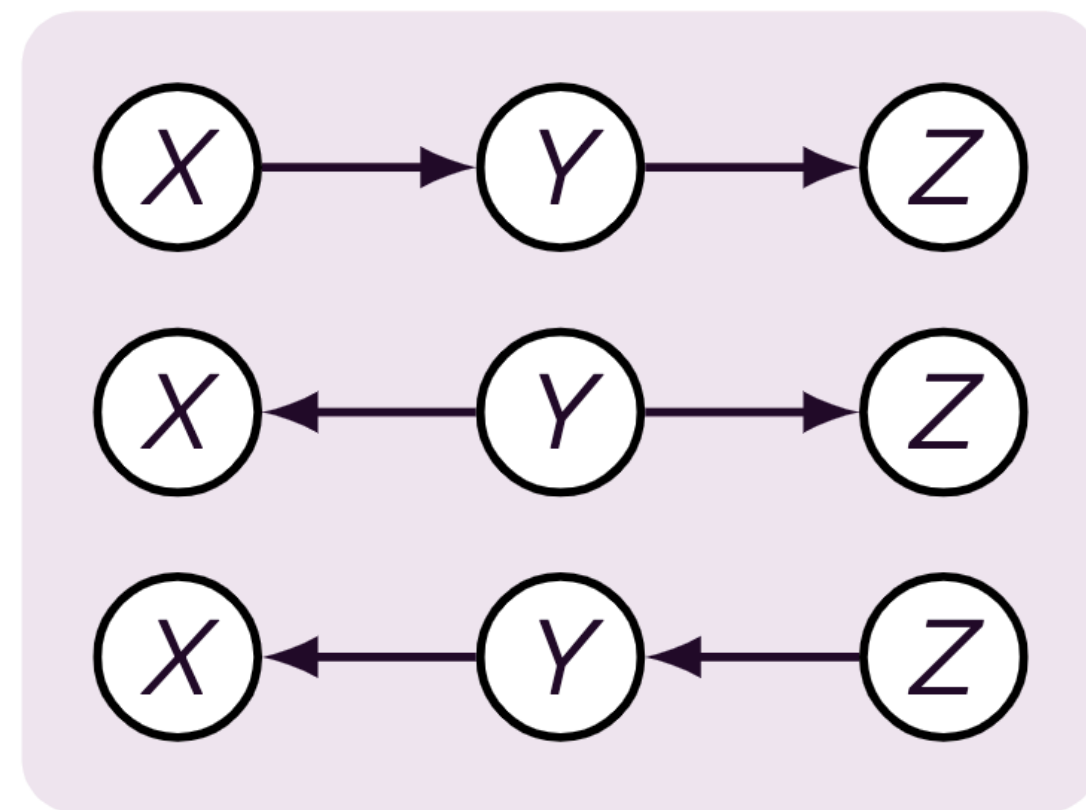
$$P(X_1, \dots, X_d) = \prod_{k=1}^d P(X_k \mid \text{Pa}_G(X_k))$$

- Structure learning:** Given a dataset of **observations**  $\mathcal{D}$ , find the graph structure  $G$ .

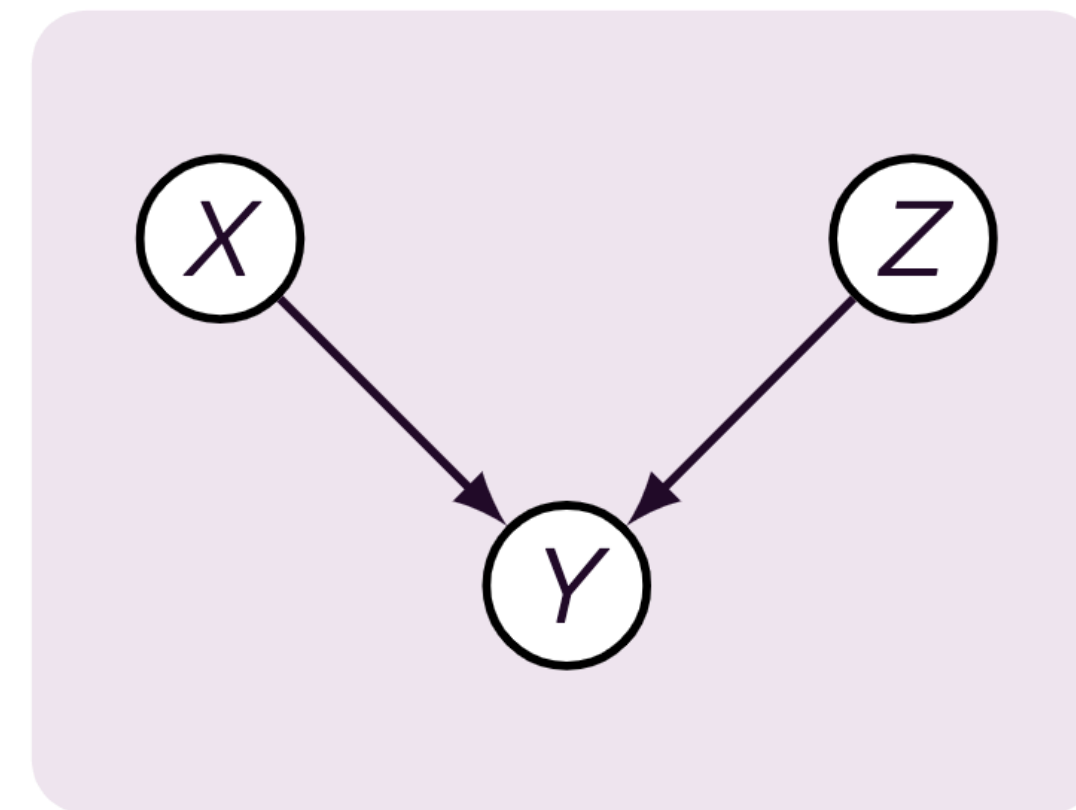


# Markov Equivalence

- Recall: A Directed Graphical Model encodes the Conditional Independence of a distribution.
- Multiple DAGs may encode the same Conditional Independence statements.



$X \not\perp\!\!\!\perp Z$  and  $X \perp\!\!\!\perp Z \mid Y$



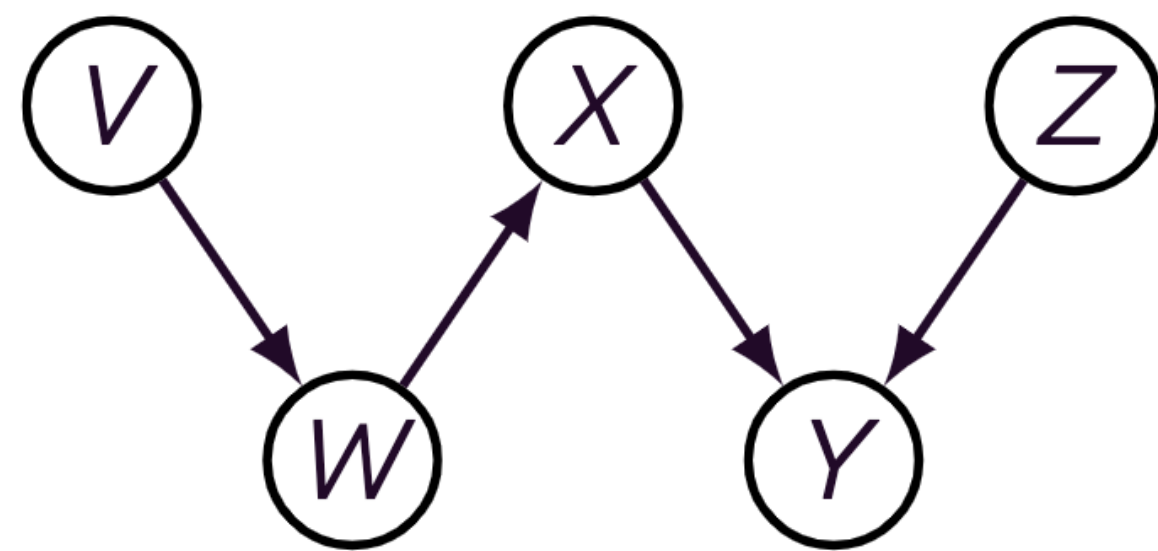
$X \perp\!\!\!\perp Z$  and  $X \not\perp\!\!\!\perp Z \mid Y$

- Two DAGs encoding the same Conditional Independence statements are called **Markov Equivalent**.

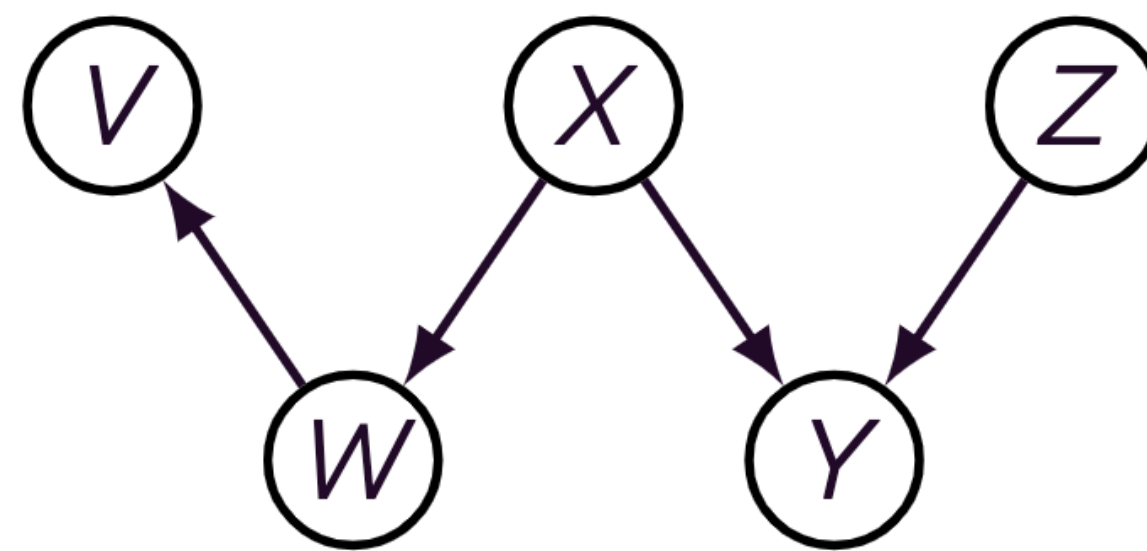
# Markov Equivalence

Theorem (Verma & Pearl, 1991)

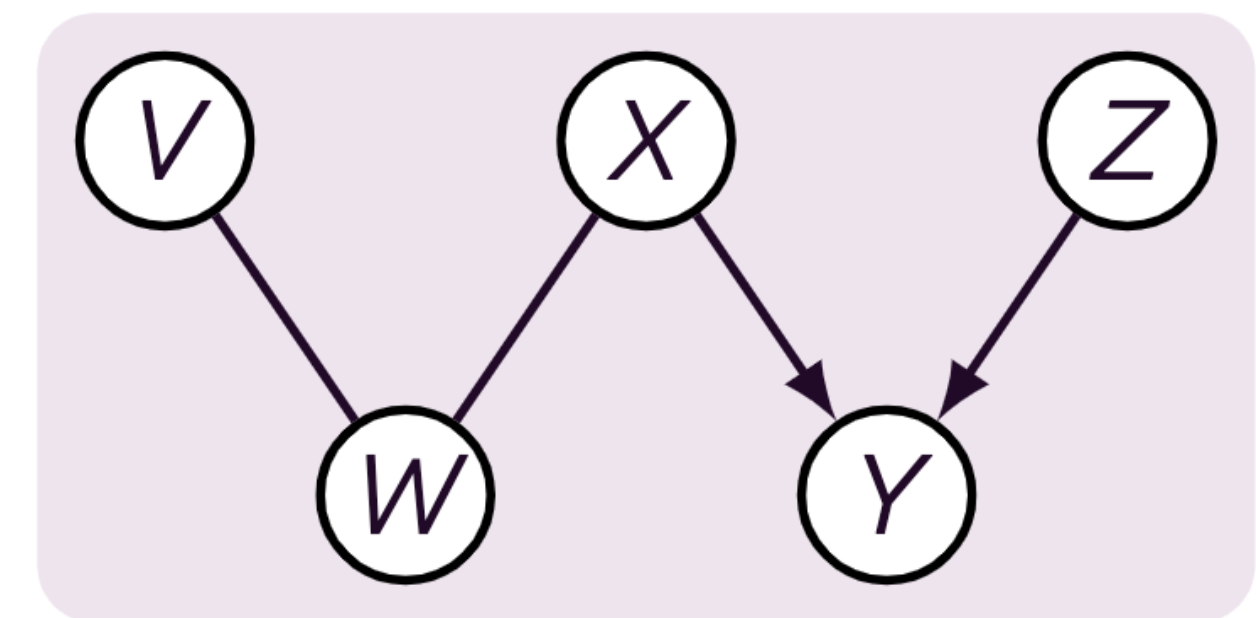
Two DAGs  $G_1$  and  $G_2$  are **Markov Equivalent** if and only if they have the same skeleton and the same  $v$ -structures.



$G_1$



$G_2$



**CPDAG**

- Markov Equivalence Classes can be represented as a **Completed Partially Directed Acyclic Graph** (CPDAG).

# Faithfulness

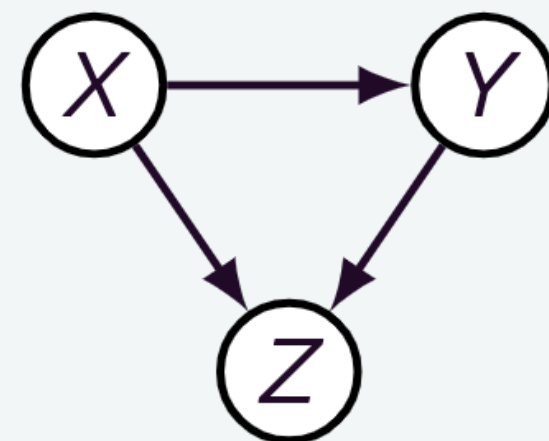
$A$  &  $B$  are d-separated  
by  $C$  in  $\mathcal{G}$

Global Markov Prop.

$$X_A \perp\!\!\!\perp X_B \mid X_C$$

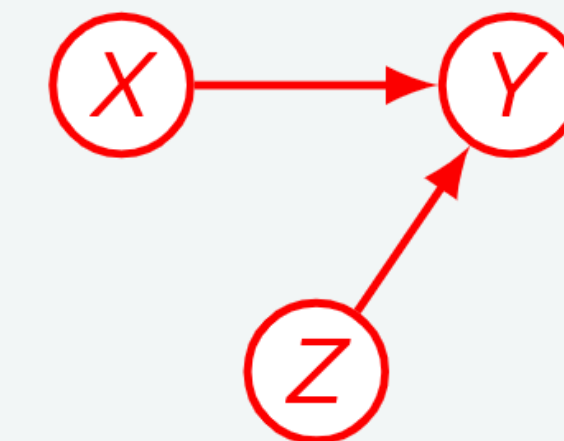
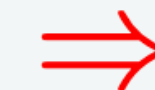
Faithfulness

## Exercise: Violation of Faithfulness



$X := N_X$   
 $Y := X + N_Y$   
 $Z := X - Y + N_Z$   
with  $N_X, N_Y, N_Z \stackrel{iid}{\sim} \mathcal{N}(0, \sigma^2)$

Structure  
Learning



$p(X, Y, Z)$  is a Multivariate Normal distribution, where the only conditional independence statements are:  $X \perp\!\!\!\perp Z$  and  $X \not\perp\!\!\!\perp Z \mid Y$ .

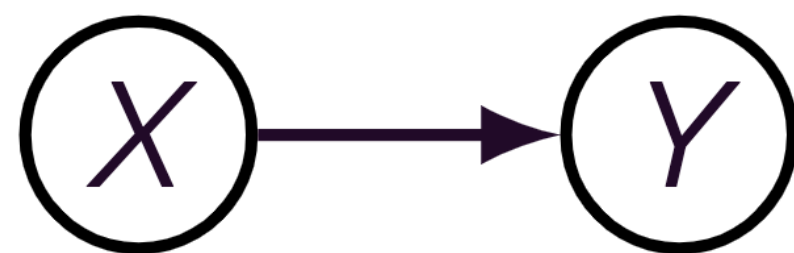


# Structure Identifiability

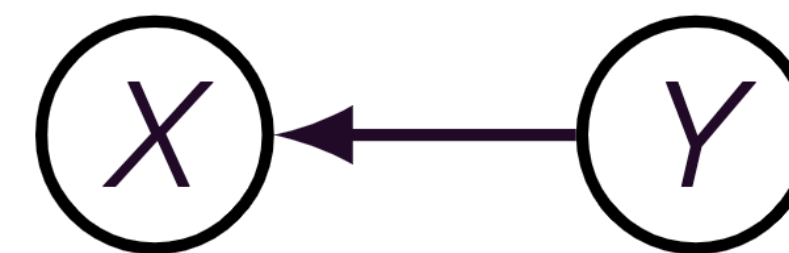
## Theorem

If  $p$  is faithful wrt.  $\mathcal{G}^0$ , then the Markov Equivalence class of  $\mathcal{G}^0$  is **identifiable** from  $p$ .

- Only the Markov Equivalence class is identifiable from observations, **not an individual graph**. Two Markov Equivalent graphs may lead to different causal conclusions!



or



- Under different assumptions, an individual DAG may be identifiable
  - Additive Noise Model (ANM):  $X_j := f_j(X_{\text{Pa}_j}) + N_j$ ,  $N_j \stackrel{iid}{\sim} \mathcal{N}(0, \sigma^2)$ , where  $f_j$  are nonlinear.
  - Using **interventional data** (i.e. data resulting from controlled experiments).

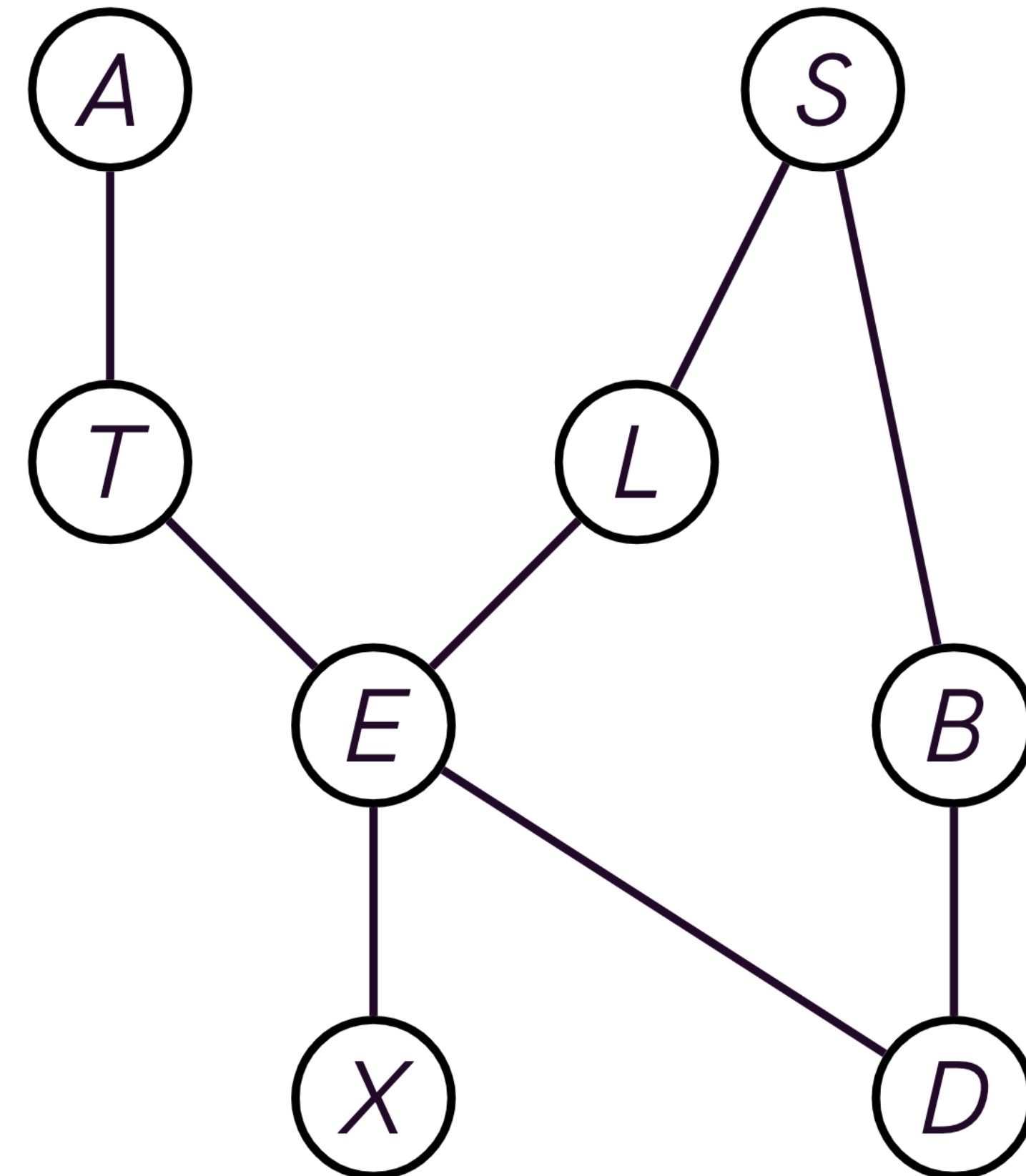


# Constraint-based methods

## Step 1: Identify the skeleton

For each pair of nodes  $X$  &  $Y$ , and  $\mathbf{A} \subseteq \mathbf{V} \setminus \{X, Y\}$ , test if  $X \perp\!\!\!\perp_{\mathcal{D}} Y \mid \mathbf{A}$ .

If there is no set  $\mathbf{A}$  s.t.  $X \perp\!\!\!\perp_{\mathcal{D}} Y \mid \mathbf{A}$ , then add an edge  $X - Y$ .



# Constraint-based methods

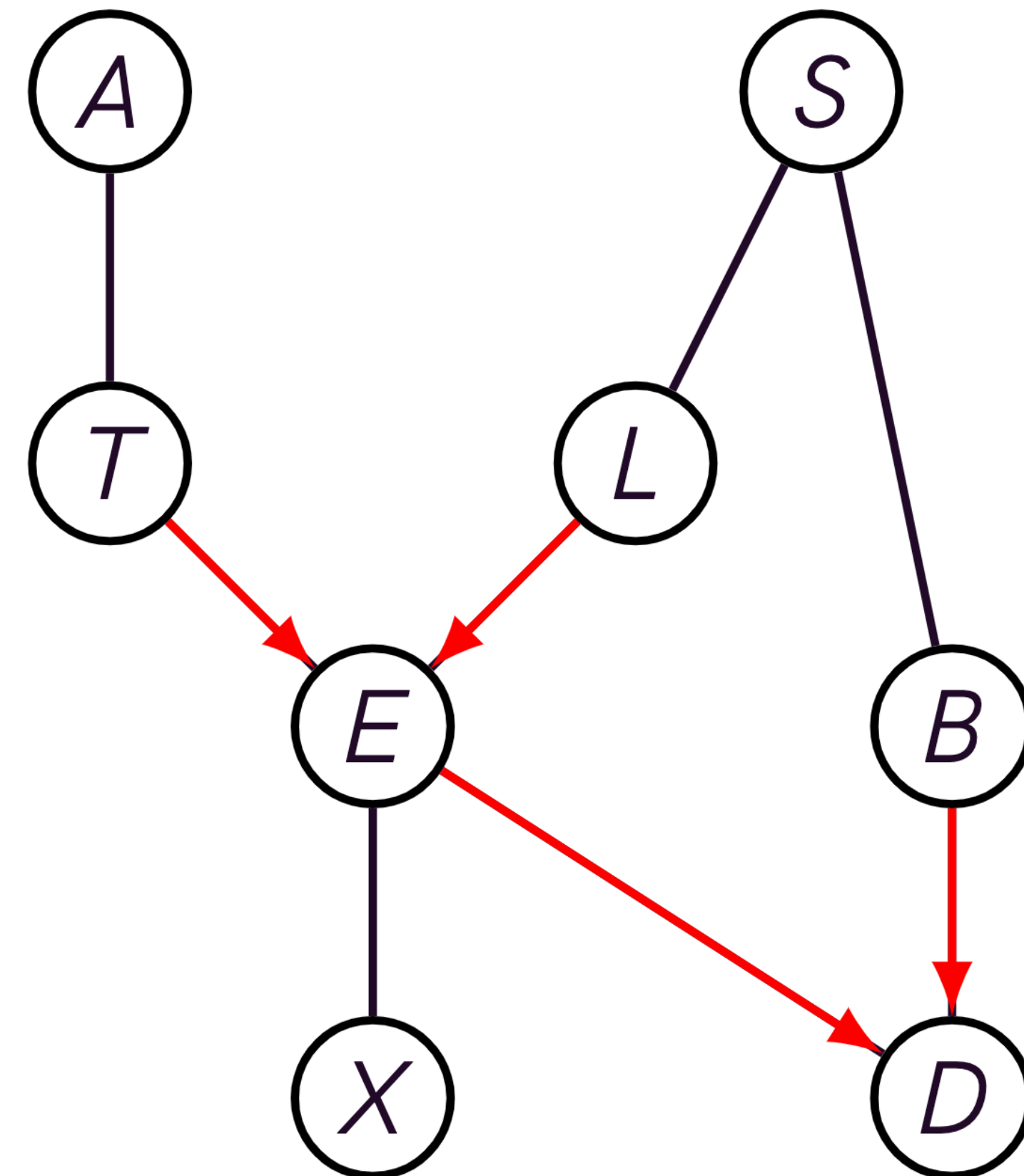
## Step 1: Identify the skeleton

For each pair of nodes  $X$  &  $Y$ , and  $\mathbf{A} \subseteq \mathbf{V} \setminus \{X, Y\}$ , test if  $X \perp\!\!\!\perp_{\mathcal{D}} Y \mid \mathbf{A}$ .

If there is no set  $\mathbf{A}$  s.t.  $X \perp\!\!\!\perp_{\mathcal{D}} Y \mid \mathbf{A}$ , then add an edge  $X - Y$ .

## Step 2: Identify the v-structures

For each structure  $X - Z - Y$  with no edge between  $X$  &  $Y$ , orient  $X \rightarrow Z \leftarrow Y$  iff  $Z \notin \mathbf{A}$ , where  $\mathbf{A}$  is such that  $X \perp\!\!\!\perp_{\mathcal{D}} Y \mid \mathbf{A}$ .

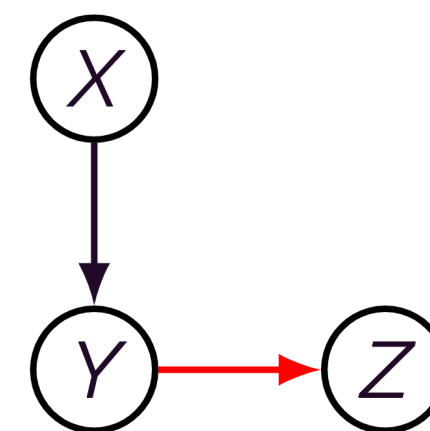
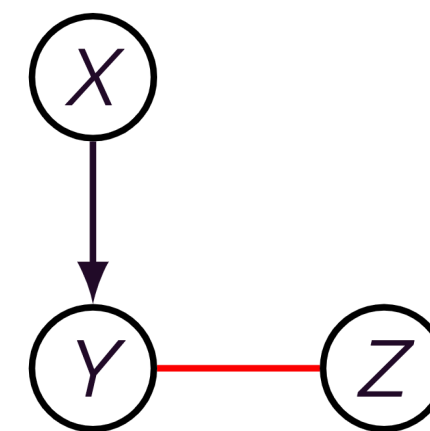


# Constraint-based methods

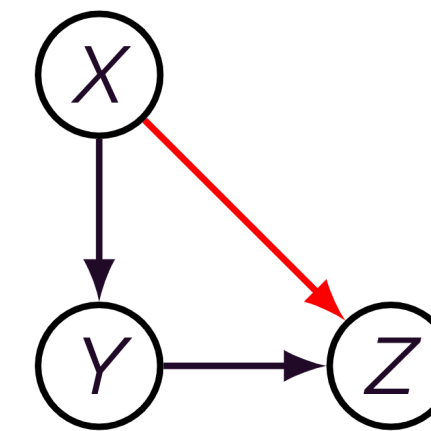
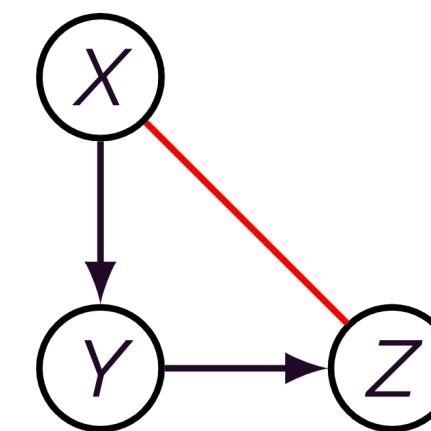
## Step 2': Additional orientations

Use **Meek's orientation rules** to orient some of the remaining edges.

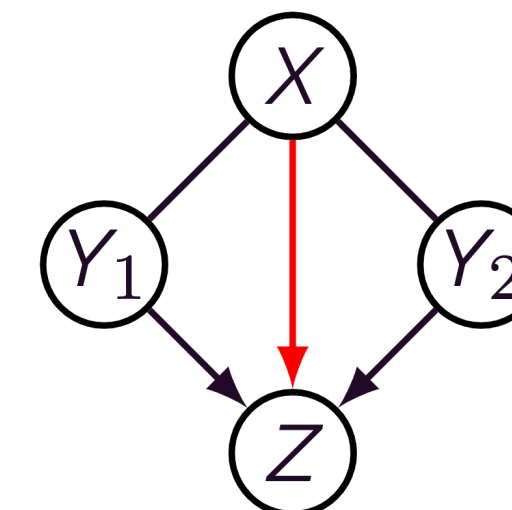
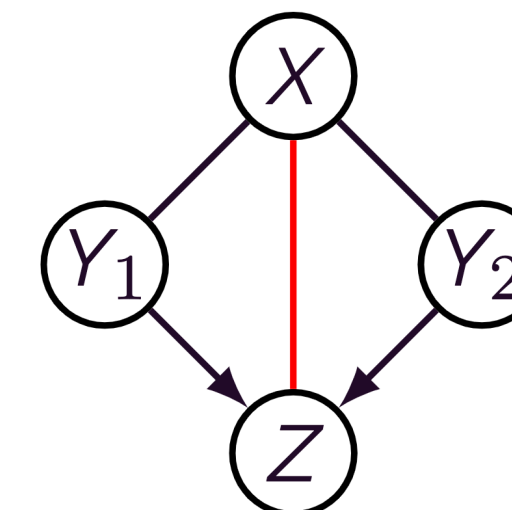
Rule 1



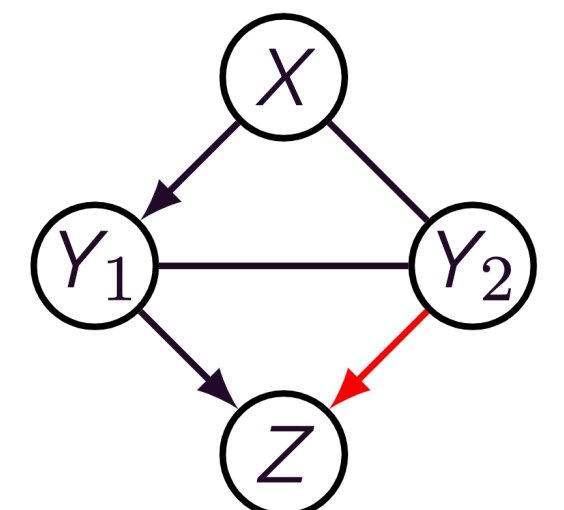
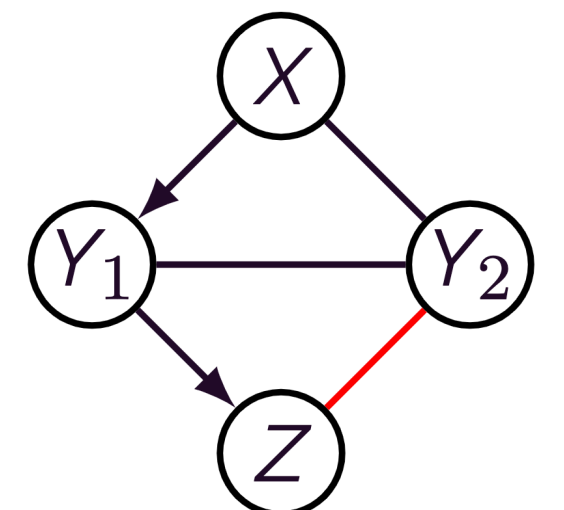
Rule 2



Rule 3



Rule 4



# Score-based methods

- Treat the problem of learning the structure of the DAG as a **model selection problem**

$$\max_{\mathcal{G} \in \text{DAG}} \text{score}(\mathcal{G} \mid \mathcal{D})$$

Choice of scores

- Likelihood score:  $\text{score}_L(\mathcal{G} \mid \mathcal{D}) = \log p(\mathcal{D} \mid \hat{\theta}_{\mathcal{G}}^{\text{MLE}}, \mathcal{G})$
- Bayesian score:  $\text{score}_B(\mathcal{G} \mid \mathcal{D}) = \log p(\mathcal{D} \mid \mathcal{G}) + \log p(\mathcal{G})$

- Bayesian Information Criterion (BIC):

$$\text{score}_{BIC}(\mathcal{G} \mid \mathcal{D}) = \log p(\mathcal{D} \mid \hat{\theta}_{\mathcal{G}}^{\text{MLE}}, \mathcal{G}) - \frac{\log N}{2} \text{Dim}[\mathcal{G}]$$

# Score-based methods

$$\max_{\mathcal{G} \in \text{DAG}} \text{score}(\mathcal{G} \mid \mathcal{D})$$

- How to search over the space of DAGs?
- The number of DAGs over  $n$  nodes is **super-exponential** in  $n$ :  $2^{\Theta(n^2)}$

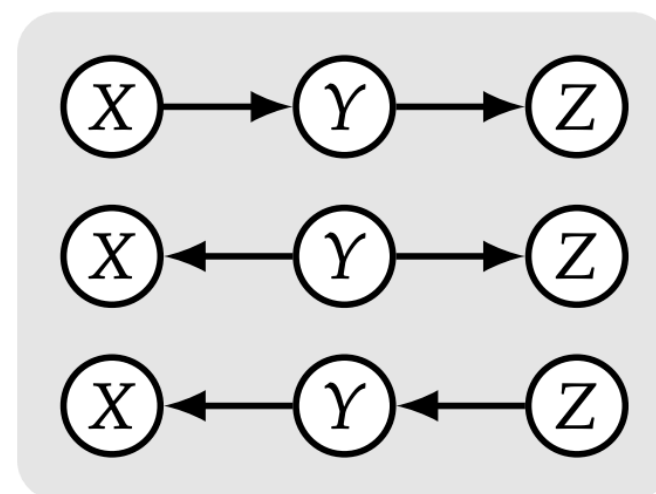
## Theorem

*Let  $G_{\leq d} = \{\mathcal{G} \text{ a DAG} \mid \text{every node has at most } d \text{ parents}\}$ . Finding a DAG in  $G_{\leq d}$  that maximizes a score is **NP-hard** for  $d \geq 2$ .*

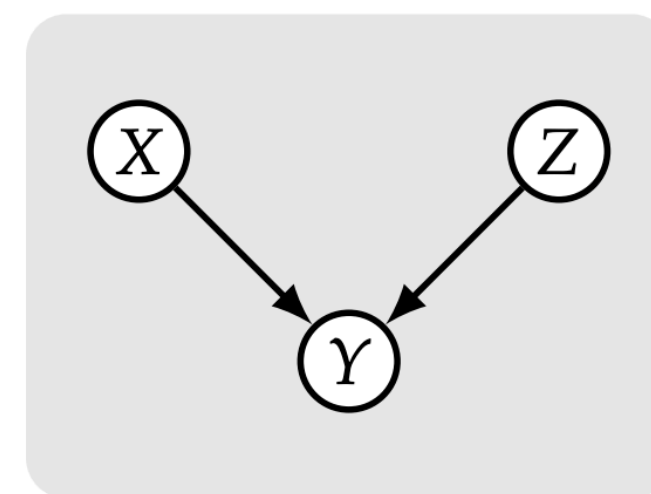
- Heuristic solutions:
  - Greedy algorithms: Hill climbing, GES
  - Genetic algorithms
  - Constrained continuous optimization: NOTEARS, Gran-DAG, DCDI, etc...

# Bayesian Structure Learning

- When the **dataset is small**, we want to take into account the **epistemic uncertainty** over the graph structures of the Bayesian Network.
- **Markov Equivalence:** There may be multiple graphs encoding the **same conditional independences**.



$X \not\perp\!\!\!\perp Z$  and  $X \perp\!\!\!\perp Z \mid Y$



$X \perp\!\!\!\perp Z$  and  $X \not\perp\!\!\!\perp Z \mid Y$

- From the point of view of observations, Markov equivalent graphs **fit the data equally well**.

## Bayesian Structure Learning:

Instead of finding a single graph from observations, characterize the whole **posterior distribution** over graphs:

$$P(G \mid \mathcal{D}) = \frac{P(\mathcal{D} \mid G)P(G)}{P(\mathcal{D})}$$



# Bayesian Structure Learning

**Bayesian Structure Learning:**  
Instead of finding a single graph from observations, characterize the whole **posterior distribution** over graphs:

$$P(G \mid \mathcal{D}) = \frac{P(\mathcal{D} \mid G)P(G)}{P(\mathcal{D})}$$

Graphs are **discrete** and **composite** objects  
*The number of DAGs is **super-exponential** in the number of nodes (eg. there are  $10^{72}$  DAGs over 15 nodes)*

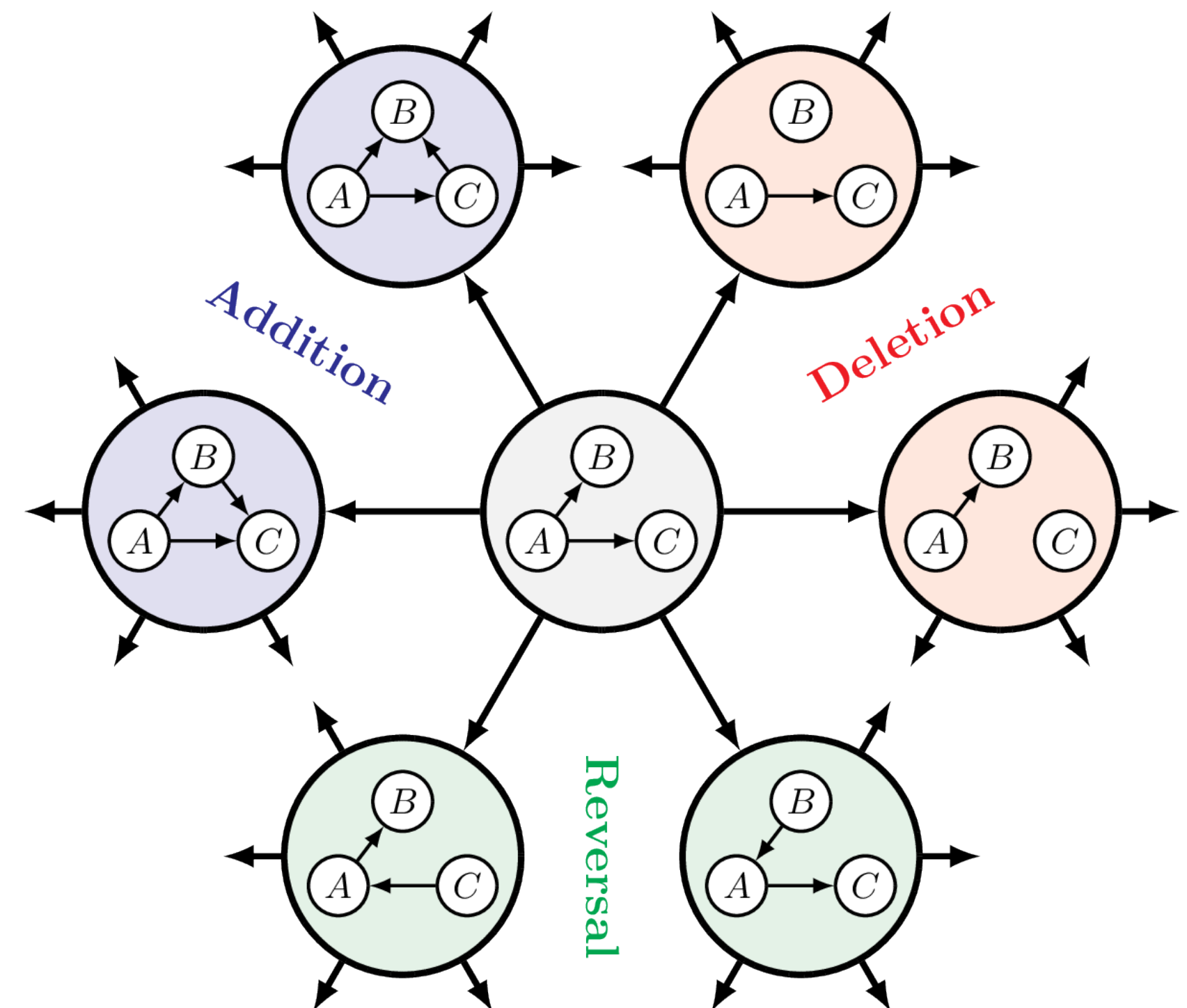
The **marginal likelihood** is in general **intractable**

$$P(\mathcal{D} \mid G) = \int_{\Theta} P(\mathcal{D} \mid \theta, G)P(\theta \mid G)d\theta$$

*We will choose models so that this can be computed efficiently in **closed form**.*

# Markov Chain Monte Carlo

- Approximate the posterior distribution using **Markov Chain Monte Carlo (MCMC)**.
- Build a Markov chain by **adding, removing, or reversing edges** uniformly at random.
- **Issue: Highly multimodal distribution** (Markov equivalence), leading to **poor mixing** of the Markov chain.

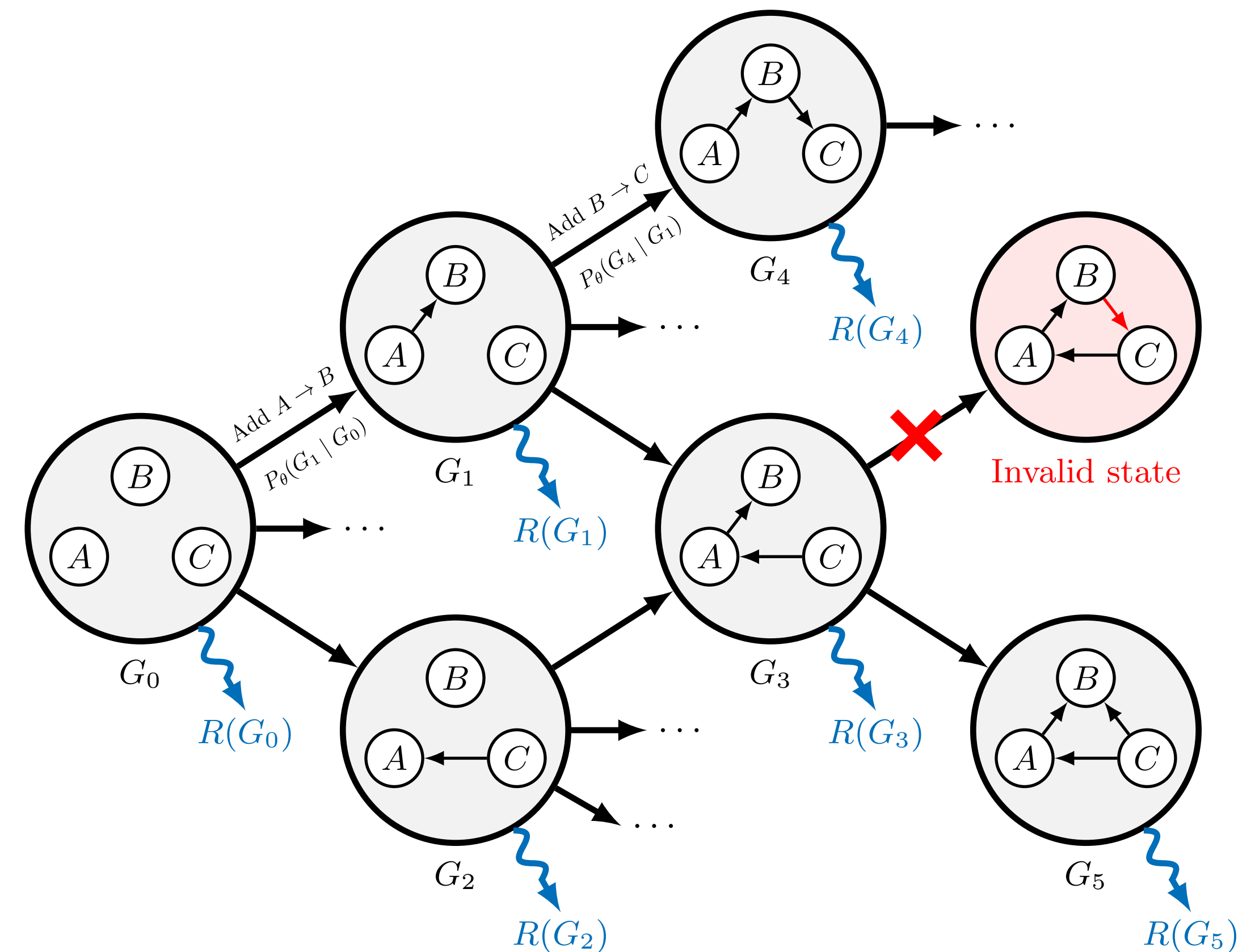




# DAG-GFlowNet

# GFlowNet over DAGs

- DAGs are constructed sequentially **one edge at a time**, starting from the **empty graph**.
- All the states of the GFlowNet are **valid DAGs**, meaning that **all the states are terminating**.
- A new edge to be added to a DAG:
  - must not **already be present**;
  - must not **introduce a cycle**.
- We can filter out invalid actions using a **mask**, that can also be **updated online**.



# Detailed balance condition

**Flow matching condition** (Bengio et al., 2021)

$$\sum_{s \in \text{Pa}(s')} F_{\theta}(s \rightarrow s') - \sum_{s'' \in \text{Ch}(s')} F_{\theta}(s' \rightarrow s'') = R(s')$$



$$P(s_{t+1} \mid s_t) \propto F_{\theta}(s_t \rightarrow s_{t+1})$$

**Detailed balance condition** (Ours)

$$R(s') P_B(s \mid s') P_{\theta}(s_f \mid s) = R(s) P_{\theta}(s' \mid s) P_{\theta}(s_f \mid s')$$



Fixed **backward**  
transition probability  
*e.g. Uniform distribution*

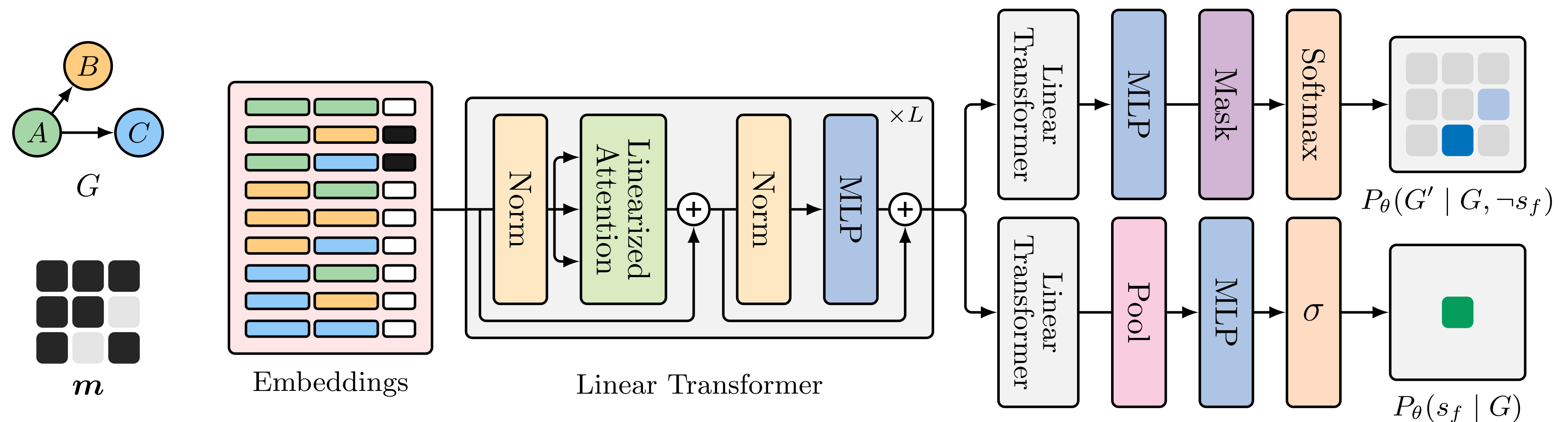


Learned **forward**  
transition probability

- ✓ Valid when **all the states** of the GFlowNet **are terminating**
- ✓ Induces a **distribution**  $P(s) \propto R(s)$
- ✓ It **does not depend on flows** anymore (flow-matching or detailed balance conditions).
- ✓ It **does not depend on the total flow**  $Z$  (trajectory balance condition).

# Forward Transition Probabilities

**Hierarchical model** for the forward transition probabilities:  $P_{\theta}(G' | G) = (1 - P_{\theta}(s_f | G))P_{\theta}(G' | G, \neg s_f)$



- ✓ Independent of the **order** of edges.
- ✓ **Set-to-set** architecture.
- ✓ The **number of parameters** does not scale too much with the size of the graph.
- ✓ **No quadratic scale** with the input size.

# Application to Bayesian Structure Learning

## Bayesian Structure Learning

Characterize the *posterior distribution* over DAGs

$$P(G \mid \mathcal{D}) = \frac{P(\mathcal{D} \mid G)P(G)}{P(\mathcal{D})}$$

## GFlowNet

A GFlowNet induces a *distribution*

$$P(s) \propto R(s)$$

## DAG-GFlowNet

$$R(G) = P(\mathcal{D} \mid G)P(G)$$

# Tools from Reinforcement Learning

The GFlowNet is trained **off-policy**

We use a **replay buffer** to store transitions over the course of training, and **sample transitions** from the replay buffer

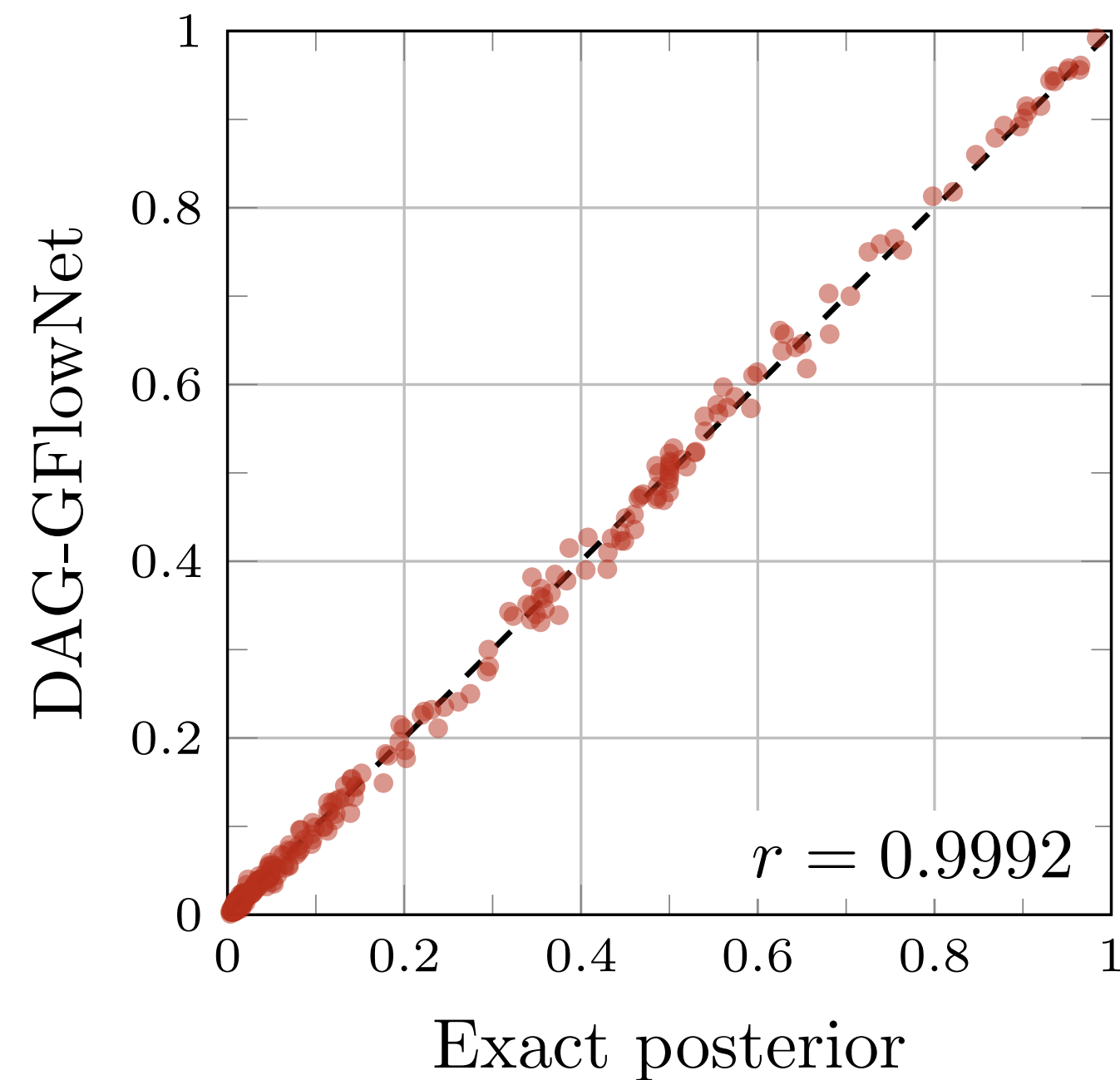
$$\mathcal{L}(\theta) = \mathbb{E}_{\pi} \left[ \left[ \log \frac{R(G') P_B(G \mid G') P_{\theta}(s_f \mid G)}{R(G) P_{\theta}(G' \mid G) P_{\theta}(s_f \mid G')} \right]^2 \right]$$

# Experimental results

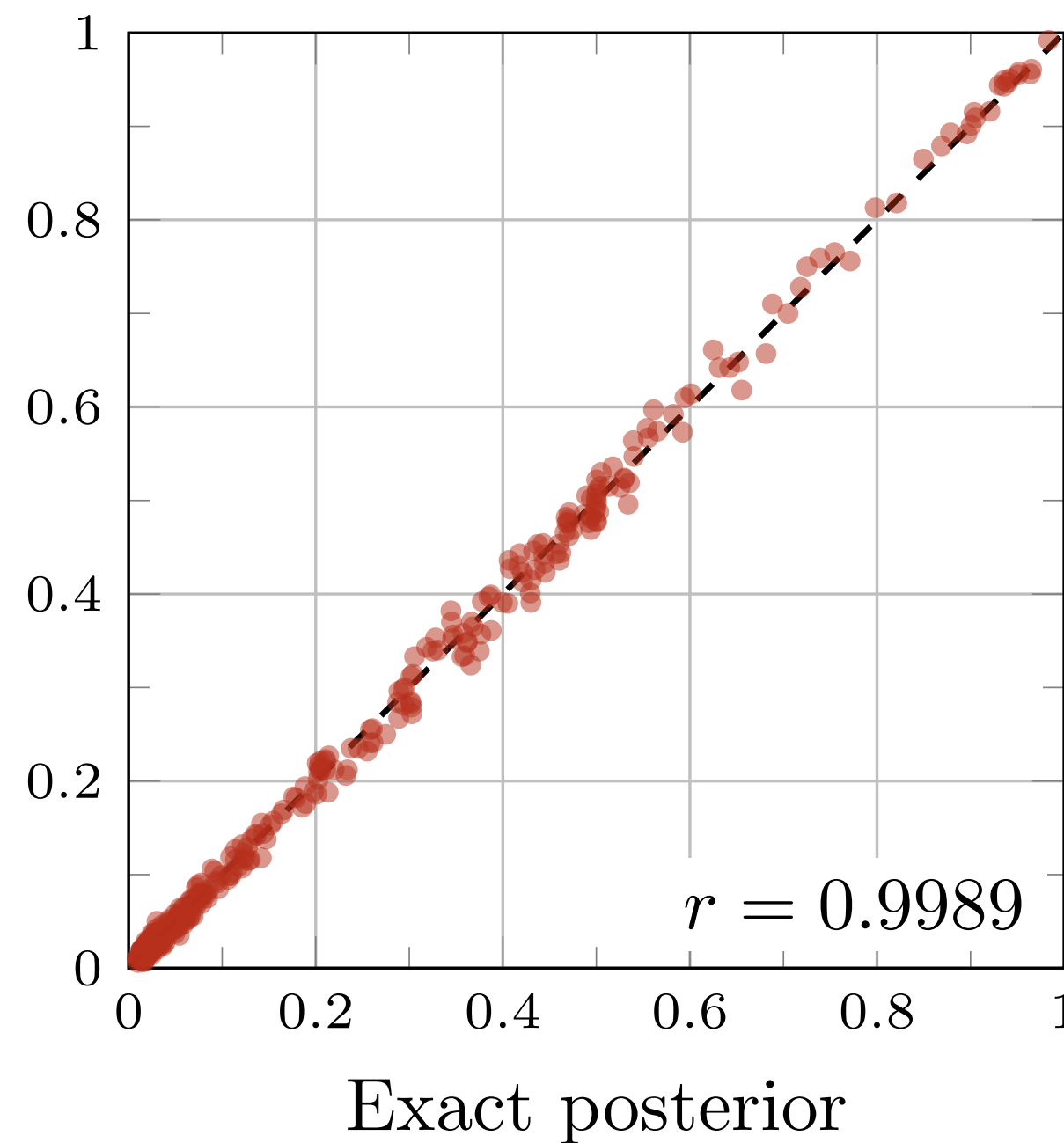


# Experimental results – Accurate approximation

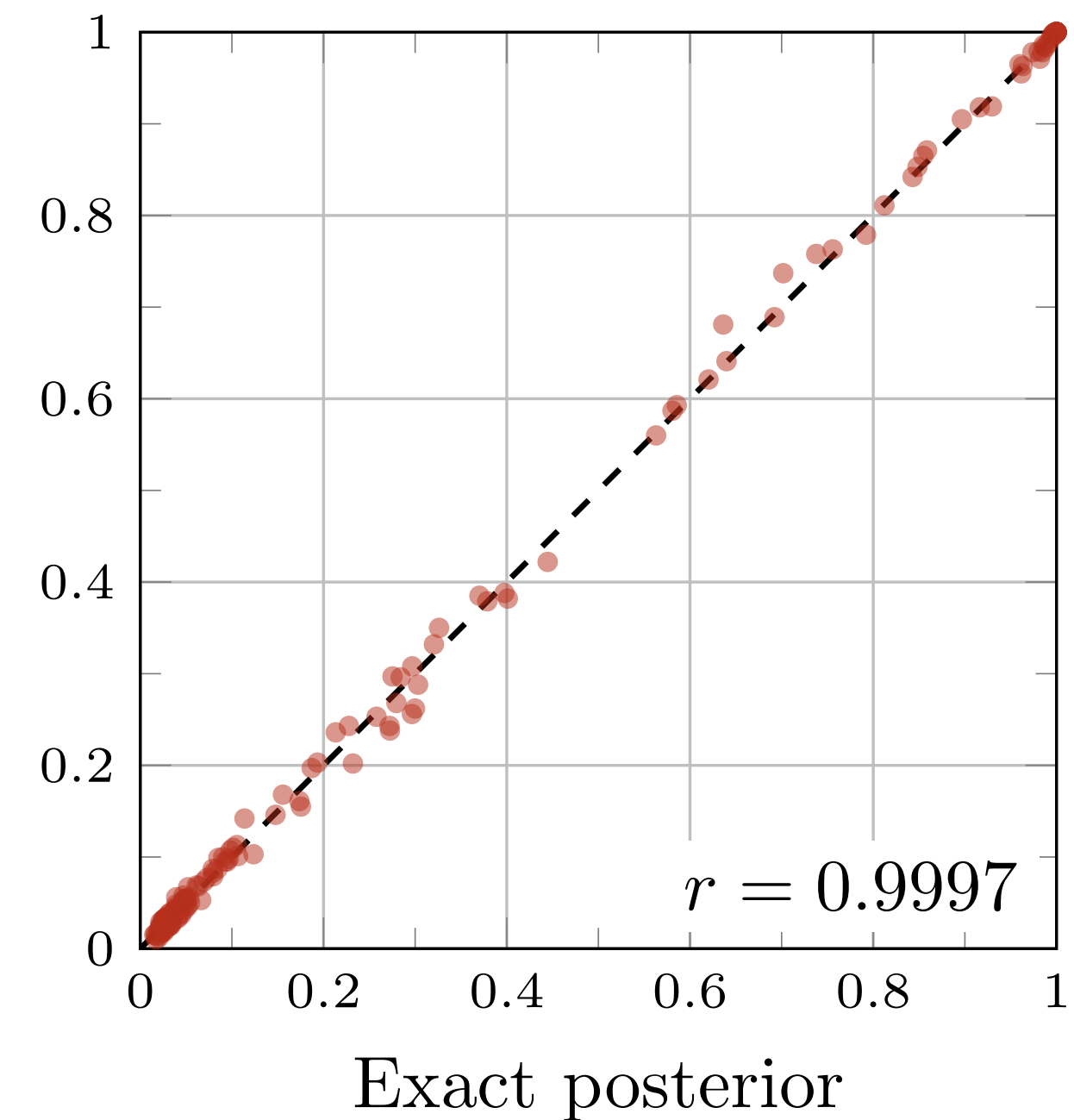
Comparison with the **exact posterior** distribution  $P(G \mid \mathcal{D})$  on graphs with  $d = 5$  nodes, computed by enumerating the **29,281 possible DAGs**.



(a) Edge features



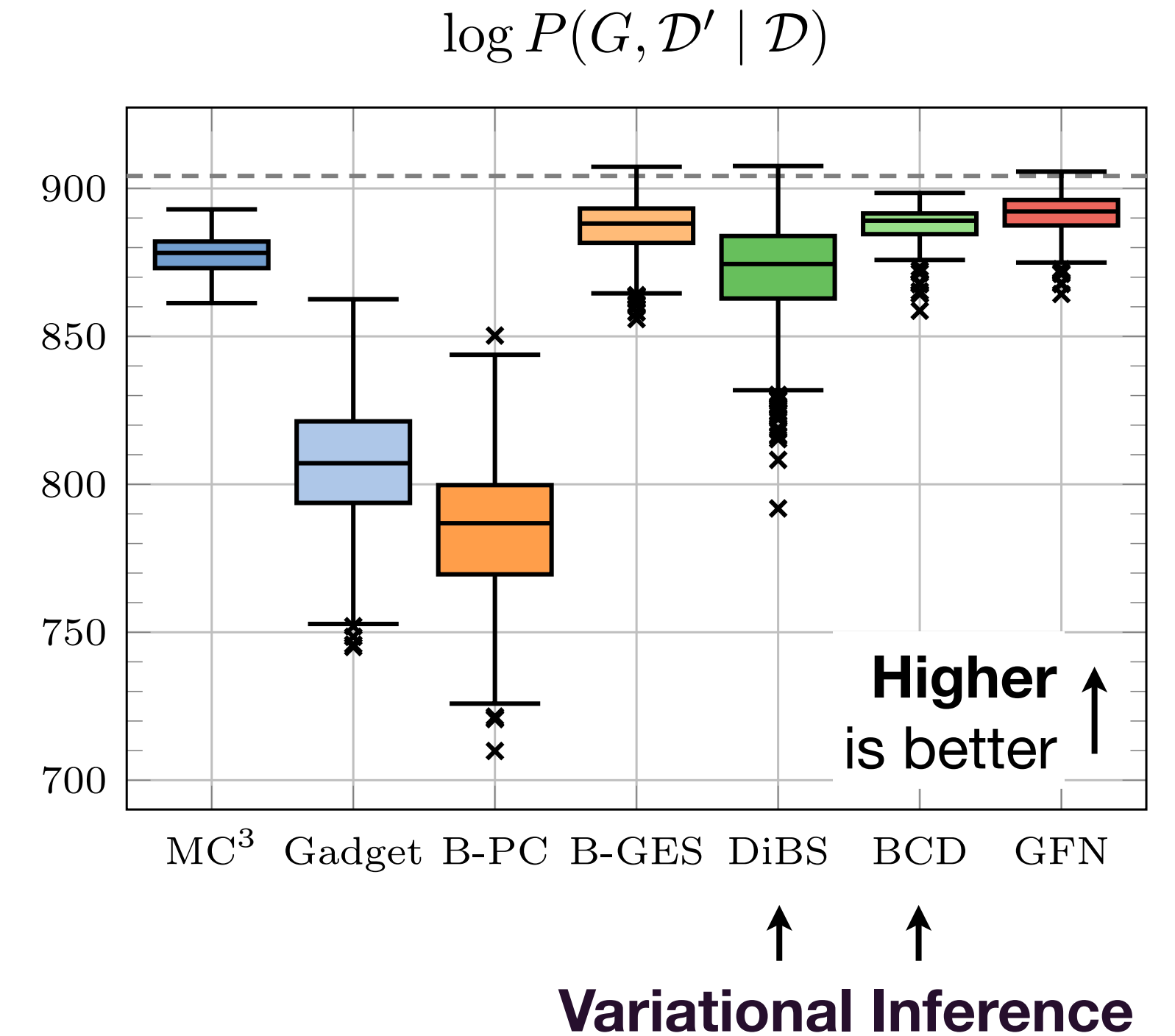
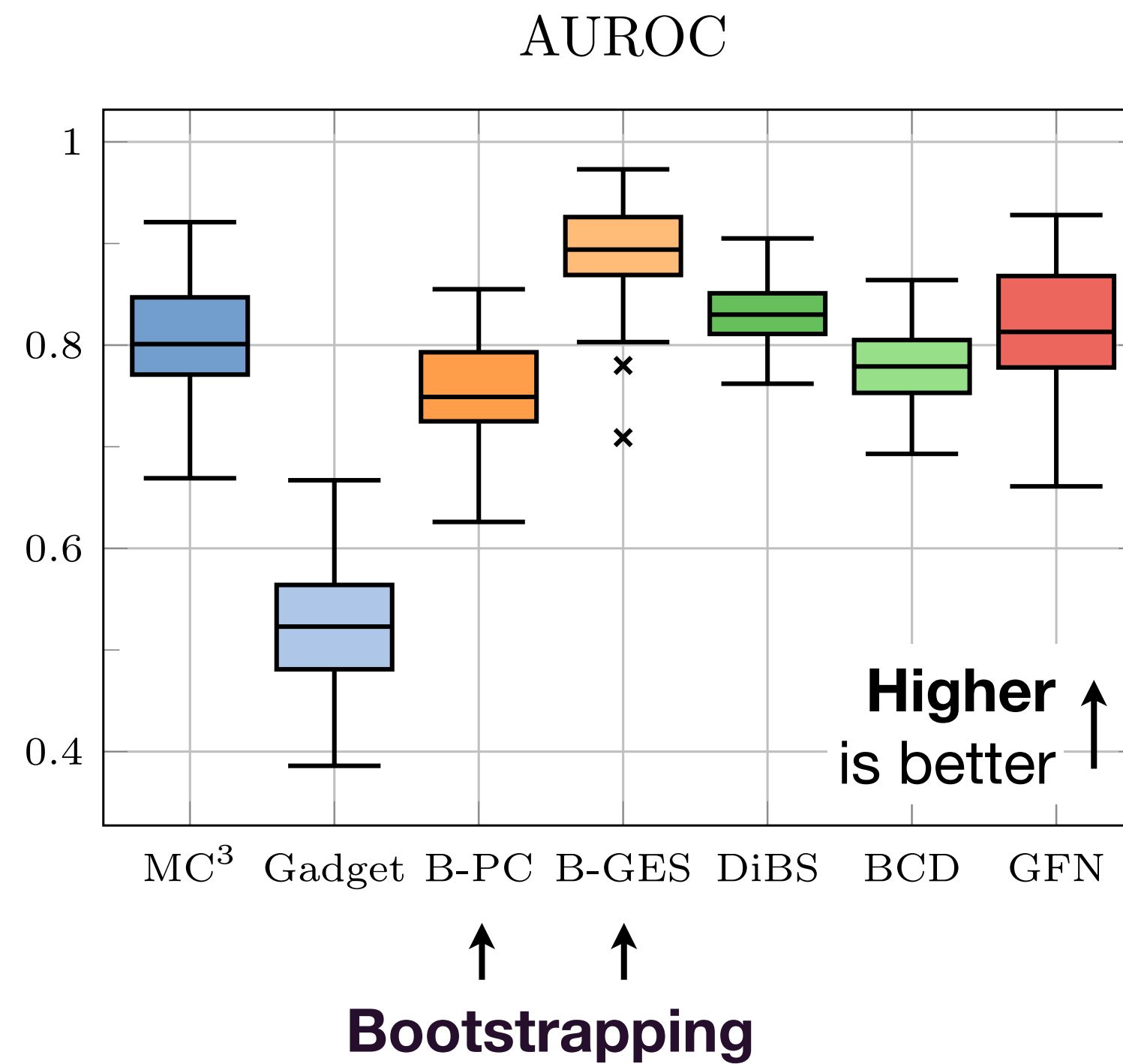
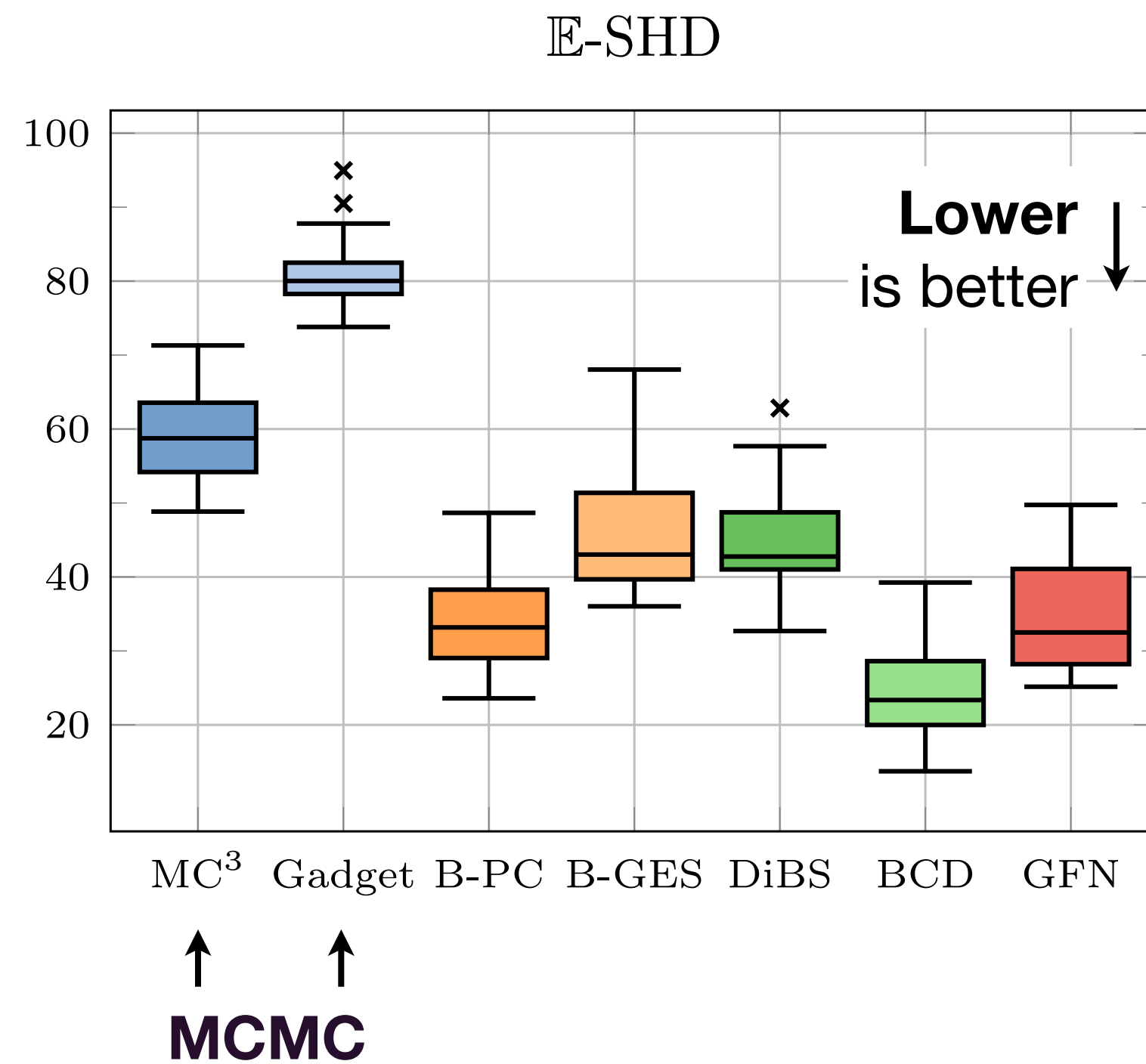
(b) Path features



(c) Markov features

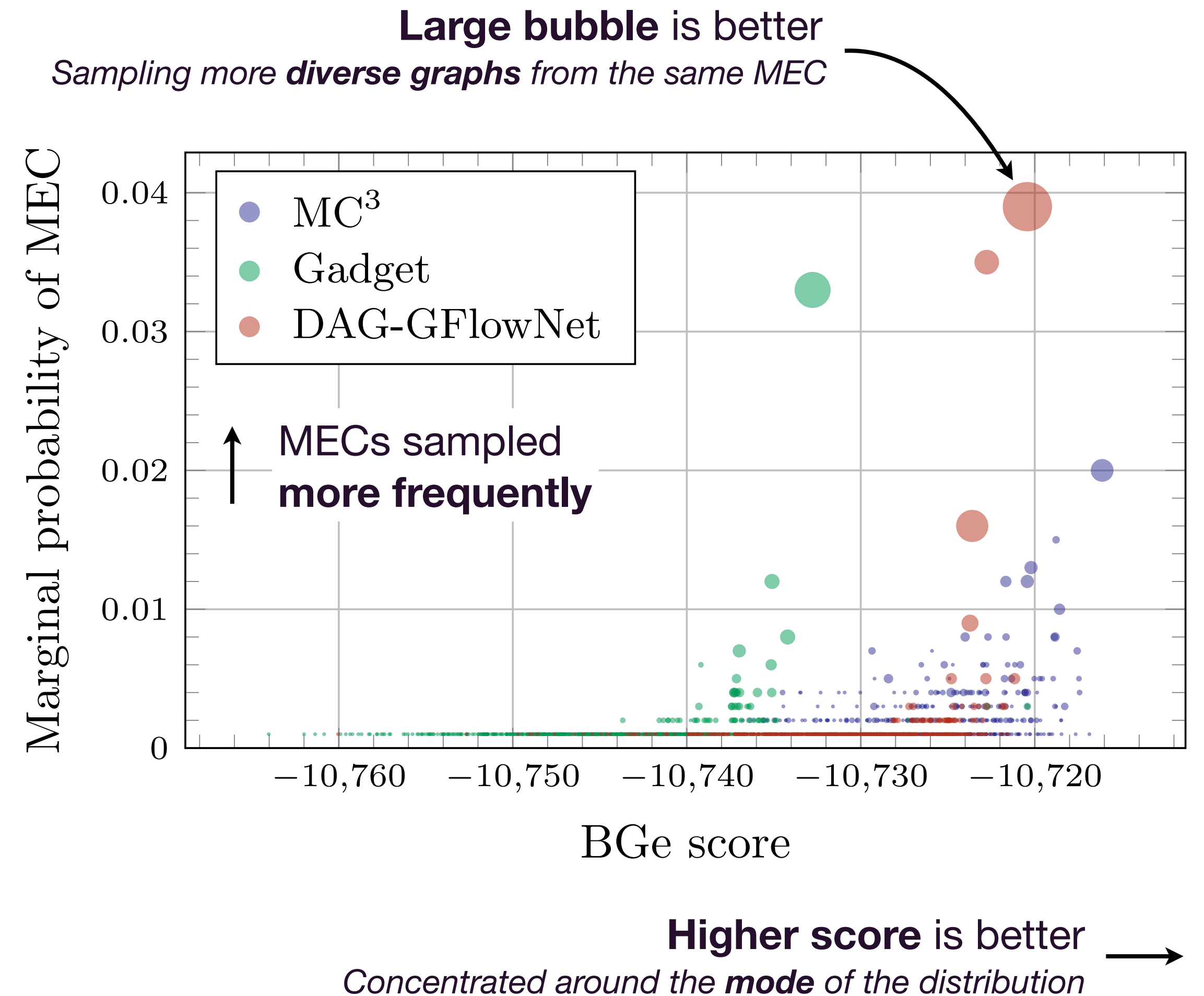


# Experimental results – Simulated data

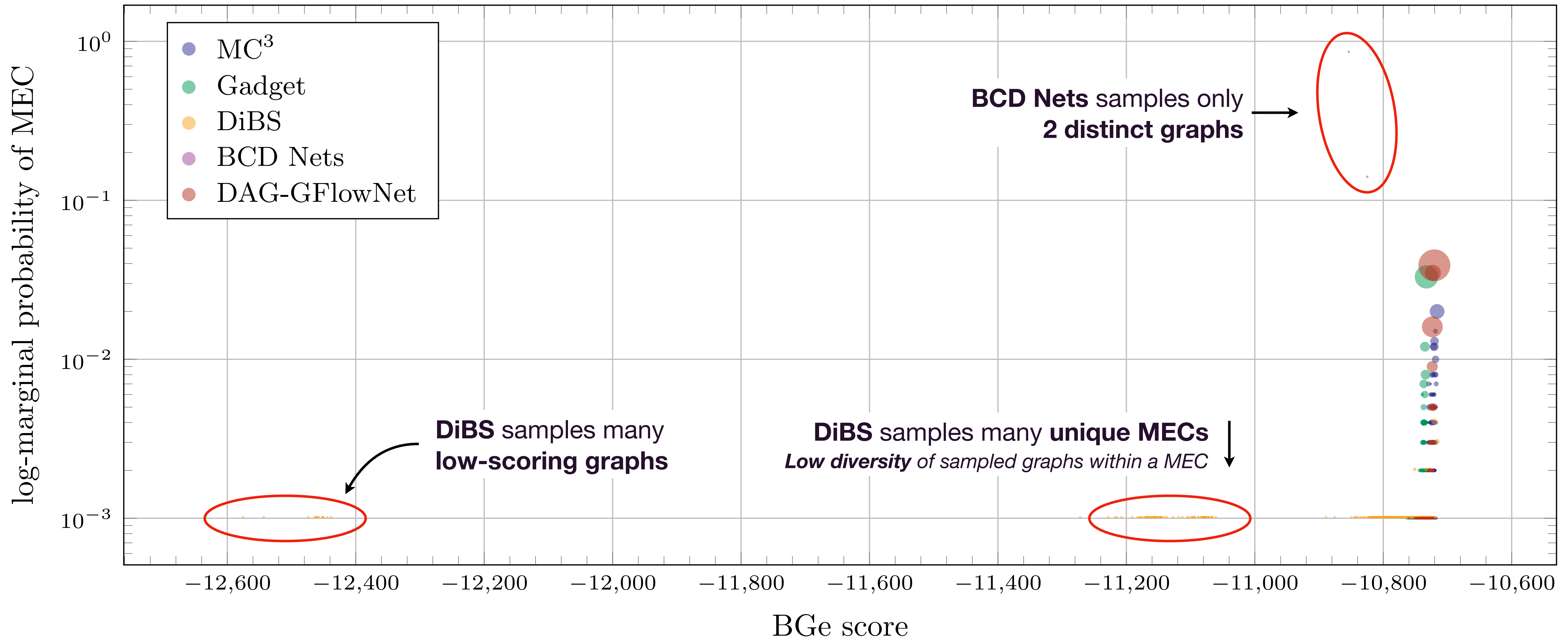


# Experimental results – Flow cytometry data

- Real-world **flow cytometry data**, to learn protein signaling pathways.
- Data: continuous measurements of **11 phosphoproteins**. There are **853 observations**.
- The *ground truth* graph contains **11 nodes** and **17 edges**.
- The consensus graph may not represent a realistic description of the system.



# Experimental results – Flow cytometry data



# Experimental results – Interventional data

- The real world flow cytometry data also contains **interventional data**, based on experimentations where some phosphoproteins are inhibited.
- We model these as **perfect interventions**, even though it may not be the case in practice.
- We know the **intervention targets**.
- We can adapt the reward function (computation of the marginal likelihood) to handle a mixture of **observational & interventional data**.
- This is a first step toward **causal discovery**.

	$\mathbb{E}$ -# Edges	$\mathbb{E}$ -SHD	AUROC
Exact posterior*	—	—	<b>0.816</b>
MC <sup>3</sup>	$25.97 \pm 0.01$	<b><math>25.08 \pm 0.02</math></b>	0.665
DAG-GFlowNet	$30.66 \pm 0.04$	$27.77 \pm 0.03$	0.700

# Thank you



[github.com/tristandeleu/jax-dag-gflownet](https://github.com/tristandeleu/jax-dag-gflownet)

