Revisiting Non-Acyclic GFlowNets in Discrete Environments

N. Morozov[1,⋆], I. Maksimov[1,⋆], D. Tiapkin[2,3], S. Samsonov[1]

[1]HSE University [2]CMAP, École Polytechnique [3]LMO, Université Paris-Saclay

GFlowNets

- **GFlowNets** are models designed to learn a sampler from a complex discrete space $\mathcal{X}$ according to a probability mass function $\mathcal{R}(x)$ (*GFlowNet reward*) given up to an unknown normalizing constant $\mathrm{Z} = \sum_{x\in\mathcal{X}} \mathcal{R}(x)$.
- Introduce a directed acyclic graph $\mathcal{G} = (\mathcal{S}, \mathcal{E})$. Non-terminal states describe "*incomplete*" objects, with an empty object denoted as s_0, edges — adding new components to them. Terminal states are "*complete*" objects and coincide with $\mathcal{X}$.
- Introduce probability distributions over complete trajectories in backward and forward directions

$$\mathcal{P}_\mathrm{B}(\tau = (s_0 \to s_1 \to \ldots \to s_{n_\tau} \to s_f)) = \frac{\mathcal{R}(s_{n_\tau})}{\mathrm{Z}} \prod_{t=1}^{n_\tau} \mathcal{P}_\mathrm{B}(s_{t-1}|s_t), \quad \mathcal{P}_\mathrm{F}(\tau = (s_0 \to s_1 \to \ldots \to s_{n_\tau} \to s_f)) = \prod_{t=1}^{n_\tau} \mathcal{P}_\mathrm{F}(s_t|s_{t-1}),$$

  where $\mathcal{P}_\mathrm{B}(s_{t-1}|s_t)$ is called a *backward policy* and $\mathcal{P}_\mathrm{F}(s_t|s_{t-1})$ is called a *forward policy*.
- **Main goal:** find a pair of policies such that $\mathcal{P}_\mathrm{F}(\tau) = \mathcal{P}_\mathrm{B}(\tau)$ for all τ. **Then by sampling $\tau \sim \mathcal{P}_\mathrm{F}$, terminal states are sampled with probabilities proportional to $\mathcal{R}(x)$.**
- Parameterize policies by neural networks and introduce a training objective that will force this constraint. E.g., *Detailed Balance* objective:

$$\mathcal{L}_\mathrm{DB}(s \to s') = \left(\log \frac{\mathcal{F}_\theta(s)\mathcal{P}_\mathrm{F}(s'|s,\theta)}{\mathcal{F}_\theta(s')\mathcal{P}_\mathrm{B}(s|s',\theta)}\right)^2, \quad \text{where} \quad \mathcal{F}_\theta(s) = \mathcal{R}(s)\ \forall s \in \mathcal{X}.$$

GFlowNets in Non-Acyclic Environments

We rigorously prove that the theory behind discrete GFlowNets can be directly extended to allow cycles in the environment graph.

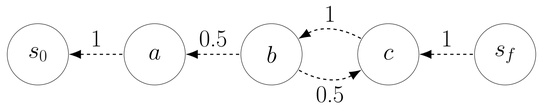


Figure 1:A graph with cycles, backward probabilities are labeled on the edges

Main theoretical contributions and differences from the acyclic case:

- **Trajectories:** We prove expected trajectory length $\mathbb{E}[n_\tau]$ is always finite. Under the assumption of non-zero backward probabilities $\mathcal{P}_\mathrm{B}(s \mid s') > 0$.
- **Flows:** We redefine state flows $\mathcal{F}(s)$ as **expected number of visits** instead of state visitation probabilities:

$$\mathcal{F}(s) = \mathcal{F}(s_f) \cdot \mathbb{E}_{\tau\sim\mathcal{P}(\tau)}\left[\sum_{t=0}^{n_\tau+1} \mathbb{I}\{s_t = s\}\right], \mathcal{F}(s_f) = Z$$

  This allows us to rewrite key GFlowNet theory consistently with acyclic case
- **Expected Trajectory Length:** The expected length $\mathbb{E}[n_\tau]$ is equal to the normalized total state flow:

$$\mathbb{E}[n_\tau] = \frac{1}{\mathcal{F}(s_f)} \sum_{s\in\mathcal{S}\setminus\{s_0,s_f\}} \mathcal{F}(s)$$

- **Constrained Optimization:** Rewriting the task and using the equation above we achieve solution with **smallest possible total flow** which allows us to control mean trajectory length of the final sampler.

$$\min_{\mathcal{F},\mathcal{P}_\mathrm{F},\mathcal{P}_\mathrm{B}} \sum_{s\in\mathcal{S}\setminus\{s_0,s_f\}} \mathcal{F}(s)$$
$$\text{subject to } \left(\log \frac{\mathcal{F}(s)\mathcal{P}_\mathrm{F}(s' \mid s)}{\mathcal{F}(s')\mathcal{P}_\mathrm{B}(s \mid s')}\right)^2 = 0, \qquad \forall s \to s' \in \mathcal{E},$$
$$\mathcal{F}(s_f)\mathcal{P}_\mathrm{B}(x \mid s_f) = \mathcal{R}(x), \qquad \forall x \to s_f \in \mathcal{E}.$$

- **Equivalence to RL:** We generalize the connection to RL, showing that training a non-acyclic GFlowNet with a fixed backward policy is equivalent to solving an **entropy-regularized RL problem** in a corresponding MDP.

State Flow Regularization

Use $\mathcal{F}_\theta(s)$ as a regularizer in the DB loss to control the expected trajectory length:

$$\left(\log \frac{\mathcal{F}_\theta(s)\mathcal{P}_\mathrm{F}(s' \mid s,\theta)}{\mathcal{F}_\theta(s')\mathcal{P}_\mathrm{B}(s \mid s',\theta)}\right)^2 + \lambda\mathcal{F}_\theta(s), \quad \text{where} \quad \mathcal{F}_\theta(s) = \mathcal{R}(s)\ \forall s \in \mathcal{X}.$$

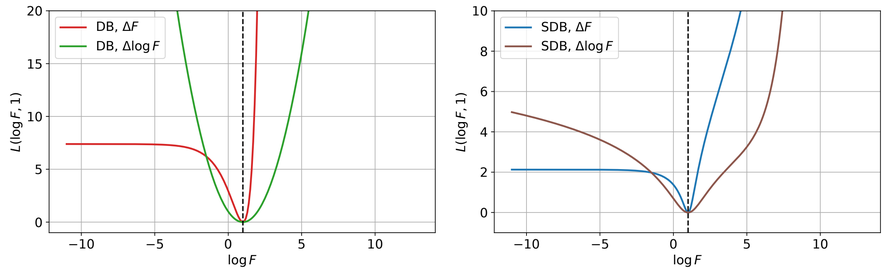
Loss Stability and Scaling Hypothesis

(Brunswic et al., 2024) introduced the concept of **stable losses:** losses that cannot be decreased by increasing the flow on a cycle, which means *expected trajectory length is bounded during training.*

$$\mathcal{L}_\mathrm{SDB}(s \to s') = \log\Big(1 + \varepsilon(\mathcal{F}_\theta(s)\mathcal{P}_\mathrm{F}(s'|s,\theta) - \mathcal{F}_\theta(s')\mathcal{P}_\mathrm{B}(s|s',\theta))^2\Big)(1 + \eta\mathcal{F}_\theta(s))$$

We put out the **scaling hypothesis:**

The main factor that plays a crucial role in loss stability in practice is the scale in which the error between flows is computed: log-flow scale $\Delta\log\mathcal{F}$ vs flow scale $\Delta\mathcal{F}$. We hypothesize that using log-flow scale losses without regularization can lead to arbitrarily large trajectory length, while flow scale losses are biased towards solutions with smaller flows and thus do not suffer from this issue.



Experimental Results and Observations

We run experiments on two non-acyclic environments. Namely hypergrids and permutations. The table below presents results on the **permutation environment**, where permutations of length n are generated. Action space consists of 1) swapping adjacent elements, 2) cyclic shift to the right. Blue indicates the best metric, red indicates the smallest expected trajectory length.

Loss	$n=8$ $C(k)$ $L^1\downarrow$	$n=8$ $\Delta\mathcal{R}\downarrow$	$n=8$ $\Delta\log\mathcal{Z}\downarrow$	$n=8$ $\mathbb{E}[n_\tau]$	$n=20$ $C(k)$ $L^1\downarrow$	$n=20$ $\Delta\mathcal{R}\downarrow$	$n=20$ $\Delta\log\mathcal{Z}\downarrow$	$n=20$ $\mathbb{E}[n_\tau]$
DB, $\Delta\mathcal{F}$	0.215 ±0.198	0.214 ±0.086	0.814 ±0.826	2.43 ±0.28	0.453 ±0.002	0.343 ±0.000	42.98 ±0.000	2.00 ±0.00
SDB, $\Delta\mathcal{F}$	0.031 ±0.012	0.046 ±0.023	0.074 ±0.025	3.32 ±0.15	0.452 ±0.001	0.343 ±0.000	42.98 ±0.000	2.01 ±0.00
DB, $\Delta\log\mathcal{F}$, $\lambda = 10^{-3}$	0.036 ±0.015	0.056 ±0.024	0.018 ±0.010	2.80 ±0.04	0.041 ±0.002	0.064 ±0.000	0.023 ±0.005	3.23 ±0.00
SDB, $\Delta\log\mathcal{F}$, $\lambda = 10^{-3}$	0.037 ±0.013	0.056 ±0.019	0.020 ±0.015	2.79 ±0.04	0.041 ±0.002	0.064 ±0.000	0.026 ±0.003	3.22 ±0.00
DB, $\Delta\log\mathcal{F}$, $\lambda = 10^{-5}$	0.005 ±0.001	0.001 ±0.000	0.005 ±0.004	4.31 ±0.05	0.017 ±0.002	0.035 ±0.002	0.003 ±0.003	7.55 ±0.50
SDB, $\Delta\log\mathcal{F}$, $\lambda = 10^{-5}$	0.005 ±0.001	0.002 ±0.000	0.006 ±0.006	4.36 ±0.09	0.014 ±0.001	0.025 ±0.001	0.005 ±0.005	7.31 ±0.07

- Our empirical results support the scaling hypothesis: even the standard DB in $\Delta\mathcal{F}$ scale is stable; however, non-acyclic GFlowNets trained with $\Delta\mathcal{F}$ scale losses often fail to accurately match the reward distribution;
- Both DB and SDB in $\Delta\log\mathcal{F}$ scale result in better matching the reward distribution but need to be utilized with state flow regularization to ensure small expected trajectory length $\mathbb{E}[n_\tau]$.
- Learning with a fixed $\mathcal{P}_\mathrm{B}$ is possible without stable losses and regularization, however, manually picking $\mathcal{P}_\mathrm{B}$ with small $\mathbb{E}[n_\tau]$ is challenging;
- λ is crucial, the optimization task is very sensitive to the choice of this parameter, changing the order of λ can significantly impact the result by extending or shrinking of mean trajectory length.