A Theory of Non-acyclic Generative Flow Networks

Leo Brunswic[1,2], Yinchuan Li[3*], Yushun Xu[1], Yijun Feng [1], Shangling Jui[1], Lizhuang Ma[2]

[1] Huawei Shanghai Research Center
[2] Shanghai Jiaotong University
[3] Huawei Noah's Ark Lab, Beijing, China
{leo.maxime.brunswic, liyinchuan, jui.shangling}@huawei.com, {xuyushun1, fengyijun1}@hisilicon.com,
ma-lz@cs.sjtu.edu.cn

Abstract

GFlowNets is a novel flow-based method for learning a stochastic policy to generate objects via a sequence of actions and with probability proportional to a given positive reward. We contribute to relaxing hypotheses limiting the application range of GFlowNets, in particular: acyclicity (or lack thereof). To this end, we extend the theory of GFlowNets on measurable spaces which includes continuous state spaces without cycle restrictions, and provide a generalization of cycles in this generalized context. We show that losses used so far push flows to get stuck into cycles and we define a family of losses solving this issue. Experiments on graphs and continuous tasks validate those principles.

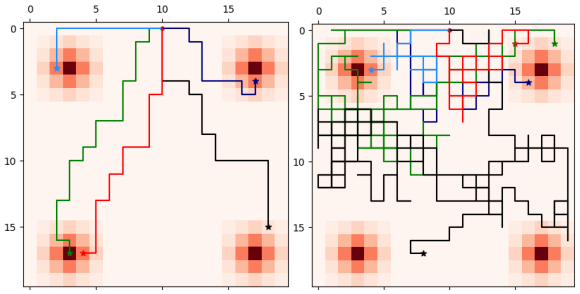


Figure 1: Paths generated on a grid using: non-acyclic loss (left, ours); FM loss (right, Bengio et al. (2021a)) .

Introduction

Bengio et al. (Bengio et al. 2021a,b; Zhang et al. 2022; Madan et al. 2022; Malkin et al. 2022) introduced GFlowNets (GFN): a new method for diverse candidate generation. The objective of GFN is to sample states of a Directed Acyclic Graph (DAG) proportionally to some reward function (see Section for quick introduction). In this way, GFN may be compared to MCMC (Brooks et al. 2011) and distributional reinforcement learning methods (Bellemare, Dabney, and Munos 2017). Experiments conducted by Bengio et al. on challenging tasks compared GFN favorably to MCMC-based methods. GFlowNets has also been successfully used in GNN (Li et al. 2023a,b), domain adaptation (Zhu et al. 2023), optimization (Zhang et al. 2023a), large language model training (Li et al. 2023c), offline data (Wang et al. 2023) and other fields.

Restricting GFN to DAG was initially motivated by specific applications in which solutions are built by applying sequences of constructors such as in the case of molecule generation. This seems however too restrictive: both GFN and MCMC methods sample an unnormalized distribution, it is tempting to try using GFN as a replacement for MCMC altogether (raytracing (Veach and Guibas 1997) is a typical industrial application of MCMC) leading to the need for a generalization of GFN to continuous state spaces. Moreover, some tasks have intrinsic symmetries that are better modeled by a non-acyclic graph, say a Cayley graph like the one of Rubick's cube. Worse, state space and transitions may be constrained in such a way that cycles are unavoidable: one may think of a game in which the adversary may force to return to a previous state. Recent work (Li et al. 2023d; Lahlou et al. 2023) made attempts at the former but the latter is still unaddressed.

The main aim of this paper is to clarify the issues brought up during the training of generative flows on spaces with cycles. In particular, we study their effects on gradient descent minimizations of the so-called Flow-Matching, Detailed Balance, and Trajectory Balance losses introduced in the previous aforementioned works to train GFN.

Considering the level of generalization of the present work is a few steps above that of the initial GFN framework, we use the term "Generative Flows" so that GFN correspond to the special case of Generative Flows on DAG.

Main Contributions & Outline of the Paper 1) We describe in section a property of losses on graphs leading flows to be trapped into loops and define a notion of stability in order to control this behavior which is detrimental to training and increases inference cost. 2) Section describes the elementary measure theoretical generalization of GFN slightly differently from that of Lahlou et al. (2023) lifting the built-in acyclicity constraint. The mathematical framework is enriched in particular with the definition of a suitable generalization of cycles. 3) In section , we reformulate Flow Matching, Detailed Balance, and Trajectory Balance losses as variations of f-divergences and show that such losses are unstable in the sense we introduced. We propose a family of

*Corresponding Author: Yinchuan Li

stable losses and regularizations to increase stability. 4) We demonstrate in section the effect of stability by solving toy problems on Hypergrids, Cayley graphs (GFN), and a continuous task from Gym (Brockman et al. 2016; Towers et al. 2023) (CFlowNets). Appendix A and B respectively provide extra theoretical results and proofs.

Background: Generative Flows on DAG

Following Bengio et al. (2021a) and Bengio et al. (2021b), let G be a *directed acyclic graph* (DAG) given by its (countable) *state set* $S := S^* \cup \{s_0, s_f\}$ and its *edges* set E[1] with source and sink states s_0 and s_f [2] For the sake of simplicity, we assume G is not a multigraph. An *edgeflow* on G is an assignment F of a non-negative value on each edge, $\forall s \to s', F(s \to s') \geq 0$.

Definition 1. *A flow is an edgeflow F satisfying the so-called flow-matching constraint (1). Given a reward $R : S^* \to \mathbb{R}_+$, an edgeflow is a R-edgeflow if it satisfies the reward constraint (2). A R-flow is an edgeflow satisfying both constraints, as the following*

$$\forall s \in S^*, \qquad \underbrace{\sum_{s' \to s} F(s' \to s)}_{\textit{Ingoing Flow}} = \underbrace{\sum_{s \to s'} F(s \to s')}_{\textit{Outgoing Flow}} \tag{1}$$

$$\forall s \in S^*, \qquad F(s \to s_f) = R(s). \tag{2}$$

Any edgeflow on such a finite graph G induces a Markov chain $(s_0, s_1, s_2, \cdots, s_f)$ starting at the source s_0, stopping at the sink s_f and at each time t such that $s_t \neq s_f$:

$$\mathbb{P}(s_{t+1} = s | s_t) = \frac{F(s_t \to s)}{\sum_{s_t \to s'} F(s_t \to s')} \tag{3}$$

whenever this formula makes sense. This Markov chain samples a state s at time τ if $s_\tau = s$ and $s_{\tau+1} = s_f$, the time τ is the *sampling time*. The fundamental result of Bengio et al. (2021a) is that equations (1)-(3) imply that at the sampling time, we have s_τ is distributed proportionally to the reward.

One trains a parameterized model of edgeflows to satisfy the flow-matching constraint while the reward constraint is enforced by the implementation. To train such a model via gradient descent, one thus defines a loss $\mathcal{L}$ that is minimized if and only if the flow-matching constraint (1) is satisfied. So far, three families of losses have been proposed: Flow-Matching loss (FM) (Bengio et al. 2021a), Detailed Balance loss (DB) (Bengio et al. 2021b) and Trajectory Balance loss (TB) (Malkin et al. 2022), respectively as the following:

$$\mathcal{L}_{\text{FM}}(F^\theta) = \mathbb{E} \sum_{t=1}^{\tau} \log^2 \frac{\sum_{s_t \to s'} F^\theta(s_t \to s')}{\sum_{s' \to s_t} F^\theta(s' \to s_t)}, \tag{4}$$

$$\mathcal{L}_{\text{DB}}(F_f^\theta, F_b^\theta) = \mathbb{E} \sum_{t=0}^{\tau-1} \log^2 \frac{F_{\text{out}}^{\theta,f}(s_t) \pi_f(s_t \to s_{t+1})}{F_{\text{out}}^{\theta,f}(s_{t+1}) \pi_b(s_t \to s_{t+1})}, \tag{5}$$

$$\mathcal{L}_{\text{TB}}(F_f^\theta, F_b^\theta) = \mathbb{E} \log^2 \frac{F_{\text{out}}^{\theta,f}(s_0) \prod_{t=0}^{\tau} \pi_f(s_t \to s_{t+1})}{R(s_\tau) \prod_{t=0}^{\tau-1} \pi_b(s_t \to s_{t+1})}. \tag{6}$$

[1] Our formulation differs slightly from that of (Bengio et al. 2021a) in which 'edges' are referred to as 'actions'.

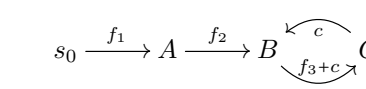
[2] A source is a state having no ongoing edge, a sink is a state having no outgoing edge.

The expectations above are taken over a distribution of paths $s_0 \to s_1 \to \cdots \to s_\tau \to s_f$ and we used $\pi_f(s \to s') := \frac{F_f(s_t \to s)}{\sum_{s_t \to s'} F_f(s_t \to s')}$ and $\pi_b(s \to s') := \mathbb{P}(s_t = s | s_{t+1} = s') = \frac{F_b(s \to s')}{\sum_{s'' \to s'} F_b(s'' \to s')}$. Obvious variations may be built by replacing $x, y \mapsto \log^2(\frac{x}{y})$ by any distance-like function.

From DAG to Measurable Spaces

Stability, Cycles and 0-Flows on Graphs

Loss Instability on Non-Acyclic Graphs All definitions considered in the case of DAG still make sense on Directed Graphs without the acyclicity assumption. Bengio's sampling theorem is still true and will be derived as a subcase of a more general statement given on measurable state spaces. However, training GFlowNets fails in the presence of cycles: one can see that training converges toward infinite edgeflow along cycles. In particular, the expected sampling time τ grows to infinity: $\mathbb{E}(\tau) \to +\infty$. For example, consider the following families of edgeflows F^θ parameterized by a free non-negative parameter $\theta := (f_1, f_2, f_3, c)$.



Flows on this graph satisfying the reward constraints are exactly the F^θ corresponding to $c \geq 0$ and $f_1 = f_2 = f_3 = 1$. However, $\mathcal{L}_{\text{FM}}$ may be minimized by letting $c \to +\infty$ and $f_1, f_2, f_3 \to 0$. This is captioned by $\partial_c \mathcal{L}_{\text{FM}}(F^\theta) < 0$, this problem also occurs for $\mathcal{L}_{\text{DB}}$ and $\mathcal{L}_{\text{TB}}$.

Linearity A typical implementation of a flow on a graph will be via an MLP with two heads: a softmax and a scalar. They respectively output the forward policy and the outgoing flow at a given state. From an analytical view point, however, edgeflows on graphs are non-negative real-valued maps on the edges set: the space of edgeflows is thus a subdomain of a vector space. Furthermore, the reward and flowmatching constraints are linear, we deduce that sums and convex combinations of edgeflows as well as directional derivatives of a functional on edgeflows in the direction of some edgeflow makes sense. Precise description of the space of flows is given in appendix.

0-Flows Cycles induce flows on graphs in a natural way: for a cycle $\gamma = (s_1 \to \cdots \to s_\ell \to s_1)$, we may define the 0-flow 1_γ by $1_\gamma(s \to s') = 1$ if the edge $s \to s'$ is in the cycle γ and 0 otherwise. Such a 1_γ is a flow but also satisfies the reward constraint for null reward: it is a 0-flow.

Definition 2. *A 0-flow is a flow satisfying the reward constraint (2) for the null reward $R = 0$.*

0-flows are a nice way to speak about cycles as edgeflows: taking the derivative of a functional on edgeflows in the direction of a cycle makes sense. Moreover, on graphs, the space of 0-flows is generated by cycles (see appendix).

Formalization of Stability Next, define an edgeflow F_0 as a subflow of an edgeflow F with $F_0 \leq F$ as functions defined on edges. We can have the following stability definition.

Definition 3 (Stability). *A loss $\mathcal{L}$ acting on families of flows $(F_1, \cdots, F_p)$ is stable if adding a 0-flow does not decrease the loss. More formally: if for any 0-flow F_0*

$$\mathcal{L}(F_1 + F_0, \cdots, F_p + F_0) \geq \mathcal{L}(F_1, \cdots, F_p),$$

then the loss is stable.

Lemma 1. *A sufficient condition for stability is that*

$$\partial_{F_0} \mathcal{L}(F_1, \cdots, F_p) \geq 0$$

for all 0-flow F_0 which is a subflow of each F_i.

Stabilizing Regularization The first consequence of the discussion above is the use of stabilizing regularizations.

Theorem 1. *Let*

$$\mathcal{L}_\alpha(F) = \mathcal{L}(F) + \alpha \mathcal{R}(F) \tag{7}$$

with $\mathcal{L}$ stable, $\alpha > 0$ and $\mathcal{R}$ such that $\partial_{F_0} \mathcal{R} > 0$ for all 0-subflows. Let $\alpha_n \to 0^+$ and let $(F_n)_{n\in\mathbb{N}}$ be a sequence of $\mathcal{L}_{\alpha_n}$-minimizing R-edgeflows.

Assume the sequence $(F_n)_{n\in\mathbb{N}}$ converges to a flow, then it converges toward an acyclic R-flow.

Theorem 1 suggests that regularizations controlling the size of the total edgeflow, such as the norm of the edgeflow matrix on graphs, might help kill cycles thus shortening the expected sampling time. Theorem 2 at the end of the following section also goes in this direction.

Beyond Graphs

In order to generalize GFlowNets from the graph intuition to a general measurable state space setting, instead of considering the outgoing flow for a single point or the edgeflow of a single edge, we consider flows on domains: $F(A \to B)$ from a domain $A \subset S$ and toward a domain $B \subset S$. It is readily interpreted in the case of graphs as $\sum_{A \ni s \to s' \in B} F(s \to s')$, In/outgoing flows are $F(S \to \cdot)$ and $F(\cdot \to S)$, respectively.

In the same way, the reward is considered as a function returning the total reward available some domain $A \subset S^*$. On graphs this means $R(A) = \sum_{s\in A} R(s)$. On finite graphs, as functions of domains, R and $F_{\text{in/out}}$ are clearly additive e.g. $R(A \cup B) = R(A) + R(B)$ if A, B are disjoint subsets of S^*, on the other hand, the edgeflow $F(\cdot \to \cdot)$ is additive in both variables. A natural extension of the notion flows is then via measures: an edgeflow is a bimeasure on S i.e. a measure on $S \times S$ while $F_{\text{in/out}}$ and R are measures on S. For technical reasons, we shall restrict to *finite non-negative measures*: all measure considered $F, F_{\text{out}}, F_{\text{in}}, R, \ldots$ are assumed finite. The usual generalization of forward policies of Markov chains in the measurable context is then via Markov kernels.

From an edgeflow F as a measure on $S \times S$, we may define the outgoing/ingoing flow from/to a given domain A by $F_{\text{out}}(A) := F(A \to S)$ and $F_{\text{in}}(A) := F(S \to A)$. Both the *reward constraint* for a reward measure R on S and the *flow-matching constraint* translate readily to equalities between measures: $F(\cdot \to s_f) = R$ and $\mathbf{1}_{S^*} F_{\text{in}} = \mathbf{1}_{S^*} F_{\text{out}}$. An edgeflow is an R-edgeflow if it satisfies the reward constraint for R and an R-flow if it satisfies both the reward constraint for R and the flow-matching constraint. In particular, a 0-flow is a flow satisfying the reward constraint for $R = 0$.

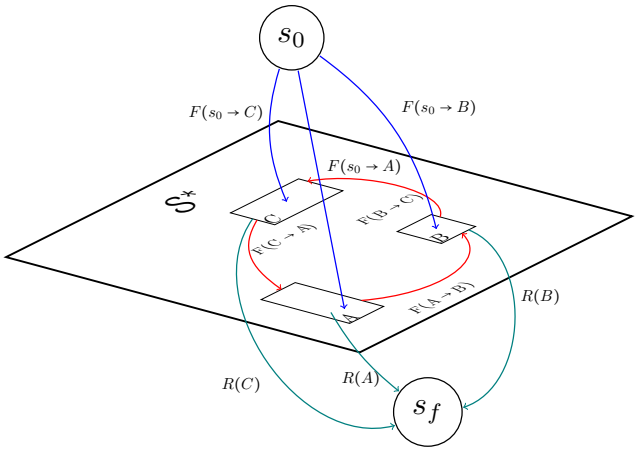


Figure 2: A depiction of an R-edgeflow on a measurable space. The initial flow $F(s_0 \to \cdot)$ is in blue and the Reward R is in green. A possible 0-flow is highlighted in red.

In order to properly understand the generalization of edges (i.e. transition restrictions) in the measurable setting, allow us to remind the reader of the following characterization of the domination relation between measures:

$$\forall \nu, \mu \text{ measures on } S, \quad \nu \ll \mu \Leftrightarrow \exists \varphi,\ \nu = \varphi\mu.$$

Say S is given by a graph, taking $\mu := \sum_{s\to s'} \delta_{s\to s'}$, the relation $F \ll \mu$ is equivalent to the existence of some function $\varphi : S \times S \to \mathbb{R}$ such that $F = \sum_{s\to s'} \varphi(s \to s')\delta_{s\to s'}$ in which case the function φ is exactly the edgeflow, i.e.

$$F = \sum_{s\to s'} F(s \to s')\delta_{s\to s'}.$$

We thus generalize edges constraining transitions to the data of a measure μ dominating the edgeflow. The source/sink property of s_0/s_f is enforced by assuming $\mu(s_f \to S^*) = \mu(S \to s_0) = 0$ and $\mu(s_f \to s_f) = 1$. The notion of cycle becomes shaky when dealing with measures in general. However, the notion of 0-flow given in the preceding section extends readily hence the definition of stability is identical.

A technical difficulty appears when linking edgeflows to their induced forward (and backward) policies. A natural way is via Radon-Nikodym differentiation and tensor product of measures, see Appendix for more details. For now, we simply state that any edgeflow F induces a forward (resp. backward) Markov transition kernel well defined on the 'support' of F_{out} (resp. F_{in}) and conversely, from (π_f, F_{out}) or (π_b, F_{in}) one may recover the edgeflow F:

$$(F_{\text{in}}, \pi_b) \underset{\otimes}{\overset{(F_{\text{in}}, \frac{d}{dF_{\text{in}}})}{\rightleftarrows}} F \underset{\otimes}{\overset{(F_{\text{out}}, \frac{d}{dF_{\text{out}}})}{\rightleftarrows}} (F_{\text{out}}, \pi_f)\ .$$

We may now recover the fundamental sampling result of Bengio et al. (2021a). Let F be an edgeflow and let π_f be its induced forward policy. Consider $(s_t)_{t\geq 0}$ the Markov chain of transition kernel π_f and starting at s_0, the sampling time

	Bengio et al. (2021b)	Lahlou et al. (2023)	Measurable (ours)
State space S	DAG	Topological space, Borel σ-algebra	Measurable space
Edgeflow	Function on edges $F(s \to s')$	Implicit $F_{\text{out}} \otimes \pi_f$	Measure F on $S \times S$
In/Outgoingflow	Functions on states $F_{\text{in/out}}(s)$	Measures $F_{\text{in}}, F_{\text{out}}$ on S	Idem
Trajectory Flow	Function on trajectories	Measure on trajectories	Idem
Reward	Function on states $R(s)$	Measure R on S	Idem
Forward policy	$\mathbb{P}(s_{t+1}\|s_t) = \frac{F(s_t \to s_{t+1})}{\sum_{s_t \to s} F(s_t \to s)}$	Markov kernel $\pi_f : S \to S$	$\pi_f(x, A) = \frac{\mathrm{d}F(\cdot \to A)}{\mathrm{d}F(\cdot \to S)}$
Backward policy	$\mathbb{P}(s_t\|s_{t+1}) = \frac{F(s_t \to s_{t+1})}{\sum_{s \to s_{t+1}} F(s \to s_{t+1})}$	Markov kernel $\pi_b : S \to S$	$\pi_b(x, A) = \frac{\mathrm{d}F(A \to \cdot)}{\mathrm{d}F(S \to \cdot)}$
Transitions	Edges	Dominating kernel $\pi_f \ll \kappa$	Dominating measure $F \ll \mu$
Cycles?	Acyclic	s_f finitely absorbing hence acyclic	0-flows
Reachable Reward	Connectivity	Ergodicity	Dominating measure $R \ll \nu_{train}$
Paths length	Finite graph: $\tau \leq \#S$	s_f finitely absorbing: $\tau \leq \tau_{max}$	Finite measure: $\mathbb{E}(\tau) \leq \frac{F_{\text{out}}(S^*)}{R(S^*)}$

Table 1: Translation from directed graphs to measurable spaces

of the edgeflow F is the last position before reaching the sink:

$$\tau = \max\{t \mid s_t \neq s_f\}. \tag{8}$$

Theorem 2. *Assuming $R \neq 0$, the sampling time τ of a R-flow is almost surely finite and the sampling distribution is proportional to R. More precisely:*

$$\mathbb{E}(\tau) \leq \frac{F_{\text{out}}(S^*)}{R(S^*)}, \text{ and } s_\tau \sim \frac{1}{R(S^*)} R. \tag{9}$$

Finally, Bengio et al. (2021b) introduced a notion of trajectory flow that is readily translated in our context as a finite non-negative measure on the space of paths from s_0 to s_f. We denote by $F^{\otimes}$ the trajectory flow induced by an edge flow F i.e. $F_{\text{out}}(s_0)$ times the probability distribution of paths obtained by applying the kernel π_f repeatedly starting at s_0 until one reaches s_f. Figure 1 summarizes the translation from graphs to measurable state spaces.

Acyclic 0-Flows Intuitively, cycles tend to induce exploding sampling time i.e. $\mathbb{E}(\tau) \to +\infty$ while training. While cycles introduce this issue in graphs, in the continuous setting, they are not the only factor causing this pathological behavior as the following example shows.

Take $S^* = \{z \in \mathbb{C} \mid |z| = 1\}$ the unit circle, we may consider a Markov chain of forward policy π_f on S^* that is perpetually wandering but never cyclic. One may even take π_f deterministic taking rotations such as $\pi_f(z) = \exp(2i\pi\theta)z$ with θ irrationnal. It is tempting to consider that cycles are negligible in the continuous case but the continuous case comes with many ergodic transformations such as irrational rotations as above, each inducing many '*almost-cycles*'.

The above is an instance of acyclic 0-flow. Controlling 0-flows is thus at the very least a theoretical necessity to control that perpetually wandering trajectories and bounding the expected sampling time.

Stable Non-Acyclic Losses

Training Losses as Generalized Divergences

Since the flow-matching property is an equality between measures, it is natural to build losses based on divergences. Define

$$\mathrm{div}_{g,\nu}(\alpha, \beta) := \int_{x \in \mathcal{X}} g\left(\frac{d\alpha}{d\beta}(x)\right) d\nu(x), \tag{10}$$

where α, β, ν are finite measures on some measurable space $\mathcal{X}$ and g is some non-negative function. Define ν_{state}, ν_{edge}, ν_{path} as training distributions on S^*, $S \times S$ and $\mathrm{Path}(s_0, s_f) := \{s_0 \to s_1 \to \cdots \to s_\tau \to s_f, \ \tau \geq 1\}$ respectively. Then, we can write generalizations of FM, DB, and TB losses (4)-(6) via divergence forms:

$$\mathcal{L}_{\text{FM}}(F) = \mathrm{div}_{g,\nu_{\text{state}}}(F_{\text{in}}, F_{\text{out}}), \tag{11}$$

$$\mathcal{L}_{\text{DB}}(F_f, F_b) = \mathrm{div}_{g,\nu_{\text{edge}}}(F_f, F_b), \tag{12}$$

$$\mathcal{L}_{\text{TB}}(F_f, F_b) = \mathrm{div}_{g,\nu_{\text{path}}}(F_f^{\otimes}, F_b^{\otimes}) \tag{13}$$

where $F_f^{\otimes}$ is the trajectory flow, non-negative finite measure on the space of paths $\mathrm{Path}(s_0, s_f)$, induced by some edge-flow F_f. Beware that the trajectory flow $F_b^{\otimes}$ is induced by a backward kernel. Direct computations on graphs show the choice $g = \log^2$ indeed yields the losses $\mathcal{L}_{\text{FM}}, \mathcal{L}_{\text{DB}}$ and $\mathcal{L}_{\text{TB}}$ given in (4)-(6).

In practice, F_f and F_b are built from $F_f := F_{\text{out}} \otimes \pi_f$ and $F_b := F_{\text{out}} \otimes \pi_b$ and with the convention that $\pi_b(s_f, \cdot) \propto R$ to enforce the reward constraint. In a similar manner, $F_f^{\otimes}$ and $F_b^{\otimes}$ are built from $F_f^{\otimes} = F_{\text{init}} \delta_{s_0} \pi_f^{\otimes\tau}$ and $F_b^{\otimes} = R(S^*)\delta_{s_f}\pi_b^{\otimes\tau}$, where we defined for any kernel π the kernel such that $\pi^{\otimes\tau(x)}$ is the distribution of paths starting at x and ending at the first encounter of s_0 or s_f. Hence, $\mathcal{L}_{\text{FM}}, \mathcal{L}_{\text{DB}}$ and $\mathcal{L}_{\text{TB}}$ are special cases of (10).

Natural variations of these losses are given by f-divergences:

$$\mathcal{L}_{\text{FM},f}(F) = \mathrm{div}_f(F_{\text{in}}, F_{\text{out}}), \tag{14}$$

$$\mathcal{L}_{\text{DB},f}(F_f, F_b) = \mathrm{div}_f(F_f, F_b), \tag{15}$$

$$\mathcal{L}_{\text{TB},f}(F_f, F_b) = \mathrm{div}_f(F_f^{\otimes}, F_b^{\otimes}) \tag{16}$$

where div_f denotes an f-divergence. We allow f-divergences to be improper in the sense that we do not assume a priori that f satisfies the usual hypotheses (Polyanskiy and Wu 2022+), i.e. f convex, strictly convex at 1 with

$f(1) = 0$. We want div_f to be distance-like on measures not only on probability measures, i.e. for all non-negative finite measures α, β: $\mathrm{div}_f(\alpha, \beta) \geq 0$ and $\mathrm{div}_f(\alpha, \beta) = 0 \Rightarrow \alpha = \beta$. Therefore, we assume that $\forall x \in \mathbb{R}_+^*, f(x) \geq 0$ with equality if and only if $x = 1$. In particular, this hypothesis excludes Kullback-Leibler cross-entropy.

Theorem 3 (Instability of divergence-based losses)**.** *If* div_f *is a proper divergence, then* $\mathcal{L}_{FM,f}, \mathcal{L}_{DB,f}$ *are* ***unstable*** *except if* div_f *is the total variation*[3].

The losses $\mathcal{L}_{FM,g,\nu}, \mathcal{L}_{DB,g,\nu}$ *and* $\mathcal{L}_{TB,g,\nu}$ *are unstable.*

A Family of Stable Losses

The problem of the divergence-like losses is their form of scale invariance: $t \mapsto \frac{d(F_1+tF_0)}{d(F_2+tF_0)}$ is converging to 1 as $t \geq 0$ grows. This suggests replacing this scale invariance with a 0-flow invariance:

$$\Delta_{f,g,\nu}(\alpha, \beta) := \int_{x \in A} f\left(\alpha(x) - \beta(x)\right) g(\alpha(x), \beta(x)) d\nu(x)$$

where α, β are measurable functions that will be taken densities of considered measures with respect to a background measure λ. Then, we can have

$$\mathcal{L}_{\mathrm{FM},\Delta,f,g,\nu}(F) = \Delta_{f,g,\nu_{\mathrm{state}}}(F_{\mathrm{in}}, F_{\mathrm{out}}), \quad (17)$$

$$\mathcal{L}_{\mathrm{DB},\Delta,f,g,\nu}(F_f, F_b) = \Delta_{f,g,\nu_{\mathrm{edge}}}(F_f, F_b). \quad (18)$$

In Theorem 4, we prove stable conditions of the above two loss functions.

Theorem 4. *If* f, g, ν, R *satisfy*

- $f \geq 0$ *and* $g \geq 1$*;*
- $f(x) = 0 \Leftrightarrow x = 0$*;*
- f *and* g *are continuous and piecewise continuously differentiable;*
- f *decreases on* $\mathbb{R}_-$ *and increases on* $\mathbb{R}_+$*;*
- $\partial_{(1,1)} g \geq 0$*,*

then for R*-edgeflows the loss* $\mathcal{L}_{FM,\Delta,f,g,\nu}, \mathcal{L}_{DB,\Delta,f,g,\nu}$ *are stable.*

According to theoretical results, we can have the following stable loss examples. It is a pity that we cannot find the stable condition of TB, because the related theoretical analysis of TB is very tricky due to the nonlinearity of the operator $F \mapsto F^{\otimes}$ which returns the trajectory flow associated with an edgeflow. We also give a stable version of CFlowNet loss (Li et al. 2023d), which is a variant of FM loss.

Example 1 (Stable FM loss)**.** *For* $\mathcal{L}_{FM,\Delta}$*, a particularly natural choice of is* $f(x) = \log(1 + \epsilon|x|^\alpha)$ *and* $g(x, y) = (1 + \eta(x + y))^\beta$*; yielding in the instance of graphs losses:*

$$\mathcal{L} := \mathbb{E} \sum_{t=1}^{\tau} \left\{ \log\left[1 + \varepsilon \left|F_{\mathrm{in}}(s_t) - F_{\mathrm{out}}(s_t)\right|^\alpha\right] \times \left(1 + \eta\left(F_{\mathrm{in}}(s_t) + F_{\mathrm{out}}(s_t)\right)\right)^\beta \right\} \quad (19)$$

for some $\epsilon, \eta, \alpha, \beta > 0$*. In particular,* $(\alpha, \beta, \epsilon, \eta) = (2, 1, 0.001, 1)$ *is straightforward as convexity should help convergence for such an under-determined linear problem and dampening too large flow matching error keep the intuitions of the original Bengio loss of not driving too much the training by nodes far from satisfying the flow-matching constraint.*

[3]see Appendix for a more precise statement.

Example 2 (Stable DB loss)**.** *One may modify the FM loss above to yield a DB loss:*

$$\mathcal{L} := \mathbb{E} \sum_{t=1}^{\tau} \left\{ \log\left[1 + \varepsilon \left|F^f(s_t \to s_{t+1}) - F^b(s_t \to s_{t+1})\right|^\alpha\right] \times \left(1 + \eta F_{\mathrm{out}}(s_t)\right)^\beta \right\}, \quad (20)$$

where $F^f(s_t \to s_{t+1}) = F_{\mathrm{out}}(s_t) \times \pi_f(s_t \to s_{t+1})$ *and* $F^b(s_t \to s_{t+1}) = F_{\mathrm{out}}(s_{t+1}) \times \pi_b(s_t \to s_{t+1})$ *when forward and backward flows are parameterized by a forward and a backward policy respectively.*

Example 3 (Stable CFlowNet loss)**.** *A continuous variation may be written for CFlowNets by replacing* $F_{\mathrm{in/out}}$ *by their densities* $f_{i/o}$ *with respect to a background measure:*

$$\mathcal{L} := \mathbb{E} \sum_{t=1}^{\tau} \left\{ \log\left[1 + \varepsilon \left|f_i(s_t) - f_o(s_t)\right|^\alpha\right] \times \left(1 + \eta\left(f_i(s_t) + f_o(s_t)\right)\right)^\beta \right\}. \quad (21)$$

Experiments Results

Results on Hypergrids

Here $S^* = [\![1, W]\!]^D$ together with transitions of the form $s \to s \pm (0, \cdots, 0, 1, 0, \cdots, 0)$ in addition to an initial transition $s_0 \to a$ for some given $a \in S^*$ and terminal transitions $\forall s \in S^*, s \to s_f$. In Bengio et al. (2021a) more restrictive transitions $s \to s + (0, \cdots, 0, 1, 0, \cdots, 0)$ were chosen to enforce acyclicity. GFlowNets for such small graphs were implemented using the adjacency matrix of the underlying graph while the edgeflow along every transition is stored in an array. We compared $\mathcal{L}_{\mathrm{FM}}$ and $\mathcal{L}_{\mathrm{TB}}$ to $\mathcal{L}_{\mathrm{FM},\Delta,f,g}$. We observed that both losses lead to a converging model but that their paths' behavior differ significantly: the expected path length induced by $\mathcal{L}_{\mathrm{FM}}$ explodes as the paths wander while for $\mathcal{L}_{\mathrm{FM},\Delta,f,g}$ the paths are relatively straight to reward yielding areas. In fact, the edgeflow for non-stable FM losses diverges to infinity reducing the relative reward signal: even if the reward at s_t is high, since the edgeflow toward non-stopping directions are high (due to cycles) then $F(s_t \to s_f) \ll F(s_t \to s')$ for $s' \neq s_f$ and the Markov chain continue to wander, see figures 1 and 3. The behavior of $\mathcal{L}_{\mathrm{TB}}$ is more nuanced: it induces wandering paths but the expected sampling time does not explode and rather stabilizes at a significantly higher value than for stable losses in some cases. In some others, the TB loss eventually reaches both the same expected reward and expected path lengths as stable losses after a long training time.

Results on Cayley Graphs

Here $S^* = \mathfrak{S}_p$, the group of permutations of $[\![0, p-1]\!]$ for some $p \geq 1$. The edges are given by a set of generators

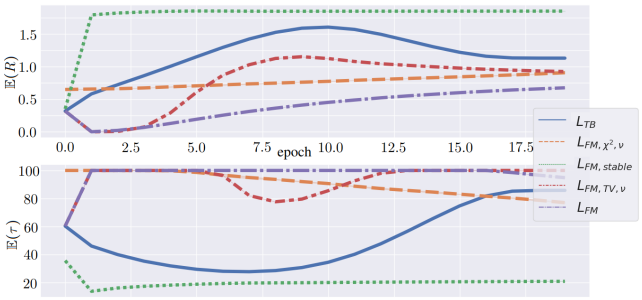


Figure 3: GFlowNets were trained with various unstable and stable losses on the 2DW20 grid: $\mathcal{L}_{FM,stable} = \mathcal{L}_{FM,\Delta,f,0,\nu}$ with $f(x) = x^2$; $\mathcal{L}_{FM,\chi^2,\nu}$ and $\mathcal{L}_{FM,TV,\nu}$ are instances of $\mathcal{L}_{FM,f,\nu}$ as above with $f(x) = (1-x)^2$ and $f(x) = |1-x|$ respectively. Top: evolution of averaged reward during training. Bottom: evolution of average lengths of sampled paths.

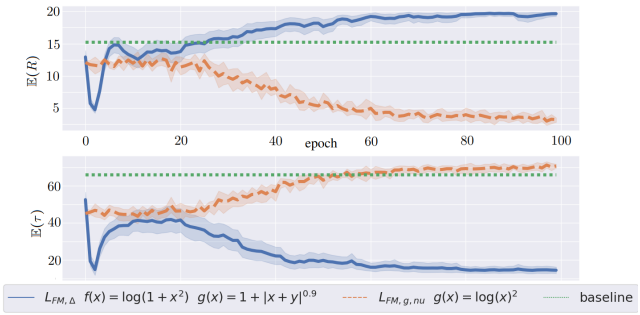


Figure 4: Comparison results of Stable-GFlowNets (ours), GFlowNets, and a Metropolis-Hasting baseline on the Cayley graph of $\mathfrak{S}_{20}$ generated by a transposition, a 20-cycle and its inverse. The reward is R_1 as above with $k = 1$ and $c = 20$. Paths are drawn with a cut-off length of 80. An optimal strategy yields an expected reward of 20 with an expected length of 5. Top: evolution of averaged reward under training. Bottom: Average lengths of sampled paths.

$(\sigma_1, \cdots, \sigma_q)$ so that for all $g \in S^*$ and all $i \in [\![1, q]\!]$, we have an edge $g \to g\sigma_i$ (more generally one may replace $\mathfrak{S}_p$ by any discrete group). $p \geq 25$ provides instances of graphs that are intractably big even for current supercomputers but still have small diameters, have tractable FM losses, are easy to implement, and provide a wide variety of structures, symmetries, and cycles. Many tasks may be reduced to partially ordering a list, we are thus led to construct an element of $\mathfrak{S}_p$ using generators corresponding to available actions: for instance, a bubble sort only uses transpositions of adjacent objects in the list L to sort, so $p = \text{length}(L)$ and $\sigma_i = (i, i+1)$.

This experiment differs from the other presented in that the initial policy $\pi_f(s_0 \to \cdot)$ is fixed as the uniform distribution on S^*. Two reasons motivate this choice. Firstly, exploratory behavior is not easy to enforce but we may leverage the highly connected and cyclic nature of Cayley graphs to allow starting from any position. Secondly, it leads to learning a generic strategy to solve a problem from any initial configuration.

We compared training using $\mathcal{L}_{\text{FM}}$ and $\mathcal{L}_{\text{FM},\Delta}$ on two families of rewards: $R_1(\sigma) := c\mathbf{1}_{\sigma \in S_1}$ and $R_2(\sigma) = d(\rho(\sigma), S_2)$ for some subsets $S_1, S_2 \subset S^*$. Taking $S_1 = \{\sigma \mid \sigma(i) = i, i \leq k\}$ leads to emulating a partial sorting algorithm, see figure 4 for $p = 20$ and R_1 with $c = 20$ and $k = 1$.

Choices of R_2 emulate common heuristics such as variations of the Manhattan distance. The policy $\pi_f(s_0 \to \cdot) = \mathcal{U}(S^*)$ is non-trainable to emulate a random initial instance, thus training the flow to solve a partial ordering problem from any position. We observed that, as theoretically predicted, the flow tends to grow uncontrollably when using unstable losses, while using stable losses, the flow stays bounded, thus mitigating the issue of exploding sampling time.

Results on Continuous Task

To show the effectiveness of the proposed Stable-CFlowNets loss, we conduct experiments on Point-Robot-Sparse continuous control tasks with sparse rewards. The goal of the agent is to navigate two different goals with coordinates $(10, 10)$ and $(0, 0)$. The agent starts at $(5, 5)$ and the maximum episode length is 12. We changed the angle range of the agent's movement from (0, 90°) to (0, 360°), that is, a cycle can be generated in the trajectory. All other experimental settings and algorithm hyperparameters in Roint-Robot-Sparse are the same as in Li et al. (2023d). We refer to Li et al. (2023d) for more details. As for Stable-CFlowNets, we only change the loss of CFlowNets to that in Example 3.

Figure 5 shows the number of valid-distinctive trajectories explored and the reward during the training process. After a certain number of training epochs, 5000 trajectories are collected. We can see that the proposed stable loss does not affect the exploration performance, while the reward has improved a lot, and it has become more stable (the variance range is smaller). It is worth noting that the two completely opposite goals are a difficult environment for RL algorithms, so the reward of the RL algorithms is not high.

Figure 6 gives an example of sampled trajectories on the Point-Robot-Sparse task of Stable-CFlowNets. To obtain these trajectories, we sample 1000 random actions per node to obtain the corresponding flow outputs (probabilities) and then randomly select one of the top-300 most probable actions to generate trajectories. We can see that there are many cycles in the trajectory at the beginning of the training, but it can still converge to the goal at the end of the training, which shows that our stable loss is very robust to cycles.

Conclusion and Discussion

We introduced a theoretical framework to generalize GFlowNets concepts and Theorems beyond their initial scope of directed acyclic graphs to measurable spaces. The analysis conducted allowed to link sampling time i.e. the length of sampled paths and the stability of the training via gradient descent. This analysis allowed us to define losses adapted to spaces with possible cycles as well as regularization, helping manage cycles.

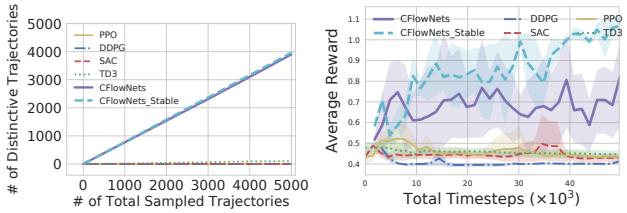


Figure 5: Comparison results of Stable-CFlowNets (ours), CFlowNets, DDPG, TD3, SAC and PPO on Point-Robot-Sparse. Left: Number of valid-distinctive trajectories generated under 5000 explorations. Right: The average reward of different methods.

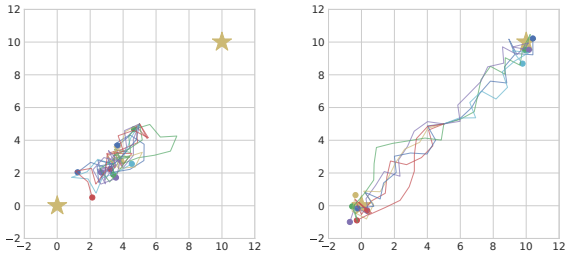


Figure 6: An example of sampled trajectories on Point-Robot-Sparse task of Stable-CFlowNets (ours). Left: at the beginning of training. Right: at the end of training.

Related Works

In the midst of the growing literature on GFlowNets (Bengio et al. 2021b; Jain et al. 2022a; Malkin et al. 2022; Madan et al. 2022; Zhang et al. 2023b; Jain et al. 2022b; Pan et al. 2023; Deleu and Bengio 2023) three, in particular, are closely related to ours. Bengio et al. (2021b) attempt at laying the foundation of the theory of GFlowNets, they discuss many openings for future applications or explorations of the method. The work of Lahlou et al. (2023) using a similar approach to generalize GFlowNets. Finally, Li et al. (2023d) made a first attempt at training GFlowNets for continuous state space. However, none of these works attack cyclic space limitations, in particular, our stability property is new. Furthermore, our framework is somewhat less involved than that of Lahlou et al. (2023) in that most of the fundamental work deals with general finite non-negative measures. Extra hypotheses enforcing acyclicity are not used nor even specified. Furthermore, we explore the structure of the space of flows on graphs, and in general, this is new and a step toward a better understanding of good practices for the training of GFlowNets.

Considering GFlowNets in such a general setting leads to a direct comparison to MCMC (Brooks et al. 2011) and reinforcement learning (Sutton and Barto 2018). On the one hand, from a sampling viewpoint, GFlowNets and MCMC are interchangeable: both work in the setting of an unnormalized distribution on a space with the goal of sampling the space proportionally to this target distribution. More precisely, the trainable nature of GFlowNets has to be compared to adaptive MCMC (Andrieu and Thoms 2008). Two key differences allow us to hope for the replacement of MCMC methods by GFlowNets at least in some situations: GFlowNets benefit from finite mixing-time by opposition of the infinite mixing-time of MCMC, the latter have difficulties moving from modes to modes even after having discovered them while the former can learn sequences of actions to perform such moves. Mode transition is a long-standing problem impairing MCMC efficiency (Andricioaei, Straub, and Voter 2001; Pompe, Holmes, and Łatuszyński 2020; Samsonov et al. 2021); one hopes that GFlowNets can provide global moves to help overcome this issue. On the other hand, GFlowNets on graphs may be thought of as a distributional Q-method (Bellemare, Dabney, and Munos 2017), the outgoing flow playing the role of the expected reward.

Limitations

First, the hypotheses we used on our instability/stability results are not optimal, and they are technical to allow the use of simple arguments. Furthermore, trajectory balance loss was only partially studied: while FM and DB losses are proved to be unconditionally unstable, TB loss is only unstable in some cases and our analysis and experiments suggest it may be stable for small flows.

Second, our theoretical analysis of Trajectory Balance losses is limited. No stable losses variant of this loss is proposed and our theoretical instability result is rather limited. It suggests that it is possible to find constraints, regularizations, or better parameterizations of flows stabilizing trajectory balance losses.

Third, the experiments conducted are rather limited compared to the theoretical range of the work. We did not compare stable versions of detailed balance losses in the continuous setting. More thorough experiments testing the influence of non-cyclic 0-flows in the continuous setting are needed.

Fourth, untrained GFlowNets may have difficulty exploring the space efficiently to learn the target distribution. For adaptive MCMC methods, criteria are known to ensure ergodicity (Andrieu and Thoms 2008). However, the ergodicity of exploration with GFlowNets is completely open and is expected to suffer from caveats similar to that of adaptive MCMC. Our naive approach was to add a constant edgeflow when sampling paths or add a small background reward.

Fifth, on Cayley graphs, we were not able to train GFlowNets so that the initial flow converges quickly toward the theoretical value. Our simple implementation using an initial flow as a trainable parameter of the model proved to be insufficient.

Finally, our experiments on Cayley graphs used the simplest feedforward neural architecture. More sophisticated architectures taking into account the local graph structure and/or learning more global properties of the graph would be desirable. Graph transformer (Dwivedi and Bresson 2020) would be a natural next step, a step already taken on DAG (Zhang et al. 2023b) using topoformers (Gagrani et al. 2022).

References

Andricioaei, I.; Straub, J. E.; and Voter, A. F. 2001. Smart darting monte carlo. *The Journal of Chemical Physics*, 114(16): 6994–7000.

Andrieu, C.; and Thoms, J. 2008. A tutorial on adaptive MCMC. *Statistics and computing*, 18(4): 343–373.

Bellemare, M. G.; Dabney, W.; and Munos, R. 2017. A distributional perspective on reinforcement learning. In *International Conference on Machine Learning*, 449–458. PMLR.

Bengio, E.; Jain, M.; Korablyov, M.; Precup, D.; and Bengio, Y. 2021a. Flow Network based Generative Models for Non-Iterative Diverse Candidate Generation. arXiv:2106.04399.

Bengio, Y.; Deleu, T.; Hu, E. J.; Lahlou, S.; Tiwari, M.; and Bengio, E. 2021b. GFlowNet Foundations. arXiv:2111.09266.

Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; and Zaremba, W. 2016. OpenAI Gym. arXiv:1606.01540.

Brooks, S.; Gelman, A.; Jones, G.; and Meng, X.-L. 2011. *Handbook of Markov chain Monte Carlo*. CRC press.

Deleu, T.; and Bengio, Y. 2023. Generative Flow Networks: a Markov Chain Perspective. *arXiv preprint arXiv:2307.01422*.

Dwivedi, V. P.; and Bresson, X. 2020. A generalization of transformer networks to graphs. *arXiv preprint arXiv:2012.09699*.

Gagrani, M.; Rainone, C.; Yang, Y.; Teague, H.; Jeon, W.; Van Hoof, H.; Zeng, W. W.; Zappi, P.; Lott, C.; and Bondesan, R. 2022. Neural Topological Ordering for Computation Graphs. *arXiv preprint arXiv:2207.05899*.

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

Jain, M.; Raparthy, S. C.; Hernandez-Garcia, A.; Rector-Brooks, J.; Bengio, Y.; Miret, S.; and Bengio, E. 2022b. Multi-Objective GFlowNets. *arXiv preprint arXiv:2210.12765*.

Lahlou, S.; Deleu, T.; Lemos, P.; Zhang, D.; Volokhova, A.; Hernández-García, A.; Ezzine, L. N.; Bengio, Y.; and Malkin, N. 2023. A theory of continuous generative flow networks. *arXiv preprint arXiv:2301.12594*.

Li, W.; Li, Y.; Li, Z.; HAO, J.; and Pang, Y. 2023a. DAG Matters! GFlowNets Enhanced Explainer for Graph Neural Networks. In *The Eleventh International Conference on Learning Representations*.

Li, Y.; Li, Z.; Li, W.; Shao, Y.; Zheng, Y.; and Hao, J. 2023b. Generative Flow Networks for Precise Reward-Oriented Active Learning on Graphs. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*. International Joint Conferences on Artificial Intelligence Organization.

Li, Y.; Luo, S.; Shao, Y.; and Hao, J. 2023c. GFlowNets with Human Feedback. In *The First Tiny Papers Track at ICLR 2023, Tiny Papers @ ICLR 2023, Kigali, Rwanda, May 5, 2023*. OpenReview.net.

Li, Y.; Luo, S.; Wang, H.; and Hao, J. 2023d. CFlowNets: Continuous Control with Generative Flow Networks. In *The Eleventh International Conference on Learning Representations*.

Madan, K.; Rector-Brooks, J.; Korablyov, M.; Bengio, E.; Jain, M.; Nica, A.; Bosc, T.; Bengio, Y.; and Malkin, N. 2022. Learning GFlowNets from partial episodes for improved convergence and stability. *arXiv preprint arXiv:2209.12782*.

Malkin, N.; Jain, M.; Bengio, E.; Sun, C.; and Bengio, Y. 2022. Trajectory Balance: Improved Credit Assignment in GFlowNets. *arXiv preprint arXiv:2201.13259*.

Pan, L.; Zhang, D.; Jain, M.; Huang, L.; and Bengio, Y. 2023. Stochastic Generative Flow Networks. *arXiv preprint arXiv:2302.09465*.

Polyanskiy, Y.; and Wu, Y. 2022+. *Information Theory From Coding to Learning*. Cambridge university press.

Pompe, E.; Holmes, C.; and Łatuszyński, K. 2020. A framework for adaptive MCMC targeting multimodal distributions. *The Annals of Statistics*, 48(5): 2930 – 2952.

Samsonov, S.; Lagutin, E.; Gabrié, M.; Durmus, A.; Naumov, A.; and Moulines, E. 2021. Local-Global MCMC kernels: the best of both worlds. *arXiv preprint arXiv:2111.02702*.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT press.

Towers, M.; Terry, J. K.; Kwiatkowski, A.; Balis, J. U.; Cola, G. d.; Deleu, T.; Goulão, M.; Kallinteris, A.; KG, A.; Krimmel, M.; Perez-Vicente, R.; Pierré, A.; Schulhoff, S.; Tai, J. J.; Shen, A. T. J.; and Younis, O. G. 2023. Gymnasium.

Veach, E.; and Guibas, L. J. 1997. Metropolis light transport. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, 65–76.

Wang, H.; Shao, Y.; Hao, J.; and Li, Y. 2023. Regularized Offline GFlowNets. In *The First Tiny Papers Track at ICLR 2023, Tiny Papers @ ICLR 2023, Kigali, Rwanda, May 5, 2023*. OpenReview.net.

Zhang, D.; Chen, R. T.; Malkin, N.; and Bengio, Y. 2022. Unifying generative models with gflownets. *arXiv preprint arXiv:2209.02606*.

Zhang, D.; Dai, H.; Malkin, N.; Courville, A.; Bengio, Y.; and Pan, L. 2023a. Let the Flows Tell: Solving Graph Combinatorial Optimization Problems with GFlowNets. *arXiv preprint arXiv:2305.17010*.

Zhang, D. W.; Rainone, C.; Peschl, M.; and Bondesan, R. 2023b. Robust scheduling with GFlowNets. *arXiv preprint arXiv:2302.05446*.

Zhu, D.; Li, Y.; Shao, Y.; Hao, J.; Wu, F.; Kuang, K.; Xiao, J.; and Wu, C. 2023. Generalized Universal Domain Adaptation with Generative Flow Networks. In *Proceedings of the 31st ACM International Conference on Multimedia*.