α-GFN: Generalized Mixing in GFlowNets for Better Exploration-Exploitation Trade-off

Anonymous authors
Paper under double-blind review

Abstract

Standard Generative Flow Network (GFlowNet) training implicitly assigns equal weights to the forward and backward policies, a consequence of the flow-matching view that constrains the exploration–exploitation dynamics. Extending the connection between GFlowNets and Markov chains, we show that this equal weighting arises from a theoretical equivalence between GFlowNet objectives and Markov chain reversibility. Building on this, we introduce **α-GFNs**, which generalize standard GFlowNet training from strictly balanced flows to imbalanced flows by mixing the forward and backward policies with a hyperparameter α in the training objectives. Through the link to reversibility, we further establish that such objectives converge to unique flows. This generalization provides a richer exploration–exploitation trade-off and, in some settings, coarse control over trajectory lengths. We also propose a simple scheduling algorithm to combine the strengths of different α values. Experiments on Set Generation, Bit Sequence Generation, and Molecule Generation demonstrate consistent performance gains and highlight the benefits of α tuning.

1 Introduction

Generative Flow Networks (GFlowNets) (Bengio et al., 2021) are generative models that sample compositional objects from high-dimensional distributions with probabilities proportional to a reward function (Bengio et al., 2021). They are sampling methods that originate from the intersection of reinforcement learning frameworks (Tiapkin et al., 2024; Mohammadpour et al., 2024; Deleu et al., 2024) and flow networks (Bengio et al., 2023), offering an alternative to traditional approaches such as Markov Chain Monte Carlo (MCMC) (Brooks,

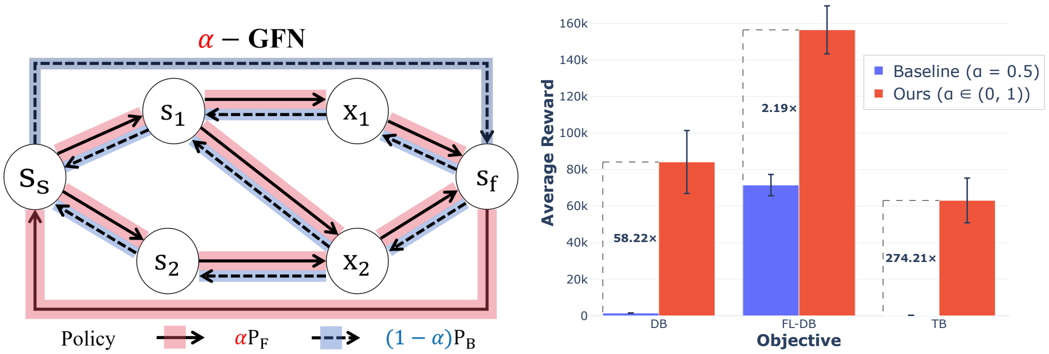


Figure 1: **(Left)** Illustration of α-GFNs, where weights α and $1-\alpha$ are assigned to forward policy P_F and backward policy P_B, respectively. **(Right)** Enhanced performance of α-GFN objectives in large sets (Pan et al., 2023). Objectives with unequal weights surpasses vanilla ones.

1998). Since their introduction, GFlowNets have been applied in various domains including molecular discovery (Bengio et al., 2021; Jain et al., 2023; Zhu et al., 2023), diffusion models (Zhang et al., 2024; Venkatraman et al., 2024; Liu et al., 2024) and large language models (Hu et al., 2024; Song et al., 2024; Yun et al., 2025; Zhu et al., 2025), demonstrating both mode-discovering and diversity-preserving abilities.

Despite their empirical success, most previous applications of GFlowNets implicitly assume fixed, equally weighted forward and backward transitions. This stems from the flow network perspective in GFlowNet theories (Bengio et al., 2021; 2023), despite their original formulation as Markov Decision Processes (MDPs) over pointed directed acyclic graph (DAG) state spaces. As a consequence, GFlowNet training objectives largely employ flow-matching (Bengio et al., 2021; 2023; Malkin et al., 2022; Madan et al., 2023; Pan et al., 2023), which inherently enforces equal transition weighting due to flow properties. Nevertheless, the equal weighting may not be ideal. Corresponding to imbalanced flows, unequal weights can enhance to the performance of GFlowNets, as illustrated in Fig. 1.

On the theory side, there are limited attempts to extend the theoretical framework of GFlowNets. Despite the link to reinforcement learning (Tiapkin et al., 2024; Deleu et al., 2024), GFlowNets' connections to Markov chains (MCs) remain relatively less explored. Previous work (Deleu & Bengio, 2023) has proposed linking Flow Matching (FM, Bengio et al. (2021)) objectives in GFlowNets to invariant measures in MCs. While this approach reflects certain properties of FM, the related conclusions do not generalize well to other objectives (Bengio et al., 2023; Malkin et al., 2022; Madan et al., 2023) where two policies are potentially mixed with equal weights.

In this work, we extend the theoretical link between GFlowNets and Markov chains to multiple training objectives. From this perspective, we propose a generalization of GFlowNets, termed α-GFNs. Compared to standard GFlowNet training, our formulation introduces a hyperparameter $\alpha \in (0, 1)$ that enables flexible mixing of forward and backward policies, leading to imbalanced flows. Our theoretical analysis shows that α-GFN objectives converge to unique flow functions. Moreover, we establish that α provides GFlowNets with a flexible exploration–exploitation trade-off and, in some tasks, a mechanism to control trajectory lengths. We also propose a scheduling algorithm that leverages different α values during training to combine their strengths. Extensive experiments on three benchmarks (Set Generation, Bit Sequence Generation, and Molecule Generation) demonstrate consistent improvements of α-GFNs over baseline methods.

Contributions. We summarize the main contributions of this paper as follows:

- We introduce **α-GFNs**, augmenting GFlowNet objectives with a hyperparameter α that enables flexible mixing of forward and backward policies (Def. 4.3). We further propose a tentative **scheduling algorithm** to combine the benefits of different α during training (Sec. 4.3).
- We extend the theoretical connections between GFlowNets and **Markov chains**, revealing the implicit equal weighting of forward and backward policies in vanilla GFlowNet objectives (Prop. 4.1, Thm. 4.2). Building on this, we show the **theoretical convergence** of α-GFN losses to unique flow functions (Prop. 4.4), and we explain the **effectiveness** of α-GFN losses by connecting their gradients with the gradients of vanilla GFlowNet losses (Prop. 4.5).
- We provide extensive experiments showing how α enables a flexible **exploration-exploitation trade-off** (Fig. 2), leading to the performance gains of our approach across multiple tasks (Sec. 5). We also show the trajectory length-controlling effect of our method in certain scenarios (Fig. 5).

2 Preliminaries

Generative Flow Network (GFlowNet, GFN) preliminaries. A GFlowNet is specified by a tuple (G, R, F, P_F, P_B). Here $G = (S, \mathbb{A})$ is a pointed directed acyclic graph with source s_s and sink s_f, $R : \mathcal{X} \subseteq S \to \mathbb{R}_+$ is a reward, $F : S \to \mathbb{R}_+$ is a state flow; and $P_F, P_B : \mathbb{A} \to [0, 1]$ are forward and backward policies. Let $\mathfrak{T}^{\text{flow}}$ be the set of complete trajectories $\mathfrak{t}^f = (s_0, \ldots, s_N)$ with

$s_0 = s_s$ and $s_N = s_f$. Samples are states $x \in \mathcal{X}$ sequentially constructed by P_F and deconstructed by P_B. The generation of a sample terminates when s_f is reached, yielding a complete trajectory $(s_0 = s_s, \dots, s_{N-1} = x, s_N = s_f)$. Ideally, the probability of generating x is proportional to its reward, $P_F^\top(x) \triangleq \sum_{\mathfrak{t} \in \mathfrak{T}^{\text{flow}}: x \in \mathfrak{t}} \prod_{i=1}^N P_F(s_i \mid s_{i-1}) \propto R(x)$. Under the convergence of its training objectives, this proportionality is satisfied and the uniqueness of flows is achieved.

GFlowNet training objectives and variants. Several losses have been introduced to train GFlowNets by enforcing flow-balance conditions, such as Flow Matching (FM, Bengio et al. (2021)), Detailed Balance (DB, Bengio et al. (2023)), Subtrajectory Balance (SubTB, Madan et al. (2023)), or Trajectory Balance (TB, Malkin et al. (2022)). In this paper, we consider mainly SubTB, which unifies DB and TB, and its variants. Given any partial trajectory $\mathfrak{t}' = (s_k, s_{k+1}, \dots, s_{k+m}) \subset \mathfrak{t}^f \in \mathfrak{T}^{\text{flow}}$, SubTB aims at ensuring that

$$F(s_k) \prod_{i=1}^{m} P_F(s_{k+i} \mid s_{k+i-1}) \;=\; F(s_{k+m}) \prod_{i=1}^{m} P_B(s_{k+i-1} \mid s_{k+i}), \tag{1}$$

with $P_B(x \mid s_f) = \frac{R(x)}{F(s_f)}$ for all $x \in \mathcal{X}$ and $F(s_f) = \sum_{x' \in \mathcal{X}} R(x') = F(s_0)$. The loss is the log-square of Eq. 1, and a convex combination of it, termed SubTB(λ), is used for training. For more details and formal definitions, see App. A.1. Although these objectives balance flows, credit assignment can be inefficient. Forward-looking (FL) variants with intermediate rewards (Pan et al., 2023; Jang et al., 2024) address this by modifying the balance with reparameterization of flows. With an intermediate energy $\mathcal{E} : \mathbb{A} \to \mathbb{R}$, SubTB becomes FL-SubTB by multiplying the right hand side of Eq. 1 with $\prod_{i=1}^m e^{-\mathcal{E}(s_{k+i-1}, s_{k+i})}$.

Reversibility. Reversibility is a classical concept in Markov chain theory, see e.g. (Douc et al., 2018). Given a probability measure over the state space $\pi : S \to [0, 1]$ and the transition kernel of the chain $P : S \times S \to [0, 1]$, the reversibility of P implies that, at any sequence of states $(s_k, s_{k+1}, \dots, s_{k+m})$

$$\pi(s_k) \prod_{i=1}^{m} P(s_{k+i} | s_{k+i-1}) = \pi(s_{k+m}) \prod_{i=1}^{m} P(s_{k+i-1} | s_{k+i}). \tag{2}$$

3 Related Works

GFlowNet theories and connections with Markov chains. Initially formalized within the flow network framework (Bengio et al., 2021), subsequent developments (e.g., the detailed balance conditions (Bengio et al., 2023)) have drawn heavily on Markov chain (MC) theory, underscoring the need for a rigorous link to classical MC theory. Deleu & Bengio (2023) proposed a GFlowNet-induced MC under the FM objective using only the forward policy; yet the connection with the backward policy is not mentioned whereas it is central to the connection we establish. Furthermore, Deleu & Bengio (2023) do not discuss the connection with other loss functions, while we consider several losses, including DB, SubTB, TB and their FL variants.

GFlowNet objective design. Standard GFlowNet losses, such as DB (Bengio et al., 2023), SubTB (Madan et al., 2023) and TB (Malkin et al., 2022), optimize with both the forward and backward policies to ensure balanced flows, which equally weights the two policies, whereas we consider more general weights in this work. Their variants include temperature-conditional GFlowNets (Zhang et al., 2023; Zhou et al., 2024; Kim et al., 2024) where a parameter is applied to scale the reward, and forward-looking GFlowNets (Pan et al., 2023; Jang et al., 2024) where intermediate energies are introduced at the transitions, and Hu et al. (2025) alters the log-square form of the losses. All of these contributions remain within the flow-matching paradigm, but our approach departs from the strict balance condition by mixing forward and backward policies via a tunable parameter, thereby allowing controlled flow imbalance.

4 GENERALIZING GFLOWNET TRAINING WITH α-GFN OBJECTIVES

This section is organized into four parts. First, we show that the target of the vanilla GFlowNet objectives corresponds to the reversibility condition of a Markov chain with the **equally mixed policy** $P_{0.5} = \frac{1}{2}P_F + \frac{1}{2}P_B$ as its transition kernel. Next, we extend the mixing to unequal weights, leading to the α-GFN objectives, whose convergence is further clarified through the underlying MC formulation. We then introduce a scheduling algorithm that combines the advantages of different α values. Finally, we illustrate the flexible exploration–exploitation trade-off enabled by α.

4.1 THE EQUALLY MIXED MARKOV CHAIN BEHIND VANILLA GFLOWNET OBJECTIVES

We begin with an intuitive comparison of Eq. 1 and Eq. 2, which reveals a structural similarity between GFlowNets objectives and the reversibility of Markov chains. Specifically, both equations share the following structure: a sequence of transitions is coupled with a probability measure on each side. Building on the equivalence between flows and probability measures (see Deleu & Bengio (2023) and Eq. 18), this resemblance actually suggests a close connection between GFN objectives and MC reversibility even though GFNs only use the forward policy P_F to generate samples. However, the policies on the two sides of Eq. 1 are not identical, which prevents a direct correspondence with Eq. 2. In reality, this obstacle can be overcome by introducing the equally mixed policy $P_{0.5}$. When applied to Eq. 2, $P_{0.5}$ naturally separates P_F and P_B onto different sides, eliminating the weights and recovering Eq. 1:

Proposition 4.1. *The reversibility of $P_{0.5}$ means for any partial trajectory $\mathfrak{t}' = (s_k, \ldots, s_{k+m})$, $\pi(s_k)\prod_{i=1}^{m} P_F(s_{k+i}|s_{k+i-1}) = \pi(s_{k+m})\prod_{i=1}^{m} P_B(s_{k+i-1}|s_{k+i})$.*

Applying Prop. 4.1 to DB, SubTB and TB, we extend the GFN-MC link in Deleu & Bengio (2023) from FM to these objectives, and turn the similarity into a theoretical equivalence between GFNs and MCs:

Theorem 4.2. *The target of SubTB is equivalent to the partial-trajectory-level reversibility of a MC with the transition kernel $P_{0.5}$, and that of DB or TB is similar. Moreover, their convergence to unique flows is related to corresponding properties in MC theory.*

The proofs for Prop. 4.1 and Thm. 4.2 appear in App. C.1 and App. C.2, respectively. App. B further clarifies link from Markov chains to GFlowNets (MC→GFN), which is not elucidated in (Deleu & Bengio, 2023).

4.2 α-GFN OBJECTIVES: FLEXIBLE MIXING OF POLICIES WITH A HYPERPARAMETER α

Now, it is clear that the objectives of GFlowNets adopt an equal weight mixing of P_F and P_B. However, the equal weights may not be ideal for all the settings. By analogy with the way two policies can be mixed by taking a convex combination of the probability distributions, we propose a flexible mixing regime with a hyperparameter $\alpha \in (0, 1)$ as the mixing ratio. To be specific, by plugging an **arbitrarily mixed policy** $P_\alpha = \alpha P_F + (1 - \alpha)P_B$ into Eq. 1, Prop. 4.1, and Thm. 4.2, we obtain objectives corresponding to reversibility of P_α, termed α-GFN objectives. The following definition provides the intuition; see App. A.1 for the rigorous α-GFN loss functions.

Definition 4.3 (**α-GFN objectives**). Given $\alpha \in (0, 1)$, the α-SubTB loss aims at ensuring that

$$\alpha^m F(s_k)\prod_{i=1}^{m} P_F(s_{k+i}|s_{k+i-1}) = (1-\alpha)^m F(s_{k+m})\prod_{i=1}^{m} P_B(s_{k+i-1}|s_{k+i}) \tag{3}$$

for any partial trajectory $\mathfrak{t}' = (s_k, \ldots, s_{k+m})$. Applying this hyperparameter to other variants, such as forward-looking ones, yields similar objectives, specifically, α-FL-SubTB aims at ensuring that

$$\alpha^m F(s_k)\prod_{i=1}^{m} P_F(s_{k+i}|s_{k+i-1}) = (1-\alpha)^m F(s_{k+m})\prod_{i=1}^{m} P_B(s_{k+i-1}|s_{k+i})e^{-\mathcal{E}(s_{k+i-1},s_{k+i})}. \tag{4}$$

Other losses are defined similarly. Although the targets in Def. 4.3 violate the balance condition of vanilla GFlowNets, their **convergence** to unique flows is described by the link to MC reversibility (proof in App. C.3):

Proposition 4.4. *The targets of α-SubTB is equivalent to the partial-trajectory-level reversibility of a MC with the transition kernel P_α, and those of α-DB, α-TB and the variants are similar. Moreover, their convergence to unique flows is similar to vanilla GFN objectives for all $\alpha \in (0, 1)$.*

4.3 Scheduling of α

Fixing the value of α may lead to undesirable effects. As shown in Fig. 3, certain choices of α negatively affect the reward-fitting ability of the forward policy. To address this, we propose a two-stage scheduling algorithm that retains the benefits of α-GFNs while preserving the fitting behavior of vanilla GFNs: (i) start with α far from 0.5 (e.g., 0.1–0.4 or 0.6–0.9) and keep it fixed for a set number of steps; (ii) gradually anneal α to 0.5 over the remaining steps. See Alg. 1 for details.

Algorithm 1 Scheduled Training of α-GFNs

```
Initialize $\alpha = \alpha_0$, select the vanilla objective $\mathcal{L}$ and
  augment it with $\alpha$ to be an $\alpha$-GFN objective $\mathcal{L}_\alpha$.
Select total steps $N$, stage 1 steps $N_1$ and the annealing
  function $f$ at stage 2.[1]
Initialize model $\theta$, forward and backward policy $P_F^\theta, P_B^\theta$,
  flow function $F^\theta$.
for $n = 1, \ldots, N$ do
  if $n > N_1$ then
    $\alpha \leftarrow f(\alpha_0, n, N_1, N)$
  end if
  Update $\theta$ by minimizing $\mathcal{L}_\alpha(P_F^\theta, P_B^\theta, F^\theta)$.
end for
```

4.4 Flexible Exploration-Exploitation Trade-Off with α

How does α contribute to the training of GFlowNets? MC theory suggests that convergence rates of α-GFN objectives to unique flows may vary exponentially for different α values (see App. B.3). Although GFNs are not optimized via MCMC, this property still has significant impact on the behavior of the forward policy.

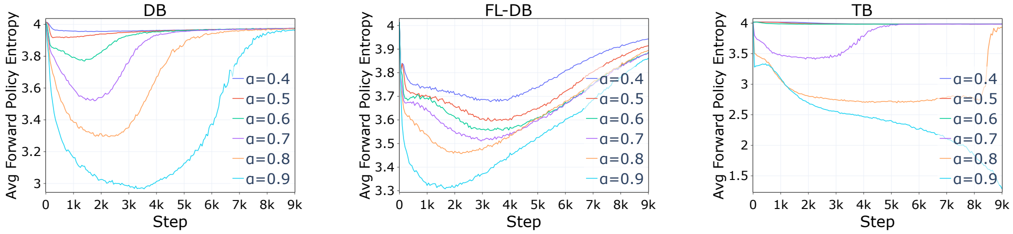


Figure 2: α-GFN entropy mechanisms in large sets with **(Left)** DB, **(Center)** FL-DB, **(Right)** TB objectives.

The hyperparameter $\alpha \in (0, 1)$ is the mixing ratio in $P_\alpha = \alpha P_F + (1-\alpha)P_B$, which controls the contribution of the forward policy P_F. In practice, α scales the training pressure on P_F: larger values of α accelerate exploitation of current estimates, while smaller values temper it. Combined with the GFlowNet target $P_F^\top(x) \propto R(x)$ and the convergence rates of different MCs, this leads to the following heuristic. When $\alpha > 0.5$, exploitation dominates, quickly suppressing low-reward or unseen actions and concentrating mass on high-reward ones. When $\alpha < 0.5$, exploitation slows down, which sustains broader exploration and produces a flatter action distribution. Empirically, the entropy dynamics in Fig. 2 follow this pattern: larger α induces an early drop in per-action entropy, while smaller α maintains higher entropy. This behavior can be explained by examining the gradient more closely:

Proposition 4.5 (Gradient of α-GFN objectives, proof in App. C.4)**.** *We take SubTB as an example and denote $P_F(\mathrm{t}') = \prod_{i=1}^{m} P_F(s_{k+i} \mid s_{k+i-1})$ for simplicity. Given the fact that GFN losses are log-square*

[1] f is a scheduling function, e.g., $f(\alpha_0, n, N_1, N) = 0.5 + (\alpha_0 - 0.5)\exp\big(-4 \cdot \frac{n-N_1}{N-N_1}\big)$.

functions of their targets, the gradient of α-SubTB loss can be expressed as a modification of the SubTB loss gradient for P_F as: $\frac{\partial L_{\alpha-\mathit{SubTB}}(\mathfrak{t}')}{\partial P_F(\mathfrak{t}')} = \frac{\partial L_{\mathit{SubTB}}(\mathfrak{t}')}{\partial P_F(\mathfrak{t}')} + \frac{2m}{P_F(\mathfrak{t}')} \log \frac{\alpha}{1-\alpha}$.

For $\alpha > 0.5$, the term $\frac{2m}{P_F(\mathfrak{t}')} \log \frac{\alpha}{1-\alpha}$ is larger when $P_F(\mathfrak{t}')$ is small (low reward) and smaller when $P_F(\mathfrak{t}')$ is large (high reward). Consequently, low-reward probabilities decay faster while high-reward ones decay more slowly, sharpening the distribution and strengthening exploitation. For $\alpha < 0.5$, $\log \frac{\alpha}{1-\alpha} < 0$ reverses the effect, which yields a flatter distribution and contributes to exploration. In addition, Proposition 4.5 shows that the exploration-exploitation effects scales with $\left|m \log \frac{\alpha}{1-\alpha}\right|$.

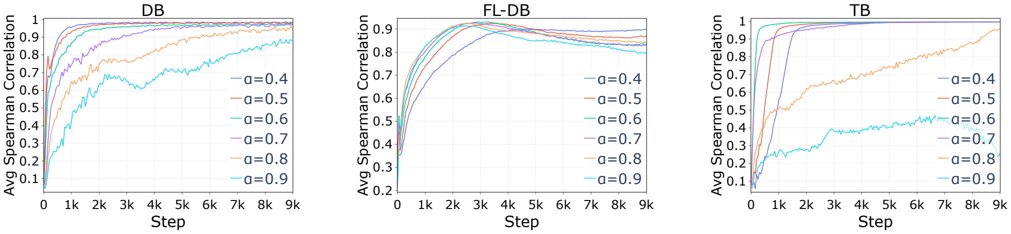


Figure 3: α-GFN Spearman correlations in large sets with **(Left)** DB **(Center)** FL-DB **(Right)** TB objectives.

Nevertheless, training P_F under the α-GFN objectives may suffer from over-exploitation when α is too large or inefficient credit assignment when α is too small. The former is reflected by a drop in the correlation between P_F and the comparison of reward in Fig. 3, and the latter is reflected by low reward of vanilla GFN objectives in Fig. 1. Thus, we schedule α with Alg. 1 to combine the advantages of different α values and avoid the undesirable effects of a fixed α.

5 Experiments

Previous sections have shown how the additional hyperparameter α shapes GFlowNet training dynamics. We now evaluate whether these dynamics yield performance gains across various domains, particularly by enhancing discovery of distinct high-reward modes and preserving sample diversity.

5.1 Datasets and Settings

We consider three popular benchmarks, Set Generation (Pan et al., 2023), non-autoregressive Bit Sequence Generation (Tiapkin et al., 2024) and Molecule Generation (Bengio et al., 2021). In contrast to former works, we separate training samples and evaluation samples in Bit Sequence Generation and Molecule Generation to avoid confusions in evaluation metrics, following the practice in Set Generation. We adopt a exponential annealing function in stage 2 of Alg. 1. Details of experimental settings are deferred to App. D.1.

Set Generation. The goal of this task is to generate sets of fixed sizes. The maximum capacity of sets $|S|$ and the size of the vocabulary vary from being small, medium to large, and task difficulty increases correspondingly. The sampling process of a set terminates when the number of elements reaches its maximum capacity. The reward function is defined as the accumulation of individual energy exponent $\exp(-\mathcal{E}(e_i))$ of each element e_i in a set, i.e. $R(x) \triangleq \prod_{i=1}^{|S|} \exp(-\mathcal{E}(e_i))$, which equips FL variants with ideal local credits (Pan et al., 2023; Jang et al., 2024). Results are reported with 10 random seeds.

Bit Sequence Generation. The goal of this task is to generate binary strings of a fixed length $n = 120$. In each step, a k-bit word is sampled to replace an existing k-bit empty word. The sampling process

Table 1: Results on Set Generation. α-GFNs perform better at reward and modes across all settings.

		DB		**FL-DB**		**TB**	
Set Size	**Metric**	Baseline	Ours	Baseline	Ours	Baseline	Ours
Small	Modes↑	26.6 ± 3.5	**51.1** ± 8.8	83.1 ± 1.9	**88.5** ± 2.7	23.2 ± 5.9	**26.8** ± 5.6
	Top-1000 Reward↑	0.184 ± 0.001	**0.203** ± 0.004	0.218 ± 0.001	**0.220** ± 0.002	0.180 ± 0.001	**0.184** ± 0.001
	Spearman Corr	0.998 ± 0.001	0.995 ± 0.001	0.989 ± 0.004	0.976 ± 0.006	1.000 ± 0.000	1.000 ± 0.000
Medium	Modes↑	0.0 ± 0.0	**79.5** ± 105.61	9.0 ± 5.57	**255.6** ± 249.8	0.0 ± 0.0	**380.2** ± 469.2
	Top-1000 Reward↑	113916 ± 7869	**575348** ± 65238	498434 ± 26370	**665591** ± 42842	44847 ± 1539	**668396** ± 74759
	Spearman Corr	0.994 ± 0.002	0.977 ± 0.004	0.971 ± 0.007	0.946 ± 0.010	0.998 ± 0.000	0.906 ± 0.268
Large	Modes↑	0.0 ± 0.0	**215.0** ± 183.1	223.9 ± 101.6	**2475.4** ± 721.8	0.0 ± 0.0	**431.0** ± 352.8
	Top-1000 Reward↑	54440 ± 3964	**580376** ± 72592	600956 ± 31384	**775196** ± 18595	11318 ± 394	**636234** ± 63003
	Spearman Corr	0.979 ± 0.010	0.945 ± 0.010	0.886 ± 0.017	0.848 ± 0.020	0.996 ± 0.000	0.996 ± 0.001

terminates when there are n/k non-empty words. The reward of a sample is defined as the negative exponent of its minimal Hamming distance to any target mode in a predefined set M with $|M| = 60$, i.e. $R(x) = \exp(-\min_{y \in M} d(x, y))$. We vary $k \in \{2, 4, 6, 8, 10\}$ as Tiapkin et al. (2024). We consider DB, SubTB(λ) and TB, and report results with 5 random seeds.

Molecule Generation. This task aims to design binders for the soluble epoxide hydrolase (sEH) protein by iteratively appending “blocks” from a fixed library onto a growing molecular graph (Jin et al., 2018). The reward is computed via a pretrained proxy from Bengio et al. (2021). We consider DB, FL-DB, SubTB(λ), FL-SubTB(λ) and TB, and report results with 5 random seeds.

5.2 Results

Enhanced Performance: More Modes and Reward. As shown in Table 1, Table 2, and Table 3, the exploration–exploitation flexibility introduced by α improves both reward and the number of modes in most settings. Because modes correspond to distinct high-reward samples, their increase also indicates that sample diversity is preserved.

In **Set Generation**, results in Table 1 shows that α-GFN objectives achieve 92%, 6%, 16% more number of modes for DB, FL-DB, TB respectively in small sets. Moreover, the performance gain scales well with increasing task difficulty. For medium and large sets, α-GFN objectives achieve at least 1000% more modes than vanilla objectives. For TB and DB, the performance gain is even larger. The average reward also greatly increases, suggesting the benefit of enhanced reward exploration-exploitation abilities of α-GFNs.

In **Bit Sequence Generation**, results in Table 2 still demonstrate the effectiveness of the hyperparameter α. α-GFNs find up to 8 more modes than vanilla GFlowNets on average and consistently surpass them in 13 settings, while vanilla GFlowNets can only find up to 2 modes than α-GFNs in 2 settings.

Finally, we evaluate α-GFNs on a real-world task, **Molecule Generation**. As shown in Table 3, α-GFN objectives substantially outperform the baselines across most evaluation metrics. In terms of the number of modes, setting $\alpha \neq 0.5$ yields gains of 44% with DB, 177% with FL-DB, 19% with SubTB(λ), 145% with FL-SubTB(λ), and 5% with TB. Corresponding improvements are also observed in the reward metric, many of which are significant. Moreover, the sample similarity metrics are either improved or slightly increased, indicating that the diversity of high-reward samples is well preserved. Taken together, these results provide strong evidence for the effectiveness of α-tuning.

Across experiments, we observe that α-GFNs are comparable to vanilla GFlowNets in Spearman correlations between sampling probabilities and rewards. In reality, α-GFNs even achieve a better fitting to the reward in

Table 2: Results on Bit Sequence Generation. In terms of number of modes on average, α-GFN objectives outperform vanilla GFlowNet objectives across 87% task settings.

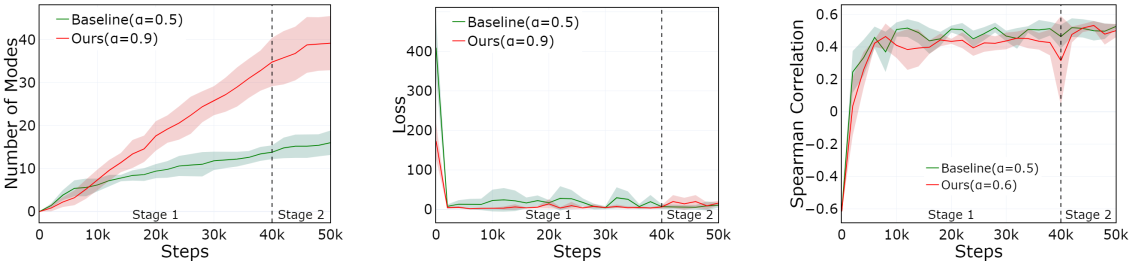
		DB		**SubTB(λ)**		**TB**	
k	**Metric**	Baseline	Ours	Baseline	Ours	Baseline	Ours
2	Modes	0.60 ± 0.89	**2.60** ± 2.07	**22.80** ± 13.27	20.80 ± 11.39	12.20 ± 2.39	**16.80** ± 3.77
	Spearman	0.32 ± 0.43	0.50 ± 0.04	0.47 ± 0.06	0.51 ± 0.19	0.47 ± 0.17	0.54 ± 0.00
4	Modes	9.80 ± 6.50	**12.00** ± 1.58	35.40 ± 4.16	**40.40** ± 2.97	38.00 ± 2.74	**41.80** ± 5.07
	Spearman	0.58 ± 0.00	0.57 ± 0.00	0.58 ± 0.00	0.60 ± 0.02	0.58 ± 0.00	0.58 ± 0.00
6	Modes	4.20 ± 1.92	**5.20** ± 1.30	20.60 ± 3.36	**22.20** ± 7.33	21.60 ± 2.70	**23.40** ± 3.71
	Spearman	0.55 ± 0.01	0.56 ± 0.00	0.55 ± 0.00	0.59 ± 0.09	0.56 ± 0.00	0.56 ± 0.00
8	Modes	36.40 ± 1.82	**44.40** ± 3.78	58.60 ± 1.67	**58.60** ± 0.89	58.80 ± 1.30	**59.00** ± 0.00
	Spearman	0.79 ± 0.00	0.79 ± 0.00	0.81 ± 0.00	0.81 ± 0.00	0.81 ± 0.00	0.81 ± 0.00
10	Modes	6.80 ± 2.05	**8.20** ± 2.39	19.20 ± 4.92	**21.60** ± 3.05	**24.40** ± 3.36	23.80 ± 1.79
	Spearman	0.57 ± 0.01	0.57 ± 0.00	0.57 ± 0.00	0.57 ± 0.00	0.57 ± 0.00	0.57 ± 0.00

Table 3: Results on Molecule Generation. α-GFNs are better at modes and reward for all objectives. Similarity metrics are improved or just slightly increased. Correlations of FL variants are omitted due to their biased target (Silva et al., 2025).

	DB		**FL-DB**		**SubTB(λ)**		**FL-SubTB(λ)**		**TB**	
Metric	Baseline	Ours	Baseline	Ours	Baseline	Ours	Baseline	Ours	Baseline	Ours
Modes↑	10.00 ± 3.08	**14.40** ± 2.41	9.00 ± 1.58	**25.00** ± 4.36	22.20 ± 2.77	**26.40** ± 2.30	16.00 ± 3.16	**39.20** ± 7.05	38.40 ± 6.11	**40.20** ± 5.07
Top-10 Reward↑	7.16 ± 0.06	**7.24** ± 0.05	7.28 ± 0.06	**7.37** ± 0.07	7.30 ± 0.07	**7.33** ± 0.06	7.39 ± 0.08	**7.46** ± 0.05	**7.45** ± 0.06	7.45 ± 0.08
Top-100 Reward↑	6.81 ± 0.03	**6.84** ± 0.03	6.97 ± 0.02	**7.03** ± 0.05	6.97 ± 0.03	**6.99** ± 0.02	6.98 ± 0.06	**7.12** ± 0.04	7.14 ± 0.05	**7.16** ± 0.05
Top-1000 Reward↑	**6.33** ± 0.01	6.32 ± 0.01	**6.69** ± 0.06	6.47 ± 0.14	6.51 ± 0.01	**6.52** ± 0.01	6.48 ± 0.13	**6.50** ± 0.05	6.70 ± 0.07	**6.73** ± 0.07
Top-10 Similarity↓	0.59 ± 0.01	**0.55** ± 0.05	0.59 ± 0.06	**0.52** ± 0.07	**0.56** ± 0.04	0.58 ± 0.02	0.55 ± 0.18	0.55 ± 0.18	**0.62** ± 0.03	0.66 ± 0.04
Top-100 Similarity↓	0.57 ± 0.01	**0.56** ± 0.01	0.74 ± 0.05	**0.51** ± 0.03	**0.57** ± 0.01	0.58 ± 0.02	**0.49** ± 0.05	0.57 ± 0.01	**0.63** ± 0.03	0.64 ± 0.04
Top-1000 Similarity↓	0.56 ± 0.00	0.56 ± 0.00	0.61 ± 0.03	**0.50** ± 0.05	0.56 ± 0.00	0.56 ± 0.01	0.54 ± 0.10	**0.50** ± 0.00	**0.60** ± 0.03	0.61 ± 0.04
Spearman Corr	0.53 ± 0.02	0.50 ± 0.05	—		0.57 ± 0.07	0.59 ± 0.05	—		0.22 ± 0.45	0.47 ± 0.16

many cases. In Molecule Generation, α-TB delivers roughly 2× improvement in Spearman correlations over vanilla TB with substantially lower variance across runs, and α-SubTB(λ) also performs better than vanilla SubTB(λ) in this metric. When the correlation drops, α-GFNs still consistently discover more modes. These results highlight the reward-fitting ability of α-GFNs.

Stability and Effects of Scheduling. Although we observe clear performance gains across tasks, the standard deviation especially in the number of modes, sometimes increases, raising stability concerns. To probe this and the effect of annealing α on reward fitting, we conduct a case study on Molecule Generation. As shown in Fig. 4(a,b), α-SubTB(λ) consistently outperforms SubTB(λ) in stage 1 (before annealing) with $\alpha = 0.9$ and maintains its advantage in stage 2 (annealing), where exponentially fast annealing induces mild loss oscillations that settle after roughly 8,000 steps, leaving a modest increase in mode variance. Nevertheless, a clear performance gain throughtout the training is still clearly observed. Meanwhile, Fig. 4(c) shows that annealing restores reward fitting. The Spearman correlation of α-DB rebounds from around 0.4 before stage 2 to about 0.5 afterward. In summary, the two-stage schedule preserves the performance gains and recovers reward fitting at the cost of a brief, limited increase in variance in some cases.



(a) Modes of α-SubTB(λ) (b) Loss of α-SubTB(λ) (c) Spearman correlation of α-DB

Figure 4: A case study on **(a)** the mode curves of α-FL-SubTB, **(b)** the loss curves of α-FL-SubTB(λ) and **(c)** the Spearman correlation curves of α-DB in Molecule Generation during training.

Length-Controlling Side Effects. With a 'stop' action allowing P_F to choose trajectory length, the trade-off in Prop. 4.5 induces an additional effect. As demonstrated in Fig. 5, as α is set larger, trajectories are lengthened since stronger exploitation may shift mass from 'stop' to constructive actions. Intriguingly, Fig. 5 also shows that average per-action entropy increases with trajectory length, plausibly reflecting accumulated uncertainty over longer horizons. A formal analysis is left to future work.

Ablation Studies. We assess the sensitivity of α-GFNs via an ablation on the number of modes across α settings for molecule generation. As shown in Table 4, performance gains persist even when α is not tuned to its optimal value: for DB, FL-DB, SubTB(λ), FL-SubTB(λ), and TB, adjusting α generally yields more modes. These results indicate that α-GFNs exhibit low sensitivity to precise α selection while still delivering consistent improvements.

Additional experimental results are in App. D.2.

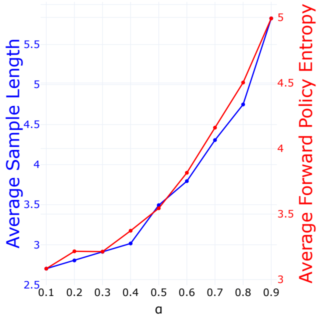


Figure 5: Sample length and entropy with α-FL-SubTB in Molecule Generation.

Table 4: Number of modes vs. α in Molecule Generation. Entries surpassing the baseline are set in **boldface**.

Modes↑	α								
Objective	0.1	0.2	0.3	0.4	0.5 (Baseline)	0.6	0.7	0.8	0.9
DB	5.40 ± 0.89	10.00 ± 4.18	**11.20** ± 4.76	**11.40** ± 3.21	10.00 ± 3.08	**14.40** ± 2.41	**11.80** ± 3.96	**11.00** ± 4.90	**13.40** ± 3.21
FL-DB	5.80 ± 1.48	5.40 ± 1.14	6.40 ± 3.13	**9.17** ± 2.14	9.00 ± 1.58	**13.20** ± 4.66	**14.60** ± 6.07	**24.00** ± 3.61	**25.00** ± 4.36
SubTB(λ)	20.20 ± 5.63	18.20 ± 4.49	**22.80** ± 3.77	**24.00** ± 6.44	22.20 ± 2.77	**26.40** ± 2.30	22.00 ± 5.43	**25.20** ± 3.63	**24.00** ± 5.79
FL-SubTB(λ)	8.40 ± 3.21	11.00 ± 3.94	12.00 ± 4.95	12.60 ± 2.61	16.00 ± 3.16	**18.20** ± 5.31	**23.40** ± 6.62	**34.20** ± 5.67	**39.20** ± 7.05
TB	19.80 ± 3.83	29.60 ± 7.30	37.00 ± 7.42	34.40 ± 6.54	38.40 ± 6.11	**40.20** ± 5.07	36.80 ± 4.44	**39.20** ± 9.81	**60.60** ± 32.39

6 CONCLUSION

In this work, we uncover the implicit equal weighting of forward and backward policies in GFlowNet objectives through an extended connection to Markov chains. Building on this finding, we introduce α-GFNs, which provide a controllable exploration–exploitation trade-off by mixing forward and backward policies with a hyperparameter α. The convergence of these mixed objectives is established via their link to Markov chains. We further explain the role of different α values through a gradient-based analysis and propose a scheduled training algorithm that combines the benefits of multiple α values. Experiments across diverse domains show that α-GFNs consistently outperform vanilla GFlowNets by discovering more high-reward modes while maintaining sample diversity, highlighting the practical value of α. These results strengthen the link between Markov chain theory and GFlowNet practice and open promising directions for future research.

References

D. Aldous and James Fill. Reversible Markov chains and random walks on graphs. 01 2002.

Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. Flow network based generative models for non-iterative diverse candidate generation. *Advances in Neural Information Processing Systems*, 34:27381–27394, 2021.

Yoshua Bengio, Salem Lahlou, Tristan Deleu, Edward J Hu, Mo Tiwari, and Emmanuel Bengio. GFlowNet foundations. *Journal of Machine Learning Research*, 24(210):1–55, 2023.

Stephen Brooks. Markov chain Monte Carlo method and its application. *Journal of the royal statistical society: series D (the Statistician)*, 47(1):69–100, 1998.

Tristan Deleu and Yoshua Bengio. Generative flow networks: a Markov chain perspective. *arXiv preprint arXiv:2307.01422*, 2023.

Tristan Deleu, Padideh Nouri, Nikolay Malkin, Doina Precup, and Yoshua Bengio. Discrete probabilistic inference as control in multi-path environments. In *The 40th Conference on Uncertainty in Artificial Intelligence*, 2024. URL https://openreview.net/forum?id=3C69sU1YkK.

Randal Douc, Eric Moulines, Pierre Priouret, Philippe Soulier, Randal Douc, Eric Moulines, Pierre Priouret, and Philippe Soulier. *Markov chains: Basic definitions*. Springer, 2018.

Edward J Hu, Moksh Jain, Eric Elmoznino, Younesse Kaddar, Guillaume Lajoie, Yoshua Bengio, and Nikolay Malkin. Amortizing intractable inference in large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

Rui Hu, Yifan Zhang, Zhuoran Li, and Longbo Huang. Beyond squared error: Exploring loss design for enhanced training of generative flow networks. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=4NTrco82W0.

Moksh Jain, Sharath Chandra Raparthy, Alex Hernández-Garcıa, Jarrid Rector-Brooks, Yoshua Bengio, Santiago Miret, and Emmanuel Bengio. Multi-objective GFlowNet. In *International conference on machine learning*, pp. 14631–14653. PMLR, 2023.

Hyosoon Jang, Minsu Kim, and Sungsoo Ahn. Learning energy decompositions for partial inference in GFlownets, 2024. URL https://openreview.net/forum?id=P15CHILQlg.

Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. In *International conference on machine learning*, pp. 2323–2332. PMLR, 2018.

Minsu Kim, Joohwan Ko, Taeyoung Yun, Dinghuai Zhang, Ling Pan, Woochang Kim, Jinkyoo Park, Emmanuel Bengio, and Yoshua Bengio. Learning to scale logits for temperature-conditional GFlowNets. In *ICML*, 2024.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Zhen Liu, Tim Z Xiao, Weiyang Liu, Yoshua Bengio, and Dinghuai Zhang. Efficient diversity-preserving diffusion alignment via gradient-informed GFlowNets. In *The Thirteenth International Conference on Learning Representations*, 2024.

Kanika Madan, Jarrid Rector-Brooks, Maksym Korablyov, Emmanuel Bengio, Moksh Jain, Andrei Cristian Nica, Tom Bosc, Yoshua Bengio, and Nikolay Malkin. Learning GFlowNets from partial episodes for improved convergence and stability. In *International Conference on Machine Learning*, pp. 23467–23483. PMLR, 2023.

Nikolay Malkin, Moksh Jain, Emmanuel Bengio, Chen Sun, and Yoshua Bengio. Trajectory balance: Improved credit assignment in GFlowNets. *Advances in Neural Information Processing Systems*, 35: 5955–5967, 2022.

Sobhan Mohammadpour, Emmanuel Bengio, Emma Frejinger, and Pierre-Luc Bacon. Maximum entropy GFlowNets with soft Q-learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 2593–2601. PMLR, 2024.

Ling Pan, Nikolay Malkin, Dinghuai Zhang, and Yoshua Bengio. Better training of GFlowNets with local credit and incomplete trajectories. In *International Conference on Machine Learning*, pp. 26878–26890. PMLR, 2023.

Tiago Silva, Rodrigo Barreto Alves, Eliezer de Souza da Silva, Amauri H Souza, Vikas Garg, Samuel Kaski, and Diego Mesquita. When do GFlownets learn the right distribution? In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=9GsgCUJtic.

Yunchong Song, Chenghu Zhou, Xinbing Wang, and Zhouhan Lin. Ordered GNN: Ordering message passing to deal with heterophily and over-smoothing. In *The Eleventh International Conference on Learning Representations*, 2023.

Zitao Song, Chao Yang, Chaojie Wang, Bo An, and Shuang Li. Latent logic tree extraction for event sequence explanation from LLMs. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 46238–46258, 2024.

Daniil Tiapkin, Nikita Morozov, Alexey Naumov, and Dmitry P Vetrov. Generative flow networks as entropy-regularized RL. In *International Conference on Artificial Intelligence and Statistics*, pp. 4213–4221. PMLR, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Siddarth Venkatraman, Moksh Jain, Luca Scimeca, Minsu Kim, Marcin Sendera, Mohsin Hasan, Luke Rowe, Sarthak Mittal, Pablo Lemos, Emmanuel Bengio, Alexandre Adam, Jarrid Rector-Brooks, Yoshua Bengio, Glen Berseth, and Nikolay Malkin. Amortizing intractable inference in diffusion models for vision, language, and control. *Advances in neural information processing systems*, 37:76080–76114, 2024.

Taeyoung Yun, Dinghuai Zhang, Jinkyoo Park, and Ling Pan. Learning to sample effective and diverse prompts for text-to-image generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 23625–23635, 2025.

David W Zhang, Corrado Rainone, Markus Peschl, and Roberto Bondesan. Robust scheduling with GFlowNets. In *The Eleventh International Conference on Learning Representations*, 2023.

Dinghuai Zhang, Nikolay Malkin, Zhen Liu, Alexandra Volokhova, Aaron Courville, and Yoshua Bengio. Generative flow networks for discrete probabilistic modeling. In *International Conference on Machine Learning*, pp. 26412–26428. PMLR, 2022.

Dinghuai Zhang, Ricky TQ Chen, Cheng-Hao Liu, Aaron Courville, and Yoshua Bengio. Diffusion generative flow samplers: Improving learning signals through partial trajectory optimization. In *The Twelfth International Conference on Learning Representations*, 2024.

Mingyang Zhou, Zichao Yan, Elliot Layne, Nikolay Malkin, Dinghuai Zhang, Moksh Jain, Mathieu Blanchette, and Yoshua Bengio. PhyloGFN: Phylogenetic inference with generative flow networks. In *The Twelfth International Conference on Learning Representations*, 2024.

Xuekai Zhu, Daixuan Cheng, Dinghuai Zhang, Hengli Li, Kaiyan Zhang, Che Jiang, Youbang Sun, Ermo Hua, Yuxin Zuo, Xingtai Lv, et al. FlowRL: Matching reward distributions for LLM reasoning. *arXiv preprint arXiv:2509.15207*, 2025.

Yiheng Zhu, Jialu Wu, Chaowen Hu, Jiahuan Yan, Tingjun Hou, Jian Wu, et al. Sample-efficient multi-objective molecular optimization with GFlowNets. *Advances in Neural Information Processing Systems*, 36:79667–79684, 2023.

A GFlowNet Objective Definitions and Related Discussions

A.1 Definitions

In this section, we provide the formal definitions of vanilla GFlowNet training objectives where both the forward policy P_F and the backward policy P_B are used, including the Detailed Balance Loss (DB, Bengio et al. (2023)), Trajectory Balance Loss (TB, Malkin et al. (2022)) and the convex combination of SubTB, SubTB(λ) (Madan et al., 2023) along with Forward-Looking (FL) (Pan et al., 2023) variants of DB (FL-DB) and SubTB(λ) (FL-SubTB(λ)).

Definition A.1. (DB, Bengio et al. (2023)) Given a state flow function $F(\cdot) : S \to \mathbb{R}_+$, a forward policy $P_F(\cdot|\cdot) : \mathbb{A} \to [0, 1]$, a backward policy $P_B(\cdot|\cdot) : \mathbb{A} \to [0, 1]$, Detailed Balance targets at

$$F(s)P_F(s'|s) = F(s)P_B(s|s'), \quad \text{for every } (s, s') \in \mathbb{A} \tag{5}$$

where $\forall s \in \mathcal{X}, F(s)P_F(s_f|s) = R(s)$. And the corresponding loss function is

$$L_{\text{DB}}(s, s') = \log^2\left(\frac{F(s)P_F(s'|s)}{F(s')P_B(s|s')}\right). \tag{6}$$

Definition A.2. (TB, Malkin et al. (2022)) Given a state flow function $F(\cdot) : S \to \mathbb{R}_+$, a forward policy $P_F(\cdot|\cdot) : \mathbb{A} \to [0, 1]$, a backward policy $P_B(\cdot|\cdot) : \mathbb{A} \to [0, 1]$, for a complete trajectory $\mathfrak{t}^f = (s_0 = s_s, s_1, \ldots, s_{n-1} = x, s_n = s_f) \in \mathfrak{T}$ where $\mathfrak{T}^{\text{flow}}$ is the set of complete trajectories in the graph, TB targets at

$$F(s_0)P_F(s_f|x)\prod_{i=1}^{n-1} P_F(s_i|s_{i-1}) = R(s_n)\prod_{i=1}^{n-1} P_B(s_{i-1}|s_i) \tag{7}$$

And the corresponding loss function is

$$L_{\text{TB}}(\mathfrak{t}^f) = \log^2\left(\frac{F(s_0)P_F(s_f|x)\prod\limits_{i=1}^{n-1} P_F(s_i|s_{i-1})}{R(s_n)\prod\limits_{i=1}^{n-1} P_B(s_{i-1}|s_i)}\right). \tag{8}$$

Note that $F(s_0) = F(s_f) = \sum_{x\in\mathcal{X}} R(x)$ and $P_B(x|s_f) = \frac{R(x)}{\sum\limits_{x\in\mathcal{X}} R(x)}$.

Definition A.3. (SubTB, Madan et al. (2023)) Given a partial trajectory $\mathfrak{t}' = (s_k, s_{k+1}, \ldots, s_{k+m}) \subset \mathfrak{t}^f \in \mathfrak{T}^{\text{flow}}$, a state flow function $F(\cdot) : S \to \mathbb{R}_+$, a forward policy $P_F(\cdot|\cdot) : \mathbb{A} \to [0, 1]$, a backward policy $P_B(\cdot|\cdot) : \mathbb{A} \to [0, 1]$, SubTB targets at Eq. 1, and the corresponding loss function is

$$L_{\text{SubTB}}(\mathfrak{t}') = \log^2\left(\frac{F(s_k)\prod_{i=1}^{m} P_F(s_{k+i}|s_{k+i-1})}{F(s_{k+m})\prod_{i=1}^{m} P_B(s_{k+i-1}|s_{k+i})}\right). \tag{9}$$

Definition A.4. (SubTB(λ), Madan et al. (2023)) Given a complete trajectory $\mathfrak{t}^f = (s_0 = s_s, s_1, \ldots, s_{n-1} = x, s_n = s_f) \in \mathfrak{T}^{\text{flow}}$, define an extracted subtrajectory $\mathfrak{t}_{i:j}$ to be

$$\mathfrak{t}_{i:j} = (s_i, s_{i+1}, \ldots, s_j), \qquad 0 \le i < j \le n. \tag{10}$$

Then, SubTB(λ) loss for the complete trajectory $\mathfrak{t}$ is a convex combination of SubTB loss at these partial trajectories, i.e.

$$L_{\text{SubTB}}(\mathfrak{t}^f, \lambda) = \frac{\sum\limits_{0\le i<j\le n} \lambda^{j-i} L_{\text{SubTB}}(\mathfrak{t}_{i:j})}{\sum\limits_{0\le i<j\le n} \lambda^{j-i}} \tag{11}$$

where $\lambda \in (0, +\infty)$.

Additionally, we consider the Forward Looking (FL) (Pan et al., 2023) variants of DB and SubTB(λ), termed FL-DB and FL-SubTB(λ) respectively. We start with an introduction to the state-level and edge-level energy function.

Definition A.5. (Pan et al., 2023) Given a state-level energy function $\mathcal{E}(\cdot) : S \to \mathbb{R}$, the edge-level energy function is an extension of its state-level counterpart, i.e.

$$\mathcal{E}(s, s') = \mathcal{E}(s') - \mathcal{E}(s) \quad \text{for all } (s, s') \in \mathbb{A}. \tag{12}$$

Applying Def. A.5 to DB and SubTB(λ), the FL variants are obtained:

Definition A.6. (FL-DB, Pan et al. (2023)) Based on the the definition of DB in Def. A.1, with an edge-level energy function $\mathcal{E}(\cdot, \cdot) : \mathbb{A} \to \mathbb{R}$, Forward-Looking Detailed Balance (FL-DB) targets at:

$$F(s)P_F(s'|s) = F(s')P_B(s|s')e^{-\mathcal{E}(s,s')} \tag{13}$$

and the corresponding loss function is

$$L_{\text{FL-DB}}(s, s') = \log^2 \left(\frac{F(s)P_F(s'|s)}{F(s')P_B(s|s')e^{-\mathcal{E}(s,s')}} \right). \tag{14}$$

Definition A.7. (FL-SubTB(λ), Pan et al. (2023)) Based on the definition of SubTB in Def. A.3, given a partial trajectory $\mathfrak{t}_{i:j} = (s_i, s_{i+1}, \dots, s_j) \subset \mathfrak{t} \in \mathfrak{T}^{\text{flow}}$ and an edge-level energy function $\mathcal{E}(\cdot, \cdot) : \mathbb{A} \to \mathbb{R}$, FL-SubTB targets at

$$F(s_k) \prod_{i=1}^{m} P_F(s_{k+i} \mid s_{k+i-1}) \;=\; F(s_{k+m}) \prod_{i=1}^{m} P_B(s_{k+i-1} \mid s_{k+i})\, e^{-\mathcal{E}(s_{k+i-1}, s_{k+i})}.$$

and the corresponding loss function is

$$L_{\text{FL-SubTB}}(\mathfrak{t}_{i:j}) = \log^2 \left(\frac{F(s_i) \prod_{k=1}^{j} P_F(s_{i+k}|s_{i+k-1})}{F(s_j) \prod_{k=1}^{j} P_B(s_{i+k-1}|s_{i+k})} \right) \tag{15}$$

Following Def. A.4, FL-SubTB(λ) is a convex combination of Fl-SubTB. Given $\lambda \in (0, +\infty)$, FL-SubTB(λ) is

$$L_{\text{FL-SubTB}}(\mathfrak{t}^f, \lambda) = \frac{\sum\limits_{0 \le i < j \le n} \lambda^{j-i} L_{\text{FL-SubTB}}(\mathfrak{t}_{i:j})}{\sum\limits_{0 \le i < j \le n} \lambda^{j-i}}. \tag{16}$$

A.2 Discussions

To further demonstrate that α-GFNs are compatible with vanilla GFlowNet training techniques, we formalize FL (Pan et al., 2023) within the Markov chain framework in a matrix form, which suggests FL is a prior to the probability measures:

Theorem A.8. *FL adds a prior to the probability measures of a GFlowNet, i.e. $\pi = \tilde{\pi}\mathcal{E}$, where $\mathcal{E} = diag(e^{-\mathcal{E}(s_0)}, \dots, e^{-\mathcal{E}(s_f)})$ is the diagonal matrix constructed with exponent of the energy function $\mathcal{E}(\cdot)$.*

Proof. Assumpation 4.1 and Prop. 4.2 in Pan et al. (2023) suggests

$$F = \tilde{F}\mathcal{E}$$

is the FL reparameterization of flows, where $F = (F(s_s), \dots, F(s_f))$ is the vector of flows, and $\tilde{F} = (\tilde{F}(s_s), \dots, \tilde{F}(s_f))$ is the reparameterized counterpart. Coupling with the fact that state flows are state-level probability measures (Deleu & Bengio, 2023), it is direct that

$$\pi = \tilde{\pi}\mathcal{E}$$

where π and $\tilde{\pi}$ are the probability measures corresponding to F and $\tilde{F}$ respectively. □

B Supporting Theoretical Framework

The following theoretical part addresses the fundamental results of our paper under the assumption that the state space is finite. This assumption is reasonable since the state space of GFlowNets are compositional (Bengio et al., 2021; 2023). Throughout, we denote by P the true transition probability of the graph to be learned, which corresponds to the forward policy P_F in GFlowNets. We first summarize the results we prove and then prove the results one by one in subsections.

(i) From GFN to MC: We show that GFlowNets can be represented as irreducible Markov Chains.

In particular, the resulting Markov chain is positive recurrent, admits a unique stationary distribution π to which the Markov chain converges from any state.

(ii) From MC to GFN under constraints: We show that every reducible Markov chain with transition probability P satisfying a specific finite set of linear constraints is a GFlowNet.

We call such Markov chains **GFNMC** (GFlowNet Markov chain). Such a Markov chain allows for a detailed balance $\tilde{P}$, which is again a GFNMC sharing the same trajectories (in the reverse direction), the same stationary distribution as well as the same eigenvalues (not necessarily the same eigenvectors).

Furthermore any α-GFNMC is again irreducible, share the same stationary distribution but not necessarily the same eigenvalues.

(iii) Convergence rate: We show that the periodicity of every GFNMC (GFlowNet Markov Chain) is equal to the greatest common denominator of all the trajectory length. It is in particular the case for any reasonable graph with various trajectory length to be aperiodic (periodicity of 1).

If not, for instance if all trajectories share the same length, we show that for any $0 < \alpha < 1$, the periodicity of the α-GFN is either 2 or 1.

This is important in terms of convergence as aperiodic Markov chains are ergodic geometric, that is the convergence to the stationary distribution also holds in total variation as exponential rate determined by the second largest eigenvalue. In terms of sampling, such results allow for a central limit theorem akin to classical Monte Carlo.

Denote with β, $\tilde{\beta}$ and β_α the second largest eigenvalues. While $\beta = \tilde{\beta}$, there is no result as β_α as a function of β as it is highly non-linear. Nevertheless, it might drastically improve by mixing. It justifies the approach to take the hyper parameter α as trainable too to achieve a good convergence and a good sampling result.

(iv) Freedom of the Choice of the Backward Policy: So far, the analysis of α-GFN has been done by considering $P_\alpha = \alpha P + (1-\alpha)\tilde{P}$ where $\tilde{P}$ is exactly the time reversal of P. However, in practice, the training to learn P is done by using a different, usually fixed backward policy P_B that is proportional to the reward in the last step of the graph. A loss function furthermore enforces some weak to strong form of reversibility. It is apriori not clear that the resulting stationary distribution will be the one of P.

We, however, show that after training P_α will share the same stationary distribution as P and P_B. Furthermore, if after training P_α set the different losses taken into account to 0, then P_α would satisfy the corresponding balanced constraints.

Prior work (Deleu & Bengio, 2023) touches on **(i)**, but it does not fully formalize the Markov-chain-to-GFlowNet (MC→GFN) connection. In particular, key Markov chain details underlying this link are left unspecified, and the remaining aspects, **(ii)**, **(iii)**, and **(iv)**, have not, to our knowledge, been investigated. We address these gaps and develop a more complete account of the relationship between GFlowNets and Markov chains.

B.1 GFNs are MCs

We start with giving a detailed version of notations in Sec. 2. Following Bengio et al. (2021), we consider a *directed graph* $(S, \mathbb{A})$ where S is a finite state space and $\mathbb{A} \subseteq S \times S$ is a set of *edges* between states where $s \to s' = (s, s') \in \mathbb{A}$ denotes an edge. A *trajectory* is a finite sequence $\mathfrak{t}^f = (s_0, \ldots, s_N)$ where $s_n \to s_{n+1}$ is an edge for any $n \leq N-1$ where $N \geq 1$. The directed graph is called *acyclic* if every such trajectory satisfies $s_0 \neq s_N$. A directed acyclic graph is called *pointed* if there exists a *source state* s_s and a *final state* s_f such that for every other state s, there exists a trajectory starting from s_s, running through s and ending in s_f. Any such finite trajectory starting in s_s and ending in s_f is called *complete*.[2] From now one we only consider pointed directed acyclic graphs where the GFlowNet theoretical framework is built (Bengio et al., 2023) and denote by $\mathfrak{T}^{\text{flow}}$ the set of complete trajectories $\mathfrak{t}^f = (s_0, \ldots, s_N)$ where $s_0 = s_s$ and $s_N = s_f$.

To realise the link with Markov chains, let $\mathfrak{T} = S^{\mathbb{N}_0}$ be the set of infinite trajectories $\mathfrak{t} = (s_0, s_1, \ldots)$ endowed with the product σ-algebra $\mathcal{T} = \otimes \mathcal{S}$ where $\mathcal{S} = 2^S$. We further denote by $X = (X_t)$ the *canonical process* on $\mathfrak{T}$ where $X_n(\mathfrak{t}) = s_n$ is the n-th state of the trajectory $\mathfrak{t}$. In other terms X is the identity on $\mathfrak{T}$ since $X(\mathfrak{t}) = \mathfrak{t}$.[3] We finally denote by $\mathcal{T}_n = \sigma(X_m \colon m \leq n)$ the filtration generated by the canonical process.

Using a finite Kolmogorov extension argument, Bengio et al. (2021) show that any Markovian probability measure $\mathbf{P}^{\text{flow}}$ on $\mathfrak{T}^{\text{flow}}$ is uniquely given by a transition probability $P(s'|s) = P_{ss'}$ for every edge $s \to s'$.

Remark B.1. Throughout, we assume that $P(s'|s) > 0$ for every edge $s \to s'$.

This transition probability can be extended to $S \times S$ by defining $P(s_s|s_f) = 1$ and $P(s'|s) = 0$ for any other pair $s \to s'$ which is not an edge. In other terms the probability of returning to the source state from the final state is equal to 1. Such an extension also defines a Markovian probability $\mathbf{P}^{s_s}$ on the space of infinite trajectories with corresponding Markov chain starting at the source state s_s. The following theorem embed Markovian GFlowNets into a subset of Markov Chains with specific properties.

Theorem B.2. *The Markovian GFlowNet probability $\mathbf{P}^{flow}$* ***coincides exactly*** *with $\mathbf{P}^{s_s}$ on the set of complete trajectories $\mathfrak{T}^{flow}$. Furthermore, the resulting Markov Chain is* ***irreducible****.*

In order to state Thm. B.2 correctly as $\mathfrak{T}^{\text{flow}}$ is not even a subset of $\mathfrak{T}$, let us consider random times and stopping times.

Definition B.3. A function $\tau \colon \mathfrak{T} \to \mathbb{N}_0 \cup \{\infty\}$ is called a *random time* if τ is measurable and a *stopping time* if $\{\tau \leq n\}$ is in $\mathcal{T}_n$ for every n.

Typical example of stopping times are the first time a trajectory $\mathfrak{t}$ satisfies a condition. In particular, we make use of

$$\tau^s(\mathfrak{t}) = \inf\{n \colon X_n(\mathfrak{t}) = s\} \quad \text{and} \quad \tau_+^s(\mathfrak{t}) = \inf\{n \geq 1 \colon X_n(\mathfrak{t}) = s, n > 0\},$$

which are the first time and first time ≥ 1 such that the trajectory $\mathfrak{t}$ visits the state s, respectively. Given a stopping time $\tau < \infty$, we denote by $X_\tau(\mathfrak{t}) = \mathfrak{t}_{\tau(\mathfrak{t})}$ the value of the trajectory at this particular random time $\tau(\mathfrak{t})$, as well as the stopped Markov chain $X^\tau := (X_{n \wedge \tau})$. The stopped Markov chain is just a slice of the trajectory $\mathfrak{t}$ between 0 and $\tau(\mathfrak{t})$ and constant afterwards:

$$X^\tau(\mathfrak{t}) = (\underbrace{s_0, \ldots, s_{\tau(\mathfrak{t})}}_{\text{slice before } \tau(\mathfrak{t})}, \underbrace{s_{\tau(\mathfrak{t})}, \ldots}_{\text{constant after}}).$$

Going back to flows, we define

$$\sigma(\mathfrak{t}) = \begin{cases} N & \text{if } (s_0, \ldots, s_N) \in \mathfrak{T}^{\text{flow}} \text{ is a complete flow trajectory for some } N \\ 0 & \text{otherwise} \end{cases}$$

[2] Note that by definition, any state in S is element of a complete trajectory.

[3] Note that any Markov chain is about defining a probability measure on the space of trajectories $\mathfrak{T}$ making the canonical process to satisfy the Markov property.

as the function returning the length of the complete flow trajectory if the infinite trajectory starts in $\mathfrak{T}^{\text{flow}}$ and 0 otherwise.

Lemma B.4. *The function σ is a uniformly bounded random time. Furthermore, $\mathfrak{T}^{flow}$ is in bijection with $\{\mathfrak{t}\colon \sigma(\mathfrak{t}) \geq 1\}$ which is a measurable subset of $\mathfrak{T}$.*

Proof. Since the state space is finite, it follows that $\sigma < \#S + 1$ and therefore uniformly bounded. It follows that the number of elementary flows is finite and therefore σ is a finite sum of simple random variables greater than 1, hence a random variable. Finally, by definition $\mathfrak{T}^{\text{flow}}$ is in bijection with $\{\sigma \geq 1\}$ and therefore measurable since σ is measurable. □

Remark B.5. If the state space is infinite, then elementary flows might be of arbitrary length. Hence, the set of elementary flows might be of uncountable cardinality. In this case, the measurability argument does not hold without further assumptions. Also, even if σ is measurable, it is not clear a priori that it is a stopping time, and in general likely not.

Even if σ is not a stopping time, it coincides with the stopping time τ^{s_f} with probability 1 for the Markov probability starting from s_s.

Proposition B.6. *Let $\mathbf{P}^{s_s}$ be the Markovian probability starting from the source state s_s. Then it follows that*

$$\mathbf{P}^{s_s}[\sigma = \tau^{s_f}] = 1$$

and $\mathbf{P}^{s_s}$ coincides with $\mathbf{P}^{flow}$ in the sense that for any $\mathfrak{t}^f$ in $\mathfrak{T}^{flow}$ it holds

$$\mathbf{P}^{s_s}[X_{[0:\tau^{s_f}]} = \mathfrak{t}^f] = \mathbf{P}^{flow}[\mathfrak{t}^f].$$

Proof. It is clear that for any trajectory $\mathfrak{t}$ such that $\sigma(\mathfrak{t}) \geq 1$, it holds that $\tau^{s_f}(\mathfrak{t}) = \sigma(\mathfrak{t})$. And by the definition of $\mathbf{P}^{s_s}$ from the same transition probability, we get that

$$\mathbf{P}^{s_s}[X_{[0:\tau^{s_f}]} = \mathfrak{t}^f] = \mathbf{P}^{s_s}[X_{[0:\sigma]} = \mathfrak{t}^f] = \mathbf{P}^{\text{flow}}[\mathfrak{t}^f]$$

In particular, $\mathbf{P}^{s_s}$ has measure 1 on the set where $\{\mathfrak{t}\colon \sigma(\mathfrak{t}) \geq 1\}$. □

Recall that a Markov chain is called **irreducible** if for every pair of states s, s' it holds that $\mathbf{P}^s[\tau^{s'} < \infty] = 1$ for any two states s and s'.

Proposition B.7. *The Markov chain resulting from P is irreducible.*

Proof. Any flow trajectory in $\{\sigma \geq 1\}$ starting in s_s will reach s_f with probability 1 in a bounded amount of steps. Furthermore, from $P(s'|s) > 0$ for any edge $s \to s'$ shows that each state is visited by a flow trajectory with strict positive probability. The assumption that $P(s_s|s_f) = 1$ and the fact that $P(s_s|s_s) = 0$ shows that starting from any state s, the probability to reach s_f and therefore s_s is equal to 1. From s_s to any other state $s' \neq s_s$ is with strict positive probability, hence from strong Markov property, it follows that any state s' is accessible from any state s with strict positive probability, that is $\mathbf{P}^s[\tau^{s'} < \infty] = 1$. □

Since the Markov chain with respect to the transition kernel P is defined over a finite discrete state space, it is also **Harris recurrent** and **positive recurrent**.

B.2 MC Constraints to be a GFN

Remark B.8. For an easy definition of pointed directed graphs, it is better to distinguish between source state s_s and final state s_f. However, since the transition from s_s to s_f is with probability one, from Markov Chain perspective, it is equivalent to identify them both by setting $s_s = s_f = \bar{s}$, which is also the practice in Deleu & Bengio (2023).

We consider a transition probability P such that the resulting Markov chain is irreducible. In particular, it is positive recurrent and Harris recurrent and has a stationary distribution π since it is defined over a finite discrete state space. For any state s, τ^s as well as τ^s_+ are finite stopping time with finite expectation.

For this Markov Chain to be a GFlowNet, it is necessary and sufficient that any finite trajectory $\mathfrak{t}^f = (s_0 = \bar{s}, s_1, \ldots, s_N = \bar{s})$ does not contain inner loop. In other terms for any state $s \neq \bar{s}$, the probability of returning to s is 1 and it shall pass first through $\bar{s}$ also with probability 1.

Theorem B.9. *A transition probability P for an irreducible Markov chain is a GFlowNet if and only if*

$$\mathbf{P}^s[\tau^{\bar{s}} < \tau^s_+] = 1 \quad \text{for every } s \neq \bar{s}. \tag{17}$$

The condition in Eq. 17 *translates into*

$$\pi_s(Z_{\bar{s}\bar{s}} - Z_{s\bar{s}}) + \pi_{\bar{s}}(Z_{ss} - Z_{\bar{s}s}) = \pi_{\bar{s}},$$

where Z is the fundamental matrix

$$Z := (I - P + \Pi)^{-1} = \sum (P - \Pi)^n$$

with Π being the matrix with each row equal to π.

Here, we use Z to align with the classical Markov chain theory. **Note that this Z is not the amount of flows in GFlowNets.**

Proof. The first statement Eq. 17 is clearly equivalent to the Markov Chain is concentrated onto a set of complete trajectories that corresponds to a pointed acyclic directed graph. In particular $\tau^{\bar{s}}_+ \leq \#S$. Now from (Aldous & Fill, 2002, Corollary 2.8), it holds that

$$\mathbf{P}^s[\tau^{\bar{s}} < \tau^s_+] = \frac{1}{\pi_s(E^s[\tau^{\bar{s}}] + E^{\bar{s}}[\tau^s])}$$

while (Aldous & Fill, 2002, Lemma 2.12) states that

$$\pi_s E^s[\tau^{\bar{s}}] = \frac{\pi_s}{\pi_{\bar{s}}}(Z_{\bar{s}\bar{s}} - Z_{s\bar{s}}) \quad \text{and} \quad \pi_s E^{\bar{s}}[\tau^s] = (Z_{ss} - Z_{\bar{s}s})$$

Together with Eq. 17, it yields

$$\pi_s(Z_{\bar{s}\bar{s}} - Z_{s\bar{s}}) + \pi_{\bar{s}}(Z_{ss} - Z_{\bar{s}s}) = \pi_{\bar{s}}, \quad \text{for all } s \neq \bar{s}.$$

□

Given an irreducible Markov chain with transition probability P and resulting stationary distribution π we can define the balanced chain $\tilde{P}$ as

$$\tilde{P}_{ss'} = \frac{\pi_{s'}}{\pi_s} P_{s's}$$

Proposition B.10. *The Markov chain $\tilde{P}$ is again irreducible with same stationary distribution π and eigenvalues as P. Furthermore, it is also a GFNMC.*

Proof. The first statement is classical. Let us show that $\tilde{P}$ is a GFNMC. Denoting with $D = \operatorname{diag}(\pi)$, it holds that $\tilde{P} = D^{-1}P^\top D$. It is easy to check that $D\Pi D^{-1} = \Pi^\top$. It follows that

$$\begin{aligned} Z = (I - \tilde{P} - \Pi)^{-1} &= (I - D^{-1}P^\top D - D^{-1}\Pi^\top D)^{-1} \\ &= D^{-1}\left((I - P - \Pi)^\top\right)^{-1} D \\ &= D^{-1}\left((I - P - \Pi)^{-1}\right)^\top D \\ &= D^{-1}Z^\top D \end{aligned}$$

showing that the fundamental matrices satisfy the same balance equation. It follows that

$$\begin{aligned} \pi_s\left(\tilde{Z}_{\bar{s}\bar{s}} - \tilde{Z}_{s\bar{s}}\right) + \pi_{\bar{s}}\left(\tilde{Z}_{ss} - \tilde{Z}_{\bar{s}s}\right) &= \pi_s\left(Z_{\bar{s}\bar{s}} - \frac{\pi_{\bar{s}}}{\pi_s}Z_{\bar{s}s}\right) + \pi_{\bar{s}}\left(Z_{ss} - \frac{\pi_s}{\pi_{\bar{s}}}Z_{s\bar{s}}\right) \\ &= \pi_s\left(Z_{\bar{s}\bar{s}} - Z_{s\bar{s}}\right) + \pi_{\bar{s}}\left(Z_{ss} - Z_{\bar{s}s}\right) \\ &= \pi_{\bar{s}} \end{aligned}$$

showing that $\tilde{P}$ is a GFNMC according to Thm. B.9. □

Taking $0 < \alpha < 1$, the α-GFN given by $\alpha P + (1-\alpha)\tilde{P}$ is clearly irreducible with the same stationary distribution. However, it is no longer of GFNMC type as some inner loop might be present in trajectories from $\bar{s}$ to $\bar{s}$. It also do not share the same set of eigenvalues. Nevertheless, it is closely related to the properties of α-GFNs, as demonstrated in the following.

B.3 Convergence Rate

Let P be the transition probability of an irreducible Markov chain of GFNMC type. Let π be the stationary distribution, $\tilde{P}$ the balanced chain. We further denote by β the largest modulus of the eigenvalue of P which is not 1 and β_α the largest modulus of the eigenvalue of $P_\alpha = \alpha P + (1-\alpha)\tilde{P}$. Let further $d(s) = \gcd\{n \colon P^n_{ss} > 0\}$ be the periodicity of the Markov chain. It is a classical result that for irreducible chains, this periodicity is the same for each state. Hence, let d, $\tilde{d}$ and d_α be the corresponding periodicities of P, $\tilde{P}$ and αP, respectively.

Proposition B.11. *It holds that the periodicity of P is equal to the greatest common divisor of the length of all complete trajectories. Furthermore $d = \tilde{d}$.*

Finally for $0 < \alpha < 1$, it holds that

$$d_\alpha = \begin{cases} 1 & \textit{if } d \textit{ is odd} \\ 1 \textit{ or } 2 & \textit{if } d \textit{ is even} \end{cases}$$

Proof. It is clear that the periodicity of P and therefore $\tilde{P}$ will coincide with the greatest common divisor of all trajectory lengths.

However for $0 < \alpha < 1$, it is then possible to have trajectories $s \to s' \to s$ with strict positive probability (since it can go with strict probability from s to s' with P and strict positive probability from s' to s with $\tilde{P}$). Hence the periodicity of $\alpha P + (1-\alpha)\tilde{P}$ divides d as well as 2 showing the result. □

Remark B.12. In the case of GFN, usually with very large graphs, either you have many possible lengths that make the periodicity already 1, or all trajectories are of the same length, and the periodicity is either 1 or 2 for the α-GFN. Although P in this case have a very large periodicity, $\tilde{P}$ will drastically reduce it to at least 2 if not 1.

Having a periodicity of 1 is important in terms of convergence, as it is geometric ergodic. In this case the convergence to the stationary distribution also holds in total variation as exponential rate determined by the second largest eigenvalue and a central limit theorem also holds that ensure correct bounds in terms of sampling akin to Monte Carlo. The rate of convergence is dominated by the second largest eigenvalue. However, as eigenvalues are highly non-linear, it is not possible to find a direct relation between β_α and β. Nevertheless, it is highly possible that mixing with different α can improve the convergence rate depending on the structure of the original Markov Chain. **In particular, setting $\alpha = 0.5$ directly yields the discussions corresponding to vanilla GFlowNets.**

C PROOFS

App. B enables formal discussions on GFlowNets from the Markov chain perspective. Building on this aspect, we now take a closer look into the objectives of α-GFNs and vanilla GFlowNets. Before diving into the corresponding theorems and propositions in the main body, we first show an equivalence between flows and probability measures with a generalization of Deleu & Bengio (2023). Although this equivalence has been used in previous parts following Deleu & Bengio (2023), we give a generalization for clarity in the following.

By defining

$$Z_{\text{state}} = \sum_{s \in S} F(s), \quad \pi(s) = \frac{F(s)}{Z_{\text{state}}}, \tag{18}$$

we directly obtain the probability measures at each state $s \in S$. Note that $\pi(\cdot)$ is a probability measure since $\sum_{s \in S} \pi(s) = 1$.

Equipped with Eq. 18 and App. B, we now derive the proofs for corresponding statements.

C.1 PROOF FOR PROP. 4.1

Since $P_{0.5} = \frac{1}{2}P_F + \frac{1}{2}P_B$, given the partial trajectory $\mathfrak{t}' = (s_k, s_{k+1}, \dots, s_{k+m})$, the reversibility of $P_{0.5}$ suggests

$$\pi(s_k) \prod_{i=1}^{m} P_{0.5}(s_{k+i}|s_{k+i-1}) = \pi(s_{k+m}) \prod_{i=1}^{m} P_{0.5}(s_{k+i-1}|s_{k+i})$$

which extends to

$$\pi(s_k) \prod_{i=1}^{m} \left(\frac{1}{2} P_F(s_{k+i}|s_{k+i-1}) + \frac{1}{2} P_B(s_{k+i}|s_{k+i-1}) \right) = \pi(s_{k+m}) \prod_{i=1}^{m} \left(\frac{1}{2} P_F(s_{k+i-1}|s_{k+i}) + \frac{1}{2} P_B(s_{k+i-1}|s_{k+i}) \right)$$

By noticing the fact that both P_F and P_B are one-directional within this partial trajectory, i.e. $\forall (s, s') \subset \mathfrak{t}'$, if $P_F(s'|s) > 0$, then $P_F(s|s') = 0$, $P_B(s|s') > 0$, $P_B(s'|s) = 0$, it follows that

$$(\frac{1}{2})^m \pi(s_k) \prod_{i=1}^{m} P_F(s_{k+i}|s_{k+i-1}) = (\frac{1}{2})^m \pi(s_{k+m}) \prod_{i=1}^{m} P_B(s_{k+i-1}|s_{k+i})$$

By eliminating $(\frac{1}{2})^m$ on both sides of the equation, one has

$$\pi(s_k) \prod_{i=1}^{m} P_F(s_{k+i}|s_{k+i-1}) = \pi(s_{k+m}) \prod_{i=1}^{m} P_B(s_{k+i-1}|s_{k+i})$$

C.2 PROOF FOR THM. 4.2

We first prove the claim that DB, SubTB and TB correspond to edge-level, partial-trajectory-level and trajectory-level reversibility of $P_{0.5}$. Using Eq. 18, we obtain that for DB,

$$\frac{F(s)}{Z_{\text{state}}} P_F(s'|s) = \frac{F(s)}{Z_{\text{state}}} P_B(s|s')$$

which translates into

$$\pi(s) P_F(s'|s) = \pi(s') P_B(s|s').$$

Obviously, this equation is the reversibility in Prop. 4.1 at an edge-level. Similarly, one can obtain the proofs for SubTB and TB by plugging Eq. 18 into Eq. 1 and Eq. 7 and comparing the equations with Prop. 4.1.

Next, we show that such objectives converge to unique state flows from the Markov chain perspective.

For **DB** and **SubTB**(λ), they both account for the edge-level reversibility (see Def. A.1 and Def. A.4), whereas reaching the edge-level reversibility is actually equivalent to the convergence of DB and, particularly, SubTB(λ). The corresponding proof for DB is straightforward since the edge-level reversibility is identically its training target (see Def. A.1). For SubTB(λ), by noticing that given a complete trajectory $\mathfrak{t}^f = (s_0 = s_s, s_1, \ldots, s_{n-1} = x, s_{n+1} = s_f)$, edge-level reversibility suggests

$$F(s_i)P_F(s_{i+1}|s_i) = F(s_{i+1})P_B(s_{i+1}|s_i), \quad \text{for all } (s_i, s_{i+1}) \subset \mathfrak{t}^f.$$

Therefore, for every partial trajectory $\mathfrak{t}' = (s_k, s_{k+1}, \ldots, s_{k+m}) \subset \mathfrak{t}^f$, by using the fact that $\forall 0 \le i \le k-1, F(s_{i+1}) = \frac{F(s_i)P_F(s_{i+1}|s_i)}{P_B(s_i|s_{i+1})}$, we have

$$F(s_k)\prod_{i=1}^{k} P_F(s_{k+i}|s_{k+i-1}) = F(s_{k+m})\prod_{i=1}^{k} P_B(s_{k+i-1}|s_{k+i}) \tag{19}$$

which suggests the target of SubTB(λ). Hence, it is only required to check whether the edge-level reversibility can be achieved from the Markov chain perspective. In fact, this edge-level reversibility is equivalent to the **detailed balance conditions** (Douc et al., 2018). If detailed balance conditions are satisfied, the probability measures are unique if the corresponding finite-state Markov chain is irreducible and positive recurrent (Douc et al., 2018). Given that flows are also unique in GFlowNets (Bengio et al., 2023), it is required to show whether our MC modeling of GFlowNets achieve unique flows as well. Since flows are unnormalized probability measures, it suffices to show that the Markov transition kernel $P_{0.5}$ yields a irreducible and positive recurrent Markov chain, which is suggested in Prop. **??**. Therefore, DB and SubTB(λ)'s convergence to unique flows is ensured from a Markov chain viewpoint.

For **TB**, we relate to the **Kolmogorov's criterion** (Douc et al., 2018), which is equivalent to the detailed balance conditions for a finite discrete Markov chain like GFlowNets. Kolmogorov's criterion suggest that for any loop $(s_0 = \bar{s}, s_1, \ldots, s_{n-1}, s_n = \bar{s})$, the reversibility at the loop is achieved, i.e.

$$\pi(s_0)\prod_{i=1}^{n} P(s_i|s_{i-1}) = \pi(s_n)\prod_{i=1}^{n} P(s_{i-1}|s_i) \tag{20}$$

if the corresponding finite discrete Markov chain is irreducible and positive recurrent. If one mergers the source state s_s and the final state s_f as Deleu & Bengio (2023), i.e. $s_s = s_f = \bar{s}$, then TB of $P_{0.5}$ targets at

$$F(\bar{s})\prod_{i=1}^{m} P_{0.5}(s_i|s_{i-1}) = F(\bar{s})\prod_{i=1}^{m} P_{0.5}(s_{i-1}|s_i)$$

which translates into

$$\pi(\bar{s})\prod_{i=1}^{m} P_{0.5}(s_i|s_{i-1}) = \pi(\bar{s})\prod_{i=1}^{m} P_{0.5}(s_{i-1}|s_i)$$

for the complete trajectory $\mathfrak{t}^f = (s_0 = s_s, s_1, \ldots, s_{n-1} = x, s_n = s_f)$ since $\pi(s) = \frac{F(s)}{Z_{\text{state}}}$ is the corresponding probability measure and $P_B(x|s_f) = \frac{R(x)}{\sum_{x' \in \mathcal{X}} R(x')}$. Therefore, the convergence of TB is a necessary condition for Kolmogorov's criterion to hold. Due to the fact that $P_{0.5}$ yields a irreducible and positive recurrent Markov chain (see Prop. **??**), if Kolmogorov's criterion holds, the uniqueness of flows is achieved by the convergence of TB as well. However, since TB is only a necessary condition, this uniqueness is relatively fragile, which potentially contribute to instability of TB training (Madan et al., 2023).

C.3 Proof for Prop. 4.4

We first prove the claim that α-DB, α-SubTB and α-TB correspond to edge-level, partial-trajectory-level and trajectory-level reversibility of P_α. The proof is similar to the proof for Thm. 4.2, and we only show the proof for α-DB. Recall the reversibility of α-DB: for any $(s, s') \in \mathbb{A}$

$$\pi(s)P_\alpha(s'|s) = \pi(s')P_\alpha(s|s'). \tag{21}$$

Since both P_F and P_B are one-directional, Eq. 21 implies

$$\alpha\pi(s)P_F(s'|s) = (1-\alpha)\pi(s')P_B(s|s'). \tag{22}$$

On the other hand, recall α-DB, which targets at

$$\alpha F(s)P_F(s'|s) = (1-\alpha)F(s')P_B(s|s'). \tag{23}$$

Plugging $\pi(s) = \frac{F(s)}{Z_{\text{state}}}$ into Equation23 yields

$$\alpha\pi(s)P_F(s'|s) = (1-\alpha)\pi(s')P_B(s|s')$$

which is identical to Eq. 22. The proofs for α-SubTB and α-SubTB(λ) are similar.

Next, we show the convergence of such objectives lead to unique state flows from the Markov chain perspective. This also follows the proof for Thm. 4.2. The convergence of α-DB and α-SubTB(λ) to unique flows are derived by the detailed balance conditions, irreducibility and positive recurrence of the Markov chain with P_α, whereas that of α-TB follows a necessary condition of Kolmogorov's criterion of P_α.

Nevertheless, App. B.3 reveals that the convergence rates to unique flows vary for different α values. Even though this aspect may not be perfectly reflected by loss curves due to the difference between GFlowNet training and MCMC algorithms, the tuning of α still contributes the training of GFlowNets, as suggested in Prop. 4.5.

C.4 Proof for Prop. 4.5

Following Def. 4.3, the loss function of α-SubTB at the partial trajectory $\mathfrak{t} = (s_k, s_{k+1}, \ldots, s_{k+m})$ is

$$L_{\alpha-\text{SubTB}}(\mathfrak{t}') = \log^2\left(\frac{\alpha^m F(s_k)\prod_{i=1}^m P_F(s_{k+i}|s_{k+i-1})}{(1-\alpha)^m F(s_{k+m})\prod_{i=1}^m P_B(s_{k+i-1}|s_{k+i})}\right).$$

We denote by $P_B(\mathfrak{t}') = \prod_{i=1}^m P_B(s_{k+i-1} \mid s_{k+i1})$. Then, the gradient to $P_F(\mathfrak{t}') = \prod_{i=1}^m P_F(s_{k+i} \mid s_{k+i-1})$ is

$$\begin{aligned}\frac{\partial L_{\alpha-\text{SubTB}}(\mathfrak{t}')}{\partial P_F(\mathfrak{t}')} &= \frac{2}{P_F(\mathfrak{t}')}\log\frac{\alpha^m F(s_k)P_F(\mathfrak{t}')}{(1-\alpha)^m F(s_{k+m})P_B(\mathfrak{t}')} \\ &= \frac{2}{P_F(\mathfrak{t}')}\log\frac{F(s_k)P_F(\mathfrak{t}')}{F(s_{k+m})P_B(\mathfrak{t}')} + \frac{2m}{P_F(\mathfrak{t}')}\log\frac{\alpha}{1-\alpha}.\end{aligned}$$

Meanwhile, taking a gradient of Eq. 9 suggests

$$\frac{\partial L_{\text{SubTB}}(\mathfrak{t}')}{\partial P_F(\mathfrak{t}')} = \frac{2}{P_F(\mathfrak{t}')}\log\frac{F(s_k)P_F(\mathfrak{t}')}{F(s_{k+m})P_B(\mathfrak{t}')}.$$

Therefore, it is direct that

$$\frac{\partial L_{\alpha-\text{SubTB}}(\mathfrak{t}')}{\partial P_F(\mathfrak{t}')} = \frac{\partial L_{\text{SubTB}}(\mathfrak{t}')}{\partial P_F(\mathfrak{t}')} + \frac{2m}{P_F(\mathfrak{t}')}\log\frac{\alpha}{1-\alpha}.$$

D Experiments

In this section, we present the detailed experimental settings, computational resource usage, ablation studies of our α-GFNs along with more numerical results and corresponding analysis to support our findings.

D.1 Detailed Experiment Setups and Computational Resource Usage

Set Generation We implement α-DB, α-TB and α-FL-DB based on the open-source code of Pan et al. (2023), where setting $\alpha = 0.5$ yields the baselines. Most details of this task follows Pan et al. (2023), including the models and hyperparameters. To be specific, the dimension of the vocabulary (the action space) is 30, 80 and 100 for small, medium and large sets, respectively. The maximum capacity of sets is capped at 20, 60, and 80 for small, medium and large sets. We employ the same intermediate energy function $\mathcal{E}(\cdot)$: energies are sampled uniformly from $[-1, 1]$, and exactly $\frac{|S|}{10}$ elements share identical energy values, resulting in multiple optimal solutions. The GFlowNet agent is parameterized by an MLP with two hidden layers of 256 units and LeakyReLU activations. We generate 16 samples per training step and use the Adam optimizer (Kingma & Ba, 2014) to optimize the objectives. All the objectives are trained with 10,000 steps. The learning rate for α-DB and α-FL-DB is set to 0.001. For α-TB, we apply a learning rate of 0.001 to the MLP parameters and 0.1 to the learnable parameter Z, which represents the total amount of flows, for TB-based objectives, which directly follows Pan et al. (2023). The backward policy P_B is set to be uniform. In particular, we notice that the parameter ϵ in the ϵ-greedy sampling trick varies across different values and set sizes. In some cases, the sampling policy is even totally uniform ($\epsilon = 1$). Therefore, we adopt a unified approach for the setting of ϵ: we first start with $\epsilon = 1$, and linearly anneal ϵ to 0.05 during the training.

The evaluation metrics are 1) **Modes**: the number of unique samples with reward greater than a fixed threshold. Since Pan et al. (2023) does not mention the reward threshold to define modes, we set the threshold to be 0.25 for small sets, following Jang et al. (2024), and 700,000 for medium and large sets. This metric measures whether the policy effectively explore different high-reward regions. 2) **Top-1000 Reward**: the average reward of unique samples with the top-1000 highest reward, a metric to test the exploitation ability of the policy. 3) **Spearman Corr**: the spearman correlation between the generation probability and the reward in a held-out test set. of size 1000 generated by the initialized policy. We additionally report the **average reward of all samples generated in on-policy evaluation in Fig. 1.**

1. remove the corresponding lines 2.

Following Pan et al. (2023), the evaluation is conducted online and independent of training samples. Training samples are generated with the ϵ-greedy policy, while evaluation samples are generated only with the forward policy P_F. Models are trained with 10,000 steps, where the first 9,000 steps are stage 1. Throughout the experiments, we fix the initialization of model weights and use different random seeds for sample generation. All the results are averaged over 10 random seeds, which are integers from 0 to 9. For each of the α-GFN objectives, it takes about 10-20 minutes, 25-50 minutes and 30-60 minutes for small, medium and large sets to run an experiment on a cluster consisting of NVIDIA RTX3090, NVIDIA RTX4090 and NVIDIA ADA6000 GPUs.

Bit Sequence Generation Here, we adopt the non-autoregressive version of this task, which is proposed in Tiapkin et al. (2024) and is more difficult than the original version in Malkin et al. (2022). We implement α-DB, α-TB and α-SubTB based on the open-source code of Tiapkin et al. (2024), where setting $\alpha = 0.5$ yields the baselines, and follow the default hyperparameters and setup as Tiapkin et al. (2024). We apply an ϵ-noisy strategy that mixes the GFlowNet forward policy P_F with a uniform distribution U, assigning weights of $(1 - \epsilon)$ and ϵ (with $\epsilon = 0.001$), while using a uniform backward policy P_B. The mode set M and test set are constructed as in (Malkin et al., 2022). A Transformer model (Vaswani et al., 2017) with 3 hidden layers, 64-dimensional hidden states, and 8 attention heads (Tiapkin et al., 2024) is employed across all objectives. Training is performed using the Adam optimizer (Kingma & Ba, 2014) with batch size 16 and learning rate 2×10^{-3}. For TB-based objectives, the learnable parameter Z uses a learning rate of 10^{-3}. For SubTB(λ) objectives we set $\lambda = 1.9$. We adopt a gradient clipping norm of 20 to stabilize the training. All models are trained for 50,000 steps, and the reward function is augmented as $\tilde{R}(x) = R(x)^{\beta}$ with $\beta = 2$. To approximate the probability of generating a sample $x \in \mathcal{X}$, which is formulated as $P_\theta(x)$, we use the Monte Carlo estimation proposed in (Zhang et al., 2022):

$$P_\theta(x) = \mathbb{E}_{P_B(\mathfrak{t}|x)} \frac{P_F(\mathfrak{t})}{P_B(\mathfrak{t}|x)} \approx \frac{1}{N} \sum_{i=1}^{N} \frac{P_F(\mathfrak{t}^i)}{P_B(\mathfrak{t}^i|x)}$$

where $\mathfrak{t}^i \in \mathfrak{T}^{\text{flow}}$ is a trajectory generated by the model. We set $N = 10$, following Tiapkin et al. (2024).

We adopt the evaluation metrics in Tiapkin et al. (2024). To be specific, the evaluation metrics of this task are 1) **Modes**: the number of unique samples with Hamming distance less than 30 to the predefined set M, ranging from 0 to 60. 2) **Spearman Corr**: the Spearman correlation between the generation probability of samples and their reward in a predefined test set.

Following the practice in Set Generation, we separate training and evaluation samples. Training samples are generated with the ϵ-greedy policy, while evaluation samples are generated with only the forward policy P_F online. Models are trained with 50,000 steps, where the first 40,000 steps are stage 1. Throughout the experiments, we fix the initialization of model weights and use different random seeds for sample generation. All the results are averaged over 5 random seeds, which are integers from 0 to 4. Each of the setting (k, objective, α) takes about 1 day to run on a cluster consisting of NVIDIA RTX3090, NVIDIA RTX4090 and NVIDIA ADA6000 GPUs.

Molecule Generation The goal of this task is to generate binders of the sEH (soluble epoxide hydrolase) proteins. In this task, we implement α-DB, α-FL-DB, α-SubTB(λ), α-SubTB(λ) and α-TB based on the open-source code of Tiapkin et al. (2024) and Pan et al. (2023). We follow the default models and hyperparameters of Tiapkin et al. (2024). We utilize a Message Passing Neural Network (MPNN) with 10 convolution steps and train it via the Adam optimizer (Kingma & Ba, 2014) using a batch size of 4 and a learning rate of 5×10^{-4}. The reward is exponentiated with $\beta = 4$ to produce $\tilde{R}(x) = R(x)^{\beta}$ as augmentation. An ϵ-greedy mixture of the GFlowNet forward policy P_F and a uniform policy U is applied, i.e. $P = \epsilon U + (1 - \epsilon) P_F$ with $\epsilon = 0.05$ is used for sampling. For TB objectives, the learnable parameter Z is initialized at $\log Z = 30$ and is optimized with a learning rate of 5×10^{-4}. SubTB(λ)-based objectives are trained with $\lambda = 0.99$ as in (Tiapkin et al., 2024). We adopt a gradient clipping norm of 2 for TB to stabilize the training.

The evaluation metrics are 1) **Modes**: the number of unique molecules with reward greater than 7 and Tanimoto similarity less than 0.7; 2) **Top-k Reward,** $k \in \{10, 100, 1000\}$: the average reward of unique samples with the top-k highest reward. 3) **Top-k Similarity,** $k \in \{10, 100, 1000\}$: the average Tanimoto similarity of samples within Top-k Reward. 4) **Spearman Corr**: the Spearman correlation between the probability of sample generation and their reward in a predefined test set.

Following the practice in Set Generation, we separate training and evaluation samples. Training samples are generated with the ϵ-greedy policy, while evaluation samples are generated with only the forward policy P_F online. We follow the defaulted setting in Pan et al. (2023) but adopt a smaller total number of steps 50,000 due to computational budgets, where the first 40,000 steps are stage 1. Training spans 50,000 steps per objective, generating 200,000 molecules for training and 200,000 molecules for evaluation. We repeat each of the settings 4 times to obtain the results. Throughout the experiments, we fix the initialization of model. Since random seed generators for sampling are set to be timestamps in the original codebase, which causes trouble in reproduction of the results, we fix the random seed generators for training with seeds ranging from 0 to 4, which are 5 integers in total, to facilitate reproduction of the results. We follow the implementation in OrderedGNN (Song et al., 2023), making message passing computation reproducible. Each of the objective settings takes about 1 day to run on a cluster consisting of NVIDIA RTX3090, NVIDIA RTX4090 and NVIDIA ADA6000 GPUs, where we run 5 settings in parallel within a single GPU.

D.2 Additional Results

Length-controling side-effects. We show in Fig. 6 how α changes sample length. Because samples in Set Generation and Bit Sequence Generation have fixed lengths, we use Molecule Generation as the representative example.

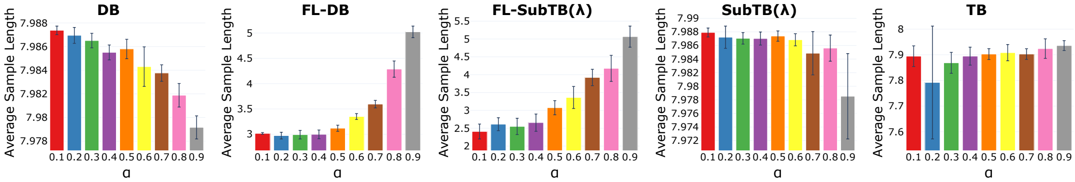
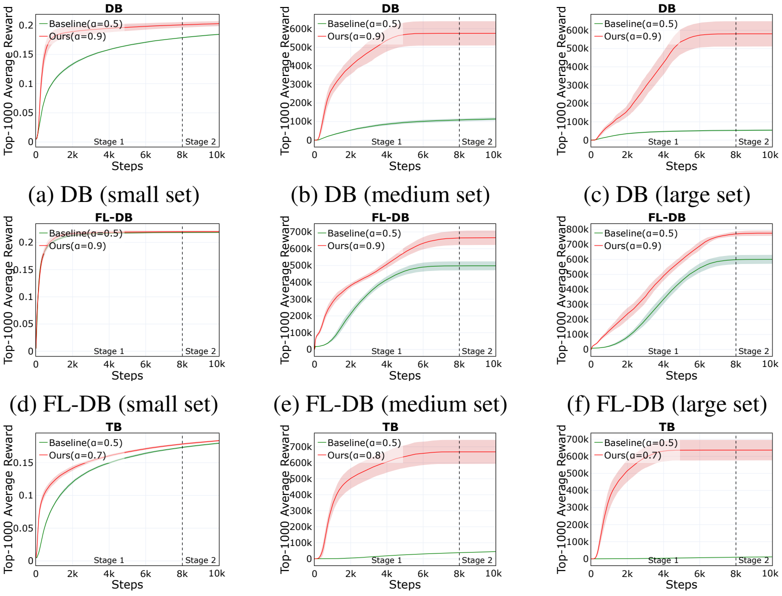


Figure 6: **Average Sample Length** vs α in Molecule Generation across different objectives.

Dynamics throughout scheduled training. We visualize how key metrics evolve across training steps under the scheduled scheme (Alg. 1). For Set Generation, see Figs. 7–11; for Bit Sequence Generation, see Figs. 12–14; and for Molecule Generation, see Figs. 15–20.

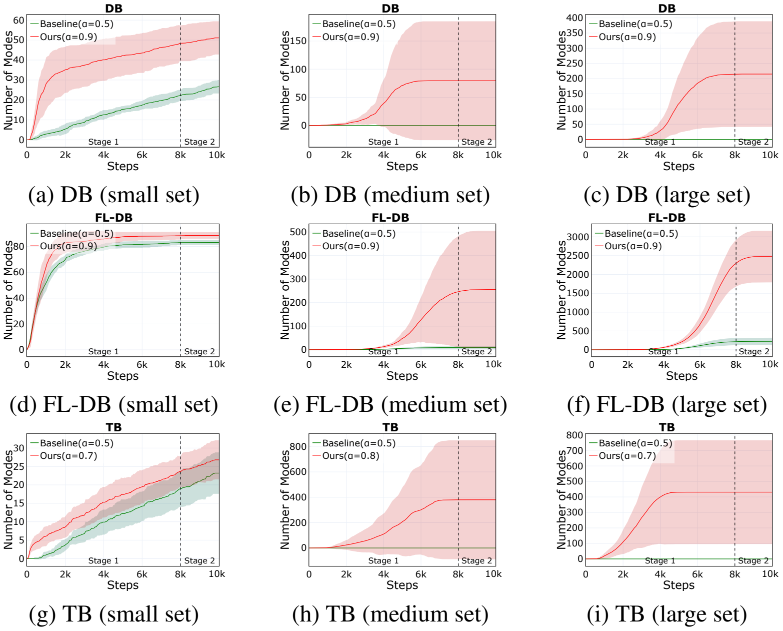


(a) DB (small set) (b) DB (medium set) (c) DB (large set)

(d) FL-DB (small set) (e) FL-DB (medium set) (f) FL-DB (large set)

(g) TB (small set) (h) TB (medium set) (i) TB (large set)

Figure 7: **Top-1000 Average Reward** vs Training Steps in **Set Generation** across different objectives and set sizes.

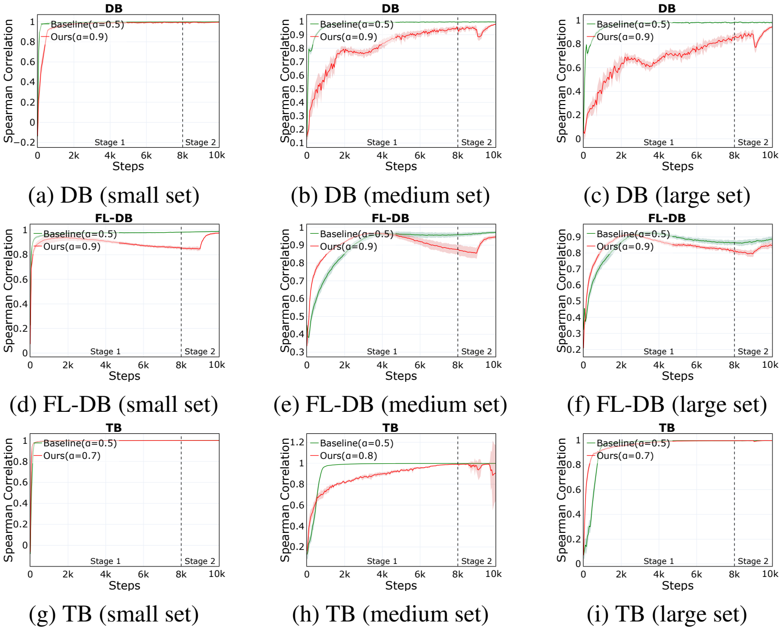


(a) DB (small set) (b) DB (medium set) (c) DB (large set)

(d) FL-DB (small set) (e) FL-DB (medium set) (f) FL-DB (large set)

(g) TB (small set) (h) TB (medium set) (i) TB (large set)

Figure 8: **Number of Modes** vs Training Steps in **Set Generation** across different objectives and set sizes.

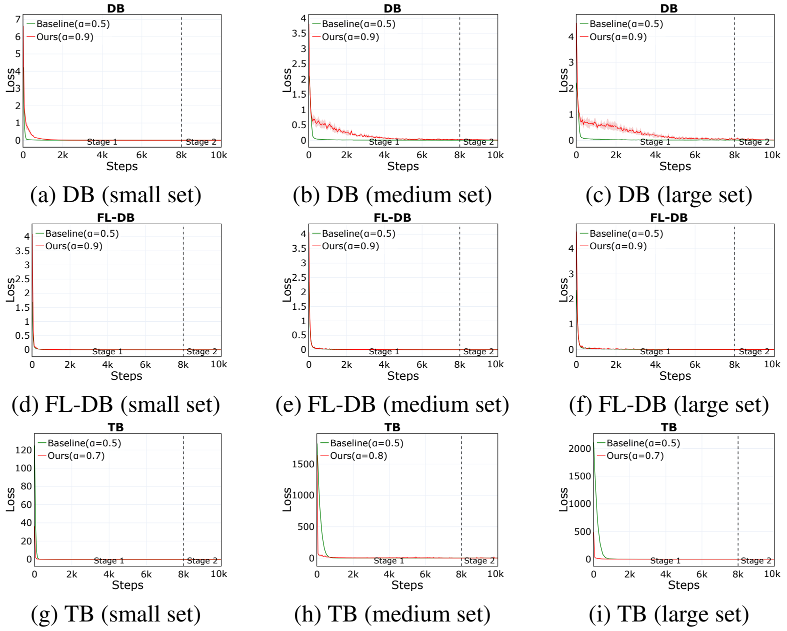


(a) DB (small set) (b) DB (medium set) (c) DB (large set)

(d) FL-DB (small set) (e) FL-DB (medium set) (f) FL-DB (large set)

(g) TB (small set) (h) TB (medium set) (i) TB (large set)

Figure 9: **Spearman Correlation** vs Training Steps in **Set Generation** across different objectives and set sizes.

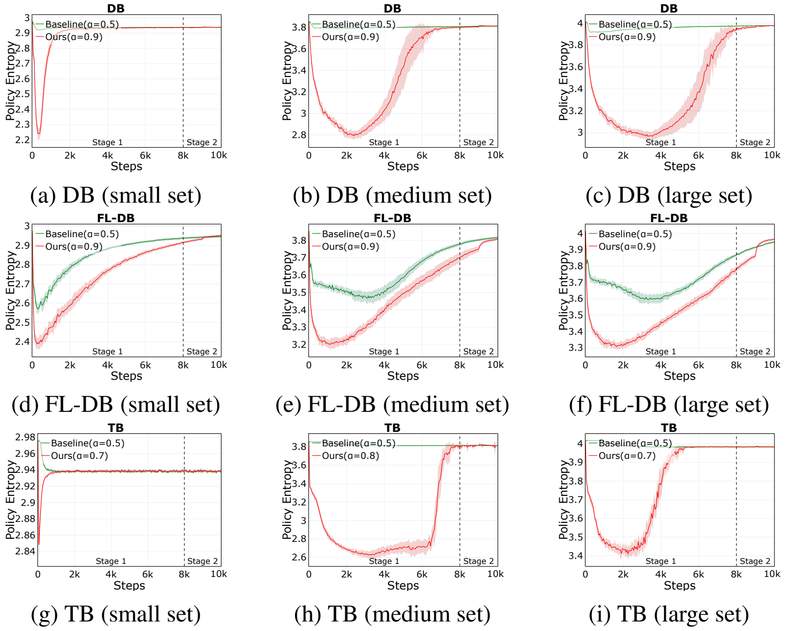


(a) DB (small set) (b) DB (medium set) (c) DB (large set)

(d) FL-DB (small set) (e) FL-DB (medium set) (f) FL-DB (large set)

(g) TB (small set) (h) TB (medium set) (i) TB (large set)

Figure 10: **Loss** vs Training Steps in **Set Generation** across different objectives and set sizes.



(a) DB (small set) (b) DB (medium set) (c) DB (large set)

(d) FL-DB (small set) (e) FL-DB (medium set) (f) FL-DB (large set)

(g) TB (small set) (h) TB (medium set) (i) TB (large set)

Figure 11: **Forward Policy Entropy** vs Training Steps in **Set Generation** across different objectives and set sizes.

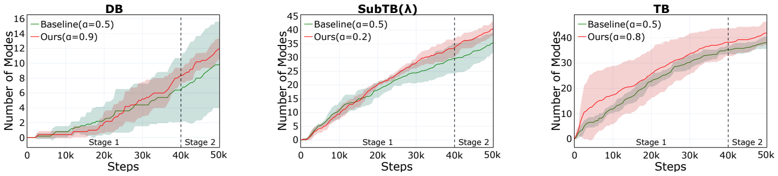


Figure 12: **Number of Modes** vs Training Steps in **Bit Sequence Generation** across different objectives.

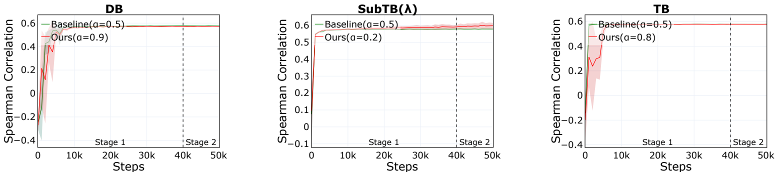


Figure 13: **Spearman Correlation** vs Training Steps in **Bit Sequence Generation** across different objectives.

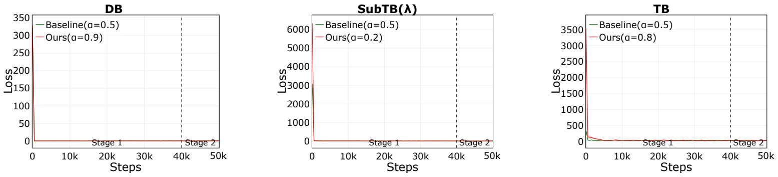


Figure 14: **Loss** vs Training Steps in **Bit Sequence Generation** across different objectives.

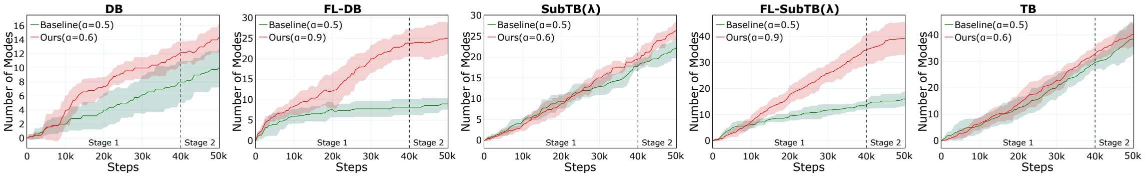


Figure 15: **Number of Modes** vs Training Steps in **Molecule Generation** across different objectives.

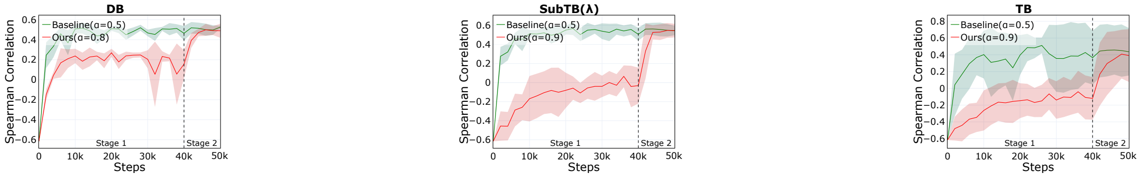


Figure 16: **Spearman Correlation** vs Training Steps in **Molecule Generation** across different objectives. FL-DB and FL-SubTB(λ) are omitted due to their biased target (Silva et al., 2025).

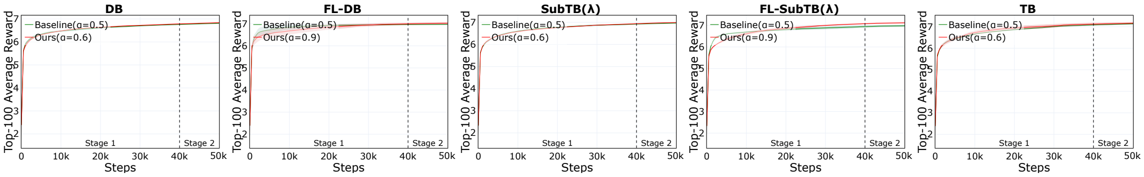


Figure 17: **Top-100 Average Reward** vs Training Steps in **Molecule Generation** across different objectives.

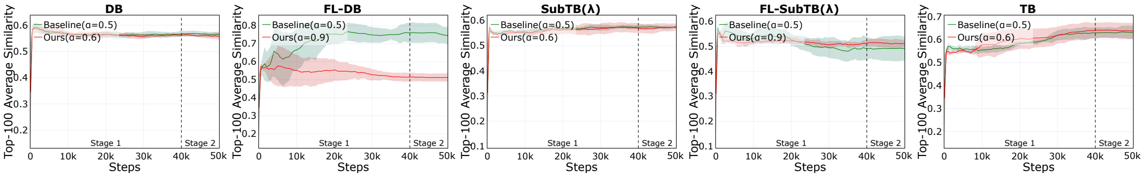


Figure 18: **Top-100 Average Similarity** vs Training Steps in **Molecule Generation** across different objectives.

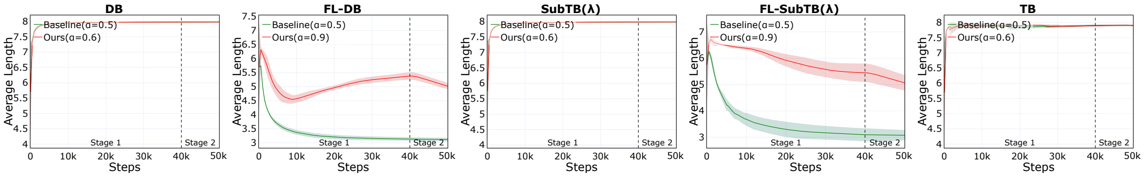


Figure 19: **Average Sample Length** vs Training Steps in **Molecule Generation** across different objectives.

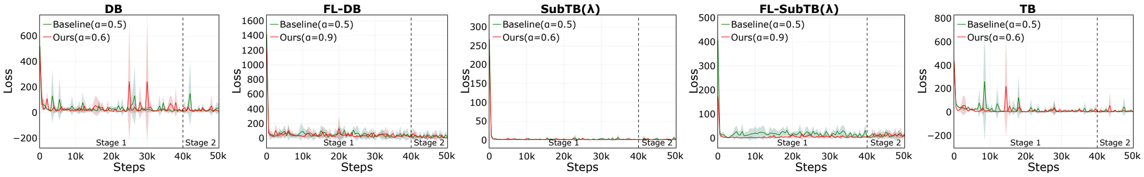


Figure 20: **Loss** vs Training Steps in **Molecule Generation** across different objectives.

E Ethics Statement

This paper presents α-GFN, a GFlowNet algorithm that improves the exploration–exploitation trade-off. By advancing GFlowNet theory and methodology, it directly supports scientific applications. We evaluate on widely used public benchmarks to ensure transparency and fairness, and our efficient configurations limit compute and environmental impact, underscoring our dedication to sustainable, reproducible research.

F Reproducibility Statement

We include implementation and dataset details in Sec. 5.1, full experimental settings in App. D.1, and the complete training procedure in Alg. 1 to ensure reproducibility. We provide all mathematical formulations, implementation details, and configuration settings required for reproducing our results in the paper and appendices.

G The Use of Large Language Models (LLMs)

LLMs are used solely to assist writing and style polishing. All fundamental research ideas, theoretical derivations, experimental setups, and algorithmic innovations are developed solely by the authors without any LLM assistance. The human researchers exclusively derive the mathematical formulations, establish the proofs, and obtain the experimental results. LLMs did not contribute to the fundamental conceptual development of α-GFN or to the core insights about exploration–exploitation trade-offs.