Fixing Truncation-Induced Mode Collapse in GFlowNets via Pruning Loss

Anonymous
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

***Abstract*—Generative Flow Networks (GFlowNets) aim to generate diverse molecular candidates by sampling proportionally to rewards through flow conservation constraints. However, they suffer from mode collapse, the very problem they were designed to solve. We identify that the root cause is forced terminal states arising from artificial trajectory truncation in vast state spaces. Unlike natural terminal states, forced terminals violate flow conservation boundary constraints, causing flow leakage that biases generation toward maximum-length trajectories and triggers mode collapse. We propose Pruning Loss, a novel training objective that enforces flow conservation at forced terminals by requiring both sink flow and total outflow to equal the reward. This dual constraint implicitly drives unnecessary action flows to zero while maintaining non-vanishing gradients for stable convergence. Our theoretical analysis demonstrates that Pruning Loss recovers proper flow conservation in truncated spaces while guaranteeing gradient persistence. Empirical evaluation on molecular generation tasks validates our theoretical predictions. On sparse-reward kinase protein targets, our method achieves substantial improvements over standard objectives. On dense-reward drug-likeness tasks, all methods perform comparably well, validating that flow leakage specifically limits performance in sparse reward landscapes where diverse exploration is critical. Our results establish that correcting boundary constraints at forced terminals is more fundamental than refining balance equations. This principle provides a new foundation for addressing mode collapse in GFlowNets.**

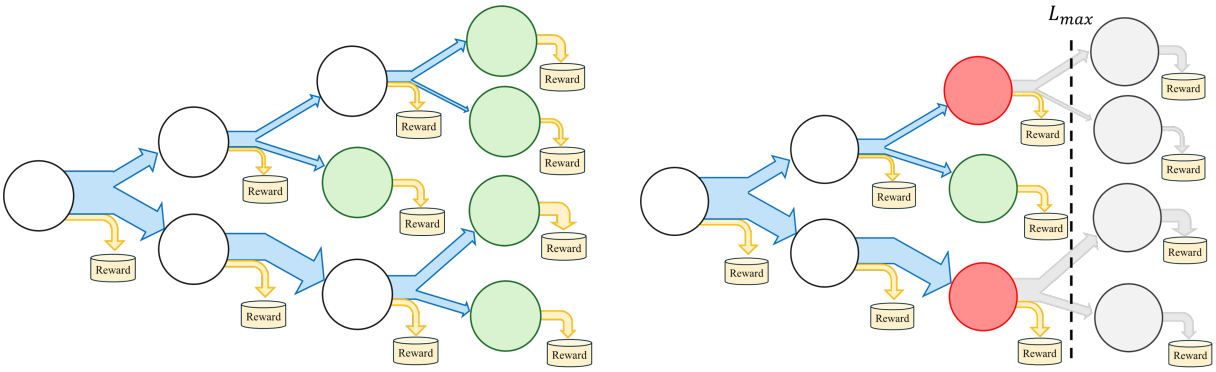
***Index Terms*—Generative Flow Networks, Mode Collapse, Molecular Generation, Flow Conservation, Trajectory Truncation**

I. Introduction

Mode collapse, where generative models repeatedly sample from a narrow subset of solutions, poses a fundamental challenge in scientific discovery applications. This issue becomes particularly critical in molecular generation, where the goal is not to find a single optimal molecule but to discover diverse candidates for downstream experimental validation [1], [2]. Since experimental hit rates against protein targets typically range only 0.01-0.14% [3], exploring diverse regions of chemical space is essential for discovering viable therapeutic candidates. A generative model that repeatedly produces duplicated molecules, even with high individual scores, fundamentally fails this exploration objective. Generative Flow Networks (GFlowNets) [1], [4] emerged to address this diversity challenge. Unlike reinforcement learning, which seeks a single optimal policy, GFlowNets learn to sample objects with probability proportional to their rewards. This is achieved by treating sequential generation as a flow network. The flow conservation principle ensures that the learned sampling distribution matches the target reward distribution, theoretically guaranteeing diverse exploration. This framework has been applied to drug design [1], biological sequence design [2], and combinatorial optimization [5], with the promise of eliminating mode collapse by design.

However, despite this theoretical promise, mode collapse persists in practical GFlowNet applications [6]. GFlowNets continue to suffer from the very diversity failures they were designed to prevent, revealing a critical gap between theory and practice. We identify trajectory length truncation as the root cause of this failure. Real-world search spaces are intractably large, with the space of drug-like molecules alone containing 10^{60} or more candidates [7]. To make computation feasible, practitioners universally impose a maximum trajectory length $L_{\max}$, truncating the state space to manageable regions. However, this truncation fundamentally violates the theoretical assumptions underlying GFlowNets. States at depth $L_{\max}$ are artificially forced to be terminal, even when actions remain structurally available. We term these forced terminal states and formally define them as states s_f where the depth limit forces termination but $\mathcal{A}(s_f) \neq \varnothing$. Unlike natural terminal states (Figure 1a), forced terminal states violate the boundary conditions of flow conservation. The learned policy assigns non-zero flow to structurally available but artificially prohibited actions, creating flow leakage (Figure 1b). This leakage biases generation toward maximum-length trajectories, as the policy learns to always reach $L_{\max}$ to maximize cumulative flow. Consequently, long trajectories dominate the generated distribution, causing systematic deviation from the target distribution and resulting in precisely the mode collapse.

Existing approaches fail to address the flow conservation violation at forced terminals. Some works have attempted reward shaping to discourage maximum-length trajectories through penalties or decay factors, but these methods distort the target reward distribution and require careful domain-specific tuning. Post-processing approaches like Local Search GFlowNets [8] can refine generated samples but do not address the training-time boundary violation. More fundamentally, even refined GFlowNet training objectives like Trajectory Balance (TB)



(a) State Space (b) Truncated State Space

Fig. 1. **Impact of trajectory length truncation on GFlowNet state space and flow structure.** Blue arrows indicate forward transitions, yellow arrows represent sink flows to terminal rewards, and gray elements denote unreachable states and flows due to truncation. (a) In the complete state space, all trajectories can reach natural terminal states (green) where sink flows properly direct to rewards. (b) With maximum trajectory length L_{max}, trajectories are forced to terminate prematurely at non-terminal states (red), creating artificial sink flows. States and transitions beyond L_{max} become unreachable (gray), resulting in incomplete state space coverage and distorted flow balance.

[9] and Detailed Balance (DB) [4] inherit the same flawed boundary assumptions under truncation.

Despite the ubiquity of state space truncation in practical GFlowNet applications, no prior work has formally characterized the flow leakage problem, proposed a theoretically grounded solution preserving flow conservation, or analyzed the gradient dynamics of potential solutions. Our work addresses this gap by providing both a rigorous problem formulation and a principled solution with theoretical guarantees. We introduce Pruning Loss, an auxiliary training objective that restores proper flow conservation at forced terminal states. The key insight is to enforce dual constraints requiring both sink flow and total outflow to equal the reward at these boundary states. This approach implicitly drives action flows to zero without direct suppression, thereby maintaining non-vanishing gradients throughout training. Critically, Pruning Loss operates orthogonally to the choice of balance equation, enabling integration with any GFlowNet training objective.

Our main contributions are as follows:

- We formally characterize flow leakage in truncated state spaces and establish that boundary condition violations at forced terminal states constitute the fundamental cause of mode collapse.
- We provide theoretical analysis proving that Pruning Loss satisfies modified flow conservation while guaranteeing gradient persistence. We show that the dual constraint structure implicitly eliminates flow leakage without the vanishing gradient problem that plagues direct suppression approaches.
- We validate our framework across tasks with varying reward densities. On sparse reward kinase targets, our method substantially outperforms all baselines. In contrast, on dense reward tasks, all methods achieve comparable performance, indicating that flow leakage constitutes the primary bottleneck in sparse regimes.

II. Background

This section provides a brief overview of GFlowNets and establishes notation for subsequent sections. For a comprehensive treatment, we refer to [4].

A. Problem Setup

State space. Let $\mathcal{S}$ be a finite state space with a distinguished initial state s_0. States are organized as a directed acyclic graph (DAG) with edges corresponding to actions.

Actions and transitions. For each state $s \in \mathcal{S}$, let $\mathcal{A}(s)$ denote the set of available actions. An action $a \in \mathcal{A}(s)$ deterministically transitions to a successor state $s' = T(s, a)$ where T is the transition function. We write $s \to s'$ when s' is reachable from s in one step.

Terminal states and rewards. A state s is terminal if no actions are available: $\mathcal{A}(s) = \varnothing$. Let $\mathcal{S}_T \subseteq \mathcal{S}$ denote the set of terminal states. Each terminal state $x \in \mathcal{S}_T$ has an associated non-negative reward $R(x) \geq 0$. The reward function $\mathcal{S}_T \to \mathbb{R}_+$ encodes domain-specific objectives. For molecular generation, rewards might represent predicted binding affinity to a target protein.

Trajectories. A trajectory $\tau = (s_0, a_1, s_1, \ldots, a_n, s_n)$ is a sequence of states and actions starting from s_0 and ending at a terminal state $s_n \in \mathcal{S}_T$. The trajectory length is n, the number of actions taken. Different trajectories may lead to the same terminal state, as the state space DAG allows multiple paths to the same destination. We denote the set of all trajectories leading to terminal state x as $\mathcal{T}(x)$.

B. Flow Networks

Forward policy. A forward policy $P_F(a|s)$ defines a probability distribution over actions at each state. It induces a distribution over complete trajectories $P_F(\tau)$.

Flow functions. The core concept in GFlowNets is the notion of flow, which comes in two forms. Edge flow, denoted

as $F(s, a)$ or equivalently $F(s' \to s)$, represents the flow through a specific action a from state s, or more generally the flow from a parent state s' to its child state s. State flow $F(s) = \sum_{a \in \mathcal{A}(s)} F(s, a)$ aggregates all outgoing flows from a given state, computed as the sum over all available actions at that state. These flow quantities provide an alternative representation of the generation process, and importantly, the forward policy can be recovered from flows through normalization:

$$P_F(a|s) = \frac{F(s, a)}{F(s)} = \frac{F(s, a)}{\sum_{a' \in \mathcal{A}(s)} F(s, a')} \tag{1}$$

This relationship establishes that the policy is simply the proportion of total outgoing flow allocated to each action.

C. Flow Conservation Principle

The fundamental constraint that enables GFlowNets to sample from the target distribution is flow conservation. This principle requires that for every non-terminal state, the total incoming flow from parent states must equal the total outgoing flow to child states. Mathematically, this constraint takes the form:

$$\sum_{s':s' \to s} F(s' \to s) = R(s) + \sum_{a \in \mathcal{A}(s)} F(s, a) \tag{2}$$

This general equation holds universally, where $R(s) = 0$ for non-terminal states and the action set $\mathcal{A}(s) = \varnothing$ for terminal states. These boundary constraints allow us to specialize the conservation equation for different types of states.

For non-terminal states ($s \in \mathcal{S} \setminus \mathcal{S}_T$), where generation continues and no reward has yet been assigned, the conservation equation simplifies to a pure flow balance condition:

$$\sum_{s':s' \to s} F(s' \to s) = \sum_{a \in \mathcal{A}(s)} F(s, a) \tag{3}$$

Here, all incoming flow must be distributed among outgoing edges to child states, with no flow being absorbed or created at the state itself.

Terminal states ($s \in \mathcal{S}_T$) require special treatment because they represent completed objects where generation stops. At these states, the flow conservation equation becomes:

$$\sum_{s':s' \to s} F(s' \to s) = R(s) \tag{4}$$

The terminal condition naturally satisfies this equation because the empty action set $\mathcal{A}(s) = \varnothing$ implies zero outgoing flow. All incoming flow is therefore absorbed by the state's reward $R(s)$, which acts as a sink that removes flow from the network.

When these flow conservation constraints are satisfied throughout the network, and the initial state flow equals the partition function $F(s_0) = Z$ where $Z = \sum_{x \in \mathcal{S}_T} R(x)$, the induced trajectory distribution provably matches the target reward distribution:

$$P_F(\tau) = \frac{R(s)}{Z} \tag{5}$$

where s is the terminal state reached by trajectory τ. This result establishes that proper flow conservation guarantees sampling proportional to rewards.

III. Method

A. Problem Formulation: Flow Leakage in Truncated Spaces

We now formalize the problem arising from state space truncation and develop our solution. Given a maximum trajectory length $L_{\max} \in \mathbb{N}$, the truncated state space consists of all states reachable within this depth limit:

$$\mathcal{S}_{L_{\max}} = \{s \in \mathcal{S} : \text{depth}(s) \leq L_{\max}\} \tag{6}$$

where $\text{depth}(s)$ denotes the length of the shortest path from the initial state s_0 to state s. In practice, exploration becomes restricted to this truncated space $\mathcal{S}_{L_{\max}}$, rendering states beyond depth $L_{\max}$ unreachable. This truncation is not merely a theoretical concern but a practical necessity. The full state space $\mathcal{S}$ is often infinite or astronomically large, computational resources fundamentally limit how long trajectories can extend, and domain-specific constraints may impose natural limits on object complexity, such as maximum molecular size in drug design.

This truncation creates a problematic new class of terminal states that violate the assumptions underlying GFlowNet theory. Natural terminal states $\mathcal{S}_n$ satisfy the condition $\mathcal{A}(s) = \varnothing$ by construction, as they represent completed objects where no further actions are structurally possible, so the zero outflow constraint is automatically met. However, forced terminal states $\mathcal{S}_f$ exist at depth $L_{\max}$ where actions remain structurally available but are artificially prohibited by the depth limit. At such a forced terminal state $s_f \in \mathcal{S}_f$, flow leakage occurs when the learned policy assigns non-zero flow to these prohibited actions:

$$\sum_{a \in \mathcal{A}(s_f)} F(s_f, a) > 0 \tag{7}$$

This leakage has a concrete interpretation. Forced terminals implicitly accumulate what we term "phantom rewards" from unexplored future states that lie beyond the truncation boundary. The flow conservation equation at these states becomes:

$$\sum_{s':s' \to s_f} F(s' \to s_f) = R(s_f) + \underbrace{\sum_{a \in \mathcal{A}(s_f)} F(s_f, a)}_{\text{phantom rewards}} \tag{8}$$

The incoming flow no longer equals the true reward $R(s_f)$ but instead includes additional flow that "should" continue forward but cannot due to truncation. This violates the terminal boundary condition that the theory relies upon.

The consequence is that the learned distribution systematically deviates from the target. Specifically, the marginal probability of reaching a forced terminal state s_f becomes inflated:

$$P_F(s_f) = \frac{\sum_{s' \to s_f} F(s' \to s_f)}{Z} > \frac{R(s_f)}{Z} \tag{9}$$

Because forced terminal states leak flow forward, the policy learns to preferentially reach depth $L_{\max}$ to accumulate maximum flow. This length-bias causes long trajectories to dominate the generated distribution, resulting in the mode collapse we observe empirically.

B. Pruning Loss via Indirect Constraint

The key challenge is to eliminate flow to prohibited actions at forced terminal states. Formally, we require:

$$F(s_f, a) = 0 \quad \forall a \in \mathcal{A}(s_f), \quad \forall s_f \in \mathcal{S}_f \tag{10}$$

A naive approach would directly penalize non-zero flows through a squared error loss:

$$\mathcal{L}_{\text{direct}} = \sum_{s_f \in \mathcal{S}_f} \sum_{a \in \mathcal{A}(s_f)} [F(s_f, a)]^2 \tag{11}$$

However, this direct constraint suffers from a critical gradient vanishing problem. As the flows approach zero, which is precisely our objective, the gradient of this loss also vanishes:

$$\frac{\partial \mathcal{L}_{\text{direct}}}{\partial \theta} = 2F(s_f, a) \cdot \frac{\partial F(s_f, a)}{\partial \theta} \to 0 \tag{12}$$

As flows approach zero, gradients vanish proportionally, preventing further optimization. This is analogous to the dying ReLU problem in neural networks [10], where neurons producing zero output lose gradient signal and cannot recover.

Indirect Constraint. Our approach avoids directly enforcing $F(s_f, a) = 0$. Instead, we impose two constraints that implicitly guarantee zero flow while maintaining non vanishing gradients.

First, the sink flow must equal the reward:

$$F(s_f) = R(s_f) \tag{13}$$

Second, the total outgoing flow, including sink flow, must also equal the reward:

$$\sum_{a \in \mathcal{A}^+(s_f)} F(s_f, a) = R(s_f) \tag{14}$$

where $\mathcal{A}^+(s_f)$ denotes the augmented action set including the sink action.

These dual constraints implicitly force action flows to zero. Since sink flow alone already equals the reward (Equation (13)) and total outflow must equal the reward (Equation (14)), no flow budget remains for other actions. Formally, if both Equations (13) and (14) are satisfied, then Equation (10) holds.

C. Pruning Loss Definition

We translate these two constraints into corresponding loss functions. The sink flow loss penalizes any deviation of the sink flow from the reward:

$$\mathcal{L}_{\text{sink}} = \sum_{s_f \in \mathcal{S}f} [F(s_f) - R(s_f)]^2 \tag{15}$$

Similarly, the total outflow loss penalizes deviation of the total outgoing flow from the reward:

$$\mathcal{L}_{\text{outflow}} = \sum_{s_f \in \mathcal{S}_f} \left[\sum_{a \in \mathcal{A}^+(s_f)} F(s_f, a) - R(s_f) \right]^2 \tag{16}$$

Combining these yields our Pruning Loss:

$$\mathcal{L}_{\text{pruning}} = \mathcal{L}_{\text{sink}} + \mathcal{L}_{\text{outflow}} \tag{17}$$

This combined objective can be integrated with any standard GFlowNet training loss to form the complete training objective:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{GFN}} + \lambda_{\text{prune}} \cdot \mathcal{L}_{\text{pruning}} \tag{18}$$

where $\mathcal{L}_{\text{GFN}}$ represents any existing GFlowNet loss function such as Trajectory Balance, Detailed Balance, or SubTrajectory Balance, and $\lambda_{\text{prune}} > 0$ is a hyperparameter controlling the strength of the pruning constraint. Importantly, this formulation means Pruning Loss operates orthogonally to the choice of base training objective, making it a general solution applicable to any GFlowNet variant.

IV. Experiments

A. Experimental Setup

Fragment-Based Molecular Generation. Following the fragment-based molecule design approach, molecular graphs are generated by sequentially attaching fragments from a predefined vocabulary of building blocks to atoms of partially constructed molecules. [1], [11], [12] This formulation naturally fits the GFlowNet framework. A state s represents a partially constructed molecule. The space is structured as a directed acyclic graph (DAG) $G = (\mathcal{S}, E)$ where edges $e \in E$ correspond to fragment attachment transitions $s \to s'$. At each state, the agent selects an attachment point (atom) in the current molecule and a fragment from the vocabulary to attached that point. The number of available actions varies between 100 and 2000 depending on the state, making the total state space size $|\mathcal{X}| \approx 10^{16}$. We set $L_{\max} = 15$ fragment attachments, creating a truncated state space where states at depth 15 become forced terminal states ($\mathcal{S}_f$).

Model Architecture. We parameterize the flow predictor F_θ using a Message Passing Neural Network (MPNN) [13] over the graph of molecular fragments. Each fragment is embedded using the MPNN, which iteratively aggregates information from neighboring fragments. And then two prediction heads, edge flow head and state flow head, edge flow head predicts $F(s \to s')$ based on node (fragment) representations. State flow head predicts $F(s)$ based on graph-level representations. Both heads are parameterized with multilayer perceptrons (MLPs). We denote the combined parameters as $\theta = (\theta_{\text{MPNN}}, \theta_{\text{edge}}, \theta_{\text{state}})$.

Training objective. We use Flow Matching (FM) loss [1] as the base objective:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{FM}} + \lambda_{\text{prune}} \cdot \mathcal{L}_{\text{pruning}} \tag{19}$$

where $\mathcal{L}_{\text{FM}}$ indicates standard Flow Matching loss. We set $\lambda_{\text{prune}} = 1.0$ as the default value for all main experiments. To assess the robustness of our method to this hyperparameter, we conduct additional experiments with $\lambda_{\text{prune}} \in \{0.1, 0.5, 1.0, 10.0\}$ on GSK3β, as detailed in Section IV-E.

Molecular Generation Tasks. We evaluate our method across four tasks with different reward landscapes. For kinase protein optimization, we target JNK3 and GSK3β [14], [15], where only 0.1% of molecules show meaningful biological activity, creating sparse reward landscapes that demand diverse

exploration. For drug-likeness optimization, we use Synthetic Accessibility (SA) [16] and Quantitative Estimate of Drug-likeness (QED) [17]. SA measures synthesis ease (0-1 scale, higher is better), while QED integrates multiple physicochemical properties for ADMET prediction (0-1 scale). Unlike kinase tasks, SA and QED represent dense reward landscapes where favorable properties are common, allowing us to test whether our method introduces biases when mode collapse is not the primary concern.

Evaluation Metrics. We assess generation quality through multiple complementary metrics. Following standard practice, we generate 1,000 molecules and analyze the top-100 by reward.

Uniqueness (Uni) measures the percentage of distinct molecules in the top-100 generated samples. This metric directly assesses mode collapse avoidance by quantifying whether the policy generates identical or nearly identical molecules. A Uni score of 100% indicates all top-100 molecules are distinct, demonstrating successful chemical space exploration. Lower percentages indicate mode collapse, where the generative policy produces redundant molecules with minor variations rather than exploring diverse chemical structures.

Score computes the mean reward of the top-100 molecules, measuring exploitation capability. For kinase targets, this reflects the ability to generate molecules with high predicted binding affinity; for SA/QED tasks, it measures favorable molecular properties. This metric quantifies how effectively the policy identifies and samples high-reward regions of chemical space.

Diversity (Div) calculates the average pairwise Tanimoto distance using Morgan fingerprints. This metric quantifies chemical diversity by measuring whether generated molecules explore different regions of chemical space or cluster around similar structures. Higher diversity values are crucial for lead optimization, as they provide broader structural options for medicinal chemistry and increase the probability of discovering molecules with complementary properties.

Harmonic Mean (HM) combines uniqueness, score, and diversity into a balanced overall metric. The harmonic mean penalizes methods excelling in only one dimension while neglecting others, requiring balanced performance across all three criteria. Higher HM values indicate superior balance between exploration (uniqueness and diversity) and exploitation (score), providing a robust quality metric for evaluating generative models in drug discovery applications.

B. Efficacy Across Sparse and Dense Reward Landscapes

Our theoretical analysis predicts that Pruning Loss provides the greatest benefit in sparse reward landscapes, where high-quality states are rare and mode collapse is most detrimental. To test this prediction, we evaluate our method across tasks with fundamentally different reward structures: kinase protein targeting (JNK3, GSK3β), where only 0.01-0.14% of molecules show activity, and drug-likeness optimization (SA, QED), where favorable properties are far more abundant. We compare against three standard GFlowNet training objectives: Flow Matching (FM), Trajectory Balance (TB), and Detailed Balance (DB).

TABLE I
COMPARATIVE PERFORMANCE ANALYSIS ACROSS OPTIMIZATION TASKS

Metric	Model	*JNK3*	*GSK3β*	*SA*	*QED*
HM	GFlowNet-TB	0.34	0.57	0.83	0.45
	GFlowNet-DB	0.41	0.53	<u>0.92</u>	0.86
	GFlowNet-FM	<u>0.53</u>	<u>0.65</u>	<u>0.92</u>	**0.92**
	Ours	**0.63**	**0.74**	**0.93**	**0.92**
score	GFlowNet-TB	0.17	0.39	0.66	0.22
	GFlowNet-DB	0.19	0.29	0.86	0.72
	GFlowNet-FM	<u>0.36</u>	<u>0.50</u>	**0.88**	**0.89**
	Ours	**0.52**	**0.58**	<u>0.87</u>	<u>0.88</u>
Uni (%)	GFlowNet-TB	57%	**100%**	**100%**	**100%**
	GFlowNet-DB	**100%**	**100%**	<u>98%</u>	**100%**
	GFlowNet-FM	70%	80%	<u>98%</u>	**100%**
	Ours	<u>99%</u>	<u>93%</u>	**100%**	**100%**
Div	GFlowNet-TB	<u>0.71</u>	0.60	0.93	<u>0.90</u>
	GFlowNet-DB	**0.84**	**0.85**	0.93	**0.91**
	GFlowNet-FM	0.68	0.74	<u>0.92</u>	0.88
	Ours	0.54	<u>0.80</u>	<u>0.92</u>	0.89

Bold: best / <u>Underline</u>: second best (best in each task/metric group)

Table I reveals a striking and counterintuitive pattern across these tasks. For the kinase targets, which represent the challenging sparse-reward regime, the performance hierarchy consistently contradicts what theory would predict. On JNK3, TB achieves only 0.34 harmonic mean, DB improves modestly to 0.41, FM reaches 0.53, and our method attains 0.63. This represents an 85% improvement over TB and a 19% gain over FM. The pattern continues on GSK3β, where TB scores 0.57, DB falls slightly to 0.53, FM achieves 0.65, and our method reaches 0.74. The fact that theoretically more sophisticated objectives like TB and DB consistently underperform the simpler FM baseline suggests something fundamental about how truncation affects different training objectives. Performance characteristics differ substantially for dense reward tasks. For SA optimization, the performance gaps nearly vanish. TB achieves 0.83, while DB, FM, and our method all cluster tightly around 0.92-0.93. The difference between the worst and best performers is only about 10%, compared to the 85% gap observed on JNK3. QED presents a more complex picture, with TB exhibiting anomalously poor performance at 0.45, while DB achieves 0.86, FM achieves 0.92, and our method also achieves 0.92, all showing strong results. The key observation is that in dense reward settings, most methods succeed reasonably well, and our improvements become marginal. This task-dependent behavior pattern directly confirms our theoretical prediction that boundary violations primarily harm performance in sparse reward landscapes.

Looking deeper into the individual metrics helps explain these performance differences. On JNK3, TB generates only 57% unique molecules with an average score of 0.17, demonstrating severe mode collapse where the model repeatedly samples duplicates with low rewards. DB takes the opposite extreme, achieving 100% uniqueness but with a score of

TABLE II
ROBUSTNESS AND PERFORMANCE COMPARISON ACROSS MAXIMUM TRAJECTORY LENGTHS ($\mathbf{L_{max}}$) ON GSK3β.

Metric	Model	$\mathbf{L_{max} = 10}$	$\mathbf{L_{max} = 15}$	$\mathbf{L_{max} = 20}$	$\mathbf{\Delta_{max}}$
HM	GFlowNet-FM	0.59	0.65	0.69	0.10
	Ours	**0.68**	**0.74**	**0.72**	**0.06**
score	GFlowNet-FM	**0.70**	0.50	0.49	0.21
	Ours	0.58	**0.57**	**0.56**	**0.02**
Uni (%)	GFlowNet-FM	47%	80%	**91%**	44%
	Ours	**76%**	**93%**	86%	**17%**
Div	GFlowNet-FM	0.67	0.74	0.78	0.11
	Ours	**0.72**	**0.80**	**0.82**	**0.10**

Bold: Best performance per $L_{\max}$ setting (in $L_{\max}$ columns) **or** lowest $\mathbf{\Delta}_{\max}$ (in $\mathbf{\Delta}_{\max}$ column).

only 0.19, suggesting the policy explores diverse regions of chemical space but fails to concentrate on high-reward areas. FM strikes a better balance with 70% uniqueness and a 0.36 score, though it still exhibits substantial mode collapse. Our method achieves 99% uniqueness while maintaining a 0.52 score, effectively balancing the dual objectives of exploration and exploitation that sparse reward optimization demands. These behavioral differences stem from how each training objective propagates boundary errors through the network. TB enforces trajectory-level consistency, requiring the product of forward probabilities along entire paths to match rewards. When forced terminals leak flow, this global constraint causes the entire trajectory distribution to skew toward maximum-length paths.

In sparse reward landscapes like JNK3, where only about 0.1% of molecules exhibit meaningful activity, this becomes catastrophic. The policy fixates on high-probability paths that reach forced terminals rather than exploring broadly for rare high-reward states. DB imposes strict local balance at every state transition, which increases diversity because the policy cannot collapse to narrow trajectories without violating these constraints. However, the rigid coupling between states prevents efficient backward propagation of reward information from rare high-quality molecules. FM uses relatively loose flow matching constraints, enabling intermediate performance between TB's collapse and DB's inefficient exploration.

Our method eliminates these violations by enforcing that forced terminal states satisfy both correct inflow, which should equal the reward, and zero outflow, as true terminal states naturally possess. This enables the policy to genuinely explore chemical space without the length-bias induced by flow leakage. On JNK3, this yields 99% uniqueness with a 0.52 score, avoiding both TB's collapse to duplicates and DB's scattered exploration. On GSK3β, we achieve 93% uniqueness with a 0.58 score, maintaining the balanced tradeoff that sparse reward optimization demands. The fundamental insight is that fixing boundary constraints matters more than refining balance equations. The practical implications are significant for drug discovery. Difficult protein targets like JNK3 and GSK3β represent precisely the sparse reward landscapes where existing drugs are lacking and discovery efforts are most urgently needed. Our substantial improvements on these targets demonstrate that Pruning Loss could accelerate therapeutic candidate discovery. The ability to generate highly unique molecules while maintaining competitive scores means experimental validation can examine diverse candidates rather than wasting resources on near-duplicates. Meanwhile, our preserved performance on SA and QED, where we remain within 1% of the best baselines, ensures universal applicability across molecular design objectives. These results establish a design principle for GFlowNets operating in truncated spaces: boundaries before balance. Rather than adopting more theoretically sophisticated variants, researchers should prioritize proper handling of forced terminal states. This also explains why TB emerged as the practical standard in molecular generation despite alternatives like DB. TB is more robust than DB in truncated spaces, though still limited by unaddressed boundary violations that our method now resolves.

C. Scalability to Longer Search Trajectories

A critical test of our hypothesis examines whether Pruning Loss maintains stable performance as trajectory lengths increase. If flow leakage at forced terminal states causes mode collapse, baseline GFlowNets should degrade as $L_{\max}$ grows, with deeper truncation causing more severe flow conservation violations. Conversely, our method should maintain stable performance by properly handling forced terminal states regardless of depth.

Table II confirms this prediction through both absolute performance and stability metrics. We quantify robustness across trajectory lengths using the maximum deviation $\Delta_{\max} = \max_{L \in \mathcal{L}}\{\text{metric}(L)\} - \min_{L \in \mathcal{L}}\{\text{metric}(L)\}$ where $\mathcal{L} = \{10, 15, 20\}$. A smaller $\Delta_{\max}$ indicates greater robustness to changes in trajectory length.

Baseline GFlowNet-FM exhibits severe instability with $\Delta_{\max} = 0.21$ for score and 44% for uniqueness, indicating a fundamental quality-diversity tradeoff. At $L_{\max} = 10$, the baseline achieves a score of 0.7 with only 47% unique molecules, showing severe mode collapse. Increasing to $L_{\max} = 20$ improves uniqueness to 91% but reduces the score to 0.49, contributing to the high $\Delta_{\max}$. This inverse relationship reveals the fundamental failure mode. As the state space expands with longer trajectories, flow leakage intensifies and impairs the model's ability to identify high-

reward regions. The policy sacrifices quality for superficial diversity, generating unique but low-value molecules.

In contrast, our method demonstrates remarkable stability with $\Delta_{\max} = 0.02$ for score compared to the baseline's 0.21, and 17% for uniqueness. At $L_{\max} = 10$, we achieve 76% uniqueness with a score of 0.58, surpassing baseline uniqueness by 62%. As $L_{\max}$ increases to 20, our score remains virtually unchanged at 0.56, while maintaining high uniqueness of 86%. The harmonic mean metric shows similar stability with $\Delta_{\max} = 0.06$ versus the baseline's 0.10, peaking at 0.74 for $L_{\max} = 15$. These results validate our theoretical analysis. The low $\Delta_{\max}$ values across all metrics confirm that Pruning Loss enforces proper flow conservation at forced terminal states, enabling stable exploration of deeper state spaces. This robustness has significant practical implications. Our method enables searching complex molecular spaces requiring long synthesis pathways where baseline GFlowNets would exhibit unstable performance, as quantified by their high $\Delta_{\max}$ values.

D. Robustness Against Mode Collapse During Training

To empirically validate that Pruning Loss addresses trajectory truncation-induced mode collapse, we track the number of unique molecules generated throughout training on the GSK3B inhibitor design task. We set the maximum trajectory length to $L_{\max} = 15$, a typical constraint for computational efficiency in molecular generation. At each iteration, we sample 1,000 molecules from the learned policy and count distinct structures, providing a direct measure of whether the model explores diverse chemical space or collapses to repetitive samples. Figure 2 reveals a striking difference between baseline Flow Matching (FM) and our method (FM+Pruning). The baseline exhibits catastrophic mode collapse that starting with approximately 900 unique molecules at iteration 2,000, diversity falls to as low as 150 by iteration 12,500. This collapse exhibits non-monotonic behavior with severe fluctuations, indicating training instability where the policy repeatedly loses previously discovered modes and converges to narrow solution subsets. By iteration 30,000, the baseline generates fewer than 200 unique molecules from 1,000 samples, demonstrating that 75% of generated candidates are duplicates. In contrast, FM+Pruning maintains stable diversity near 1,000 unique molecules throughout all 30,000 iterations. The model consistently generates almost entirely unique samples with over 90% uniqueness, showing only minor fluctuations around 900-1,000 modes. This stability demonstrates that Pruning Loss successfully prevents the flow leakage mechanism we identified: by enforcing proper flow conservation at forced terminal states, the model avoids the length-bias that drives trajectories toward $L_{\max}$ and causes mode collapse.

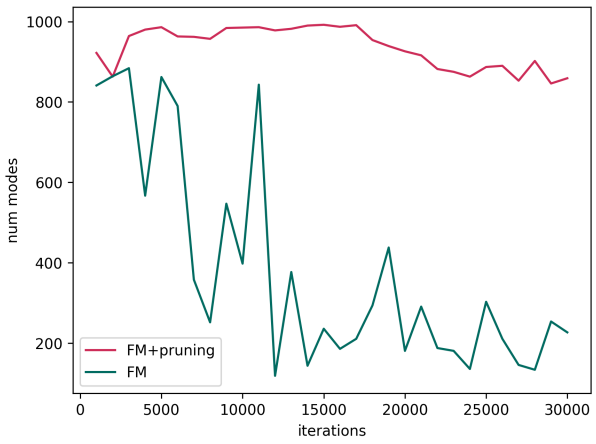


Fig. 2. Evolution of molecular diversity during training. Number of unique molecules (modes) among 1,000 generated samples at each iteration. FM (green) exhibits severe mode collapse, with diversity dropping from 900 to 150 unique molecules. FM+Pruning (red) maintains stable diversity near 1,000 throughout training, demonstrating robustness against trajectory truncation-induced mode collapse.

These results directly confirm our theoretical diagnosis. Without Pruning Loss, trajectory truncation creates forced terminal states that leak flow, biasing the policy toward maximum-length trajectories and causing the diversity collapse predicted by our analysis. With Pruning Loss, flow conservation is restored even in truncated spaces, enabling the model to fulfill GFlowNet's original promise: generating diverse candidates proportional to their rewards. Our method produces 5-6 times more unique molecules than the baseline, transforming GFlowNets from mode-collapsed generators into truly diverse exploration tools for molecular discovery.

E. Sensitivity to Pruning Loss Weight

To assess the robustness of our method, we investigate the sensitivity of the pruning loss weight λ_{prune}, which controls the strength of the pruning constraint. Table III presents the results on GSK3β in different values of λ_{prune} ranging from 0.1 to 10.0. The results demonstrate that our method maintains strong performance across a wide range of λ_{prune} values, with harmonic mean scores varying between 0.66 and 0.74. The optimal setting of λ_{prune} = 1.0 achieves the best overall balance, with 0.74 harmonic mean, 93% uniqueness, 0.58 score, and 0.80 diversity. Notably, even at the extremes of this range, performance remains competitive. At the lower end with λ_{prune} = 0.1, the method achieves the highest uniqueness of 96% but somewhat lower score of 0.49, suggesting insufficient boundary enforcement allows more exploration at the cost of reward optimization. At the higher end with λ_{prune} = 10.0, score remains strong at 0.56 but uniqueness drops to 80%, indicating that over-emphasizing boundary constraints may

TABLE III
SENSITIVITY ANALYSIS OF PRUNING LOSS WEIGHT λ_{PRUNE} ON GSK3β.

λ_{prune}	HM	Unique(%)	Score	Diversity
0.1	0.68	**96%**	0.49	0.75
0.5	0.66	89%	0.49	0.10
1.0	**0.74**	93%	**0.58**	**0.80**
10.0	0.69	80%	0.56	0.77

Bold: Best performance across all λ_{prune} values.

slightly restrict exploration. The relatively flat performance curve across this order-of-magnitude range indicates that the method is not overly sensitive to the precise choice of λ_{prune}. This robustness is practically valuable, as it reduces the burden of extensive hyperparameter tuning when applying the method to new tasks. The consistent improvements over baselines across all tested values confirm that the core mechanism of enforcing proper boundary conditions, rather than the specific weighting, drives the performance gains.

V. Conclusion

In this paper, we identify and resolve a fundamental obstacle to deploying GFlowNets for scientific discovery. Mode collapse induced by trajectory length truncation prevents effective deployment in real applications. We establish that forced terminal states, where computational constraints necessitate termination despite available actions, violate the boundary conditions underlying GFlowNet theory. This violation creates flow leakage that biases generation toward maximum-length trajectories. The boundary violation affects all existing training objectives, explaining why theoretically refined variants such as Trajectory Balance and Detailed Balance fail in practice, often underperforming simpler baselines on sparse reward tasks. Our proposed Pruning Loss addresses this problem at its source by enforcing proper flow conservation at forced terminals through a dual constraint. This approach implicitly eliminates leakage while maintaining gradient persistence. Experimental results across molecular generation tasks validate our theoretical predictions. Our method maintains stable training dynamics even in scenarios where baseline approaches collapse. It scales robustly to longer trajectories without introducing artificial tradeoffs between quality and diversity. Furthermore, it achieves substantial improvements in sparse reward landscapes, which is particularly important for therapeutic discovery that demands extensive exploration. These results establish a fundamental design principle for truncated GFlowNets. Correcting boundary constraints takes precedence over refining balance equations. Our work provides practitioners with a principled solution applicable to any domain requiring trajectory truncation due to computational constraints. Beyond molecular generation, our framework extends naturally to materials design, biological sequence engineering, and combinatorial optimization, where generating diverse high-quality candidates remains essential for experimental validation and real-world impact.

Acknowledgment

Anonymous

References

[1] E. Bengio, M. Jain, M. Korablyov, D. Precup, and Y. Bengio, "Flow network based generative models for non-iterative diverse candidate generation," *Advances in neural information processing systems*, vol. 34, pp. 27 381–27 394, 2021.

[2] M. Jain, E. Bengio, A. Hernandez-Garcia, J. Rector-Brooks, B. F. Dossou, C. A. Ekbote, J. Fu, T. Zhang, M. Kilgour, D. Zhang *et al.*, "Biological sequence design with gflownets," in *International Conference on Machine Learning*. PMLR, 2022, pp. 9786–9801.

[3] T. Zhu, S. Cao, P.-C. Su, R. Patel, D. Shah, H. B. Chokshi, R. Szukala, M. E. Johnson, and K. E. Hevener, "Hit identification and optimization in virtual screening: Practical recommendations based on a critical literature analysis: Miniperspective," *Journal of medicinal chemistry*, vol. 56, no. 17, pp. 6560–6572, 2013.

[4] Y. Bengio, S. Lahlou, T. Deleu, E. J. Hu, M. Tiwari, and E. Bengio, "Gflownet foundations," *Journal of Machine Learning Research*, vol. 24, no. 210, pp. 1–55, 2023.

[5] D. Zhang, H. Dai, N. Malkin, A. C. Courville, Y. Bengio, and L. Pan, "Let the flows tell: Solving graph combinatorial problems with gflownets," *Advances in neural information processing systems*, vol. 36, pp. 11 952–11 969, 2023.

[6] M. W. Shen, E. Bengio, E. Hajiramezanali, A. Loukas, K. Cho, and T. Biancalani, "Towards understanding and improving gflownet training," in *International conference on machine learning*. PMLR, 2023, pp. 30 956–30 975.

[7] C. A. Lipinski, F. Lombardo, B. W. Dominy, and P. J. Feeney, "Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings," *Advanced drug delivery reviews*, vol. 23, no. 1-3, pp. 3–25, 1997.

[8] M. Kim, T. Yun, E. Bengio, D. Zhang, Y. Bengio, S. Ahn, and J. Park, "Local search gflownets," *International Conference on Learning Representations (ICLR)*, 2024.

[9] N. Malkin, M. Jain, E. Bengio, C. Sun, and Y. Bengio, "Trajectory balance: Improved credit assignment in gflownets," *Advances in Neural Information Processing Systems*, vol. 35, pp. 5955–5967, 2022.

[10] L. Lu, Y. Shin, Y. Su, and G. E. Karniadakis, "Dying relu and initialization: Theory and numerical examples," *arXiv preprint arXiv:1903.06733*, 2019.

[11] W. Jin, R. Barzilay, and T. Jaakkola, "Junction tree variational autoencoder for molecular graph generation," in *International conference on machine learning*. PMLR, 2018, pp. 2323–2332.

[12] Y. Xie, C. Shi, H. Zhou, Y. Yang, W. Zhang, Y. Yu, and L. Li, "Mars: Markov molecular sampling for multi-objective drug discovery," *arXiv preprint arXiv:2103.10432*, 2021.

[13] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *International conference on machine learning*. Pmlr, 2017, pp. 1263–1272.

[14] Y. Li, L. Zhang, and Z. Liu, "Multi-objective de novo drug design with conditional graph generative model," *Journal of cheminformatics*, vol. 10, no. 1, p. 33, 2018.

[15] W. Jin, R. Barzilay, and T. Jaakkola, "Multi-objective molecule generation using interpretable substructures," in *International conference on machine learning*. PMLR, 2020, pp. 4849–4859.

[16] P. Ertl and A. Schuffenhauer, "Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions," *Journal of cheminformatics*, vol. 1, no. 1, p. 8, 2009.

[17] G. R. Bickerton, G. V. Paolini, J. Besnard, S. Muresan, and A. L. Hopkins, "Quantifying the chemical beauty of drugs," *Nature chemistry*, vol. 4, no. 2, pp. 90–98, 2012.