

Generalization and Distributed Learning of GFlowNets

Tiago da Silva

August, 29, 2025

ICLR, 2025.

Approximate Discrete Bayesian Inference

Let $\mathcal{X} = \{x_1, \dots, x_n\}$ be a finite and large space.

Let \mathcal{D} be the data space and $f : \mathcal{D} \times \mathcal{X} \rightarrow \mathbb{R}$ be a probabilistic model indexed by \mathcal{X} .

Let $\pi : \mathcal{X} \rightarrow \mathbb{R}$ be a prior over \mathcal{X} . Examples:

1. \mathcal{X} = phylogenetic trees, \mathcal{D} = nucleotide sequences.
2. \mathcal{X} = drugs, \mathcal{D} = pharmacological properties.
3. \mathcal{X} = sentences, \mathcal{D} = answers (for a given question).

Our objective is to estimate the **posterior** $\pi(\cdot \mid D)$ over \mathcal{X} given a subset $D \subset \mathcal{D}$.

$$\pi(x \mid D) = f(D \mid x)\pi(x) \left(\sum_{y \in \mathcal{X}} \pi(y \mid D) \right)^{-1}$$

Due to \mathcal{X} 's intractable size, the **partition function**

$$Z = \sum_{y \in \mathcal{X}} \pi(y \mid D)$$

cannot be directly computed.

In doing so, our goal is to both compute expectations over and estimate the modes of $\pi(x \mid D)$. That is,

$$\mathbb{E}_{x \sim \pi(\cdot \mid D)}[\kappa(x)]$$

for a decision function $\kappa : \mathcal{X} \rightarrow \mathbb{R}$, and

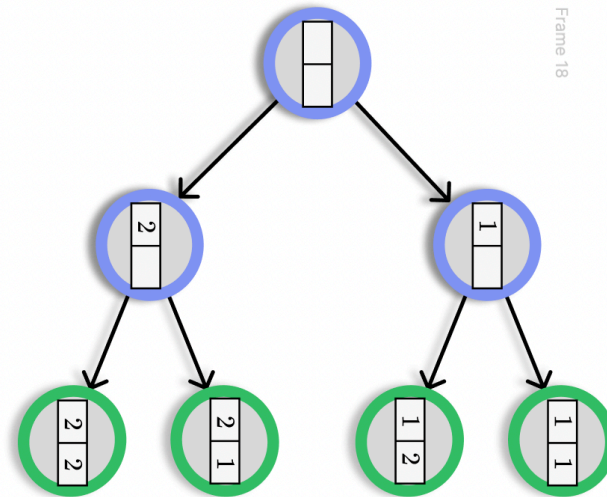
$$\max_{x \in \mathcal{X}} \pi(x \mid D).$$

Owing to \mathcal{X} 's lack of a differential structure, traditional approaches (e.g., HMC) often fail to properly approximate $\pi(\cdot \mid D)$. This is the problem we will address.

Generative Flow Networks (GFlowNets)

GFlowNets cast the problem of approximating $\pi(\cdot \mid D)$ as that of learning a stochastic inductive process on an extended space \mathcal{S} of sub-instances of \mathcal{X} .

When perfectly learned, this process reaches each $x \in \mathcal{X}$ proportionally to $f(x \mid D)\pi(x)$. That is, it generates **exact samples** from the **posterior**. We refer to \mathcal{X} as the set of terminal states.



2-sized sentence-generation over the vocabulary $\{1, 2\}$.

\mathcal{X} = terminal states, $\mathcal{S} \setminus \mathcal{X}$ = extended states. We call this is a state graph, which is a pointed directed acyclic graph.

Generative Flow Networks (GFlowNets)

From a learning perspective, we define parametric models

1. for a forward policy, $p_F : \mathcal{S} \times \mathcal{S} \rightarrow [0, 1]$, and
2. for a backward policy, $p_B : \mathcal{S} \times \mathcal{S} \rightarrow [0, 1]$.

$p_F(s_2 \mid s_1)$ measures the probability of going from s_1 to s_2 .

$p_B(s_1 \mid s_2)$ measures the probability of reverting the above action (when traversing the state graph backwards).

We parameterize both p_F and p_B with neural networks.

Generative Flow Networks (GFlowNets)

Recall that our objective is for p_F to satisfy

$$\sum_{\tau: s_o \rightsquigarrow x} \underbrace{\prod_{(s_i, s_{i+1}) \in \tau} p_F(s_{i+1} \mid s_i)}_{p_F(\tau \mid s_o)} \propto f(x \mid D) \pi(x)$$

(s_o is the source of the state graph). This is satisfied when

$$\mathcal{V}_{p_E} \left(\frac{p_F(\tau \mid s_o)}{p_B(\tau \mid x) f(x \mid D) \pi(x)} \right) = 0$$

for a distribution p_E such that $p_E(\tau) > 0$ for every τ .

Generative Flow Networks (GFlowNets)

We remark on two facts from this setting.

1. p_B is an artifact that enables tractable learning of p_F .
2. Learning is based on adaptive stochastic gradient-based estimates of \mathcal{V}_{p_E} , i.e.,

$$p_F^{(t+1)} \leftarrow p_F^{(t)} - \gamma_t \nabla \mathcal{V}_{p_E}^K,$$

in which $\mathcal{V}_{p_E}^K$ is a Monte Carlo estimate of \mathcal{V}_{p_E} based on samples $\{\tau_1, \dots, \tau_K\} \sim p_E$ and a learning rate schedule $\{\gamma_t\}$.

Learning & Generalization of GFlowNets

In conclusion, a GFlowNet learns a stochastic policy over a fixed state graph via stochastic gradient descent.

1. Learning is based on partial observations of the state graph. Does the learned policy provably generalize beyond the observed states?
2. Which algorithmic improvements can speed up the learning of a generalizable policy?

Crash course on PAC-Bayes bounds

Probably Approximately Correct (PAC) Bayes techniques provide a principled approach for obtaining non-vacuous statistical bounds on the generalization of a model.

1. Let $L : \Theta \times \mathcal{X} \rightarrow \mathbb{R}$ be a loss function and \hat{L} be its empirical estimate of $\mathbb{E}_x[L]$. The **generalization gap** is

$$g(\theta) = \mathbb{E}_x[L(\theta, x)] - \hat{L} \text{ for each } \theta \in \Theta.$$

2. PAC-Bayes bounds limit the expected gap,

$$\mathbb{E}_{\theta \sim Q}[g(\theta)] \leq \varphi(L, \Theta, P, Q) \text{ for a functional } \varphi \text{ and distributions } P, Q.$$

3. Assumptions: boundedness of L and stationarity distributed observations.

- The former can be relaxed by redefining L as $L'(\theta, x) = \min(L(\theta, x), B)$ for a sufficiently large $B \in \mathbb{R}$.
- Stationarity is a good first approximation when dealing with static datasets.

Important: our bound does not take into account the learning algorithm. This is the reason we bound $\mathbb{E}_{\theta}[g(\theta)]$ instead of $g(\hat{\theta})$ for the learned $\hat{\theta}$.

PAC-Bayes for GFlowNets

- As explained earlier, GFlowNet learning is based on a sequence $\{S_t\}_{t=1}^N$ of b -sized trajectory datasets, $S_t = \{\tau_1^{(t)}, \dots, \tau_b^{(t)}\}$. We assume $\{S_t\}_{t=1}^N$ is stationary.
- Based on the PAC-Bayes framework, our understanding of GFlowNet generalization should be grounded on:
 1. A bounded loss function, L .
 2. Distributions $P, Q : \Theta \rightarrow \mathbb{R}$ over the parameters Θ of the neural network parameterizing p_F .

PAC-Bayes for GFlowNets

- We adopt FCS as a performance metric for GFlowNets, which we have introduced in an earlier work.
- In a nutshell, for $S = \{\tau_1, \dots, \tau_K\}$,

$$\text{FCS} (p_F, S) = \frac{1}{2} \sum_{\tau \in S} | p_T(x) - \hat{p}_T(x) |,$$

1. x is the terminal state associated with $\tau \in S$ and
2. p_T (resp. \hat{p}_T) is the true (resp. estimated) marginal distribution of $x \in S$.

PAC-Bayes for GFlowNets

- Both p_T and \hat{p}_T can be efficiently computed on S :

$$p_T(y) = \frac{f(y \mid D)\pi(y)}{\sum_{\tau \in S} f(\tau \mid D)\pi(\tau)} \text{ and } \hat{p}_T(y) = \mathbb{E}_{p_B} \left[\frac{p_F(\tau \mid s_o)}{p_B(\tau \mid y)} \right].$$

- FCS is clearly bounded by $[0, 1]$ (it is a total variation distance restricted to S).
- We define the **generalization gap** of a GFlowNet as

$$g(p_F) := \mathbb{E}_{S \sim p_E} [\text{FCS}(p_F, S)] - \frac{1}{N} \sum_{1 \leq t \leq N} \text{FCS}(p_F, S_t).$$

PAC-Bayes for GFlowNets

- Building upon this, a PAC-Bayes bound for g is:

$$\mathbb{E}_P[g(p_F)] \leq \mathcal{O}\left(\frac{K}{N^{\frac{1}{2}}}\right) + \text{KL} (P\|Q),$$

with $\text{KL} (P\|Q) = \mathbb{E}_P \left[\log \frac{dP}{dQ} \right]$ as the Kullback-Leibler divergence between P and Q .

(we will skip the technical details)

- When the right-hand side of the above formula is larger than 1, the bound is said to be **vacuous**.

PAC-Bayes for GFlowNets

- Building upon this, a PAC-Bayes bound for g is:

$$\mathbb{E}_P[g(p_F)] \leq \mathcal{O} \left(\sqrt{\frac{\text{KL} (P\|Q)}{N}} \right),$$

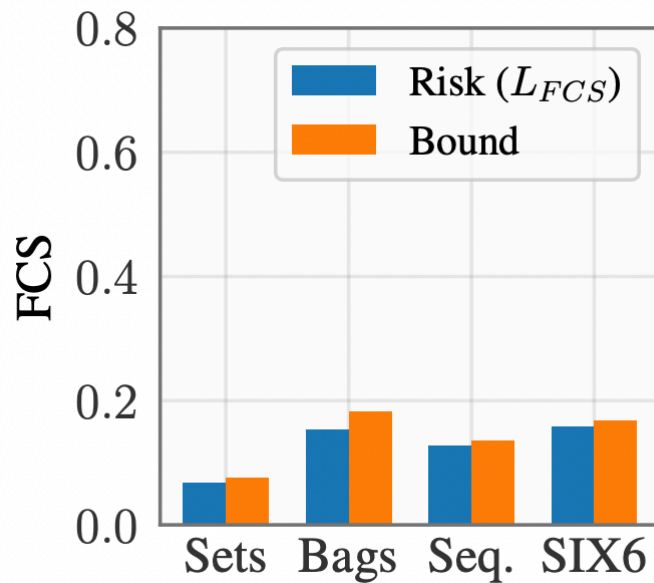
with $\text{KL} (P\|Q) = \mathbb{E}_P \left[\log \frac{dP}{dQ} \right]$ as the Kullback-Leibler divergence between P and Q .

- When the right-hand side of the above formula is larger than 1, the bound is said to be **vacuous**.

PAC-Bayes for GFlowNets

- Remaining ingredients: P and Q .
- As per traditional work, we let both P and Q be isotropic Gaussian distributions with fixed variance.
- We also split $\{S_t\}_{t=1}^N$ into two datasets, \mathcal{S}_1 and \mathcal{S}_2 .
 - P is learned by minimizing the variance loss, \mathcal{V}_{p_E} , defined above (smooth proxy for FCS) on \mathcal{S}_1 .
 - Q is learned by minimizing the upper bound for the learned P on \mathcal{S}_2 .

PAC-Bayes for GFlowNets



This approach consistently produces non-vacuous bounds on consolidated benchmark tasks.

PAC-Bayes and Distributed Learning

- We show that, for an adequate loss function $L : \Theta \rightarrow \mathbb{R}$,

$$\mathbb{E}_P[L(p_F)] \lesssim \mathbb{E}_P[\hat{L}(p_F)] + \mathcal{O}\left(\frac{\text{KL}(P \parallel Q) + \log M}{N}\right),$$

in which M is the maximum trajectory length within the state graph. Examples:

1. Phylogenetic inference: M is the tree size.
2. Sentence generation: M is the maximum sentence size.

PAC-Bayes and Distributed Learning

$$\mathbb{E}_P[L(p_F)] \lesssim \mathbb{E}_P[\hat{L}(p_F)] + \mathcal{O}\left(\frac{\text{KL}(P \parallel Q) + \log M}{N}\right).$$

This formula provides two actionable solutions for improving generalization.

1. By reducing model size, we reduce $\text{KL}(P \parallel Q)$ (larger for high-dimensional P, Q).
2. By shrinking the state graph, we reduce $\log M$.

PAC-Bayes and Distributed Learning

$$\mathbb{E}_P[L(p_F)] \lesssim \mathbb{E}_P[\hat{L}(p_F)] + \mathcal{O}\left(\frac{\text{KL}(P \parallel Q) + \log M}{N}\right).$$

This formula provides two actionable solutions for improving generalization.

1. By reducing model size, we reduce $\text{KL}(P \parallel Q)$ (larger for high-dimensional P, Q).
2. By shrinking the state graph, we reduce $\log M$.

Distributed Learning of GFlowNets

We propose a distributed algorithm that breaks up the state graph into pieces learned by smaller GFlowNets.

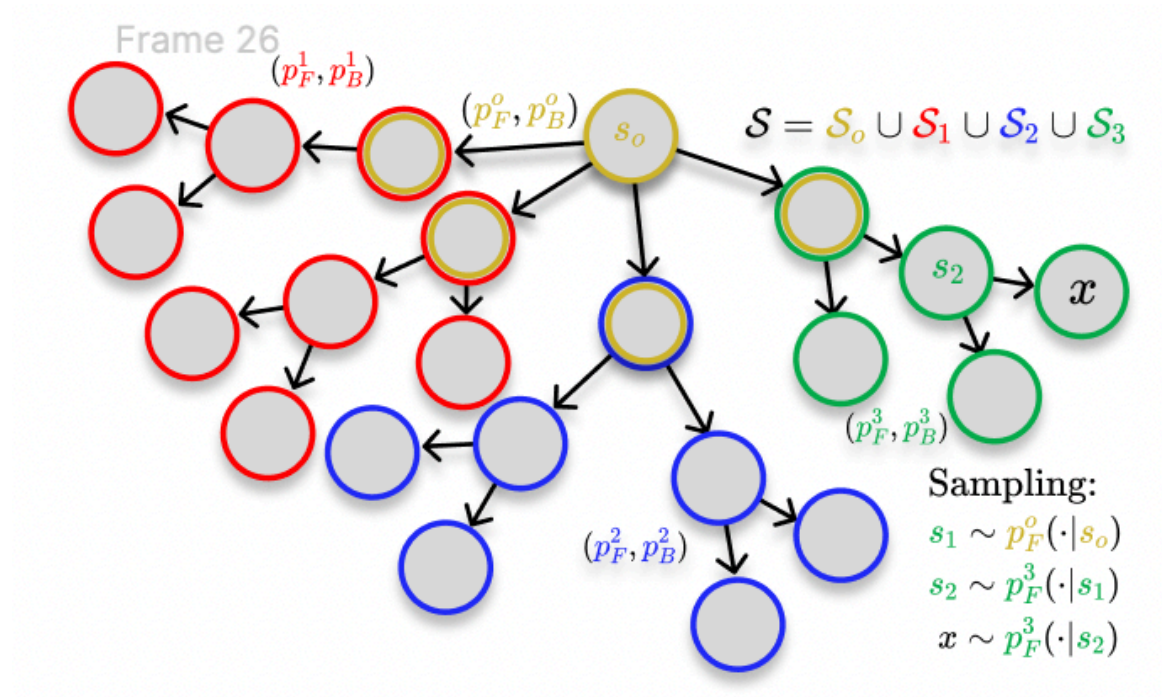
Each GFlowNet parallelly addresses a simpler problem:

$$\downarrow \text{KL} (P\|Q) \text{ and } \downarrow M.$$

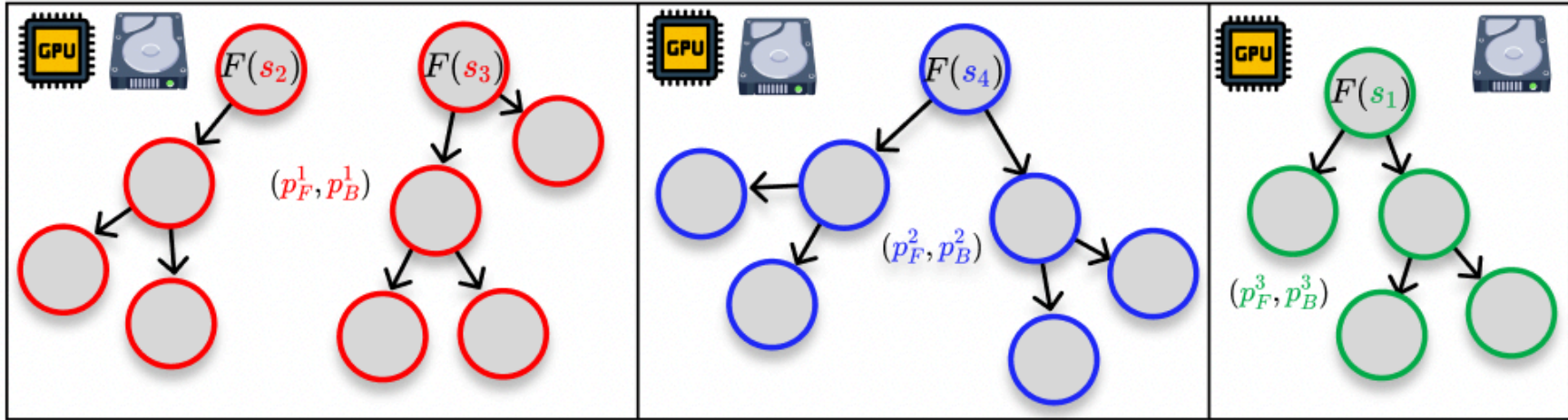
The trained GFlowNets are then be efficiently aggregated.

Our algorithm, Subgraph Asynchronous Learning (SAL), can be easily implemented in computer clusters.

Subgraph Asynchronous Learning

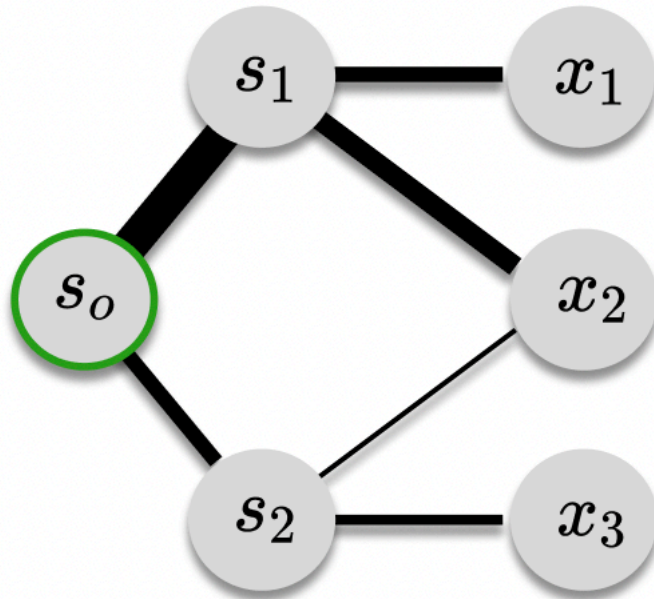


State graph.



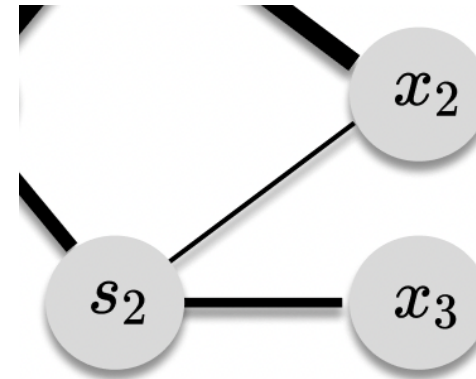
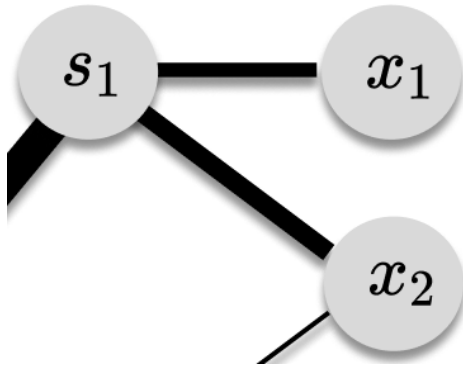
State graph divided into independent chunks.
A GFlowNet is parallelly learned for each chunk.

SAL: Intuition & Flow Assignment



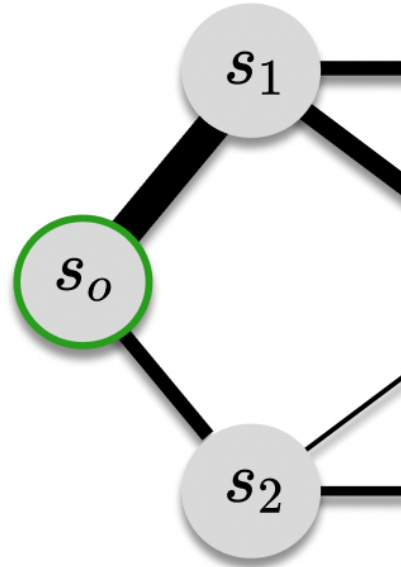
Analogy: A GFlowNet learns a flow assignment in a flow network (this is the reason for the model's name).

SAL: Intuition & Flow Assignment



From this viewpoint, SAL learns a flow assignment for subnetworks in parallel.
When a subnetwork has multiple initial states, we learn an amortized solution.

SAL: Intuition & Flow Assignment



In a final step, a centralized model assigns the appropriate amount of flow to each subnetwork.

SAL: Empirical Analysis

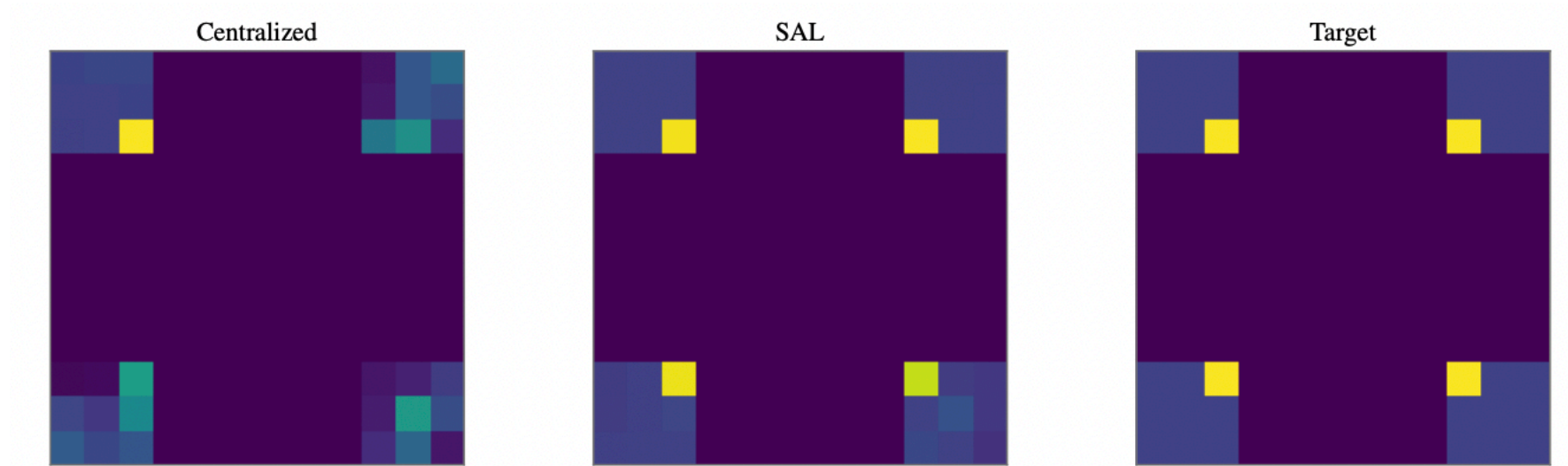
Recall our initial objectives:

1. Estimate posterior expectations.
2. Approximately find posterior modes.

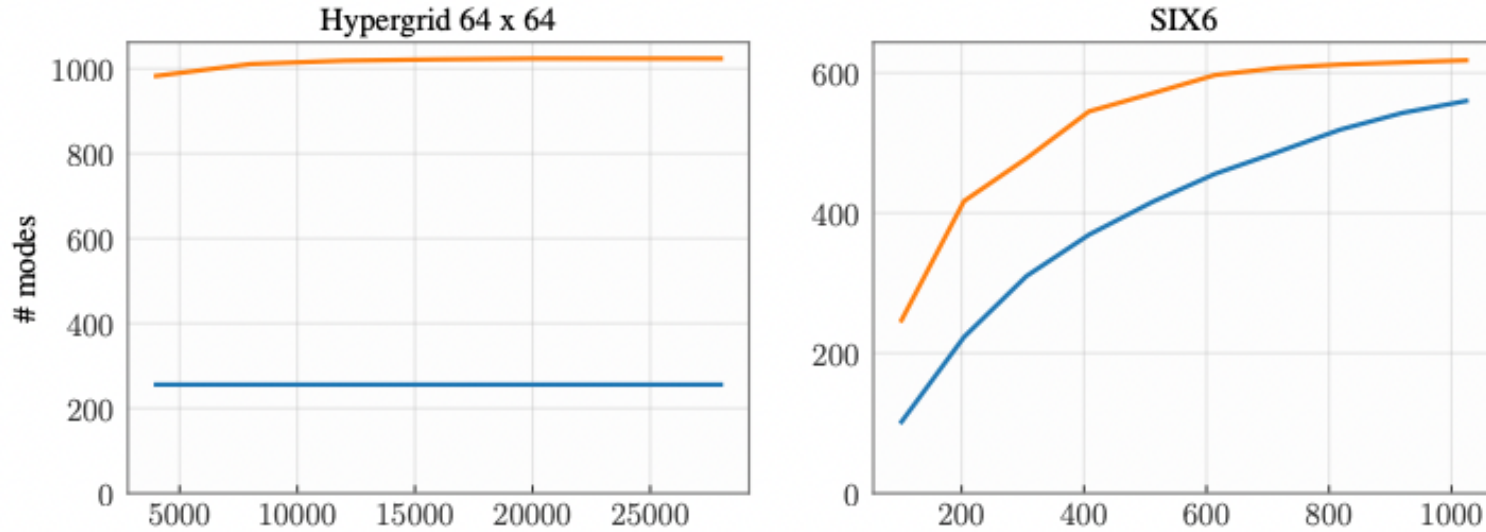
When assessing the former, we evaluate the **goodness-of-fit** of the learned distribution to the posterior.

When assessing the latter, we evaluate the value of $\max_{x \in X} \pi(x) f(D \mid x)$ for a **sampled set** $X \subseteq \mathcal{X}$.

SAL **performs best** in both cases, as we illustrate next.



SAL finds a better approximation to the target distribution than a monolithic model.



SAL finds high-valued states faster than a monolithic model for benchmark tasks.

(A **mode** is defined as a terminal state $x \in \mathcal{X}$ with $\pi(x)f(D|x) > \rho$ for a prescribed ρ)

Take home message

GFlowNets are state-of-the-art models for approximate inference over discrete and compositional distributions.

Our PAC-Bayes bounds rigorously demonstrate that a GFlowNet learns a generalizable policy function.

Our distributed algorithm (SAL) improves generalization by reducing both the model and problem complexities.

SAL also leads to faster mode discovery and better posterior approximations than a centralized model.