A Theory of Multi-Agent Generative Flow Networks

Anonymous Author(s)
Affiliation
Address
email

Abstract

Generative flow networks utilize a flow-matching loss to learn a stochastic policy for generating objects from a sequence of actions, such that the probability of generating a pattern can be proportional to the corresponding given reward. However, a theoretical framework for multi-agent generative flow networks (MA-GFlowNets) has not yet been proposed. In this paper, we propose the theory framework of MA-GFlowNets, which can be applied to multiple agents to generate objects collaboratively through a series of joint actions. We further propose four algorithms: a centralized flow network for centralized training of MA-GFlowNets, an independent flow network for decentralized execution, a joint flow network for achieving centralized training with decentralized execution, and its updated conditional version. Joint Flow training is based on a local-global principle allowing to train a collection of (local) GFN as a unique (global) GFN. This principle provides a loss of reasonable complexity and allows to leverage usual results on GFN to provide theoretical guarantees that the independent policies generate samples with probability proportional to the reward function. Experimental results demonstrate the superiority of the proposed framework compared to reinforcement learning and MCMC-based methods.

1 Introduction

Generative flow networks (GFlowNets) [1] can sample a diverse set of candidates in an active learning setting, where the training objective is to approximate sampling of the candidates proportionally to a given reward function. Compared to reinforcement learning (RL), where the learned policy is more inclined to sample action sequences with higher rewards, GFlowNets can perform exploration tasks better. The goal of GFlowNets is not to generate a single highest-reward action sequence, but rather is to sample a sequence of actions from the leading modes of the reward function [2]. However, based on current theoretical results, GFlowNets cannot support multi-agent systems.

A multi-agent system is a set of autonomous interacting entities that share a typical environment, perceive through sensors, and act in conjunction with actuators [3]. Multi-agent reinforcement learning (MARL), especially cooperative MARL, is widely used in robot teams, distributed control, resource management, data mining, etc [4, 5, 6]. There two major challenges for cooperative MARL: scalability and partial observability [7, 8]. Since the joint state-action space grows exponentially with the number of agents, coupled with the environment's partial observability and communication constraints, each agent needs to make individual decisions based on the local action observation history with guaranteed performance [9, 10, 11]. In MARL, to address these challenges, a popular centralized training with decentralized execution (CTDE) paradigm [12, 13] is proposed, in which the agent's policy is trained in a centralized manner by accessing global information and executed in a decentralized manner based only on the local history. However, extending these techniques to GFlowNets is not straightforward, especially in constructing CTDE-architecture flow networks and finding IGM conditions for flow networks need investigating.

In this paper, we propose the multi-agent generative flow networks (MA-GFlowNets) framework for cooperative decision-making tasks. Our framework can generate more diverse patterns through sequential joint actions with probabilities proportional to the reward function. Unlike vanilla GFlowNets, the proposed method analyzes the interaction of multiple agent actions and shows how to sample actions from multi-flow functions. Our approach consists of building a virtual global GFN capturing the policies of all agents and ensuring consistency of local (agent) policies. Variations of this approach yield different flow-matching losses and training algorithms.

Furthermore, we propose the Centralized Flow Network (CFN), Independent Flow Network (IFN), Joint Flow Network (JFN), and Conditioned Joint Flow Network (CJFN) algorithms for multi-agent GFlowNets framework. CFN considers multi-agent dynamics as a whole for policy optimization regardless of the combinatorial complexity and demand for independent execution, so it is slower; while IFN is faster, but suffers from the flow non-stationary problem. In contrast, JFN and CJFN, which are trained based on the local-global principle, takes full advantage of CFN and IFN. They can reduce the complexity of flow estimation and support decentralized execution, which are beneficial to solving practical cooperative decision-making problems.

Main Contributions: 1) We first generalize the measure GFlowNets framework to the multi-agent setting, and propose a theory of multi-agent generative flow networks for cooperative decision-making tasks; 2) We propose four algorithms under the measure framework, namely CFN, IFN, JFN and CJFN, for training multi-agent GFlowNets, which are respectively based on centralized training, independent execution, and the latter two algorithms are based on the CTDE paradigm; 3) We propose a local-global principle and then prove that the joint state-action flow function can be decomposed into the product form of multiple independent flows, and that a unique Markovian flow can be trained based on the flow matching condition; 4) Control tasks experiments demonstrate that the proposed algorithms can outperform current cooperative MARL algorithms in terms of exploration capabilities.

1.1 Preliminaries and Notations

Measurable GFlowNets [14, 15, 16], extending the original definition of GFlowNets [17, 1] to non-acyclic continuous and mixed continuous-discrete statespaces, are defined by a tuple $(\mathcal{S}, \mathcal{A}, S, T, R, F_{\text{out}})$ in the single-agent setting, where $\mathcal{S}$ and $\mathcal{A}$ denote the state and action space, S and T are the state and transition map, π and F_{out} are the forward policy and outflow respectively.

$$\begin{tikzcd} \mathcal{A} \arrow[r, bend left, "S"] \arrow[r, bend right, "T"'] & \mathcal{S} \arrow[l, "\pi"'] \arrow[r, shift left, "R"] \arrow[r, shift right, "F_{\text{out}}"'] & \mathbb{R}_+ \end{tikzcd}$$

More precisely, the state space $\mathcal{S}$ and the state-dependent action spaces $\mathcal{A}_s$ are measurable spaces; for each state $s \in \mathcal{S}$, the environment comes with a stochastic transition map[1] $\mathcal{A}_s \xrightarrow{T_s} \mathcal{S}$. We formalize this dependency on state by bundling (packing) state and action together into a bundle $\{(s, a) \mid s \in \mathcal{S}, a \in \mathcal{A}_s\} = \mathcal{A} \xrightarrow{S,T} \mathcal{S}$ where $S(s, a) := s$ and $T(s, a) := T_s(a)$. For graphs, a bundled action is an edge $s \to s'$; the state map S returns the origin s while the transition map returns the destination s'. The forward policy π is a section of S, i.e., a kernel $\mathcal{S} \xrightarrow{\pi} \mathcal{A}$ such that $S \circ \pi$ is identity on $\mathcal{S}$. The outflow (or state-flow) F_{out} and the reward R are non-negative finite measure on $\mathcal{S}$. The state space $\mathcal{S}$ has two special states s_0 and s_f such that $T(s_0, a) \neq s_0$ and $T(s_f, a) = s_f$ for all actions a; furthermore, there is a special action STOP such that $T(s, \text{STOP}) = s_f$ for all state s. The reward R is generally non-trainable and unknown but implicitly a component of F_{out} and π; since the reward may not be tractable in the multi-agent setting, we favor a reward-free parameterization of GFlowNets, ie we restrict all objects to $\mathcal{S}^* := \mathcal{S} \setminus \{s_0, s_f\}$. Therefore, we parameterize them by triplets $\mathbb{F} = (\pi^*, F^*_{\text{out}}, F_{\text{init}})$ where $\pi^*(a|s) = \pi(a|s, a \neq \text{STOP})$, $F^*_{\text{out}} := F_{\text{out}} - R$ and $F_{\text{init}} = F_{\text{out}}(s_0) T \circ \pi(s_0)$. We define a Markov chain $(s_t)_{t \geq 1}$ by sampling a first state s_1 from F_{init}, then $a_t = \pi(s_t)$ and $s_{t+1} = T(a_t)$. The sample generated by the GFlowNet is the last position before hitting s_f. The sampling time τ is then the first t such that $a_t = \text{STOP}$. The distribution of s_τ is controlled by the so-called flow-matching constraint

$$F_{\text{out}} = F_{\text{in}} := F_{\text{init}} + F^*_{\text{out}} \pi^* T, \tag{1}$$

as measures on $\mathcal{S}$, and the sampling Theorem first proved in [1]:

[1]We adopt the naming convention of [18]. The kernel $K : \mathcal{X} \to \mathcal{Y}$ is a stochastic map which is formalized as follows: for all $x \in \mathcal{X}$, $K(x \to \cdot)$ is a probability distribution on $\mathcal{Y}$. In addition, $K(x \to \cdot)$ varies measurably with x in the sense that for all measurable set $A \subset \mathcal{Y}$, the real valued map $x \mapsto K(x \to A)$ is measurable.

Theorem 1 ([15] Theorem 2)**.** *Let $\mathbb{F} := (\pi, F^*_{\text{out}}, F_{\text{init}})$ be a GFlowNets on $(\mathcal{S}, \mathcal{A}, S, T, R)$. If the reward R is non-zero and $\mathbb{F}$ satisfies the flow-matching constraint, then its sampling time is almost surely finite and the sampling distribution is proportional to R. More precisely:*

$$\mathbb{P}(\tau < +\infty) = 1, \mathbb{E}(\tau) \leq \frac{F_{\text{out}}(\mathcal{S})}{R(\mathcal{S})} - 1, \textit{ and } s_\tau \sim \frac{1}{R(\mathcal{S})}R. \tag{2}$$

In passing we introduce $\hat{R} := F_{\text{in}} - F^*_{\text{out}}$, $F^*_{\text{in}} := F^*_{\text{out}}\pi^*T$ and $F_{\text{action}} := F_{\text{out}} \otimes \pi$.

Flow-matching losses (FM), denoted by $\mathcal{L}_{\text{FM}}$, are used to enforce the flow-matching constraint 1. They compare the outflow F_{out} with the inflow $F_{\text{in}} := F_{\text{init}} + F^*_{\text{out}}\pi^*T$; and are minimized when $F_{\text{in}} = F_{\text{out}}$ so that a gradient descent on GFlowNets parameters may enforce equation 1. The previous works [2, 19] used divergence-based FM losses valid as long as the state space is acyclic while [15, 20, 14] introduced stable FM losses and regularization allowing training in the presence of cycles:

$$\mathcal{L}^{\text{div}}_{\text{FM}}(\mathbb{F}^\theta) = \mathbb{E}_{s\sim\nu_{\text{state}}} g \circ \log\left(\frac{dF^\theta_{\text{in}}}{dF^\theta_{\text{out}}}(s)\right) \quad \mathcal{L}^{\text{stable}}_{\text{FM}}(\mathbb{F}^\theta) = \mathbb{E}_{s\sim\nu_{\text{state}}} g\left(\frac{dF^\theta_{\text{in}}}{d\lambda}(s) - \frac{dF^\theta_{\text{out}}}{d\lambda}(s)\right), \tag{3}$$

where g is some positive function, decreasing on $]-\infty, 0]$, $g(0) = 0$ and increasing on $[0, +\infty[$. The simplest choices are $g(x) = x^2$ or $g(x) = \log(1 + \alpha|x|^\beta)$.

1.2 Multi-agent Problem Formulation

The **multi-agent setting** formalizes the data of state, actions, and transitions for multiple agents. Each agent $i \in I$ in the finite agent set I has its own observation $o^{(i)}$ in its observation space $\mathcal{O}^{(i)}$; it depends on the state via the projection $\mathcal{S} \xrightarrow{p^{(i)}} \mathcal{O}^{(i)}$. For simplicity sake, we identify $\mathcal{S} = \prod_{i\in I} \mathcal{O}^{(i)}$. Each agent has its own action space $\mathcal{A}^{(i)}$ and each of the agent observation-dependent action space $\mathcal{A}_o$ contains a special action STOP; the environment is such that once an agent chooses STOP, it is put on hold until all agents do as well. The game finishes when all agents have chosen STOP; a reward is given based on the last state. The reward received is formalized by a non-negative function $r : \mathcal{S} \to \mathcal{S}$. We assume that each agent may freely choose its own action independently from the actions chosen by other agents: this is formalized via $\mathcal{A}_s = \prod_{i\in I} \mathcal{A}^{(i)}_{o^{(i)}} / \sim$ ie the Cartesian product of agent actions space up to the identification of the STOP actions. A trajectory of the system of agents is a, possibly infinite, sequence of states $(s_t)_{t<\tau+1}$ with $\tau \in \mathbb{N} \cup \{\infty\}$ starting at the source state $s_0 \in \mathcal{S}$ and may eventually calling STOP; the space of trajectories is $\mathcal{T}$. A policy on $\mathcal{S}$ induces a Markov chain hence a distribution on trajectories.

$$\begin{tikzcd}
 & \mathcal{A} \arrow[dl, "p^{(i)}"'] \arrow[dr, "p^{(j)}"] \arrow[d, bend right, "S"'] \arrow[d, bend left, "T"] & \\
\mathcal{A}^{(i)} \arrow[d, "S^{(i)}"'] & \mathcal{S} \arrow[u, "\pi"] \arrow[dl, "p^{(i)}"'] \arrow[d, "R"] \arrow[dr, "p^{(j)}"] & \mathcal{A}^{(j)} \arrow[d, "S^{(j)}"] \\
\mathcal{O}^{(i)} & \mathbb{R}_+ & \mathcal{O}^{(j)}
\end{tikzcd}$$

Figure 1: Multi-agent formalism

MA-GFlowNets are tuples $((\mathbb{F}^{(i)})_{i\in I}, \mathbb{F})$, where each *local* GFlowNets $\mathbb{F}^{(i)}$ is defined on $(\mathcal{O}^{(i)}, \mathcal{A}^{(i)}, S^{(i)}, T^{(i)}, R^{(i)})$ for $i \in I$ and the *global* GFlowNets $\mathbb{F}$ is defined on $(\mathcal{S}, \mathcal{A}, S, T, R)$. **The objective of MA-GFlowNets**, similarly to GFlowNets, is to build a policy π so that the induced trajectories are finite and s_τ is distributed proportionally to $R := r\lambda$ where λ is some fixed measure on $\mathcal{S}$ and $\int_{s\in\mathcal{S}} r(s)d\lambda(s)$ is finite. In general, some GFlowNets (local or global) may be virtual, i.e. not implemented.

2 Multi-Agent GFlowNets

This section is devoted to details and theory regarding the variations of algorithms for MA-GFlowNets training. If resources allow, the most direct approach is included in the training of the global model directly, leading to a centralized training algorithm in which the local GFlowNets are virtual. As expected, such an algorithm suffers from high computational complexity, hence, demanding decentralized algorithms. Decentralized algorithms require the agents to collaborate to some extent. We achieve such a collaboration by enforcing consistency rules between the local and global GFlowNets. The global GFlowNets is virtual and is used to build a training loss for the local models ensuring the global model is GFlowNets, so that the sampling Theorem applies. The sampling properties of the MA-GFlowNets are then deduced from the flow-matching property of the virtual global model.

2.1 Centralized Training

Centralized training consists in training of the global flow directly. Here, the local flows are virtual:they are theoretically recovered from the global flow as image by the observation maps but not implemented. We use FM-losses as given in equations 3 applied to the flow on $(\mathcal{S}, \mathcal{A})$. See Algorithm 1. Implicitly, F_{out} contains a parameterizable component from F^*_{out}, while F_{in} contains the parameterization of π^* and F_{init}.

Algorithm 1 Centralized Flow Network Training Algorithm for MA-GFlowNets

---

Require: A multi-agent environment $(\mathcal{S}, \mathcal{A}, \mathcal{O}^{(i)}, \mathcal{A}^{(i)}, p_i, S, T, R)$, a parameterized GFlowNets $\mathbb{F} := (\pi, F^*_{\text{out}}, F_{\text{init}})$ on $(\mathcal{S}, \mathcal{A})$.

while not converged **do**

  Sample and add trajectories $(s_t)_{t \geq 0} \in \mathcal{T}$ to replay buffer with policy $\pi(s_t \to a_t)$.

  Generate training distribution ν_{state}.

  Apply minimization step of the FM loss $\mathcal{L}^{\text{stable}}_{\text{FM}}(\mathbb{F}^\theta)$.

end while

From the algorithmic viewpoint, the CFN algorithm is identical to a single GFlowNets. As a consequence, the usual results on the measurable GFlowNets apply as is. There are, however, a number of key difficulties: 1) even on graphs, the computational complexity increases as $O(|\mathcal{A}_s|^N)$ at any given explored state; 2) centralized training requires all agents to share observations, which is impractical since in many applications the agents only have access to their own observations.

2.2 Local Training: Independent

The dual training method is embodied in the training of local GFlowNets instead of the global one. In this case, the local flows $\mathbb{F}^{(i)}$ are parameterized and the global flow is virtual. In the same way, a local FM loss is used for each client. In order to have well-defined local GFlowNets, we need a local reward, for which a natural definition is $R^{(i)}(o_t^{(i)}) := \mathbb{E}(R(s_t)|o_t^{(i)})$. The local training loss function can be written as: $\mathcal{L}(\mathbb{F}^{(i)}) = \mathbb{E} \sum_{t=1}^{\tau} g\left(F^{\theta_i}_{\text{in}}\left(o^i_t\right) - F^{\theta_i}_{\text{out}}\left(o^i_t\right)\right)$.

The algorithm 3 in Appendix B describes the simplest training method, which solves the issue of exponential action complexity with an increasing number of agents. In this formulation, however, two issues arise: the evaluation of ingoing flow $F^{(i)}_{\text{in}}(o^{(i)})$ becomes harder as we need to find all transitions leading to a given local observation (and not to a given global state). This problem may be non-trivial as it is also related to the actions of other agents. More importantly, in this case, the local reward is intractable, so we cannot accurately estimate the reward $R^{(i)}(o^{(i)})$ of each node. Falling back to using the stochastic reward $R^{(i)}(o^{(i)}) := R(s_t|o_t^{(i)})$ instead leads to transition uncertainty and spurious rewards, which can cause non-stationarity and/or mode collapse as shown in Figure 2.

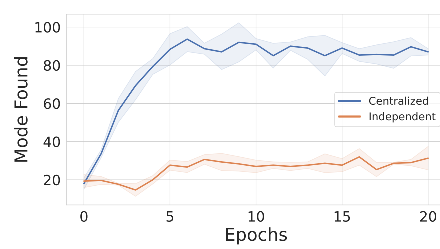


Figure 2: Performance comparison on Hypergrid task.

2.3 Local-Global Training

2.3.1 Local-Global Principle: Joint Flow Network

Local-global training is based upon the following local-global principle, which combined with Theorem 1 ensures that the MA-GFlowNet has sampling distribution proportional to the reward R.

Theorem 2 (Joint MA-GFlowNets)**.** *Given local GFlowNets $\mathbb{F}^{(i)}$ on some environments $(\mathcal{O}^{(i)}, \mathcal{A}^{(i)}, S^{(i)}, T^{(i)})$ there exists a global GFlowNets $\mathbb{F}^{\text{joint}}$ on a multi-agent environment $(\prod_{i \in I} \mathcal{O}^{(i)}, \mathcal{A}, S, \tilde{T})$ consistent with the local GFlowNets $\mathbb{F}^{(i)}$, such that*

$$F^*_{\text{out}} = \prod_{i \in I} F^{(i),*}_{\text{out}}, \quad F_{\text{in}} = \prod_{i \in I} F^{(i)}_{\text{in}}. \tag{4}$$

Moreover, if $\mathbb{F}^{\text{joint}}$ *satisfies equation 1 for a reward* R *and each* $\hat{R}^{(i)} \geq 0$ *then* $R = \prod_{i \in I} \hat{R}^{(i)}$.

Theorem 2 states that if the $\tilde{T}$ guided by the local transition map $T^{(i)}$ is consistent with the true transition map T, and the global reward R is the product of the local rewards, then the local and global flow function satisfies the (4). Based on this conclusion, our Joint Flow Network (JFN) algorithm leverages Theorem 2 by sampling trajectories with policy

$$o_t^{(i)} = p_i(s_t^{(i)}) \text{ and } \pi^{(i)}(o_t^{(i)} \to a_t^{(i)}), \;\; i \in I \tag{5}$$

with $a_t = (a_t^{(i)} : i \in I)$ and $s_{t+1} = T(s_t, a_t)$, build formally the (global) joint GFlowNet from local GFlowNets and train the collection of agent via the FM-loss of the joint GFlowNet. Equation 4 ensures that the inflow and outflow of the (global) joint GFlowNet are both easily computable from the local inflows and outflows provided by agents. See algorithm 2.

Algorithm 2 Joint Flow Network Training Algorithm for MA-GFlowNets

Require: Number of agents N, A multi-agent environment $(\mathcal{S}, \mathcal{A}, \mathcal{O}^{(i)}, \mathcal{A}^{(i)}, p_i, S, T, R)$.
Require: Local GFlowNets $(\pi^{(i),*}, F_{\text{out}}^{(i),*}, F_{\text{init}}^{(i)})_{i \in I}$.
while not converged **do**
  Sample and add trajectories $(s_t)_{t \geq 0} \in \mathcal{T}$ to replay buffer with policy according to (5).
  Generate training distribution ν_{state} from replay buffer
  Apply minimization step of $\mathcal{L}_{\text{FM}}^{\text{stable}}(\mathbb{F}^{\theta,\text{joint}})$ for R
end while

This training regiment presents two key advantages: over centralized training, the action complexity is linear w.r.t. the number of agents and local actions as in the independent training; over independent training, the reward is not spurious. Indeed, in $\mathcal{L}_{\text{FM}}^{\text{stable}}(\mathbb{F}^{\theta,\text{joint}})$, by equation 4, the computation of F_{in} and F_{out}^{*} reduces to computing the inflow and star-outflow for each local GFlowNets. Also, only the global reward R appears. The remaining, possibly difficult, challenge is the estimation of local ingoing flows from the local observations as it depends on the local transitions $T^{(i)}$, see first point below. At this stage, the relations between the global/joint/local flow-matching constraints are unclear, and furthermore, the induced policy of the local GFlowNets still depends on the yet undefined local rewards. The following point clarify those links.

First, the collection of local GFlowNets induces local transitions kernels $T^{(i)} : \mathcal{O}^{(i)} \to \mathcal{O}^{(i)}$ which are not uniquely determined in general by a single GFlowNets. Indeed, the local policies induce a global policy $\pi(s_t \to a_t) := \prod_{i \in I} \pi(o_t^{(i)} \to a_t^{(i)})$. Then, the (virtual) transition kernel $T^{(i)}(a_t^{(i)}) = p^{(i)}(T(a_t)|a_t^{(i)})$ of the GFlowNets i depends on the distribution of states and the corresponding actions of **all** local GFlowNets. See appendix A.5 for details. Note that $T^{(i)}$ are derived from the actual environment T and the joint GFlowNets on the multi-agent environment with the true transition T, while the Theorem above ensures splitting of star-inflows and virtual rewards only for the approximated $\tilde{T}$. Furthermore, local rewards may be formalized as stochastic rewards to take into account the lack of information of a single agent, but they are never used during training: the allocation of rewards across agents is irrelevant. Only the virtual rewards $\hat{R}^{(i)} = F_{\text{out}}^{(i),*} - F_{\text{in}}^{(i)}$ are relevant but they are effectively free. As a consequence, Algorithm 2 effectively trains both the joint flow as well as a product environment model. But since in general $T \neq \tilde{T}$ Algorithm 2 may fail to reach satisfactory convergence.

Second, beware that in our construction of the joint MA-GFlowNets, there is no guarantee that the global initial flow is split as the product of the local initial flows. In fact, we favor a construction in which F_{init} is non-trivial to account for the inability of local agents to assess synchronization with another agent. See Appendix A.8 for formalization details.

Third, we may partially link local and global flow-matching properties.

Theorem 3. *Let* $(\mathbb{F}^{(i)})_{i \in I}$ *be local GFlowNets and let* $\mathbb{F}$ *be their joint GFlowNets. Assume that none of the local GFlowNets are zero and that each* $\hat{R}^{(i)} \geq 0$*. If* $\mathbb{F}$ *satisfies equation 1, then there exists an "essential" subdomain of each* $\mathcal{O}^{(i)}$ *on which local GFlowNets satisfy the flow-matching constraint.*

The restriction regarding the domain on which local GFlowNets satisfy the flow-matching constraint is detailed in Appendix A.8, this sophistication arises because of the stopping condition of the multi-agent system. The essential domain may be informally formulated as "where the local agent is still playing": an agent may decide (or be forced) to stop playing, letting other agents continue playing, the forfeited player is then on hold until the game stops and rewards are actually awarded.

To conclude, the joint GFlowNets provides an approximation of the target global GFlowNets, this approximation may fail if the transition kernel T is highly coupled or if the reward is not a product.

2.3.2 Conditioned Joint Flow Network

Training of MA-GFlowNets via training of the virtual joint GFlowNets is an approximation of the centralized training. In fact, the space of joint GFlowNets is smaller than that of the general MA-GFlowNets, as only rewards that splits into the product $R(s) = \prod_{i \in I} R^{(i)}(o^{(i)})$ may be exactly sampled. If the rewards are not of this form, the training may still be subject to a spurious reward or mode collapse. One may easily build more sophisticated counter-examples based on this one.

Our proposed solution is to build a conditioned JFN inspired by augmented flows [21, 22] methods, which allow the bypass of architectural constraints for Normalization flows [23]. The trick is to add a shared "hidden" state to the joint MA-GFlowNets allowing the agent to synchronize. This hidden state is constant across a given episode and may be understood as a cooperative strategy chosen beforehand by the agents. Formally, this simply consist in augmenting the state space and the observation spaces by a strategy space Ω to get $\tilde{\mathcal{S}} = \mathcal{S} \times \Omega$ and $\tilde{\mathcal{O}}^{(i)} = \mathcal{O}^{(i)} \times \Omega$, F_{init} is augmented by a distribution $\mathbb{P}$ on Ω, the observation projections as well as transition kernel act trivially on Ω ie $T(s;\omega) = T(s)$ and $p^{(i)}(s;\omega) = (p^{(i)}(s), \omega)$. The joint MA-GFlowNets theorem applies the same way, beware that the observation part of $T^{(i)}$ now have a dependency on Ω even though T does not. In theory, Ω may be big enough to parameterize the whole trajectory space $\mathcal{T}$, in which case it is possible to have decoupled conditioned local transition kernels $T^{(i)}(\cdot;\omega)$ so that $\tilde{T} = T$ on a relevant domain. Furthermore, the limitation on the reward is also lifted if the flow-matching property is enforced on the expected joint flow $\mathbb{E}_\omega \mathbb{F}^{\text{joint}}$. Two possible losses may be considered: $\mathbb{E}_\omega \mathcal{L}_{\text{FM}}^{\text{stable}}(\mathbb{F}^{\theta,\text{joint}}(\cdot;\omega))$ or $\mathcal{L}_{\text{FM}}^{\text{stable}}(\mathbb{E}_\omega \mathbb{F}^{\theta,\text{joint}}(\cdot;\omega))$. The former, which we use in our experiments, is simpler to implement but does not a priori lift the constraint on the reward.

The training phase of the Conditioned Joint Flow Network (CJFN) is shown in Algorithm 4 in the appendix. We first sample trajectories with policy $o_t^{(i)} = p_i(s_t^{(i)})$ and $\pi_\omega^{(i)}(o_t^{(i)} \to a_t^{(i)}),\ \ i \in I$ with $a_t = (a_t^{(i)} : i \in I)$ and $s_{t+1} = T(s_t, a_t)$. Then we train the sampling policy by minimizing the FM loss $\mathbb{E}_\omega \mathcal{L}_{\text{FM}}^{\text{stable}}(\mathbb{F}^{\theta,\text{joint}}(\cdot;\omega))$.

3 Related Works

Generative Flow Networks:

Nowadays, GFlowNets has achieved promising performance in many fields, such as molecule generation [2, 19, 24], discrete probabilistic modeling [25], structure learning [26], domain adaptation [27], graph neural networks training [28, 29], and large language model training [30, 31, 32]. This network could sample the distribution of trajectories with high rewards and can be useful in tasks where the reward distribution is more diverse. GFlowNets is similar to reinforcement learning (RL) [33]. However, RL aims to maximize the expected reward favoring mode collapse onto the single highest reward yielding action sequence, while GFlownets favor diversity. Tiapkin et al. [34] bridged GFlowNets to entropy-RL.

Comprehensive distributed GFlowNets framework is still lacking. Previously, the meta GFlowNets algorithm [35] was proposed to solve the problem of GFlowNets distributed training but it requires the observation state and task objectives of each agent to be the same, which is not suitable for multi-agent problems. Later, a multi-agent GFlowNets algorithm was proposed in [36], but lacked theoretical support and general framework. Connections between MA-GFlowNets and multi-agent RL are discussed in Appendix C.

Cooperative Multi-agent Reinforcement Learning: There exist many MARL algorithms to solve collaborative tasks. Two extreme algorithms for thus purpose are independent learning [37] and centralized training. Independent training methods regard the influence of other agents as part of the

(a) Win Rate (b) Episode 1 (c) Episode 2

Figure 3: The performance comparison results on the 3m map of StarCraft

environment, but the team reward function often has difficulty in measuring the contribution of each agent, resulting in the agent facing a non-stationary environment [9].

On the contrary, centralized training treats the multi-agent problem as a single-agent counterpart. However, this method has high combinatorial complexity and is difficult to scale beyond dozens of agents [7]. Therefore, the most popular paradigm is centralized training and decentralized execution (CTDE), including value-based [9, 11, 38, 10] and policy-based [39, 40, 41] methods. The goal of value-based methods is to decompose the joint value function among the agents for decentralized execution. This requires satisfying the condition that the local maximum of each agent's value function should be equal to the global maximum of the joint value function.

The methods, VDN [9] and QMIX [11] employ two classic and efficient factorization structures, additivity and monotonicity, respectively, despite their strict factorization method. In QTRAN [38] and QPLEX [10], extra design features are introduced for decomposition, such as the factorization structure and advantage function. The policy-based methods extend the single-agent TRPO [42] and PPO [43] into the multi-agent setting, such as MAPPO [40], which has shown surprising effectiveness in cooperative multi-agent games. These algorithms maximize the long-term reward, however, it is difficult for them to learn more diverse policies in order to generate more promising results.

4 Experiments

We first verify the performance of CFN on a multi-agent hyper-grid domain where partition functions can be accurately computed. We then compare the performance of CFN and CJFN with standard MCMC and some RL methods to show that our proposed sampling distributions better match normalized rewards. All our code is done using the PyTorch [44] library. We re-implemented the multi-agent RL algorithms and other baselines.

4.1 Hyper-grid Environment

We consider a multi-agent MDP where states are the cells of a N-dimensional hypercubic grid of side length H. In this environment, all agents start from the initialization point $x = (0, 0, \cdots)$, and are only allowed to increase coordinate i with action a_i. In addition, each agent has a stop action. When all agents choose the stop action or reach the maximum H of the episode length, the entire system resets for the next round of sampling. The reward function is designed as

$$R(x) = R_0 + R_1 \prod_i \mathbb{I}\left(0.25 < |x_i/H - 0.5|\right) + R_2 \prod_i \mathbb{I}\left(0.3 < |x_i/H - 0.5| < 0.4\right), \quad (6)$$

where $x = [x_1, \cdots, x_I]$ includes all agent states and the reward term $0 < R_0 \ll R_1 < R_2$ leads a distribution of modes. The specific details about the environments and experiments can be found in the appendix.

We compare CFN and CJFN with a modified MCMC and RL methods. In the modified MCMC method [45], we allow iterative reduction of coordinates on the basis of joint action space and cancel the setting of stop actions to form an ergodic chain. As for the RL methods, we consider the maximum entropy algorithm, i.e., multi-agent SAC [46], and a previous cooperative multi-agent algorithm, i.e., MAPPO, [40]. To measure the performance of these methods, we use the normalized L1 error as $\mathbb{E}[|p(s_f) - \pi(s_f)| \times N]$ with $p(s_f) = R(s_f)/Z$ being the sample distribution computed by the true reward, where N is cardinality of the space of s_f.

Moreover, we can consider the mode found theme to demonstrate the superiority of the proposed algorithm.

Figure 4 illustrates the performance superiority of our proposed algorithm compared to other methods in the L1 error and Mode Found. We find that on small-scale environments shown in Figure 4-Left, CFN can achieve the best performance because CFN can accurately estimate the flow of joint actions when the joint action space dimension is small. There are two main reasons for the large l1-error index. First, we normalized the standard L1 error and multiplied it by a constant to avoid the inconvenience of visualization of a smaller magnitude. Secondly, when evaluating L1-error, we only sampled 20 rounds for calculation, and with the increase of the number of samples, L1-error will further decrease. As the complexity of the estimation of action flow increases, we find that the performance of CFN degrades while the joint-flow-based methods still achieve good estimation and maintain the speed of convergence, as shown in Figure 4-Middle.

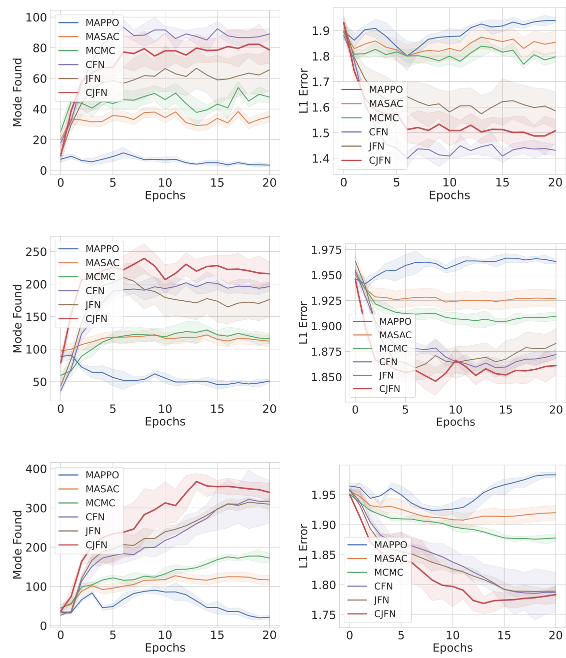


Figure 4: Mode Found (Left, higher is better) and L1 error (Right, lower is better) performance of different algorithms on hyper-grid environments. **Top:** Hyper-Grid v1, **Middle:** Hyper-Grid v2, **Bot:** Hyper-Grid v3.

4.2 StarCraft

Figure 3 shows the performance of the proposed algorithm on the StarCraft 3m map [47], where (a) shows the win rate comparison with different algorithms, and (b) and (c) show the decision results sampled using the proposed algorithm. In the experiment, the outflow flow is calculated when the flow function is large, and the maximum flow is used to calculate the win rate when sampling. It can be found that since the experimental environment is not a sampling environment with diversified rewards, although the proposed algorithm is not significantly better than other algorithms, it still illustrates its potential in large-scale decision-making. In addition, the proposed algorithm can sample results with more diverse rewards, such as (b) and (c), and the number of units left implies the trajectory reward.

5 Conclusion

In this paper, we discussed the policy optimization problem when GFlowNets meets the multi-agent systems. Different from RL, the goal of MA-GFlowNets is to find diverse samples with probability proportional to the reward function. Since the joint flow is equivalent to the product of independent flow of each agent, we designed a CTDE method to avoid the flow estimation complexity problem in a fully centralized algorithm and the non-stationary environment in the independent learning process, simultaneously. Experimental results on Hyper-Grid environments and StarCraft task demonstrated the superiority of the proposed algorithms.

Limitation and Future Work: Our theory is incomplete as it does not apply to non-cooperative agents and has limited support of different game/agent terminations or initialization. A local-global principle beyond independent agent policies would also be particularly interesting. Our experiments do not cover the whole range of the theory in particular regarding continuous tasks and CJFN loss on projected GFN. An ablation study analyzing the tradeoff of small versus big condition space Ω would enlighten its importance. Finally, a metrization of the space of global GFlowNet would allow a more precise functional and optimization analysis of JFN/CJFN and their limitations.

References

[1] Yoshua Bengio, Salem Lahlou, Tristan Deleu, Edward J Hu, Mo Tiwari, and Emmanuel Bengio. Gflownet foundations. *JMLR*, 24(1):10006–10060, 2023.

[2] Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. Flow network based generative models for non-iterative diverse candidate generation. In *NeurIPS*, 2021.

[3] Lucian Busoniu, Robert Babuska, and Bart De Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(2):156–172, 2008.

[4] Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of Reinforcement Learning and Control*, pages 321–384, 2021.

[5] Lorenzo Canese, Gian Carlo Cardarilli, Luca Di Nunzio, Rocco Fazzolari, Daniele Giardino, Marco Re, and Sergio Spanò. Multi-agent reinforcement learning: A review of challenges and applications. *Applied Sciences*, 11(11):4948, 2021.

[6] Amal Feriani and Ekram Hossain. Single and multi-agent deep reinforcement learning for ai-enabled wireless networks: A tutorial. *IEEE Communications Surveys & Tutorials*, 23(2):1226–1252, 2021.

[7] Yaodong Yang, Rasul Tutunov, Phu Sakulwongtana, Haitham Bou Ammar, and Jun Wang. α^α-rank: Scalable multi-agent evaluation through evolution. 2019.

[8] Matthijs TJ Spaan. Partially observable markov decision processes. In *Reinforcement Learning*, pages 387–414. Springer, 2012.

[9] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z Leibo, Karl Tuyls, et al. Value-decomposition networks for cooperative multi-agent learning. In *AAMAS*, 2018.

[10] Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. Qplex: Duplex dueling multi-agent q-learning. *arXiv preprint arXiv:2008.01062*, 2020.

[11] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *ICML*, 2018.

[12] Frans A Oliehoek, Matthijs TJ Spaan, and Nikos Vlassis. Optimal and approximate q-value functions for decentralized pomdps. *Journal of Artificial Intelligence Research*, 32:289–353, 2008.

[13] Frans A Oliehoek and Christopher Amato. *A concise introduction to decentralized POMDPs*. Springer, 2016.

[14] Leo Maxime Brunswic, Mateo Clémente, Rui Heng Yang, Adam Sigal, Amir Rasouli, and Yinchuan Li. Ergodic generative flows. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, Proceedings of Machine Learning Research. PMLR, 2025.

[15] Leo Brunswic, Yinchuan Li, Yushun Xu, Yijun Feng, Shangling Jui, and Lizhuang Ma. A theory of non-acyclic generative flow networks. In *AAAI Conference on Artificial Intelligence*, 2024.

[16] Salem Lahlou, Tristan Deleu, Pablo Lemos, Dinghuai Zhang, Alexandra Volokhova, Alex Hernández-Garcıa, Léna Néhale Ezzine, Yoshua Bengio, and Nikolay Malkin. A theory of continuous generative flow networks. In *ICML*, 2023.

[17] Tristan Deleu and Yoshua Bengio. Generative flow networks: a markov chain perspective. *arXiv preprint arXiv:2307.01422*, 2023.

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

[19] Nikolay Malkin, Moksh Jain, Emmanuel Bengio, Chen Sun, and Yoshua Bengio. Trajectory balance: Improved credit assignment in gflownets. In *NeurIPS*, 2022.

[20] Nikita Morozov, Ian Maksimov, Daniil Tiapkin, and Sergey Samsonov. Revisiting non-acyclic gflownets in discrete environments. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, Proceedings of Machine Learning Research. PMLR, 2025.

[21] Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented neural odes. *Advances in neural information processing systems*, 32, 2019.

[22] Chin-Wei Huang, Laurent Dinh, and Aaron Courville. Augmented normalizing flows: Bridging the gap between generative flows and latent variable models. *arXiv preprint arXiv:2002.07101*, 2020.

[23] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021.

[24] Moksh Jain, Emmanuel Bengio, Alex Hernandez-Garcia, Jarrid Rector-Brooks, Bonaventure FP Dossou, Chanakya Ajit Ekbote, Jie Fu, Tianyu Zhang, Michael Kilgour, Dinghuai Zhang, et al. Biological sequence design with gflownets. In *ICML*, 2022.

[25] Dinghuai Zhang, Nikolay Malkin, Zhen Liu, Alexandra Volokhova, Aaron Courville, and Yoshua Bengio. Generative flow networks for discrete probabilistic modeling. In *ICML*, 2022.

[26] Tristan Deleu, António Góis, Chris Emezue, Mansi Rankawat, Simon Lacoste-Julien, Stefan Bauer, and Yoshua Bengio. Bayesian structure learning with generative flow networks. In *Uncertainty in Artificial Intelligence*, 2022.

[27] Didi Zhu, Yinchuan Li, Yunfeng Shao, Jianye Hao, Fei Wu, Kun Kuang, Jun Xiao, and Chao Wu. Generalized universal domain adaptation with generative flow networks. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 8304–8315, 2023.

[28] Yinchuan Li, Zhigang Li, Wenqian Li, Yunfeng Shao, Yan Zheng, and Jianye Hao. Generative flow networks for precise reward-oriented active learning on graphs. *arXiv preprint arXiv:2304.11989*, 2023.

[29] Wenqian Li, Yinchuan Li, Zhigang Li, Jianye Hao, and Yan Pang. Dag matters! gflownets enhanced explainer for graph neural networks. *arXiv preprint arXiv:2303.02448*, 2023.

[30] Yinchuan Li, Shuang Luo, Yunfeng Shao, and Jianye Hao. Gflownets with human feedback. *arXiv preprint arXiv:2305.07036*, 2023.

[31] Edward J Hu, Moksh Jain, Eric Elmoznino, Younesse Kaddar, Guillaume Lajoie, Yoshua Bengio, and Nikolay Malkin. Amortizing intractable inference in large language models. *arXiv preprint arXiv:2310.04363*, 2023.

[32] Dinghuai Zhang, Yizhe Zhang, Jiatao Gu, Ruixiang Zhang, Josh Susskind, Navdeep Jaitly, and Shuangfei Zhai. Improving gflownets for text-to-image diffusion alignment. *arXiv preprint arXiv:2406.00633*, 2024.

[33] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[34] 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*, pages 4213–4221. PMLR, 2024.

[35] Xinyuan Ji, Xu Zhang, Wei Xi, Haozhi Wang, Olga Gadyatskaya, and Yinchuan Li. Meta generative flow networks with personalization for task-specific adaptation. *Information Sciences*, 672:120569, 2024.

[36] Shuang Luo, Yinchuan Li, Shunyu Liu, Xu Zhang, Yunfeng Shao, and Chao Wu. Multi-agent continuous control with generative flow networks. *Neural Networks*, 174:106243, 2024.

[37] Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *ICML*, 1993.

[38] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *ICML*, 2019.

[39] Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *NeurIPS*, 2017.

[40] Chao Yu, Akash Velu, Eugene Vinitsky, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of ppo in cooperative, multi-agent games. In *NeurIPS*, 2022.

[41] Jakub Grudzien Kuba, Ruiqing Chen, Munning Wen, Ying Wen, Fanglei Sun, Jun Wang, and Yaodong Yang. Trust region policy optimisation in multi-agent reinforcement learning. In *ICLR*, 2022.

[42] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *ICML*, 2015.

[43] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[44] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019.

[45] Yutong Xie, Chence Shi, Hao Zhou, Yuwei Yang, Weinan Zhang, Yong Yu, and Lei Li. {MARS}: Markov molecular sampling for multi-objective drug discovery. In *ICLR*, 2021.

[46] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *ICML*, 2018.

[47] Oriol Vinyals, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani, Heinrich Küttler, John Agapiou, Julian Schrittwieser, et al. Starcraft ii: A new challenge for reinforcement learning. *arXiv preprint arXiv:1708.04782*, 2017.

[48] Olav Kallenberg et al. *Random measures, theory and applications*, volume 1. Springer, 2017.

[49] Martin Riedmiller, Roland Hafner, Thomas Lampe, Michael Neunert, Jonas Degrave, Tom Wiele, Vlad Mnih, Nicolas Heess, and Jost Tobias Springenberg. Learning by playing solving sparse reward tasks from scratch. In *ICML*, 2018.

[50] Alexander Trott, Stephan Zheng, Caiming Xiong, and Richard Socher. Keeping your distance: Solving sparse reward tasks using self-balancing shaped rewards. In *NeurIPS*, 2019.

A Joint Flow Theory

The goal of this section is to lay down so elementary points on the measurable theory of MA-GFlowNets as well as prove the main theorem on the joint GFlowNet.

A.1 Notations on Measures and Kernels

We mostly use notations from [18] regarding kernels and measures. The measurable GFlowNet formalism is that of [14] In the whole section, since we deal with technicalities, we use kernel type notations for image by kernels and maps (seen as deterministic kernels). So that for a kernel $K : X \to Y$ and a measure μ on X we denote by μK the measure on Y defined by $\mu K(B) = \int_{x \in X} K(x \to B) d\mu(x)$ for $B \subset Y$ measurable and $\mu \otimes K$ is the measure on $X \times Y$ so that $\mu \otimes K(A \times B) = \int_{x \in A} K(x \to B) d\mu(x)$. Recall that a measure ν dominates a measure μ which is denoted $\mu \ll \nu$, if for all measurable A, $\nu(A) = 0 \Rightarrow \mu(A) = 0$. The Radon-Nykodim Theorem ensures that if $\mu \ll \nu$ and μ, ν are finite then there exists $\varphi \in L^1(\nu)$ so that $\mu = \varphi \nu$. This function φ is called the Radon-Nykodim derivative and is denoted $\frac{d\mu}{d\nu}$. We favor notations $\mu(A \to B)$ when μ is a measure on $X \times Y$ and $A \subset X$ and $B \subset Y$; also $\mu(A \to \cdot)$ means the measure $B \mapsto \mu(A \to B)$. We provide the notations in Table 1.

Table 1: The Descriptions of Different Notations

Notation	Descriptions
$\mathcal{S}$	state space, any measurable space with special elements s_0, s_f
λ	background measure on $\mathcal{S}$; usually the counting measure on discrete spaces, or the Lebesgue measure on continuous spaces
$\mathcal{A}$	action space, a measurable space with special element STOP, it is a bundle over $\mathcal{S}$
$\mathcal{S}^{(i)} = \mathcal{O}^{(i)}$	state/observation space of agent i, same properties as $\mathcal{S}$
$\mathcal{A}^{(i)}$	action space of agent i, has a special element $\mathrm{STOP}^{(i)}$, it is a bundle over $\mathcal{S}^{(i)}$
$\mathcal{A}_s$	action space on state s, ie the fiber above s of the bundle $\mathcal{A} \xrightarrow{S} \mathcal{S}$
$\mathcal{A}^{(i)}_{o^{(i)}}$	action space on observation $o^{(i)}$, ie the fiber above $o^{(i)}$ of the bundle $\mathcal{A}^{(i)} \xrightarrow{S^{(i)}} \mathcal{O}^{(i)}$
S	state map $S(s, a) = s$, i.e., return the current state s
$S^{(i)}$	state map of agent i
T	Transition map $T(s, a) = T_s(a) = s'$, i.e., transfer the current state s into next state s'
$T^{(i)}$	Transition map of agent i, depend in general of the chosen action of all agent and is thus stochastic
R	the reward measure on $\mathcal{S} \setminus \{s_0, s_f\}$, usually known via its density r with respect to the background measure λ
$R^{(i)}$	the perceived reward measure of agent i, usually intractable and stochastic
$\hat{R}$	the GFlowNet reward measure on $\mathcal{S} \setminus \{s_0, s_f\}$ used at inference time instead of R for stop decision making
$\hat{R}^{(i)}$	the GFlowNet reward measure of agent i on $\mathcal{O}^{(i)} \setminus \{s_0, s_f\}$ used at inference time instead of $R^{(i)}$ for stop decision making
F_{out}	out-flow or state-flow, non-negative measure on $\mathcal{S} \setminus \{s_0, s_f\}$
$F^{(i)}_{\text{out}}$	out-flow or state-flow of agent (i), non-negative measure on $\mathcal{O}^{(i)} \setminus \{s_0, s_f\}$
F^{*}_{out}	star out-flow, non-negative measure on the space $\mathcal{S} \setminus \{s_0, s_f\}$ such that $F^{*}_{\text{out}} := F_{\text{out}} - R$
$F^{(i),*}_{\text{out}}$	star out-flow of agent i, non-negative measure on the space $\mathcal{O}^{(i)} \setminus \{s_0, s_f\}$ such that $F^{(i),*}_{\text{out}} = F^{(i)}_{\text{out}} - R^{(i)}$
π	the forward policy, can call STOP action
π^{*}	the forward policy defined on the space $\mathcal{S} \setminus \{s_0, s_f\}$, does not call STOP action
$\pi^{(i)}$	the forward policy of agent i, can call STOP action
$\pi^{(i),*}$	the star forward policy of agent i defined on the space $\mathcal{S} \setminus \{s_0, s_f\}$, does not call STOP action
F_{init}	the unnormalized distribution used to sample s_1 while moving $s_0 \to s_1$
$F^{(i)}_{\text{init}}$	the unnormalized distribution used by agent i to sample $s^{(i)}_1$ while moving $s_0 \to s_1$
$r(s)$	the density of reward at s on a continuous statespace

A.2 An Introduction for Notations

We understand that our formalism is abstract, this section is devoted justifying our choices and providing examples.

A.2.1 Motivations

To begin with, our motivation to formalize the action space as a measurable bundle $\mathcal{A} := \{(s,a) \mid s \in \mathcal{S}, a \in \mathcal{A}_s\} \xrightarrow{S} \mathcal{S}$ is three fold:

1. The available actions from a state may depend on the state itself: on a grid, the actions available while on the boundary of the grid are certainly more limited than while in the middle. More generally, on a graph, actions are typically formalized by edges $s \xrightarrow{a} s'$ of the graph, the data of an edge contains both the origin s and the destination s'. In other words, on graphs, actions are bundled with an origin state. It is thus natural to consider the actions as bundled with the origin state. When an agent is transiting from a state to another via an action, the state map tells where it comes from while the transition map tells where it is going.
2. We want our formalism to cover as many cases as possible in a unified way: Graphs, vector spaces with linear group actions or mixture of discrete and continuous state spaces. Measures and measurable spaces provide a nice formalism to capture the quantity of reward on a given set or a probability distribution.
3. We want a well-founded and possibly standardized mathematical formalism. In particular, the policy takes as input a state and returns a distribution of actions. the actions should correspond to the input state. To avoid having a cumbersome notion of policy as a family of distributions π_s each on $\mathcal{A}_s$, we prefer to consider the union of the state-dependent action spaces $\mathcal{A} := \bigcup_{s\in\mathcal{S}} \mathcal{A}_s$ and define the policy as Markov kernel $\mathcal{S} \to \mathcal{A}$. However, we still need to satisfy the constraint that the distribution $\pi(s)$ is supported by $\mathcal{A}_s$. Bundles are usual mathemcatical objects formalizing such situations and constraints and are thus well suited for this purpose and the constraint is easily expressed as $S \circ \pi(s) = s, \forall s \in \mathcal{S}$.

Our synthetic formalism comes with a few drawbacks due to the level of abstraction:

1. The notation $\pi(s)$ differs from the more common notation $\pi(s,a)$ as the action already contains s implicitly.
2. We need to use Radon-Nikodym derivative. At a given state, on a graph, a GFlowNets has a probability of stopping
$$\mathbb{P}(\mathrm{STOP}|s) = \frac{R(s)}{F_{\mathrm{out}}(s)}.$$
On a continuous statespace with reference measure λ, the stop probability is
$$\mathbb{P}(\mathrm{STOP}|s) = \frac{r(s)}{f_{\mathrm{out}}(s)}$$
where $r(s)$ is the density of reward at s and $f_{\mathrm{out}}(s)$ is the density of outflow at s. A natural measure-theoretic way of writing these equations as one is via Radon-Nikodym derivation: given two measures μ, ν; if $\mu(X) = 0 \Rightarrow \nu(X) = 0$ for any measurable $X \subset \mathcal{S}$ then μ is said to dominate ν and, by Radon-Nikodym Theorem, there exists a measurable function $\varphi \in L^1(\mu)$ such that $\nu(X) = \int_{x\in X} \varphi(x) d\nu(x)$ for all measurable $X \subset \mathcal{S}$. This φ is the Radon-Nikodym derivative $\frac{d\nu}{d\mu}$.
If one has a measure λ dominating both R and F_{out} and if F_{out} dominated R then
$$\mathbb{P}(\mathrm{STOP}|s) := \frac{dR}{dF_{\mathrm{out}}}(s) = \frac{dR}{d\lambda}(s) \times \left(\frac{dF_{\mathrm{out}}}{d\lambda}\right)^{-1}.$$
When $\mathcal{S}$ is discrete, we choose λ as the counting measure, and we recover the graph formula above. When $\mathcal{S}$ is continuous, we choose λ as the Lebesgue measure, and we recover the second formula.

A.2.2 Example

Consider the D-dimensional W-width hypergrid case with agent set I, see Figure 5. The state space is the finite set $\mathcal{S} = \left(\{1, \cdots, W\}^D\right)^I$, say each agent only observes its own position on the grid so that $\mathcal{O}^{(i)} = \{1, \cdots, W\}^D$. the observation-dependent action space of the i-th agent $\mathcal{A}^{(i)}_{o^{(i)}}$ is a subset of $H := \{\pm \mathbf{1}_k : 1 \leq k \leq W\}$ where $\mathbf{1}_k$ is the hot-one array $(0, \cdots, 0, 1, 0, \cdots, 0)$ with a one at the k-th coordinate. The set $\mathcal{A}^{(i)}_{o^{(i)}}$ depends on s: if $1 < s_k < W$ then $\mathcal{A}^{(i)}_{o^{(i)}} = \{\pm \mathbf{1}_k : 1 \leq k \leq W\} \cup \{\text{STOP}\}$ but if $s_k = 1$ then $-\mathbf{1}_k \notin \mathcal{A}^{(i)}_{o^{(i)}}$ and similarly if $s_k = W$. The local total action space is then

$$\mathcal{A}^{(i)}_{o^{(i)}} = \{(s, a) \mid 1 \leq s_k \leq W \text{ and } 1 \leq s_k + a_k \leq W\} \cup \{\text{STOP}\} \subset \{1, \cdots, W\}^D \times H \cup \{\text{STOP}\}.$$

The local state maps $S^{(i)}$ is $S^{(i)}(o^{(i)}, a) = o^{(i)}$. Since each agent may choose its action freely, for any $s \in \mathcal{S}, \mathcal{A}_s = \prod_{i \in I} \mathcal{A}_{o^{(i)}} / \sim$ however, since $\mathcal{A}_{o^{(i)}}$ depends on i and s then $\mathcal{A} \neq \prod_{i \in I} \mathcal{A}_{o^{(i)}} / \sim$.

The local transition kernel $T^{(i)}$ depends both on the global transition kernel and the policies of all the agents. Two possible choices of transitions depend on whether the agent interacts or not. In the non-interacting case $T_1(s, a) = s + a$. If agents may not occupy the same position then the transition rejects the action if the agent moving would put them in the same position; so $T_2(s, a) = s + a$ if $s + a$ is legal, otherwise $p^{(i)} \circ T_2(s, a) = o^{(i)}$ for some i. The simplest T_2 is to choose $T_2(s, a) = s$ if $s + a$ is illegal. In this case

$$T_2^{(i)}(o^{(i)}, a^{(i)}) = \mathbb{P}(s + a \text{ is legal} | o^{(i)}, a^{(i)}) \delta_{o^{(i)} + a^{(i)}} + \mathbb{P}(s + a \text{ is illegal} | o^{(i)}, a^{(i)}) \delta_{o^{(i)}}.$$

Clearly, $\mathbb{P}(s + a \text{ is legal} | o^{(i)}, a^{(i)})$ depends on the policies and positions of all the agents, then so does the local transition kernels $T_2^{(i)}$.

A non-negative measure μ on $\mathcal{S}$ is any function of the form $\mu(X) = \sum_{x \in X} f(x)$ with $f : \mathcal{S} \to \mathbb{R}_+$ any function. Defining the counting measure $\lambda(X) := \sum_{x \in X} 1 = \text{Card}(X)$ we have $\mu = f\lambda$ as measures on $\mathcal{S}$, or equivalently, $\frac{d\mu}{d\lambda} = f$. We may thus translate any reward or probability distribution on such a hypergrid as a measure.

A policy is a Markov kernel $\mathcal{S} \to \mathcal{A}$ such that $S \circ \pi = \text{Identity}$. More concretely, it means we have a function that associates to any state s a probability distribution on $\mathcal{A}$ with support on elements of the form (s, a) with $a \in \mathcal{A}_s$. From the description of measures, such a policy is fully described by a function $q : \mathcal{A} \to \mathbb{R}_+$ such that

$$\forall s \in \mathcal{S}, \sum_{a \in \mathcal{A}_s} q(s, a) = 1.$$

The policy is then $\pi(s) = \sum_{a \in \mathcal{A}_s} q(s, a) \delta_{(s,a)}$.

A GFlowNet on this hypergrid in reward-less notations is given by $(F_{\text{init}}, \pi^*, F^*_{\text{out}})$. Now, F_{init} is any measure on $\mathcal{S}$, it may be given by a pre-chosen family of categorical distribution of the finite set $\mathcal{S}$. For big W, D and I, since F_{init} have limited number of parameters, we may choose $F_{\text{init}} = C\delta_{s_1}$ for some s_1, and some trainable constant C. The star-policy is similar to π except that the STOP action is absent:

$$\pi^*(s) = \sum_{a \in \mathcal{A}_s \setminus \text{STOP}} q^*(s, a), \qquad Z(s) := \sum_{a' \in \mathcal{A}_s \setminus \text{STOP}} q(s, a').$$

Finally, F_{out} is measure and is thus of the form

$$F^*_{\text{out}}(X) := \sum_{x \in X} f^*_{\text{out}}(x)$$

for some function $f^*_{\text{out}} : \mathcal{S} \to \mathbb{R}_+$.

Standard notation GFlowNet is then recovered, given a reward $r : \mathcal{S} \to \mathbb{R}_+$, via:

- $R(X) = \sum_{x \in X} r(x)$;
- $F_{\text{out}}(X) = \sum_{x \in X} f_{\text{out}}(x)$ with $\forall s \in \mathcal{S}, f_{\text{out}}(s) = f^*_{\text{out}}(s) + r(s)$;
- $q(s, a) = \frac{f^*_{\text{out}}(s)}{f_{\text{out}}(s)} q^*(s, a)$ if $a \neq \text{STOP}$ and $q(s, \text{STOP}) = \frac{r(s)}{f_{\text{out}}(s)}$.

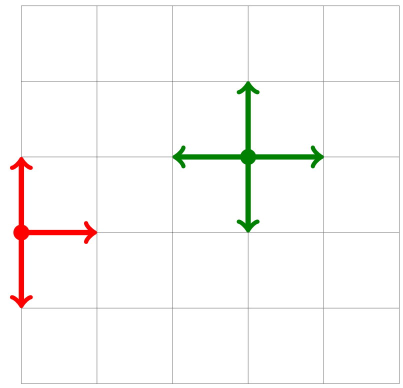


Figure 5: 2 agents on the 2D6W grid with available moves depicted.

A.3 Environment structures

We introduce first a hierarchy of single-agent environment structures.

- An action environment is a triplet $(\mathcal{S}, \mathcal{A}, S)$ with $\mathcal{A} \xrightarrow{S} \mathcal{S}$ a measurable map between measurable space is called of state space $\mathcal{S}$, action space $\mathcal{A}$ and State map S. We denote $\mathcal{A}_s := \{a \in \mathcal{A} \mid aS = s\}$.
- An interactive environment is a quadruple $(\mathcal{S}, \mathcal{A}, S, T)$ where $(\mathcal{S}, \mathcal{A}, S)$ is an action environment and $T : \mathcal{A} \to \mathcal{S}$ is a quasi-Markov kernel.
- A Game environment is a quintuple $(\mathcal{S}, \mathcal{A}, S, T, R)$ where $(\mathcal{S}, \mathcal{A}, S, T)$ is an interactive environment and R is a finite non-negative non-zero measure on $\mathcal{S}$. We may allow the reward to be stochastic so formally, R is allowed to be random measure instead [48].

For multi-agent environment, we have a similar hierarchy:

- A multi-agent action environment is a tuple $(\mathcal{S}, \mathcal{A}, S, \mathcal{O}^{(i)}, \mathcal{A}^{(i)}, S^{(i)}, p^{(i)})_{i \in I}$ with $(\mathcal{S}, \mathcal{A}, S)$ and each $(\mathcal{O}^{(i)}, \mathcal{A}^{(i)}, S^{(i)})$ being mono-agent action environments. Furthermore, we assume $\mathcal{S} = \prod_{i \in I} \mathcal{O}^{(i)}$ and $p^{(i)} : \mathcal{S} \to \mathcal{O}^{(i)}$ are the natural projection maps. Also
$$\forall s \in \mathcal{S}, \quad \mathcal{A}_s \setminus \{\text{STOP}\} = \prod_{i \in I} \left(\mathcal{A}^{(i)}_{p^{(i)}(s)} \setminus \{\text{STOP}\} \right).$$
- A multi-agent interactive environment is a tuple $(\mathcal{S}, \mathcal{A}, S, T, \mathcal{O}^{(i)}, \mathcal{A}^{(i)}, S^{(i)}, p^{(i)})_{i \in I}$ where $(\mathcal{S}, \mathcal{A}, S, \mathcal{O}^{(i)}, \mathcal{A}^{(i)}, S^{(i)}, p^{(i)})_{i \in I}$ is a multi-agent action environment and $(\mathcal{S}, \mathcal{A}, S, T)$ is a mono-agent interactive environment.
- A multi-agent game environment is a tuple $(\mathcal{S}, \mathcal{A}, S, T, R, \mathcal{O}^{(i)}, \mathcal{A}^{(i)}, S^{(i)}, p^{(i)})_{i \in I}$ such that $(\mathcal{S}, \mathcal{A}, S, T, \mathcal{O}^{(i)}, \mathcal{A}^{(i)}, S^{(i)}, p^{(i)})_{i \in I}$ is multi-agent interactive environment and $(\mathcal{S}, \mathcal{A}, S, T, R)$ is a mono-agent game environment.

A.4 GFlowNet in a Game Environment

A generative flow networks may be formally defined on an action environment $(\mathcal{S}, \mathcal{A}, S)$, as a triple $(\pi^*, F^*_{\text{out}}, F_{\text{init}})$ where $\pi^* : \mathcal{S} \to \mathcal{A}$ is a Markov kernel such that $\pi^* S = Id_{\mathcal{S}}$, F^*_{out} and F_{init} are a finite non-negative measures on $\mathcal{S}$. Furthermore, we assume that for all $s \in \mathcal{S}, \pi^*(s \to \text{STOP}_s) = 0$.

On an interactive environment $(\mathcal{S}, \mathcal{A}, S, T)$, given a GFlowNet $(\pi^*, F^*_{\text{out}}, F_{\text{init}})$, we define the ongoing flow as $F_{\text{in}} := F^*_{\text{out}} \pi^* T + F_{\text{init}}$ and the GFlowNet induces an virtual reward $\hat{R} := F_{\text{in}} - F^*_{\text{out}}$. Note that the virtual reward is always finite as the star-outflow and the initial flow are both finite and π^* and T are Markovian.

Definition 1 (Weak Flow-Matching Constraint)**.** *The weak flow-matching constraint is defined as*
$$\hat{R} \geq 0 \tag{7}$$

If the GFlowNet satisfies the weak flow-matching constraint, we may define a virtual GFlowNet policy as

$$\hat{\pi} := \frac{dF^*_{\text{out}}}{dF_{\text{in}}}\pi^* \tag{8}$$

where δ_{STOP} is the deterministic Markov kernel sending any s to STOP_s. The virtual action and edge flows are:

$$\hat{F}_{\text{action}} := F_{\text{in}} \otimes \hat{\pi} \in \mathcal{M}^+(\mathcal{S} \times \mathcal{A}); \tag{9}$$
$$\hat{F}_{\text{edge}} := F_{\text{in}} \otimes (\hat{\pi}T) \in \mathcal{M}^+(\mathcal{S} \times \mathcal{S}). \tag{10}$$

In a game environment, a GFlowNet comes with an outgoing flow, a natural policy, a natural action flow and a natural edge flow

$$F_{\text{out}} := F^*_{\text{out}} + R \tag{11}$$
$$\pi := \frac{dF^*_{\text{out}}}{dF_{\text{out}}}\pi^* \tag{12}$$
$$F_{\text{edge}} := F_{\text{out}} \otimes (\pi T) \in \mathcal{M}^+(\mathcal{S} \times \mathcal{S}) \tag{13}$$
$$F_{\text{action}} := F_{\text{out}} \otimes \pi \in \mathcal{M}^+(\mathcal{S} \times \mathcal{A}). \tag{14}$$

By abuse of notation we also write F_{action} (resp. $\hat{F}_{\text{action}}$) for $F_{\text{out}}\pi$ (resp. $F_{\text{in}}\pi$). and the flow-matching property may be rewritten as follows.

Definition 2 (Flow-Matching Constraint)**.** *The flow-matching constraint on a Game environment $(\mathcal{S}, \mathcal{A}, S, T, R)$ is defined as*

$$\hat{R} = \mathbb{E}(R). \tag{15}$$

Remark 1. *In an interactive environment $(\mathcal{S}, \mathcal{A}, S, T, \mathcal{O}^{(i)}, \mathcal{A}^{(i)}, S^{(i)}, p^{(i)})_{i\in I}$, a GFlowNet satisfying the weak flow-matching constraint satisfies the (strong) flow-matching constraint on the Game environment $(\mathcal{S}, \mathcal{A}, S, T, \hat{R}, \mathcal{O}^{(i)}, \mathcal{A}^{(i)}, S^{(i)}, p^{(i)})_{i\in I}$.*

We may recover part of the GFlowNets $(\pi^*, F^*_{\text{out}}, F_{\text{init}})$ from any of $F_{\text{action}}, \hat{F}_{\text{action}}$ as in general:

$$\pi^*(x \to A) = \frac{dF_{\text{action}}(\cdot \to A \setminus \text{STOP})}{dF_{\text{action}}(\cdot \to \mathcal{A} \setminus \text{STOP})} = \frac{d\hat{F}_{\text{action}}(\cdot \to A \setminus \text{STOP})}{d\hat{F}_{\text{action}}(\cdot \to \mathcal{A} \setminus \text{STOP})} \tag{16}$$
$$R = F_{\text{action}}(\cdot \to \text{STOP}) \qquad \hat{R} = \hat{F}_{\text{action}}(\cdot \to \text{STOP}) \tag{17}$$
$$F^*_{\text{out}} = F_{\text{action}}(\cdot \to \mathcal{A}) - R = \hat{F}_{\text{action}}(\cdot \to \mathcal{A}) - \hat{R} \tag{18}$$
$$F_{\text{init}} = F^*_{\text{out}}T + \hat{R} \tag{19}$$

If the flow-matching constraint is satisfied, then

$$F_{\text{init}} = F^*_{\text{out}}T + R. \tag{20}$$

Before going further, the presence densities.

Definition 3. *Let $\mathbb{F} = (\pi^*, F_{\text{out}}, F_{\text{init}})$ be a GFlowNet in an interactive environment $(\mathcal{S}, \mathcal{A}, S, T, \mathcal{O}^{(i)}, \mathcal{A}^{(i)}, S^{(i)}, p^{(i)})_{i\in I}$.*

The initial density of $\mathbb{F}$ is the probability distribution

$$\nu_{\mathbb{F},\text{init}} := \frac{1}{F_{\text{init}}(\mathcal{S})}F_{\text{init}}$$

The virtual presence density of $\mathbb{F}$ is the probability distribution $\hat{\nu}_{\mathbb{F}}$ defined by

$$\hat{\nu}_{\mathbb{F}} \propto \sum_{t=0}^{\infty} \nu_{\mathbb{F},\text{init}}\hat{\pi}^t.$$

The anticipated presence density of $\mathbb{F}$ is the probability distribution $\overline{\nu}_{\mathbb{F}}$ defined by

$$\overline{\nu}_{\mathbb{F}} := \frac{1}{F_{\text{in}}(\mathcal{S})}F_{\text{in}}.$$

In a game environment, the presence density of $\mathbb{F}$ is the probability distribution $\nu_{\mathbb{F}}$ defined by

$$\nu_{\mathbb{F}} \propto \sum_{t=0}^{\infty} \nu_{\mathbb{F},\text{init}}\pi^t.$$

Lemma 1. *Let $\mathbb{F}$ be a GFlowNet in an interactive environment satisfying the weak flow-matching constraint. If $\hat{\nu}_{\mathbb{F}} \gg \overline{\nu}_{\mathbb{F}}$, then $\hat{\nu}_{\mathbb{F}} = \overline{\nu}_{\mathbb{F}}$.*

Proof. Let $(\mathcal{S}, \mathcal{A}, S, T, \mathcal{O}^{(i)}, \mathcal{A}^{(i)}, S^{(i)}, p^{(i)})_{i \in I}$ be the interactive environment and let $\mathbb{F} = (\pi^*, F_{\text{out}}, F_{\text{init}})$. To begin with, $\mathbb{F}' := (\pi^*, F_{\text{init}}(\mathcal{S})\hat{\nu}_{\mathbb{F}} - \hat{R}, F_{\text{init}})$ is a GFlowNet satisfying the strong flow-matching constraint for reward $\hat{R}$, its edgeflow F'_{edge} may be compared to the edgeflow F_{edge} of $\mathbb{F}$: by Proposition 2 of [15], we have $F_{\text{edge}} \geq F'_{\text{edge}}$, and the difference $F_{\text{edge}} - F'_{\text{edge}}$ is a 0-flow in the sense this same article. Also, the domination hypothesis implies that $F'_{\text{edge}} \gg F_{\text{edge}} \gg F^0_{\text{edge}} := F_{\text{edge}} - F'_{\text{edge}}$. Since the edge-policy of F_{edge} is the same as that of F'_{edge} we deduce that it is also the same as F^0_{edge}. By the same Proposition 2, we have $F'_{\text{out}}\pi^t \xrightarrow{t \to +\infty} 0$, therefore, $\mu\pi^t \xrightarrow{t \to +\infty} 0$ for any $\mu \ll F'_{\text{out}}$. Again by domination, $F'_{\text{edge}} \gg F^0_{\text{edge}}$ we deduce that $F'_{\text{out}} \gg F^0_{\text{out}}$. Therefore, $F^0_{\text{out}}\pi^t \xrightarrow{t \to +\infty} 0$. Finally, since $\mathbb{F}^0$ is a 0-flow, $F^0_{\text{out}}\pi = F^0_{\text{out}}$, we deduce that $F^0_{\text{out}} = 0$ and thus $F_{\text{edge}} = F'_{\text{edge}}$ ie $\hat{\nu}_{\mathbb{F}} = \overline{\nu}_{\mathbb{F}}$. □

Remark 2. *As long as the GFlowNets considered are trained using an FM-loss on a training training distribution ν_{state} extracted from trajectory distributions $\hat{\nu}_{\mathbb{F}}$ or $\nu_{\mathbb{F}}$ of the GFlowNets themselves, we may assume that $\hat{\nu}_{\mathbb{F}} \gg \overline{\nu}_{\mathbb{F}}$ as flows are only evaluated on a distribution dominated by $\nu_{\mathbb{F}}$. The singular part with respect to $\nu_{\mathbb{F}}$ is irrelevant for training purposes as well as inference purposes. Therefore, we may generally assume that $\hat{\nu} = \overline{\nu}$*

Remark 3. *The main interest of the virtual reward $\hat{R}$ is for cases where the reward is not accessible or expensive to compute. Since a GFlowNet satisfying the weak flow-matching property always satisfies the strong flow-matching property for the reward $\hat{R}$, the sampling Theorem usually applies to $\hat{R}$. Therefore, $\hat{R}$ may be used as a reward during inference instead of the true reward R so that we actually sample using the policy $\hat{\pi}$ instead of π.*

A.5 MA-GFlowNets in multi-agent environments (I): Preliminaries

To begin with, let us define a MA-GFlowNet on a multi-agent environment.

Definition 4. *An MA-GFlowNet on a multi-agent action environment is the data of a global GFlowNet $\mathbb{F}$ on $(\mathcal{S}, \mathcal{A}, S)$ and a collection of local GFlowNets $\mathbb{F}^{(i)}$ on $(\mathcal{O}^{(i)}, \mathcal{A}^{(i)}, S^{(i)})$ for $i \in I$.*

We give ourselves a multi-agent interactive environment $(\mathcal{S}, \mathcal{A}, S, T, \mathcal{O}^{(i)}, \mathcal{A}^{(i)}, S^{(i)}, p^{(i)})$. We wish to clarify the links between local and global GFlowNet.

- A priori, there the local GFlowNets are merely defined on an action environment, they lack both the local transition kernel $T^{(i)}$ and the reward $R^{(i)}$.
- Given a global GFlowNet, we wish to define local GFlowNets.
- Given a family of local GFlowNets, we wish to define a global GFlowNet.

For simplicity sake, for any GFlowNet $\mathbb{F}$ defined on an interactive environment satisfying the weak flow-matching constraint, we set $R = \hat{R}$ and apply remark 2 assume that $\hat{\nu}_{\mathbb{F}} = \overline{\nu}_{\mathbb{F}} = \nu_{\mathbb{F}}$.

Definition 5. *Let $(\mathcal{S}, \mathcal{A}, S, T, \mathcal{O}^{(i)}, \mathcal{A}^{(i)}, S^{(i)}, p^{(i)})$ be a multi-agent interactive environment and let $\mathbb{F} = (\pi^*, F^*_{\text{out}}, F_{\text{init}})$ be a GFlowNet on $(\mathcal{S}, \mathcal{A})$ satisfying the weak flow-matching constraint. We introduce the following:*

- *the local presence probability distribution $\nu^{(i)}_{\mathbb{F}} := \nu_{\mathbb{F}} p^{(i)}$;*
- *the measure map $o^{(i)} \mapsto \nu_{\mathbb{F}|o^{(i)}}$ as the disintegration of $\nu_{\mathbb{F}}$ by $p^{(i)}$*
- *the Markov kernel $\tilde{\pi}^{(i)} : \mathcal{O}^{(i)} \to \mathcal{A}$ by $\delta_{o^{(i)}}\tilde{\pi}^{(i)} := \nu_{\mathbb{F}|o^{(i)}}\pi$;*
- *the Markov kernel $\pi^{(i)} : \mathcal{O}^{(i)} \to \mathcal{A}^{(i)}$ by $\pi^{(i)} = \tilde{\pi}^{(i)} p^{(i)}$;*
- *the Markov kernel $T^{(i)} : \mathcal{A}^{(i)} \to \mathcal{O}^{(i)}$ by $T^{(i)} = S^{(i)}\tilde{\pi}^{(i)} T p^{(i)}$;*

The situation may be summarized by the following diagram:

$$\begin{tikzcd}
(\mathcal{S},\nu_{\mathbb{F}}) \arrow[r, shift left, "\pi"] \arrow[d, "p^{(i)}"] & (\mathcal{A},\nu_{\mathbb{F}}\pi) \arrow[l, shift left, "S"] \arrow[l, bend left, "T"] \arrow[d, "p(i)"] \\
(\mathcal{O}^{(i)},\nu_{\mathbb{F}}^{(i)}) \arrow[r, shift left, "\pi^{(i)}"] \arrow[ur, "\tilde{\pi}^{(i)}"'] & (\mathcal{A}^{(i)},\nu_{\mathbb{F}}^{(i)}\pi^{(i)}) \arrow[l, shift left, "S^{(i)}"] \arrow[l, bend left, "T^{(i)}"]
\end{tikzcd}$$

Before going further, we need to check that these definitions are somewhat consistent.

Lemma 2. *The following diagrams are commutative in the category of probability spaces.*

$$\begin{tikzcd}
(\mathcal{S},\nu_{\mathbb{F}}) \arrow[r, shift left, "\pi"] \arrow[d, "p^{(i)}"] & (\mathcal{A},\nu_{\mathbb{F}}\pi) \arrow[l, shift left, "S"] \arrow[d, "p(i)"] \\
(\mathcal{O}^{(i)},\nu_{\mathbb{F}}^{(i)}) \arrow[r, shift left, "\pi^{(i)}"] \arrow[ur, "\tilde{\pi}^{(i)}"'] & (\mathcal{A}^{(i)},\nu_{\mathbb{F}}^{(i)}\pi^{(i)}) \arrow[l, shift left, "S^{(i)}"]
\end{tikzcd}
\qquad
\begin{tikzcd}
(\mathcal{S},\nu_{\mathbb{F}}\pi T) \arrow[d, "p^{(i)}"] & (\mathcal{A},\nu_{\mathbb{F}}\pi) \arrow[l, bend left, "T"] \arrow[d, "p(i)"] \\
(\mathcal{O}^{(i)},\nu_{\mathbb{F}}^{(i)}\pi^{(i)}T^{(i)}) & (\mathcal{A}^{(i)},\nu_{\mathbb{F}}^{(i)}\pi^{(i)}) \arrow[l, bend left, "T^{(i)}"]
\end{tikzcd}$$

Proof. For the left diagram, with the definition chosen, we only need to check that $\nu_{\mathbb{F}}^{(i)}\tilde{\pi}^{(i)} = \nu_{\mathbb{F}}\pi$. For all $\varphi \in L^1(\mathcal{A}, \nu_{\mathbb{F}}\pi)$ we have

$$\begin{aligned}
\int_{s\in\mathcal{A}} \varphi(a)d(\nu_{\mathbb{F}}\pi)(a) &= \int_{s\in\mathcal{S}}\int_{a\in\mathcal{A}} \varphi(a)d\pi(s,a)d\nu_{\mathbb{F}}(s) \\
&= \int_{o^{(i)}\in\mathcal{O}^{(i)}}\int_{s\in(p^{(i)})^{-1}(o^{(i)})}\int_{a\in\mathcal{A}} \varphi(a)d\pi(s,a)d\nu_{\mathbb{F}|o^{(i)}}(s)d\nu_{\mathbb{F}}^{(i)}(o^{(i)}) \\
&= \int_{o^{(i)}\in\mathcal{O}^{(i)}}\int_{a\in\mathcal{A}} \varphi(a)d\tilde{\pi}^{(i)}(a)d\nu_{\mathbb{F}}^{(i)}(o^{(i)}) \\
&= \int_{a\in\mathcal{A}} \varphi(a)d(\nu_{\mathbb{F}}^{(i)}\tilde{\pi}^{(i)})(a).
\end{aligned}$$

For the right diagram, we need to check that $\nu_{\mathbb{F}}\pi p^{(i)} = \nu^{(i)}\pi^{(i)}$ and that $\nu_{\mathbb{F}}\pi T p^{(i)} = \nu_{\mathbb{F}}^{(i)}\pi^{(i)}T^{(i)}$. We already proved the first equality for the left diagram and for the second:

$$\nu_{\mathbb{F}}\pi p^{(i)}T^{(i)} := \nu_{\mathbb{F}}\underbrace{\pi p^{(i)}S^{(i)}}_{=p^{(i)}}\tilde{\pi}^{(i)}Tp^{(i)} = \underbrace{\nu_{\mathbb{F}}p^{(i)}}_{\nu_{\mathbb{F}}^{(i)}}\tilde{\pi}^{(i)}Tp^{(i)} = \nu_{\mathbb{F}}^{(i)}\pi^{(i)}T^{(i)}$$

□

We see that from a global GFlowNet, we may build local policies as well as local transition kernels. These policies and transitions are natural in the sense that of local the induced local agent policy an transition are exactly the one wed would have if the observations of the other agents were provided as a random external parameter. The local rewards are then stochastics depending on the state of the global GFlowNet.

A.6 MA-GFlowNets in multi-agent environments (II): from local to global

We would like to settle construction of global GFlowNet from local ones, key difficulties arise:

- the global distributions induce local ones but the coupling of the local distributions may be non trivial;

- the defining the star-outflow and initial flow requires to find proportionality constants

$$F_{\text{in}}(\mathcal{O}^{(i)}) \propto \nu_{\mathbb{F}}^{(i)} \qquad F_{\text{init}}^{(i)} \propto \nu_{\mathbb{F}^{(i)},\text{init}};$$

- The coupling of the local transition kernels $T^{(i)}$ and the global one is in general non-trivial.

We try to solve these issues by looking at the simplest coupling: independent local agents. Recall that $\mathcal{A}_s^* = \prod_{i\in I} \mathcal{A}_s^{(i),*}$ therefore, independent coupling means that $\pi^*(s \to \cdot) = \prod_{i\in I} \pi^{(i),*}(o^{(i)} \to \cdot)$. We may generalize this relation to a coupling of GFlowNets writing $F_{\text{action}}(\prod_{i\in I} O^{(i)} \to \prod_{i\in I} A^{(i)}) = \prod_{i\in I} F_{\text{action}}^{(i)}(O^{(i)} \to A^{(i)})$. We are led to following the definition:

Definition 6. *Let $(\mathcal{S}, \mathcal{A}, S, T, \mathcal{O}^{(i)}, \mathcal{A}^{(i)}, S^{(i)}, p^{(i)})$ be a multi-agent interactive environment and let $\mathbb{F} = (\pi^*, F_{\text{out}}^*, F_{\text{init}})$ be a global GFlowNet on it satisfying the weak flow-matching constraint. The GFlowNet $\mathbb{F}$ is said to be*

- *star-split if for some local GFlowNets $\mathbb{F}^{(i)}$ and $\forall A^{(i)} \subset \mathcal{A}^{(i)} \setminus \text{STOP}$ we have:*

$$F_{\text{action}}(\prod_{i\in I} A^{(i)}) = \prod_{i\in I} F_{\text{action}}^{(i)}(A^{(i)}). \tag{21}$$

- *strongly star-split if for some local GFlowNets $\mathbb{F}^{(i)}$ and $\forall A^{(i)}, B^{(i)} \subset \mathcal{O}^{(i)}$ we have:*

$$F_{\text{edge}}(\prod_{i\in I} A^{(i)} \to \prod_{i\in I} B^{(i)}) = \prod_{i\in I} F_{\text{edge}}^{(i)}(A^{(i)} \to B^{(i)}). \tag{22}$$

The local GFlowNets $\mathbb{F}^{(i)}$ are called the components of the global GFlowNet $\mathbb{F}$.

However we encounter an additional difficulty: what happens when an agent decides to stop the game ? Indeed, local agents have their own STOP action, we then have at least three behaviors.

1. Unilateral Stop: if any agent decides to stop, the game stops and reward is awarded.
2. Asynchronous Unanimous Stop: if an agent decides to stop, it does not act anymore, waits for the other to leave the game and then reward is awarded only when all agents stopped.
3. Synchronous Unanimous Stop: if an agent decides to stop but some other does not, then the stop action is rejected and the agent plays a non-stopping action.

Similar variations may be considered for how the initialization of agents:

1. Asynchronous Start: the game has a free number of player, agents may enter the game while other are already playing.
2. Synchronous Start: the game has a fixed number of players, and agents all start at the same time.

These 6 possible combinaisons leads to slight variations on the formalization of MA-GFlowNets from local GFlowNets.

A.7 Initial local-global consistencies

Let us formalize Asynchronous and Synchronous starts. In synchronous case, the agents are initially distributed according to their own initial distributions and independently. Therefore, ν_{init} is a product and

$$F_{\text{init}} \propto \nu_{\text{init}} = \prod_{i\in I} \nu_{\text{init}}^{(i)} \propto \prod_{i\in I} F_{\text{init}}^{(i)}.$$

Also, by strong star-splitting property, $F_{\text{in}}^* = \prod_{i\in I} F_{\text{in}}^{(i),*}$. By $F_{\text{in}} = F_{\text{init}} + F_{\text{in}}^*$ we obtain the definition below.

Definition 7. *A strongly star-split global GFlowNet is said to have Synchronous start if*

$$F_{\text{in}} = \prod_{i\in I} F_{\text{init}}^{(i)} + \prod_{i\in I} F_{\text{in}}^{(i),*}$$

On the other hand, in the asynchronous case, an incoming agent may "bind" to agent arriving at the same time and other already there hence, the initial flow is a combination of any of the products

$$F_{\text{init}} = \sum \prod_{i\in\{\text{incoming}\}} F_{\text{init}}^{(i)} \prod_{j\in\{\text{already in}\}} F_{\text{init}}^{(j),*} = \prod_{i\in I}(F_{\text{init}}^{(i)} + F_{\text{in}}^{(i),*}) - \prod_{i\in I} F_{\text{in}}^{(i),*}.$$

Definition 8. *A strongly star-split global GFlowNet is said to have Asynchronous start if*

$$F_{\text{in}} = \prod_{i\in I}(F_{\text{init}}^{(i)} + F_{\text{in}}^{(i),*}).$$

A.8 Terminal local-global consistencies

We focus on terminal behaviors 1 and 2 which we formalize as follows. Local-global consistency consists in describing the formal structure linking local environments with global ones. The product structure of the action space is slightly different depending on the terminal behavior. It happens that we may up to formalization, we may cast Asynchronous Unanimous STOP as a particular case of Unilateral STOP local-global consistency. More precisely:

Definition 9 (Unilateral STOP Local-Global Consistency)**.** *With the same notations as above, we say that a multi-agent action environment has unilateral STOP if*

$$\mathcal{A}_s := \left(\prod_{i\in I} \mathcal{A}_{o^{(i)}}\right) / \sim \qquad a_1 \sim a_2 \Leftrightarrow \exists i, j \in I, a_1^{(i)} = \text{STOP}^{(i)}, a_2^{(j)} = \text{STOP}^{(j)}. \tag{23}$$

Definition 10 (Asynchronous Unanimous STOP Local-Global Consistency)**.** *With the same notations as above, we say that a multi-agent game environment has Asynchronous Unanimous STOP if is has Unilateral STOP and the observation space $\mathcal{O}^{(i)}$ may be decomposed into $\mathcal{O}^{(i)} = \mathcal{O}^{(i)}_{\textit{life}} \cup \mathcal{O}^{(i)}_{\textit{purgatory}}$ and for any observation $o^{(i)} \in \mathcal{O}^{(i)}_{\textit{life}}$ we have some $\tilde{o}^{(i)} \in \mathcal{O}^{(i)}_{\textit{purgatory}}$ such that :*

$$\begin{tikzcd} o^{(i)} \arrow[r] \arrow[rr, bend right, "0", "\text{STOP}^{(i)}"'] & \tilde{o}^{(i)} \arrow[loop above, "\varepsilon"] \arrow[r, "R^{(i)}(\tilde{o}^{(i)})", "\text{STOP}^{(i)}"'] & s_f \end{tikzcd}$$

where the value on top of arrows are constrained flow values.

The formal definition of Unilateral STOP is straightforward as any local STOP activates the global STOP so that any combination of local actions that contains at least one STOP is actually a global STOP. The quotient by the equivalence relation formalizes this property. Regarding Asynchronous Unanimous STOP, the chosen formalization allows to store the last observation of an agent while it is put on hold until global STOP. Indeed, a standard action ($\neq$ STOP) is invoked to enter purgatory, the reward is supported on purgatory and as long as all the agent are not in purgatory its value is zero (recall that from the viewpoint of a given agent, $R^{(i)}$ is stochastic but in fact depends on the whole global state). The local STOP action is then never technically called on an "alive" observation, once in purgatory the ε self-transition is called by default as long as the reward is non zero, hence until all agents are in purgatory. When the reward is activated, the policy at a purgatory state $\tilde{o}^{(i)}$ is then $\frac{d\varepsilon}{d(\varepsilon+R^{(i)})}\delta_{\tilde{o}^{(i)}} + \frac{dR^{(i)}}{d(\varepsilon+R^{(i)})}\delta_{\text{STOP}}$. As $\varepsilon \to 0^+$, the policy becomes equivalent to "if reward then STOP, else WAIT". This behavior is exactly the informal description of Asynchronous Unanimous STOP, the formalization is rather arbitrary and does not limit the applicability as it simply helps deriving formulas more easily.

We now prove Theorem 2 and 3, which have been integrated into the following theorem:

Theorem 4. *Let $(\mathcal{S}, \mathcal{A}, S, T, \mathcal{O}^{(i)}, \mathcal{A}^{(i)}, S^{(i)}, p^{(i)})$ be a multi-agent interactive environment. Let $\mathbb{F}^{(i)}$ be non-zero GFlowNets on $(\mathcal{O}^{(i)}, \mathcal{A}^{(i)}, S^{(i)})$ for $i \in I$ satisfying the weak flow-matching constraint, then there exists a transition kernel $\tilde{T}$ and a star-split GFlowNet on $(\mathcal{S}, \mathcal{A}, S, \tilde{T}, \mathcal{O}^{(i)}, \mathcal{A}^{(i)}, S^{(i)}, p^{(i)})$ whose components are the $\mathbb{F}^{(i)}$.*

Furthermore,

- *if the multi-agent environment is a game environment with Asynchronous Unanimous STOP and if the global GFlowNet satisfies the strong flow-matching constraint on* $\prod_{i\in I}\mathcal{O}^{(i)}_{\mathrm{life}}$ *then each local GFlowNet satisfies the strong flow-matching constraint on* $\mathcal{O}^{(i)}_{\mathrm{life}}$*;*
- *if the multi-agent environment is a game environment with Asynchronous Unanimous STOP and if each local GFlowNets satisfy the strong flow-matching constraint on* $\mathcal{O}^{(i)}_{\mathrm{life}}$ *then* $\hat{R}=\prod_{i\in I}\hat{R}^{(i)}$.

Proof. We simply define $\mathbb{F}=(\pi^*, F^*_{\mathrm{out}}, F_{\mathrm{init}})$ by $\pi^*(s):=(\prod_{i\in I}\pi^{(i),*}(o^{(i)}))/\sim$ ie the projection on $\mathcal{A}$ of the policy toward $\prod_{i\in I}\mathcal{A}^{(i)}$, then F^*_{out} as the product of the measures $F^{(i),*}_{\mathrm{out}}$. Then we define $\tilde{T}=\prod_{i\in I}T^{(i)}$ so that $F^*_{\mathrm{in}}(\prod_{i\in I}A^{(i)})=\prod_{i\in I}F^{(i),*}_{\mathrm{in}}(A^{(i)})$ and $F_{\mathrm{init}}:=\prod_{i\in I}(F^{(i),*}_{\mathrm{in}}+F^{(i)}_{\mathrm{init}})-\prod_{i\in I}F^{(i),*}_{\mathrm{in}}$ as the product measure of the $F^{(i)}_{\mathrm{init}}$. By construction this GFlowNet is star-split.

Assume that $\mathbb{F}$ satisfies the strong flow-matching constraint. It follows that for any $A^{(i)}\subset\mathcal{O}^{(i)}_{\mathrm{life}}$ we have

$$\prod_{i\in I}F^{(i)}_{\mathrm{in}}(A^{(i)})=\prod_{i\in I}F^{(i)}_{\mathrm{out}}(A^{(i)})=\prod_{i\in I}F^{(i),*}_{\mathrm{out}}(A^{(i)}).$$

Since, by assumption, all local GFlowNets satisfy the weak flow-matching constraint, all terms in the left-hand side product are bigger than those in the right-hand side product. Equality may only occur if some term is zero on both sides or if for all $i\in I$, We conclude that the strong flow-matching constraint is satisfied for all local GFlowNets on $\mathcal{O}^{(i)}_{\mathrm{life}}$.

If the strong flow-matching constraint is satisfied on $\mathcal{O}^{(i)}_{\mathrm{life}}$, then $\hat{R}^{(i)}=R^{(i)}=0$ on $\mathcal{O}^{(i)}_{\mathrm{life}}$. By construction, $F^{(i),*}_{\mathrm{out}}=F^{(i),*}_{\mathrm{init}}=0$ on $\mathcal{O}^{(i)}_{\mathrm{purgatory}}$. Therefore, on purgatory, we have

$$\hat{R}=F_{\mathrm{in}}-F_{\mathrm{out}}=F^*_{\mathrm{in}}-F^*_{\mathrm{out}}=\prod_{i\in I}F^{(i),*}_{\mathrm{in}}-\prod_{i\in I}F^{(i),*}_{\mathrm{out}}=\prod_{i\in I}F^{(i),*}_{\mathrm{in}}=\prod_{i\in I}\hat{R}^{(i)}.$$

□

B Algorithms

Algorithm 3 shows the training phase of the independent flow network (IFN). In the each round of IFN, the agents first sample trajectories with policy

$$o^{(i)}_t=p_i(s^{(i)}_t)\text{ and }\pi^{(i)}(o^{(i)}_t\to a^{(i)}_t),\quad i\in I \tag{24}$$

with $a_t=(a^{(i)}_t:i\in I)$ and $s_{t+1}=T(s_t,a_t)$. Then we train the sampling policy by minimizing the FM loss $\mathcal{L}^{\mathrm{stable}}_{\mathrm{FM}}(\mathbb{F}^{(i),\theta})$ for $i\in I$.

Algorithm 3 Independent Flow Network Training Algorithm for MA-GFlowNets

Require: Number of agents N, A multi-agent environment $(\mathcal{S},\mathcal{A},\mathcal{O}^{(i)},\mathcal{A}^{(i)},p_i,S,T,R)$.
Require: Local GFlowNets $(\pi^{(i),*},F^{(i),*}_{\mathrm{out}},F^{(i)}_{\mathrm{init}})_{i\in I}$ parameterized by θ.
while not converged **do**
  Sample and add trajectories $(s_t)_{t\geq 0}\in\mathcal{T}$ to replay buffer with policy according to (24)
  Generate training distribution of observations $\nu^{(i)}_{\mathrm{state}}$ for $i\in I$ from train buffer
  Apply minimization step of FM-loss $\mathcal{L}^{\mathrm{stable}}_{\mathrm{FM}}(F^{(i),\theta}_{\mathrm{action}},R^{(i)})$ for $i\in I$.
end while

Algorithm 4 shows the training phase of Conditioned Joint Flow Network (CJFN). In the each round of CJFN, we first sample sample trajectories with policy

$$o^{(i)}_t=p_i(s^{(i)}_t)\text{ and }\pi^{(i)}_\omega(o^{(i)}_t\to a^{(i)}_t),\quad i\in I \tag{25}$$

with $a_t=(a^{(i)}_t:i\in I)$ and $s_{t+1}=T(s_t,a_t)$. Then we train the sampling policy by minimizing the FM loss $\mathbb{E}_\omega\mathcal{L}^{\mathrm{stable}}_{\mathrm{FM}}(F^{\theta,\mathrm{joint}}_{\mathrm{action}}(\cdot;\omega),R)$.

Algorithm 4 Conditioned Joint Flow Network Training Algorithm for MA-GFlowNets

Require: Number of agents N, A multi-agent environment $(\mathcal{S}, \mathcal{A}, \mathcal{O}^{(i)}, \mathcal{A}^{(i)}, p_i, S, T, R)$.
Require: Simple Random distribution $(\Omega, \mathbb{P})$
Require: Local GFlowNets $(\pi^{(i),*}, F_{\text{out}}^{(i),*}, F_{\text{init}}^{(i)})_{i\in I}$ parameterized by θ and $\omega \in \Omega$.
  while not converged **do**
    Sample $\omega_1, \cdots, \omega_b \sim \mathbb{P}$ and then trajectories $(s_t^{\omega})_{t\geq 0} \in \mathcal{T}$ to replay buffer with policy according to (25) for $\omega \in \{\omega_1, \cdots, \omega_b\}$
    Generate training distribution of states/omega $\nu_{\text{state}}^{\Omega}$ from the train buffer
    Apply minimization step of the FM loss $\mathbb{E}_{\omega}\mathcal{L}_{\text{FM}}^{\text{stable}}(\mathbb{F}^{\theta,\text{joint}}(\cdot;\omega))$ under the constraint of Weak flow-matching.
  end while

C Discussion: Relationship with MARL

Interestingly, there are similar independent execution algorithms in the multi-agent reinforcement learning scheme. Therefore, in this subsection, we discuss the relationship between flow conservation networks and multi-agent RL. The value decomposition approach has been widely used in multi-agent RL based on IGM conditions, such as VDN and QMIX. For a given global state s and joint action a, the IGM condition asserts the consistency between joint and local greedy action selections in the joint action-value $Q_{\text{tot}}(s, a)$ and individual action values $[Q_i(o_i, a_i)]_{i=1}^k$:

$$\arg\max_{a\in\mathcal{A}} Q_{\text{tot}}(s,a) = \left(\arg\max_{a_1\in\mathcal{A}_1} Q_1(o_1,a_1), \cdots, \arg\max_{a_k\in\mathcal{A}_k} Q_k(o_k,a_k)\right), \forall s \in \mathcal{S}. \tag{26}$$

assumption 1. *For any complete trajectory in an MADAG $\tau = (s_0, ..., s_f)$, we assume that $Q_{tot}^{\mu}(s_{f-1}, a) = R(s_f)f(s_{f-1})$ with $f(s_n) = \prod_{t=0}^{n} \frac{1}{\mu(a|s_t)}$.*

Remark 1. *Although Assumption 1 is a strong assumption that does not always hold in practical environments. Here we only use this assumption for discussion analysis, which does not affect the performance of the proposed algorithms. A scenario where the assumption directly holds is that we sample actions according to a uniform distribution in a tree structure, i.e., $\mu(a|s) = 1/|\mathcal{A}(s)|$. The uniform policy is also used as an assumption in [2].*

Lemma 3. *Suppose Assumption 1 holds and the environment has a tree structure, based on Theorem 2 and IGM conditions we have:*
1) $Q_{tot}^{\mu}(s,a) = F(s,a)f(s)$;
2) $(\arg\max_{a_i} Q_i(o_i,a_i))_{i=1}^k = (\arg\max_{a_i} F_i(o_i,a_i))_{i=1}^k$.

Based on Assumption1, we have Lemma 3, which shows the connection between Theorem 2 and the IGM condition. This action-value function equivalence property helps us better understand the multi-flow network algorithms, especially showing the rationality of Theorem 2.

C.1 Proof of Lemma 3

Proof. The proof is an extension of that of Proposition 4 in [2]. For any (s, a) satisfies $s_f = T(s, a)$, we have $Q_{\text{tot}}^{\mu}(s, a) = R(s_f)f(s)$ and $F(s, a) = R(s_f)$. Therefore, we have $Q_{\text{tot}}^{\mu}(s, a) = F(s, a)f(s)$. Then, for each non-final node s', based on the action-value function in terms of the action-value at the next step, we have by induction:

$$\begin{aligned} Q_{\text{tot}}^{\mu}(s,a) &= \hat{R}(s') + \mu(a|s') \sum_{a'\in\mathcal{A}(s')} Q_{\text{tot}}^{\mu}(s',a';\hat{R}) \\ &\overset{(a)}{=} 0 + \mu(a|s') \sum_{a'\in\mathcal{A}(s')} F(s',a';R)f(s'), \end{aligned} \tag{27}$$

where $\hat{R}(s')$ is the reward of $Q_{\text{tot}}^{\mu}(s,a)$ and (a) is due to that $\hat{R}(s') = 0$ if s' is not a final state. Since the environment has a tree structure, we have

$$F(s,a) = \sum_{a'\in\mathcal{A}(s')} F(s',a'), \tag{28}$$

which yields

$$Q^{\mu}_{\text{tot}}(s,a) = \mu(a|s')F(s,a)f(s') = \mu(a|s')F(s,a)f(s)\frac{1}{\mu(a|s')} = F(s,a)f(s).$$

According to Theorem 2, we have $F(s_t, a_t) = \prod_i F_i(o^i_t, a^i_t)$, yielding

$$\begin{aligned}\arg\max_a Q_{\text{tot}}(s,a) &\overset{(a)}{=} \arg\max_a \log F(s,a)f(s)\\ &\overset{(b)}{=} \arg\max_a \sum_{i=1}^{k} \log F_i(o_i,a_i)\\ &\overset{(c)}{=} \left(\arg\max_{a_1\in\mathcal{A}_i} F_1(o_1,a_1),\cdots,\arg\max_{a_k\in\mathcal{A}_k} F_k(o_k,a_k)\right),\end{aligned} \tag{29}$$

where (a) is based on the fact F and $f(s)$ are positive, (b) is due to Theorem 2. Combining with the IGM condition

$$\arg\max_{a\in\mathcal{A}} Q_{\text{tot}}(s,a) = \left(\arg\max_{a_1\in\mathcal{A}_1} Q_1(o_1,a_1),\cdots,\arg\max_{a_k\in\mathcal{A}_k} Q_k(o_k,a_k)\right), \forall s\in\mathcal{S}. \tag{30}$$

we can conclude that

$$\left(\arg\max_{a_i\in\mathcal{A}_i} F_i(o_i,a_i)\right)_{i=1}^{k} = \left(\arg\max_{a_1\in\mathcal{A}_1} Q_i(o_i,a_i)\right)_{i=1}^{k}.$$

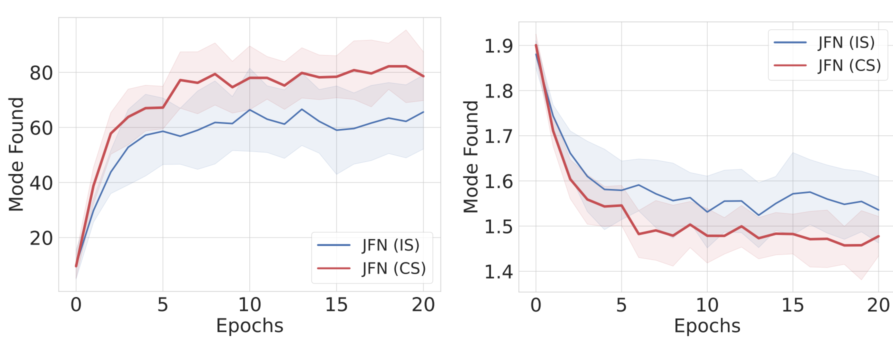
Then we complete the proof. □

D Additional Experiments

D.1 Hyper-Grid Environment

D.1.1 Effect of Sampling Method:

We consider two different sampling methods of JFN; the first one is to sample trajectories using the flow function F_i of each agent independently, called JFN (IS), and the other one is to combine the policies π_i of all agents to obtain a joint policy π, and then performed centralized sampling, named JFN (CS). As shown in Figure 6, we found that the JFN (CS) method has better performance than JFN (IS) because the error of the policy π estimated by the combination method is smaller, and several better samples can be obtained during the training process. However, the JFN (IS) method can achieve decentralized sampling, which is more in line with practical applications.



(a) Mode Found (b) L1-Error

Figure 6: The performance of JFN with different methods.

D.1.2 Effect of Different Rewards:

We study the effect of different rewards in Figure 7. In particular, we set $R_0 = \{10^{-1}, 10^{-2}, 10^{-4}\}$ for different task challenge. A smaller value of R_0 makes the reward function distribution more sparse, which makes policy optimization more difficult [2, 49, 50]. As shown in Figure 7, we found that our proposed method is robust with the cases $R_0 = 10^{-1}$ and $R_0 = 10^{-2}$. When the reward distribution becomes sparse, the performance of the proposed algorithm degrades slightly.

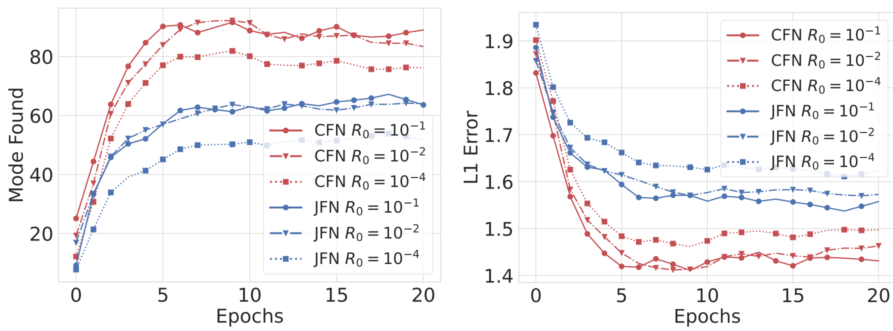


Figure 7: The effect of different reward R_0 on different algorithm.

D.1.3 Flow Match Loss Function:

Figure 8 shows the curve of the flow matching loss function with the number of training steps. The loss of our proposed algorithm gradually decreases, ensuring the stability of the learning process. For some RL algorithms based on the state-action value function estimation, the loss usually oscillates. This may be because RL-based methods use experience replay buffer and the transition data distribution is not stable enough. The method we propose uses an on-policy based optimization method, and the data distribution changes with the current sampling policy, hence the loss function is relatively stable. Then we present the experimental details on the Hyper-Grid environments. We set the same number of training steps for all algorithms for a fair comparison. Moreover, we list the key hyperparameters of the different algorithms in Tables 3-7.

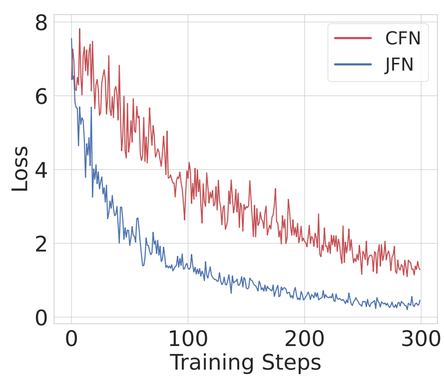


Figure 8: The flow matching loss of different algorithm.

In addition, as shown in Table 2, both the reinforcement learning methods and our proposed method can achieve the highest reward, but the average reward of reinforcement learning is slightly better for all found modes. Our algorithms do not always have higher rewards compared to RL, which is reasonable since the goal of MA-GFlowNets is not to maximize rewards.

D.2 StarCraft

We present a visual analysis based on 3m with three identical entities attacking to win. All comparison experiments adopted PyMARL framework and used default experimental parameters. Figure 9 shows

Environment	MAPPO	MASAC	MCMC	CFN	JFN
Hyper-Grid v1	2.0	1.84	1.78	2.0	2.0
Hyper-Grid v2	1.90	1.76	1.70	1.85	1.85
Hyper-Grid v3	1.84	1.66	1.62	1.82	1.82

Table 2: The best reward found using different methods.

the decision results of different algorithms on the 3m map. It can be found that the proposed algorithm can obtain results under different reward distributions, that is, win at different costs. The costs of other algorithms are often the same, which shows that the proposed algorithm is suitable for scenarios with richer rewards. Figure 10 shows the performance of the different algorithms on 2s3z, which shows a similar conclusion that the algorithm based on GFlowNets may be difficult to get the best yield, but the goal is not to do this, but to fit the distribution better. Moreover, on StarCraft missions, we did not use a clear metric to indicate the diversity of different trajectories, mainly because the status of each entity includes multiple aspects, its movement range, health, opponent observation, etc., which can easily result in different trajectories, but these differences do not indicate a good fit for the reward distribution. As a result, it is not presented in the same way as Hyper-Grid and Simple-Spread. Therefore, we used a visual method to compare the results. The maximized reward-oriented algorithms such as QMIX will improve the reward by reducing the death of entities, while the GFlowNets method can better fit the distribution on the basis of guaranteeing higher rewards.

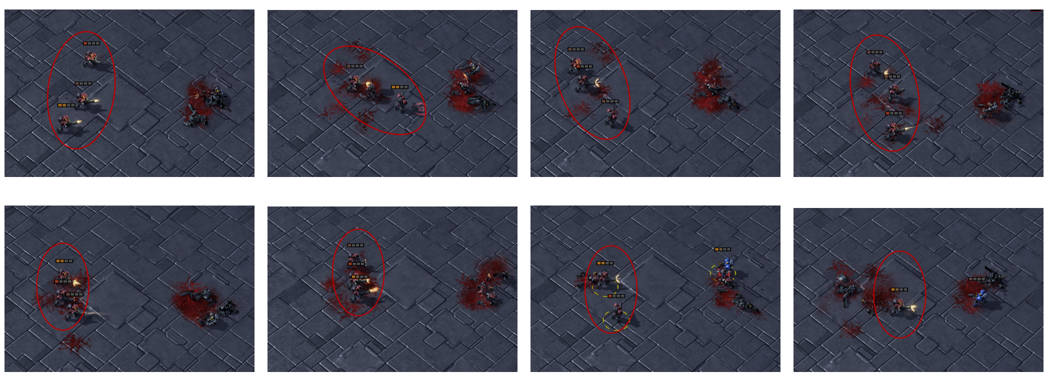


Figure 9: The sample results of different algorithm on 3m map. **Upper**: QMIX, **Bottom**: JFN

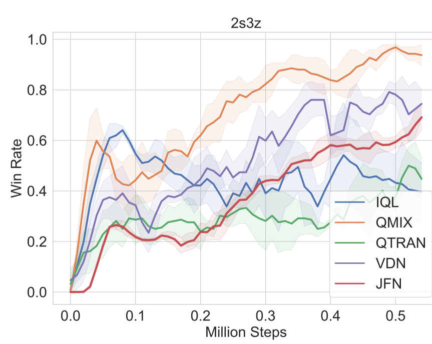


Figure 10: Average rate on 2s3z

D.3 Sparse-Simple-Spread Environment

In order to verify the performance of the CFN and JFN algorithms more extensively, we also conducted experiments on Simple-Spread in the multi-agent particle environment. We compared two classic

Multi-agent RL algorithms, QMIX [11] and MAPPO [40], which have achieved State-of-the-Art performance in the standard simple-spread environment. Since the decision-making problems solved by GFlowNets are usually the setting of discrete state-action space, we modified Simple-Spread to meet the above conditions and named it discrete Sparse-Simple-Spread. Specifically, we set the reward function such that if the agent arrives at or near a landmark, the agent will receive the highest or second-highest reward. And this reward is given to the agent only after each trajectory ends. In addition, we fix the speed of the agent to keep the state space discrete and all agents start from the origin.

We adopt the average return and the number of distinguishable trajectories as performance metrics. When calculating the average return, JFN and CFN select the action with the largest flow for testing. As shown in Figure 11-Left, although the MAPPO and QMIX algorithms converge faster than the JFN, the JFN eventually achieves comparable performance. The performance of JFN is better than that of the CFN algorithm, which also shows that the method of flow decomposition can better learn the flow F_i of each agent. In each test round, we collect 16 trajectories and calculate the number of trajectories, which can be accumulated for comparison. The number of different trajectories found by JFN is 4 times that of MAPPO in Figure 11-Right, which shows that MA-GFlowNets can obtain more diverse results by sampling with the flow function. Moreover, the performance of JFN is not as good as that of the RL algorithm. This is because JFN lacks a guarantee for monotonic policy improvement [42, 43]. It pays more attention to exploration and does not fully use the learned policy, resulting in fewer high-return trajectories collected. MAPPO finds more high-return trajectories in Figure 11-Right, but it still struggles to generate more diverse results. In each sampling process, the trajectories found by MAPPO are mostly the same, while JFN does better.

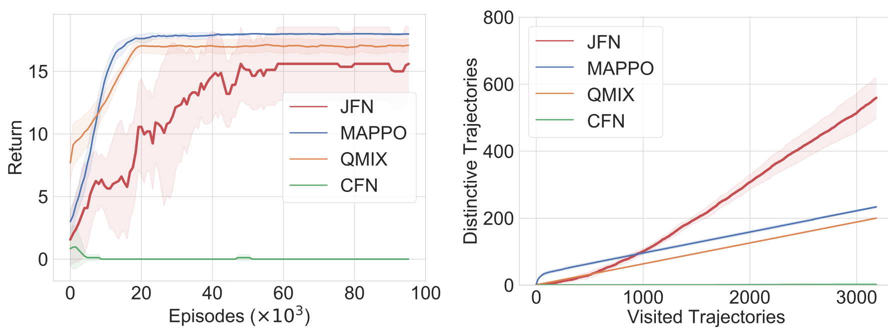


Figure 11: Average return and the number of distinctive trajectories performance of different algorithms on Sparse-Simple-Spread environments.

Table 3: Hyper-parameter of MAPPO under different environments

	Hyper-Grid-v1	Hyper-Grid-v2	Hyper-Grid-v3
Train Steps	20000	20000	20000
Agent	2	2	3
Grid Dim	2	3	3
Grid Size	[8,8]	[8,8]	[8,8]
Actor Network Hidden Layers	[256,256]	[256,256]	[256,256]
Optimizer	Adam	Adam	Adam
Learning Rate	0.0001	0.0001	0.0001
Batchsize	64	64	64
Discount Factor	0.99	0.99	0.99
PPO Entropy	1e-1	1e-1	1e-1

A Technical Appendices and Supplementary Material

Technical appendices with additional results, figures, graphs and proofs may be submitted with the paper submission before the full submission deadline (see above), or as a separate PDF in the

Table 4: Hyper-parameter of MASAC under different environments

	Hyper-Grid-v1	Hyper-Grid-v2	Hyper-Grid-v3
Train Steps	20000	20000	20000
Grid Dim	2	3	3
Grid Size	[8,8]	[8,8]	[8,8]
Actor Network Hidden Layers	[256,256]	[256,256]	[256,256]
Critic Network Hidden Layers	[256,256]	[256,256]	[256,256]
Optimizer	Adam	Adam	Adam
Learning Rate	0.0001	0.0001	0.0001
Batchsize	64	64	64
Discount Factor	0.99	0.99	0.99
SAC Alpha	0.98	0.98	0.98
Target Network Update	0.001	0.001	0.001

Table 5: Hyper-parameter of JFN under different environments

	Hyper-Grid-v1	Hyper-Grid-v2	Hyper-Grid-v3
Train Steps	20000	20000	20000
R_2	2	2	2
R_1	0.5	0.5	0.5
Grid Dim	2	3	3
Grid Size	[8,8]	[8,8]	[8,8]
Trajectories per steps	16	16	16
Flow Network Hidden Layers	[256,256]	[256,256]	[256,256]
Optimizer	Adam	Adam	Adam
Learning Rate	0.0001	0.0001	0.0001
ϵ	0.0005	0.0005	0.0005

Table 6: Hyper-parameter of CJFN under different environments

	Hyper-Grid-v1	Hyper-Grid-v2	Hyper-Grid-v3
Train Steps	20000	20000	20000
R_2	2	2	2
R_1	0.5	0.5	0.5
Grid Dim	2	3	3
Grid Size	[8,8]	[8,8]	[8,8]
Trajectories per steps	16	16	16
Flow Network Hidden Layers	[256,256]	[256,256]	[256,256]
Optimizer	Adam	Adam	Adam
Learning Rate	0.0001	0.0001	0.0001
ϵ	0.0005	0.0005	0.0005
Number of ω	4	4	4

Table 7: Hyper-parameter of CFN under different environments

	Hyper-Grid-v1	Hyper-Grid-v2	Hyper-Grid-v3
Train Steps	20000	20000	20000
Trajectories per steps	16	16	16
R_2	2	2	2
R_1	0.5	0.5	0.5
Grid Dim	2	3	3
Grid Size	[8,8]	[8,8]	[8,8]
Flow Network Hidden Layers	[256,256]	[256,256]	[256,256]
Optimizer	Adam	Adam	Adam
Learning Rate	0.0001	0.0001	0.0001
ϵ	0.0005	0.0005	0.0005

ZIP file below before the supplementary material deadline. There is no page limit for the technical appendices.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer:[Yes]

   Justification: Section 1 provides the MA-GFN formulation, section 2 provides theoretical motivations for the Multi-agent loss, section 4 provides experimental support.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: They are discussed in section 5

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Comprehensive justifications and proofs are provided in appendix A and C

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The code is not disclosed but the pseudo-code is provided.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: Only pseudo-code and environment description are provided.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Only the Starcraft experiment requires particular Hyperparamter tuning effort due to the difference between the reward maximization objective and the GFlowNet diversity objective. Manual tuning was sufficient using standard reward temperature tuning method for similar GFlowNets training.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Standard deviations at 2-sigma are provided on most plots.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Even though details of hardware used are not provided, all experiments were conducted on consumer grade hardware. Moreover, the work focuses on relative performance between algorithms.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: All contributors to the work are accounted for, no non-public dataset, environment or code were used. The theoretical nature of the work does not exclude military applications or societal consequences, but they are not the main expected outcome.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: The work is mainly theoretical and would only help scaling existing applications.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: Theoretical work.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The starcraft asset is cited.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: No new asset are introduced.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.