This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

GFlowVLM: Enhancing Multi-step Reasoning in Vision-Language Models with Generative Flow Networks

Haoqiang Kang[1,2*] Enna Sachdeva[1] Piyush Gupta[1] Sangjae Bae[1] Kwonjoon Lee[1]
[1]Honda Research Institute USA; [2]University of California San Diego

Abstract

Vision-Language Models (VLMs) have recently shown promising advancements in sequential decision-making tasks through task-specific fine-tuning. However, common fine-tuning methods, such as Supervised Fine-Tuning (SFT) and Reinforcement Learning (RL) techniques like Proximal Policy Optimization (PPO), present notable limitations: SFT assumes Independent and Identically Distributed (IID) data, while PPO focuses on maximizing cumulative rewards. These limitations often restrict solution diversity and hinder generalization in multi-step reasoning tasks. To address these challenges, we introduce a novel framework, GFlowVLM, a framework that fine-tune VLMs using Generative Flow Networks (GFlowNets) to promote generation of diverse solutions for complex reasoning tasks. GFlowVLM models the environment as a non-Markovian decision process, allowing it to capture long-term dependencies essential for real-world applications. It takes observations and task descriptions as inputs to prompt chain-of-thought (CoT) reasoning which subsequently guides action selection. We use task based rewards to fine-tune VLM with GFlowNets. This approach enables VLMs to outperform prior fine-tuning methods, including SFT and RL. Empirical results demonstrate the effectiveness of GFlowVLM on complex tasks such as card games (NumberLine, BlackJack) and embodied planning tasks (ALFWorld), showing enhanced training efficiency, solution diversity, and stronger generalization capabilities across both in-distribution and out-of-distribution scenarios. Project page is available at https://mk322.github.io/gflowvlm/.

1. Introduction

Vision-Language Models (VLMs) have achieved remarkable results in generalized tasks such as image captioning and visual question answering [15, 20, 45]. However, they struggle with structured reasoning in sequential decision making tasks that require causal understanding [3], especially in long horizon planning for tasks such as embodied AI, where agent must capture long term dependencies.

Recent advancements in LLMs and VLMs demonstrate emergent reasoning capabilities by leveraging Chain-of-Thought (CoT) reasoning, that enhances decision-making in multi-step interactive environments [7, 25, 35]. Typically, these models are fine-tuned using specialized visual instruction-following datasets through Supervised Fine Tuning (SFT) methods [18, 20], without active interaction with the environment, or optimized through Reinforcement Learning (RL) approaches, such as Proximal Policy Optimization (PPO) [29, 41]. However, SFT approaches often limits generalization to unseen scenarios, as training relies on maximizing the likelihood over a limited, specialized dataset, thereby restricting diversity in the solution space [17]. Furthermore, RL methods like PPO tend to prioritize short-term rewards, which can hinder the model's ability to consider long-term outcomes. Consequently, limited exploration in these models may lead to the oversight of more optimal long-term strategies, resulting in suboptimal performance in complex tasks [44].

In contrast to traditional reinforcement learning (RL) methods, which focus on maximizing cumulative rewards [5, 6], Generative Flow Networks (GFlowNets) [1] train stochastic policies to sample diverse, high-reward sequences (e.g., token sequences) with probabilities proportional to a specified reward function $R(x)$ [2]. This approach samples sequences based on the reward function's distribution, enabling it to find a broader range of high-reward solutions beyond those typically identified by reward-maximizing techniques. Recent studies have applied GFlowNets to multi-step reasoning within the Large Language Models (LLMs) framework, demonstrating their effectiveness over maximum likelihood training and traditional reward-maximization methods [8, 33, 40]. However, these methods lack the multimodal capabilities which are crucial for embodied AI tasks requiring the integration of visual and textual information. Additionally, related works, such as FoR [40], relies on Markovian structures within this framework, which may fail to capture the long-range dependencies necessary for complex, real-world reasoning tasks.

*Work was done during Haoqiang Kang's internship at Honda Research Institute.

To address these limitations, we introduce GFlowVLM, a novel approach integrating GFlowNets with VLMs in an end-to-end fine-tuning framework. It explicitly models non-Markovian flows, enabling richer multimodal reasoning suitable for complex sequential decision-making. To our knowledge, GFlowVLM is the first to fuse GFlowNets with VLMs directly, addressing the distinct challenges posed by multimodal, sequential reasoning crucial in structured planning environments. Our approach initializes a policy with a pretrained VLM and fine-tunes it using GFlowNets, guiding VLMs toward structured reasoning processes that capture logical dependencies between successive states. By implicitly representing reasoning as a tree structure—where nodes correspond to states with prior actions and observations, and edges represent actions leading to the next state—GFlowVLM enhances efficient learning of diverse and complex reasoning sequences. Empirical results demonstrate that GFlowVLM outperforms standard fine-tuning techniques including SFT and RL methods including PPO, by enhancing structured multimodal reasoning capabilities.

Our main contributions are as follows:

- We introduce a novel framework that integrates GFlowNets with common-sense capabilities of VLMs for multi-step decision making tasks, enhancing their reasoning abilities. To the best of our knowledge, this is the first work to explore this integration.
- By fine-tuning VLMs with GFlowNets, we improve their capacity to handle complex reasoning tasks, enabling better exploration of reasoning paths, generating diverse solutions, achieving stronger generalization to out of distribution tasks.
- Through extensive experimentation, we demonstrate that our framework achieves better training efficiency, higher success rate and diversity in solution generation tasks compared to existing methods.

2. Related Works

Multi-Step Reasoning with Vision Language Models Recent research has advanced the reasoning capabilities of large foundation models through specialized prompting techniques [4, 28, 34–37, 39] and fine-tuning methods [3, 25, 32] that often add MLP or transformer layers to frozen models to interface with action spaces. Reinforcement learning from human feedback (RLHF) also aids in developing reward models [26]. The RL4VLM approach [41] uses PPO to train VLMs but lacks the structured reasoning enabled by our GFlowNets method, which is designed for deeper understanding of complex tasks. VLM reasoning in interactive environments, particularly embodied AI, has gained attention [25, 35, 38, 41], but our GFlowNets approach uniquely enables structured reasoning, enhancing task comprehension.

GFlowNets GFlowNets [1] were originally created to learn sampling policies from unnormalized distributions, primarily aiding scientific discovery by generating diverse, high-reward samples [10, 11, 30]. They have since been applied in recommendation systems [21], domain adaptation [46], combinatorial optimization [14, 42], and enhancing neural network interpretability [16]. GFlowNets also support sampling from complex posteriors [9], sparse reward RL [27], and multi-objective optimization [12]. Recent adaptations fine-tune LLMs for multi-step reasoning tasks [40], yet lack the multimodal capability for embodied AI planning, which we address in this paper.

3. Preliminaries

GFlowNets GFlowNets are models that amortize the cost of sampling from a target distribution over terminal states $\mathcal{X}$ by learning an approximation of this distribution based on its reward function. Given a directed acyclic graph (DAG) $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ with states $\mathcal{S}$ and directed actions $\mathcal{A}$, there is an initial state s_0 and terminal states $\mathcal{X} \subset \mathcal{S}$. A trajectory $\tau = (s_0 \to \ldots \to s_n = x)$ represents a complete sequence ending in a terminal state $x \in \mathcal{X}$. The trajectory flow $F : \mathcal{T} \to \mathbb{R}_{\geq 0}$ defines flows over trajectories, with state flow $F(s) = \sum_{s \in \tau} F(\tau)$. A forward policy $P_F(\cdot|s)$, often parametrized by a neural network, induces a distribution over trajectories and a marginal distribution over terminal states, with probabilities given by: $P_F(\tau) = P_F(s_0 \to \ldots \to s_n) = \prod_{t=0}^{n-1} P_F(s_{t+1}|s_t) \quad \forall \tau \in \mathcal{T}$. Similarly, a backward policy $P_B(\tau) = P_B(s_n \to \ldots \to s_0) = \prod_{t=0}^{n-1} P_B(s_t|s_{t+1}) \quad \forall \tau \in \mathcal{T}$. Given a non-negative reward function $R : \mathcal{X} \to \mathbb{R}_{\geq 0}$, GFlowNets aim to estimate a policy where the likelihood of sampling $x \in \mathcal{X}$ is proportional to $R(x)$. Thus, there exists a constant Z such that: $R(x) = Z \sum_{\tau=(s_0 \to \ldots \to s_n = x)} P_F(\tau) \quad \forall x \in \mathcal{X}$, where $Z = F(s_0) = \sum_{\tau \in \mathcal{T}} F(\tau)$ is total flow at the initial state. See Appendix A for more details.

Off-Policy Training An advantage of GFlowNets is their ability to leverage off-policy training data by reusing transitions from past trajectories to update the forward policy P_F[1]. Unlike on-policy reinforcement learning methods, GFlowNets handle diverse, multimodal distributions effectively, using off-policy samples to approximate $R(x)$. This approach improves sample efficiency and accelerates convergence, especially in settings where generating trajectories is costly or where leveraging prior data is beneficial.

3.1. Motivating Experiment

To demonstrate the limitations of traditional approaches and highlight the dependencies captured by GFlowNets, we design a toy experiment combining two types of numerical sequences: (i) the Fibonacci sequence, defined by $F(n) = F(n-1) + F(n-2)$, where each term is the sum

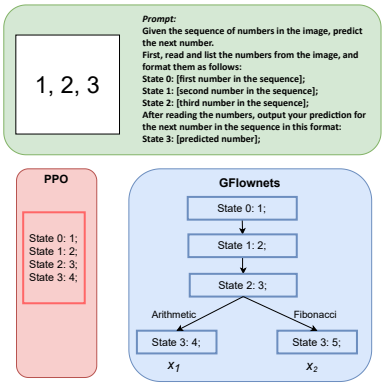


Figure 1. Overview of the prediction of diverse sequence using Gflownets as compared to PPO. The model takes the image of sequence and prompt as input, and generates the next number of sequence by implicitly modeling the causality. See Fig. 4 for a practical example.

Methods	Temp. α	SR (%)	# Solutions
w/o fine tuning	1	15.7	1.10
w/o fine tuning	1.2	16.1	1.12
SFT	1	21.7	1.03
SFT	1.2	22.0	1.09
PPO	1	50.2	1.13
PPO	1.2	49.8	1.15
GFlowVLM	1	**76.4**	**1.60**
GFlowVLM	1.2	**77.9**	**1.61**

Table 1. Results of motivating experiments. α denotes the temperature parameter of decoding.

of the previous two, and (ii) an arithmetic sequence with a constant increment, $S(n) = S(n-1) + k$, where k is a fixed step size (e.g., $k = 2$ for sequences like $[2, 4, 6, \dots]$). The task presents the model with an image of a partial sequence and a prompt (as shown in Fig. 1), to predict the next number in the sequence. We evaluate the performance of fine-tuning VLM (LLAVA-v1.6-Mistral-7B [19]) using SFT, PPO, and GFlowNets, with temperature parameters $\alpha = 1$ and $\alpha > 1$ to assess stochastic performance, as shown in Tab. 1. Success rate (SR), measured as the percentage of correct next-number predictions across 1,000 samples, shows GFlowVLM outperform PPO by 26% and generate 40% more diverse solutions. Compared to SFT, GFlowVLM achieves a 54% higher success rate and yield 59% more diversity in responses, underscoring their strength in learning and generalizing causal structures. This advantage stems from GFlowNets' ability to infer underlying causal reasoning structure of sequence by sequentially sampling reasoning paths, in contrast to the limited diversity observed with SFT and PPO. While this toy example highlights key conceptual benefits of GFlowNets, we include a practical example in Sec. A.2 of Supplementary Material demonstrating how GFlowVLM can be applied to embodied AI tasks in ALFWorld, showcasing its real-world reasoning capabilities. This addition provides further evidence of the method's utility beyond synthetic settings and illustrates its effectiveness in a more grounded, task-oriented scenario.

4. Methodology

This work utilizes GFlowNet's structure learning to enhance the VLM's ability to obtain high-quality, diverse solutions whose distribution is proportional to the reward function. By fine tuning VLMs using GFlowNets, it allows the solutions to be sampled from the distribution of the reward function, which prevents learning policies settled around a small number of modes. Fig. 2 shows the overall pipeline of our proposed framework. The model takes current observation image o_t and designed task specific prompt p_t as the input. p_t contains the description of the goal, history actions $a_{1:t-1}$, history states $s_{1:t-1}$ and admissible action space corresponding to the current observation o_t. To incorporate non-Markovian assumption, input $z_{0:t}$ include history actions $a_{0:t}$ and states $s_{0:t}$, respectively along with the input image o_t*. The desired output format includes the CoT reasoning c_t and action a_t, where a_t directly interacts with the environment.

4.1. VLM as a policy: Fine tuning VLMs using GFlowNets to estimate actions

We use a non-Markovian approach, essential for reasoning tasks that depend on multiple past states to capture long-term dependencies, and tackle longer sequences—challenges that the Markovian assumption cannot adequately address. We fine tune VLM of LLaVA [19] as a policy for structured reasoning, where VLM serves as the forward policy P_F, selecting the next action a_t that advance the reasoning chain at every step t. For each task $\mathcal{W}$, the model takes the visual observation o_t and prompt p_t as inputs, and outputs the CoT and action.

Prompt Design We retain the same prompt format as [41] for a fair comparison. However, to incorporate historical context in decision-making, we modify the prompt template to include the history of states and actions predicted by the VLM, as shown in Tab. 2. The textual prompt p_t contains the goal description g, the history of states $s_{0:t}$ and actions $a_{0:t}$, and the action space $\mathcal{A}_{t+1}$ available after interacting with the environment. For certain tasks q that may contain observation-dependent information, such as the textual description $d(o_{t+1})$ of the obser-

*Only the current input image o_t is used, as including every intermediate image would be computationally costly, and current VLMs do not perform optimally with multiple images as input.

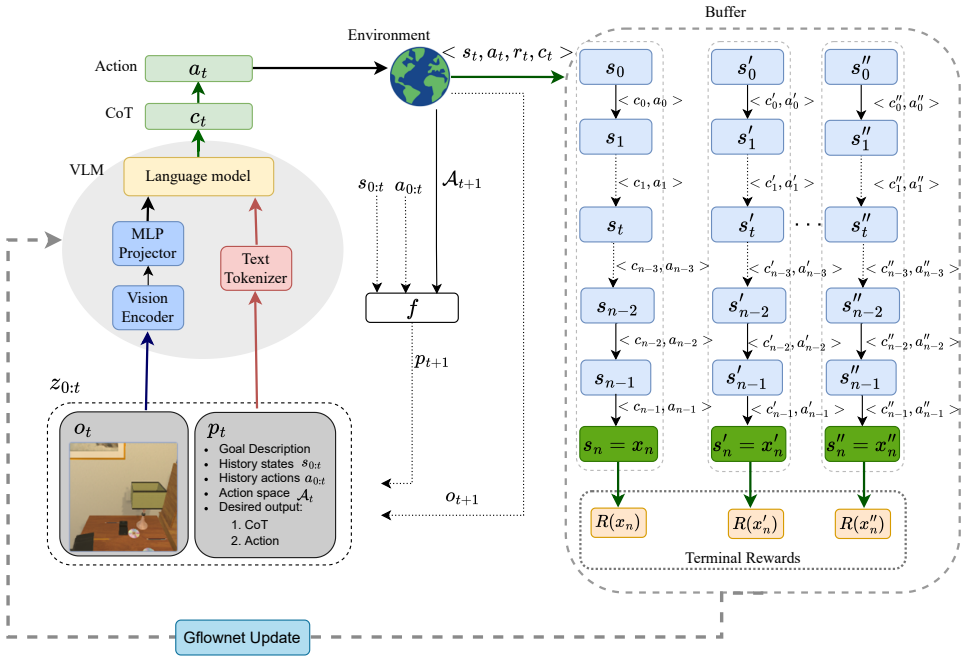


Figure 2. **Overall framework of proposed method:** The input $z_{0:t}$ at time step t consists of a visual observation o_t and an input prompt p_t containing goal description, history states $s_{0:t}$, history actions $a_{0:t}$, and admissible actions $\mathcal{A}_t$, and outputs CoT reasoning c_t, and action a_t. The a_t is executed in the environment to obtain reward $r_t(s_t, a_t)$, next observation o_{t+1}, and action space $\mathcal{A}_{t+1}$. f generates the next prompt p_{t+1} using description of next observation o_{t+1} (if applicable), history of states $s_{0:t}$ and actions $a_{0:t}$ and next admissible actions $\mathcal{A}_{t+1}$. This generates multiple trajectories. The transitions $< s_t, a_t, r_t, c_t >$, $< s'_t, a'_t, r'_t, c'_t >$ and $< s''_t, a''_t, r''_t, c''_t >$ across different trajectories are added to buffer to update the forward policy P_F using GFlowNets. $\{x, x', x''\} \in \mathcal{X}$ represent the terminal states of sequences. $R(x)$ represents the non-negative reward obtained from the environment (after reward shaping, if applicable) at terminal state x of a trajectory.

vation o_{t+1}, the function f generates the prompt p_{t+1} as: $p_{t+1} = f(d(o_{t+1}) \cdot \mathbb{I}_{\{\text{Task=q}\}}, s_{0:t}, a_{0:t}, \mathcal{A}_{t+1})$, where $\mathbb{I}$ is an indicator function which is 1 only for a certain task q if the observation-dependent information is available.

Action Selection Before selecting an action at each step t, we incorporate a CoT reasoning mechanism, where the model generates intermediate reasoning steps to guide the action selection process. At time t, the VLM first generates a reasoning CoT c_t, which includes a description of the image and intermediate thoughts. Since VLMs are pre-trained on large-scale image-caption data, CoT steps provide additional context and help the model explicitly consider dependencies between different states before selecting the next action. The CoT then guides the action selection. We define z_t as the structured state at time t in the trajectory, which includes the action a_t, environment state s_t, and visual observation o_t. More precisely, the trajectory $z_{0:t}$ is composed of the current visual observation o_t and an input prompt p_t that contains the goal description g, the history of environment states $s_{0:t-1}$, actions $a_{0:t-1}$, and the set of admissible actions $\mathcal{A}_t$. See Appendix C.1 for details.

The probabilities for the CoT and action sequences of tokens are defined as follows:

$$P_{\text{CoT}}(c_t | z_{0:t}, g; \theta) = \tag{1}$$

$$\prod_{j=1}^{n_c} P_{\text{VLM}}(w_j | w_{<j}, z_{0:t}, g; \theta) \tag{2}$$

$$P_{\text{Action}}(a_t | c_t, z_{0:t}, g; \theta) = \tag{3}$$

$$\prod_{i=1}^{n_a} P_{\text{VLM}}(w_i | w_{<i}, c_t, z_{0:t}, g; \theta) \tag{4}$$

where n_c and n_a represent the number of tokens in the CoT sequence c_t and action sequence a_t, respectively, and w_i represents the i-th text token in a sequence. Here, $P_{\text{VLM}}(w_i | w_{<i}, c_t, z_{0:t}, g; \theta)$ and $P_{\text{VLM}}(w_j | w_{<j}, z_{0:t}, g; \theta)$ denote the VLM's token-level likelihoods for the action and CoT sequences, conditioned on previous tokens, the history of states $z_{0:t}$, and goal description g. The log forward policy $\log P_F(z_{t+1} | z_{0:t}, g; \theta)$ is then computed as a weighted sum of the log probabilities of CoT tokens

$\log P_{\text{CoT}}(c_t|z_{0:t}, g; \theta)$, and the original log action probabilities $\log P_{\text{Action}}(a_t|z_{0:t}, g; \theta)$:

$$\log P_F(z_{t+1}|z_{0:t}, g; \theta) = \log P_{\text{Action}}(a_t|z_{0:t}, c_t, g; \theta) + \lambda \log P_{\text{CoT}}(c_t|z_{0:t}, g; \theta), \tag{5}$$

where $\lambda \in [0, 1]$ is a weighting factor that controls the influence of the CoT reasoning on the final action selection. The CoT probabilities $P_{\text{CoT}}(c_t|z_{0:t}, g; \theta)$ provide a structured, intermediate reasoning context that refines the decision-making process, ensuring that the final action is selected with consideration of both direct state information and the model's internal thought process. We perform an ablation study on the effect of λ, and we selected $\lambda = 0.4$ in our work, as discussed in the Sec. D.5 of Supplementary Material.

CoT prompt p_t for task $\mathcal{M}$
You are trying to solve a task $\mathcal{M}$. {Description of the task}. The action space of $\mathcal{M}$ is {all legal actions $a \in \mathcal{A}$}. Use `[DONE]` when you think you have completed the task.
Task: {Task description}
State 0: {Initial observation}
Action 0: {First action}
State 1: {Observation for step 1}
Admissible Next Actions: {"action1", "action2", " `[DONE]`" (if applicable)}

Your response should be a valid JSON file in the following format:
{ "thoughts": "first describe what you see in the image using the text description, then carefully think about which action to take to complete the task.",
"action": "an admissible action." }

Formatted text output
{ "thoughts": "Given the current state and previous steps, I should choose [a_t] as the next action.",
"action": "a_t" }

Table 2. A template showing the input prompt and corresponding output. The green text highlights the chain-of-thought reasoning which may contain task-specific descriptions, while the red text indicates the action based on the description.

4.2. Training Objectives

We adopt three different objective functions of GFlowNets, *Variance Trajectory-Balanced (TB)* [23], *Subtrajectory-Balanced (SubTB)* [22], and *Detailed-Balanced (DB)* [2], to finetune a VLM. We define z_t as the state in the trajectory sequence that includes both a_t, s_t, and o_t. See the Sec. C of the Supplementary Material for more details.

4.2.1. Variance Trajectory Balanced (Var-TB) Loss

The *Trajectory-Balanced* (TB) objective ensures that the probability of generating a complete trajectory $\tau = (z_0 \to z_1 \to \cdots \to z_n = x)$ is proportional to the reward $R(x)$. This objective is given by:

$$\mathcal{L}_{\text{VarTB}}(\boldsymbol{\tau}; \theta) = \frac{1}{K} \sum_{k=1}^{K} \left(\zeta(\tau_k; \theta) - \mathbb{E}_{\boldsymbol{\tau}}[\zeta(\boldsymbol{\tau}; \theta)] \right)^2, \tag{6}$$

where $\zeta(\cdot)$ is the estimated initial flow (see Eq. (13) in Supplementary Material for details), τ_k is k^{th} sampled trajectory during training, and K represents the total number of trajectories. This loss ensures that the high-reward trajectories are sampled more frequently by the policy.

4.2.2. Subtrajectory Balanced (SubTB) Loss

The *Subtrajectory-Balanced* (SubTB) loss operates on subtrajectories of the form $z_{0:m} = (z_0 \to z_1 \to \cdots \to z_m)$. It ensures that each segment of the reasoning path or structure remains consistent, where the flows are balanced locally between forward and backward transitions. We use a modified version of SubTB loss [8] as follows:

$$\mathcal{L}_{\text{SubTB}}(z_{0:m}, g; \theta) = \sum_{0 \le i < j \le m} \left(\log \frac{R(z_{0:i}\top) \prod_{k=i+1}^{j} P_F(z_k|z_{0:k-1}, g; \theta) P_F(\top|z_{0:j}, g; \theta)}{R(z_{0:j}\top) P_F(\top|z_{0:i}, g; \theta)} \right)^2 \tag{7}$$

where i and j are two time steps along a subtrajectory, and $\top$ is the `[DONE]` symbol to terminate the trajectory. $\top$ is generated similar to [8]. The reward $R(z_{0:i}\top)$ is computed as a cumulative reward given by the environment from step 0 to step i. This loss penalizes discrepancies in local transitions and ensures that all subsegments of a trajectory follow the correct balance conditions, reducing variance in smaller parts of the trajectory.

4.2.3. Detailed Balanced (DB) Loss

The *Detailed-Balanced* (DB) loss is used to ensure that each transition $z_t \to z_{t+1}$ between two states is balanced by matching the forward and backward flows at every step of the trajectory. Since it takes transition as input, we need dense rewards. The DB loss is formulated as:

$$\mathcal{L}_{\text{DB}}(z_{0:t} \to z_{t+1}, g; \theta) = \left(\log \frac{R(z_{0:t}\top) P_F(z_{t+1}|z_{0:t}, g; \theta) P_F(\top|z_{0:t+1}, g; \theta)}{R(z_{0:t+1}\top) P_F(\top|z_{0:t}, g; \theta)} \right)^2. \tag{8}$$

This loss ensures that every state-to-state transition follows the correct flow, preventing inconsistencies in the trajectory construction.

Remark One challenge when implementing both the SubTB and DB losses is accurately estimating the termination probability, $P_F(\top|z_{0:t}, g; \theta)$, which represents the likelihood of reaching a terminal state at any point in the trajectory. Incorrect estimation of this probability can lead to

Algorithm 1 Training VLM with GFlowNets

Input: An environment `env`, an initial VLM with parameters θ_0, a CoT reasoning scaling factor as λ, maximum episode length as T, number of tasks as W, number of collected trajectories per task as K.

for $w = 1, \ldots, W$ **do**
  $\mathcal{B}_w = \emptyset$
  for $k = 1, \ldots, K$ **do**
    $t = 0$
    $g, o_t, \mathcal{A}_t =$ `env.reset()`
    $p_t = f(o_t, \mathcal{A}_t)$
    while $t \leq T$ **do**
      $z_{0:t} = \langle o_t, p_t \rangle$
      $c_t, a_t = \arg\max P_F(z_{t+1} | z_{0:t}, g; \theta_{w-1})$
      $r_t, o_{t+1}, \mathcal{A}_{t+1} =$ `env.step`(a_t)
      $\mathcal{B}_w = \mathcal{B}_w \cup \{(s_t, c_t, a_t, r_t\}$
      $p_{t+1} = f\big(d(o_{t+1}) \cdot \mathbb{I}_{\{\mathrm{q}\}}, s_{0:t}, a_{0:t}, \mathcal{A}_{t+1}\big)$
      $t = t + 1$
      if $t = T$ **or** task w is completed **then**
        break
      end if
    end while
  end for
  Update θ_{w-1} on the collected trajectories $\mathcal{B}_w$ for task w to obtain θ_w
end for

Output: Updated parameters θ_W after W tasks.

suboptimal training and unbalanced flows. To address this, we introduce a new token, `[DONE]`, into the tokenizer to explicitly model the terminal state, and use distinct prompt designs as shown in the Tab. 2. Moreover, we perform an additional SFT step on correctly labeled examples before applying GFlowNets training. This initialization helps the model better estimate termination probabilities, resulting in improved overall performance (See ablation study in Sec. 6).

5. Experiments

We evaluate the performance of GFlowVLM on three distinct tasks that require multi-step visual-language reasoning. The Numberline and Blackjack tasks assess GFlowVLM's arithmetic reasoning capabilities with maximum steps set as 10, while Alfworld focuses on decision-making tasks that demand visual-semantic understanding with max steps set as 35 in a sequence. We mainly compare the performance with RL4VLM [41] and SFT methods. For fair comparison, we use the same base VLM of LLAVA-v1.6-Mistral-7B [19]. We conduct 4 independent runs with different random seeds, reporting mean and standard deviation. Episode success rate measures reasoning performance across tasks, while the diversity metric (Div@N) [40] quantifies unique correct solutions across N samples. The minimum for *Div@N* is 1. NumberLine and Blackjack have discrete negative rewards, but GFlowNets inherently do *not* support negative rewards (Sec. 3). Thus, we apply reward shaping as outlined in Sec. 5.2 on these two tasks.

5.1. Baselines

SFT We employ two versions of SFT in our baseline: SFT-w/o-`[DONE]` and SFT-w/-`[DONE]`. SFT-w/o-`[DONE]` uses the same GPT-4o dataset as in [41]. For SFT-w/-`[DONE]`, we include the `[DONE]` action in the training inputs and add correct examples where outputs explicitly contain `[DONE]`. We fine-tune the *LLaVA-1.6-7B* model on this dataset for 1 epoch using the official script. To ensure consistency, downstream GFlowNets training for both SubTB and DB losses starts from the same SFT checkpoint that includes `[DONE]`.

RL4VLM We compare with RL4VLM [41], which uses PPO to fine-tune the VLM. RL4VLM follows the same environment reward scheme as used in the GymCards tasks, where rewards are set to $[0, -1, 1]$. Additionally, it employs a Markovian approach by excluding history information from the prompt. To ensure a fair comparison, we modify the original setup with two additional configurations: one that replaces the default environment reward function with our custom reward function, and another that includes history information in the prompt in a non-Markovian manner. These adjustments allow us to evaluate the model's performance under different reward functions and prompt history settings.

5.2. Environments

Numberline. This task involves moving a current number "current: y_t" to a "target: c". The agent's goal is to align y_t with c by outputting an action a_t from the discrete set {"+", "-", `[DONE]`(if applicable)}. In-distribution examples include numbers from 0 to 5, and OOD examples range from 10 to 50. We revise the reward function to replace the original discrete rewards of -1, 0, and 1 with non-negative values as follows: $R(x) = R(c, y_t) = \frac{l}{|c - y_t| + 1}$, where l is a scaling constant set to 100. This reward incentivizes the model to bring the current number closer to the target, progressively increasing the reward as the gap decreases. For fair comparison, we run RL4VLM [41] with revised reward structure.

Blackjack. The Blackjack task requires VLM to reason with visual information and adapt to stochastic outcomes. The agent aims to win by selecting an action a_t from {"stand", "hit", `[DONE]`(if applicable)}. We revise the reward function to replace the original discrete rewards of -1, 0, and 1 with non-negative values as follows: $R(x) = \max(1 \times 10^{-10}, (r(x) + 1) \times 10)$, where $r(x)$ represents the environment's original reward for terminal state x. This scales the rewards and ensures they are strictly non-negative. For fair comparison, we run RL4VLM [41] with revised reward structure.

Method	Train Data	Assump.	SFT Init.	NL	NL-OOD	BJ
SFT-w/o-[DONE]	Off	-	-	24.8	0.0	23.1
SFT-w/-[DONE]	Off	-	-	24.0	0.0	20.2
RL4VLM [41]	On	M	✓	89.4	3.1	40.2
RL4VLM [41]†	On	NM	✓	90.3	4.4	41.0
RL4VLM [41]*	On	M	✓	34.8	1.9	23.5
GFlowVLM w/ Var-TB	On	NM	✓	**100.0**	6.2	41.4
GFlowVLM w/ SubTB	On	NM	✓	**100.0**	7.0	41.7
GFlowVLM w/ DB	On	NM	✓	**100.0**	9.1	42.2
Ablations - w/ Off-Policy Training data						
GFlowVLM w/ Var-TB	Off	NM	✓	**100.0**	17.3	43.0
GFlowVLM w/ SubTB	Off	NM	✓	**100.0**	16.7	42.4
GFlowVLM w/ DB	Off	NM	✓	**100.0**	**18.6**	**43.8**
Ablations - w/o SFT Initialization						
GFlowVLM w/ SubTB	On	NM	×	23.0	0.0	8.4
GFlowVLM w/ DB	On	NM	×	24.3	0.0	6.8
GFlowVLM w/ SubTB	Off	NM	×	**34.4**	0.0	**17.4**
GFlowVLM w/ DB	Off	NM	×	33.1	0.0	13.8

Table 3. Performance comparisons across baseline models for NumberLine (NL) and BlackJack (BJ) tasks for in-distribution and out-of-distributions (OOD) tasks. *We use the same reward function as ours. †We use the same prompt as ours to include history information for non-Markovian setting. NL-OOD stands for Number line with out-of-distribution tasks. On and Off represent On-Policy and Off-Policy, respectively. M and NM stands for Markovian and non-Markovian assumption respectively.

ALFWorld. ALFWorld [31] is an embodied AI environment combining a text-based interactive setup with a vision-language planning dataset. ALFWorld has a state-dependent action space $\mathcal{A}_t$; Our prompt instructs the VLM to choose from the admissible actions $\mathcal{A}_t$, and we evaluate out-of-distribution (OOD) performance using a test set of previously unseen scenes. We use the same non-negative reward function as used in [41], making it suitable for Var-TB and SubTB losses. However, since it lacks dense rewards for every transition, DB does not perform effectively.

6. Results Analysis

Improved VLM Reasoning abilities on In-distribution samples. Our experiments show that GFlowVLM significantly enhances VLM reasoning in tasks like NumberLine, Blackjack, and ALFWorld. As shown in Tab. 3, it improves success rates on in-distribution examples by 12% over RL4VLM, with an 8% gain in Blackjack due to high-quality, off-policy trajectories. For ALFWorld, GFlowVLM achieves a 29% success rate improvement (Tab. 4), highlighting GFlowNets' role in generating accurate trajectories crucial for VLM reasoning.

Diverse Solutions Our method generates more diverse solutions than other baselines. In ALFWorld, GFlowNets achieve 25% and 33% higher diversity than RL4VLM and SFT (Tab. 4), as measured by diversity metric, capturing a wider range of plausible solutions, offering a distinct advantage in scenarios that benefit from broader strategy exploration. In contrast to PPO, which optimizes a single best policy for long-term planning, GFlowNet finds multiple diverse high-reward solutions, making it better suited for structured generation (see Section E for detailed qualitative results).

Improved Generalization on OOD samples Our method enhances VLM reasoning on OOD examples, with GFlowNets achieving higher OOD success rates than RL4VLM in NumberLine and ALFWorld tasks by 322% and 156%, respectively. This demonstrates GFlowNets' capacity for robust generalization through diverse, accurate trajectory sampling, enabling effective handling of complex, unseen scenarios.

Benefits from Off-policy data Since GFlowNets allow for off-policy [13, 24, 40] along with on-policy learning unlike PPO [29], we adopt an off-policy data generation approach to evaluate the impact of using more accurate trajectories during training (see Section D.1 for the details of data generation). Tab. 3 shows results for NumberLine (NL), out-of-distribution NumberLine (NL-OOD), and BlackJack (BJ) tasks. GFlowVLM with Var-TB, SubTB, and DB, demonstrate improvements with offline data, averaging a 36.2% performance increase over online-only approaches. These results indicate that each loss function benefits from off-policy high-quality data, leveraging both correct and incorrect solutions to enhance performance, even in challenging OOD scenarios.

Training Efficiency. As shown in Fig. 3, GFlowNets

Method	Assump.	Pick	Look	Clean	Heat	Cool	Pick2	Avg.	OOD	Div@16
SFT-w/o- [DONE]	-	39.2	0	**14.4**	11.1	0	**28.6**	17.1	3.3	1.06
SFT-w/- [DONE]	-	32.7	0	10.3	10.8	0	21.8	15.9	3.0	1.02
RL4VLM [41]	M	47.4	14.7	10.4	14.4	18.8	18.0	21.7	4.8	1.12
RL4VLM [41]	NM	49.1	13.5	9.8	15.2	20.1	20.6	22.1	6.1	1.11
GFlowVLM w/ SubTB	NM	**50.0**	**23.1**	10.0	**18.7**	**24.3**	23.7	**26.1**	**12.3**	1.40
GFlowVLM w/ Var-TB	NM	**50.0**	22.2	10.2	16.1	22.7	21.9	25.7	10.9	**1.41**

Table 4. Results of ALFWorld. Since Alfworld does not provide dense rewards, we can not not using DB loss here. Furthermore, while RL4VLM and GFVLM with SubTB are trained with SFT initialization, GFVLM with TB-Var is without STF initialization since we do not need to model the flow.

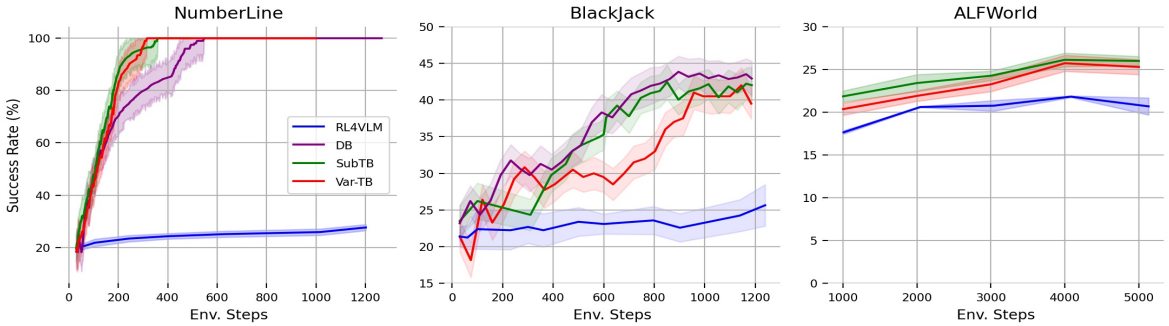


Figure 3. Training curves showing in-distribution episode success rates (%) across three tasks. For Numberline and BlackJack, RL4VLM is trained with the original reward, while GFlowVLM variants use a revised reward function, as RL4VLM serves as a strong baseline under original rewards. In ALFWorld, all methods use the same (original) reward without revision. Models are trained using on-policy sampling.

converge faster than RL4VLM on NumberLine, Blackjack, and ALFWorld, reaching optimal performance with significantly fewer environment steps—about 10,000 fewer than RL4VLM. This efficiency reduces training time and computational demands, supporting scalability for complex reasoning tasks.

Comparisons with different loss functions. As shown in the training curves, all three loss functions—DB, Var-TB, and SubTB-converge at a similar rate. DB, which requires dense rewards for every transition since it utilize transitions as input, demonstrates the best generalization, as evidenced in Tab. 3, in the NumberLine and Blackjack tasks. Both SubTB and TB achieve comparable performance in terms of in-distribution and OOD generalization, making them equally effective for a wide range of reasoning tasks.

SFT Initialization for SubTB and DB Losses. The termination probability $P_F(\top|\cdot)$ in DB and SubTB losses estimates the modified flow in GFlowNet [27], which Var-TB lacks. To enable VLMs to accurately model this for DB and SubTB, we first apply SFT on correctly completed trajectories before fine tuning with GFlowNets. As shown in Tab. 3, SFT initialization significantly boosts SubTB and DB performance on the NumberLine and Blackjack tasks. Without SFT, both losses perform poorly, especially on NL and BJ tasks. With SFT, SubTB and DB improve by 50% and 36% for NumberLine and by 107% and 103% for Blackjack, largely due to better estimation of terminal probability $P_F(\top|\cdot)$.

Markovian and non-Markovian assumptions. GFlowVLM outperforms RL4VLM in non-Markovian settings, excelling in complex, long-horizon tasks. In ALFWorld (Tab. 4), GFlowVLM achieves higher average performance by 18%, OOD robustness by 100%, and diversity by 27%. It also achieves better success rates in gym tasks (Tab. 3), where history aids decision-making, underscoring GFlowNets' advantage over PPO-based methods.

7. Conclusion, Limitation, Future Works

We introduce a novel framework using GFlowNets to enhance structured reasoning in VLMs, to capture relationships among reasoning steps for improved generalization. Unlike traditional methods like SFT and PPO, which are limited by certain assumptions, our approach supports complex, long-term reasoning tasks. Experiments in card games and embodied planning showed enhanced training efficiency, diversity, and generalization. We focus on a single-agent task setting, leaving multi-agent task and alternative prompting methods as future directions. Limited computational resources led us to use small-sized VLMs, but larger models may further benefit from GFlowNets.

References

[1] Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. Flow network based generative models for non-iterative diverse candidate generation. *Advances in Neural Information Processing Systems*, 34:27381–27394, 2021. 1, 2

[2] Yoshua Bengio, Salem Lahlou, Tristan Deleu, Edward J Hu, Mo Tiwari, and Emmanuel Bengio. Gflownet foundations. *The Journal of Machine Learning Research*, 24(1):10006–10060, 2023. 1, 5, 12, 15

[3] William Chen, Oier Mees, Aviral Kumar, and Sergey Levine. Vision-language models provide promptable representations for reinforcement learning. *arXiv preprint arXiv:2402.02651*, 2024. 1, 2

[4] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Tianyu Liu, et al. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*, 2022. 2

[5] Piyush Gupta and Vaibhav Srivastava. Deterministic sequencing of exploration and exploitation for reinforcement learning. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 2313–2318. IEEE, 2022. 1

[6] Piyush Gupta, Demetris Coleman, and Joshua E Siegel. Towards physically adversarial intelligent networks (pains) for safer self-driving. *IEEE Control Systems Letters*, 7:1063–1068, 2022. 1

[7] Piyush Gupta, David Isele, Enna Sachdeva, Pin-Hao Huang, Behzad Dariush, Kwonjoon Lee, and Sangjae Bae. Generalized mission planning for heterogeneous multi-robot teams via llm-constructed hierarchical trees. *arXiv preprint arXiv:2501.16539*, 2025. 1

[8] 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. 1, 5, 15

[9] Edward J Hu, Nikolay Malkin, Moksh Jain, Katie E Everett, Alexandros Graikos, and Yoshua Bengio. Gflownet-em for learning compositional latent variable models. In *International Conference on Machine Learning*, pages 13528–13549. PMLR, 2023. 2, 15

[10] 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 *International Conference on Machine Learning*, pages 9786–9801. PMLR, 2022. 2

[11] Moksh Jain, Tristan Deleu, Jason Hartford, Cheng-Hao Liu, Alex Hernandez-Garcia, and Yoshua Bengio. Gflownets for ai-driven scientific discovery. *Digital Discovery*, 2(3):557–577, 2023. 2

[12] Moksh Jain, Sharath Chandra Raparthy, Alex Hernández-Garcıa, Jarrid Rector-Brooks, Yoshua Bengio, Santiago Miret, and Emmanuel Bengio. Multi-objective gflownets. In *International conference on machine learning*, pages 14631–14653. PMLR, 2023. 2

[13] Hyeonah Kim, Minsu Kim, Taeyoung Yun, Sanghyeok Choi, Emmanuel Bengio, Alex Hernández-García, and Jinkyoo Park. Improved off-policy reinforcement learning in biological sequence design. *arXiv preprint arXiv:2410.04461*, 2024. 7

[14] Minsu Kim, Sanghyeok Choi, Jiwoo Son, Hyeonah Kim, Jinkyoo Park, and Yoshua Bengio. Ant colony sampling with gflownets for combinatorial optimization. *arXiv preprint arXiv:2403.07041*, 2024. 2

[15] Junnan Li, Dongxu Li, Caiming Xiong, and Steven C. H. Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *ICML*, 2022. 1

[16] 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. 2

[17] Xuanlin Li, Yunhao Fang, Minghua Liu, Zhan Ling, Zhuowen Tu, and Hao Su. Distilling large vision-language model with out-of-distribution generalizability. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2492–2503, 2023. 1

[18] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26296–26306, 2024. 1

[19] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llava-next: Improved reasoning, ocr, and world knowledge, 2024. 3, 6

[20] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024. 1

[21] Shuchang Liu, Qingpeng Cai, Zhankui He, Bowen Sun, Julian McAuley, Dong Zheng, Peng Jiang, and Kun Gai. Generative flow network for listwise recommendation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1524–1534, 2023. 2

[22] Kanika Madan, Jarrid Rector-Brooks, Maksym Korablyov, Emmanuel Bengio, Moksh Jain, Andrei Cristian Nica, Tom Bosc, Yoshua Bengio, and Nikolay

Malkin. Learning gflownets from partial episodes for improved convergence and stability. In *International Conference on Machine Learning*, pages 23467–23483. PMLR, 2023. 5, 15

[23] Nikolay Malkin, Moksh Jain, Emmanuel Bengio, Chen Sun, and Yoshua Bengio. Trajectory balance: Improved credit assignment in gflownets. *Advances in Neural Information Processing Systems*, 35:5955–5967, 2022. 5, 14

[24] Nikolay Malkin, Salem Lahlou, Tristan Deleu, Xu Ji, Edward Hu, Katie Everett, Dinghuai Zhang, and Yoshua Bengio. Gflownets and variational inference. *arXiv preprint arXiv:2210.00580*, 2022. 7

[25] Yao Mu, Qinglong Zhang, Mengkang Hu, Wenhai Wang, Mingyu Ding, Jun Jin, Bin Wang, Jifeng Dai, Yu Qiao, and Ping Luo. Embodiedgpt: Vision-language pre-training via embodied chain of thought. *Advances in Neural Information Processing Systems*, 36, 2024. 1, 2

[26] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022. 2

[27] Ling Pan, Nikolay Malkin, Dinghuai Zhang, and Yoshua Bengio. Better training of gflownets with local credit and incomplete trajectories. In *International Conference on Machine Learning*, pages 26878–26890. PMLR, 2023. 2, 8, 15

[28] Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pages 1–22, 2023. 2

[29] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 1, 7

[30] Tony Shen, Mohit Pandey, and Martin Ester. Tacogfn: Target conditioned gflownet for structure-based drug design. *arXiv preprint arXiv:2310.03223*, 2023. 2

[31] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*, 2020. 7, 12, 16

[32] Andrew Szot, Max Schwarzer, Harsh Agrawal, Bogdan Mazoure, Rin Metcalf, Walter Talbott, Natalie Mackraz, R Devon Hjelm, and Alexander T Toshev. Large language models as generalizable policies for embodied tasks. In *The Twelfth International Conference on Learning Representations*, 2023. 2

[33] Ryoichi Takase, Masaya Tsunokake, Yuta Tsuchiya, and Shota Inuzuka. Gflownet fine-tuning for diverse correct solutions in mathematical reasoning tasks. *arXiv preprint arXiv:2410.20147*, 2024. 1

[34] Zihao Wang, Shaofei Cai, Guanzhou Chen, Anji Liu, Xiaojian Ma, and Yitao Liang. Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents. *arXiv preprint arXiv:2302.01560*, 2023. 2

[35] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022. 1, 2

[36] Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. The rise and potential of large language model based agents: A survey. *arXiv preprint arXiv:2309.07864*, 2023.

[37] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022. 2

[38] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: synergizing reasoning and acting in language models (2022). *arXiv preprint arXiv:2210.03629*, 2023. 2

[39] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[40] Fangxu Yu, Lai Jiang, Haoqiang Kang, Shibo Hao, and Lianhui Qin. Flow of reasoning: Efficient training of llm policy with divergent thinking. *arXiv preprint arXiv:2406.05673*, 2024. 1, 2, 6, 7, 16

[41] Yuexiang Zhai, Hao Bai, Zipeng Lin, Jiayi Pan, Shengbang Tong, Yifei Zhou, Alane Suhr, Saining Xie, Yann LeCun, Yi Ma, et al. Fine-tuning large vision-language models as decision-making agents via reinforcement learning. *arXiv preprint arXiv:2405.10292*, 2024. 1, 2, 3, 6, 7, 8, 12, 14, 16, 18, 24

[42] Dinghuai Zhang, Hanjun Dai, Nikolay Malkin, Aaron Courville, Yoshua Bengio, and Ling Pan. Let the flows tell: Solving graph combinatorial optimization problems with gflownets. *arXiv preprint arXiv:2305.17010*, 2023. 2

[43] David W Zhang, Corrado Rainone, Markus Peschl, and Roberto Bondesan. Robust scheduling with gflownets. *arXiv preprint arXiv:2302.05446*, 2023. 14

[44] Junwei Zhang, Zhenghao Zhang, Shuai Han, and Shuai Lü. Proximal policy optimization via enhanced exploration efficiency. *Information Sciences*, 609: 750–765, 2022. 1

[45] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *ArXiv*, abs/2304.10592, 2023. 1

[46] 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. 2