Set Reinforcement Learning: Principles and Approaches

Jubayer Ibn Hamid, Ifdita Hasan Orney, Dorsa Sadigh, Chelsea Finn and Noah Goodman

Stanford University

Abstract

In this note, we discuss the set reinforcement learning framework, various objectives that can be optimized in this setting, and develop apparatus for theoretical analysis.

Contents

1 Introduction

Reinforcement learning (RL) provides a general framework for training agents through interaction with an environment enabling them to learn from their own experiences. Its flexibility stems both from the diversity of algorithmic approaches and from the ability to specify the environment and reward function. In standard formulations, the reward is defined over individual actions or trajectories, and the goal is to learn a policy that selects an optimal action at each state. However, in many settings, the objective is not to produce a single optimal action, but rather a *set* of actions or trajectories with desirable collective properties. For example, in language model (LM) post-training, one often samples multiple candidate generations before applying verification, ranking, search procedures or self-aggregation. In such cases, performance depends on properties of the entire set rather than on individual generations.

Set reinforcement learning (set RL), introduced in [HOX+26], generalizes the RL framework by optimizing policies with respect to objectives defined over sets of trajectories. Concretely, given a policy that induces a distribution over sets of trajectories, set RL aims to maximize a set-level objective function that couples elements within the set under a shared learning signal. This perspective provides a high degree of flexibility in designing objectives, analogous to reward design in standard RL, while enabling fundamentally different forms of learning signals.

A central feature of set RL is the *coupling* between elements of a set. Unlike standard RL, where each trajectory contributes independently to the objective, set-level objectives can induce dependencies across trajectories, leading to learning signals that reflect collective properties. Understanding how this coupling manifests is a primary focus of this note.

In this note, we investigate the structure and implications of set RL objectives. We analyse several classes of objectives, develop tools for reasoning about their learning signals, and highlight key phenomena arising from coupling within sets. Our discussion is motivated primarily by applications in LM post-training, where set-based objectives naturally arise in the context of sampling, search, and test-time compute.

2 Set Reinforcement Learning Framework

We start by recalling the standard reinforcement learning framework. Consider a Markov decision process defined by state space $\mathcal{S}$, action space $\mathcal{A}$, transition dynamics distribution $p(s_{t+1} \mid s_t, a_t)$, reward function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, initial state distribution ρ_0 and discount factor $\gamma \in (0, 1)$. In reinforcement learning, we aim to find the policy π_θ parametrized by $\theta \in \Theta$ that solves the following problem:

$$\max_\theta \mathbb{E}_{s_0 \sim \rho} \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 \right]. \tag{1}$$

Here, a trajectory τ refers to a tuple $(s_0, a_0, r_0, s_1, a_1, r_1, \cdots)$. Note that the notation here is overloaded; we write $\tau \sim \pi_\theta(\cdot \mid s_0)$ to mean that each action $a_t \sim \pi_\theta(\cdot \mid s_t)$ is sampled from the policy while the states are sampled from the MDP's transition dynamics distribution, i.e. $s_{t+1} \sim P(\cdot \mid s_t, a_t)$. Intuitively, the problem here is to find the policy that maximizes the expected returns from a trajectory τ sampled under the policy. One can think of this problem as essentially aiming to solve $\max_\theta \mathbb{E}_{s_0 \sim \rho, \tau \sim \pi_\theta(\cdot \mid s_0)} [f(\tau)]$ where $f(\tau)$ is the (discounted) sum of rewards in the trajectory τ. We will refer to this framework as *standard reinforcement learning* for clarity.

Set reinforcement learning is a framework that, at a high level, attempts to find that policy that maximizes the expected value of a *set* of trajectories sampled under the policy with respect to some objective function. We first look at the idealised sequential formulation setting and then study the more practical setting.

2.1 Sequential Formulation

In the idealised sequential formulation, at each visited state s, the policy samples a set of n actions independently from $\pi_\theta(\cdot \mid s)$ where $n > 1$ is a fixed integer. Given $a_{1:n} := (a_1, \ldots, a_n) \overset{\text{i.i.d.}}{\sim} \pi_\theta(\cdot \mid s)$, a

set-level reward $f(s, a_{1:n})$ is assigned to the entire set. Repeating this procedure recursively at each timestep generates an n-ary rollout tree induced by our policy. At depth t, the tree contains n^t states, which we denote by $s_t^{(1)}, \ldots, s_t^{(n^t)}$. For each such state $s_t^{(i)}$, let $(a_t)_{1:n}^{(i)}$ denote the n-tuple of actions sampled from $\pi_\theta(\cdot \mid s_t^{(i)})$. The goal is to solve the following optimization problem defined using set value function:

$$\max_\theta \mathbb{E}_{s_0 \sim \rho} \mathbb{E}_{\pi_\theta} \left[\sum_{t=0}^{\infty} \sum_{i=1}^{n^t} \gamma^t f\left(s_t^{(i)}, (a_t)_{1:n}^{(i)}\right) \mid s_0 \right]. \tag{2}$$

To be clear, at each state $s_t^{(i)}$, the set of n actions are sampled independently, i.e. $a_{t,1}^{(i)}, \cdots, a_{t,n}^{(i)} \overset{i.i.d}{\sim} \pi_\theta(\cdot \mid s_t^{(i)})$. In particular, we require that f is an objective such that $\mathbb{E}_{a_{1:n} \sim \pi_\theta(\cdot|s)}\left[f(s, a_{1:n})\right]$ cannot be written as $\mathbb{E}_{a \sim \pi_\theta(\cdot|s)}\left[g(s, a)\right]$ where g is independent of θ for all policy parameters $\theta \in \Theta$. In other words, we cannot collapse the set RL problem to a standard RL one. This is an important feature in the definition of set RL.

This last point is particularly important. For example, if the objective is defined as $f(s, a_{1:n}) = \frac{1}{n}\sum_{i=1}^n r(s, a_i)$, then the set RL problem in eq. (2) becomes equivalent to the standard RL problem in eq. (1). However, one can consider several objectives that satisfy the requirements for set RL; for example, $f(s, a_{1:n}) = \max_{i \in \{1, \cdots, n\}} r(s, a_i)$, which is the pass@$n$ objective.

To understand what set RL is attempting to optimize, it is useful to construct a notion of value functions in this framework.

Definition 2.1 (Set Value Functions)**.** *Given a policy π generating a state-visitation tree and a set objective $f : \mathcal{S} \times \mathcal{A}^n \to \mathbb{R}$, the set state value function $V_\pi^\sharp(s; f)$ and the set state-action value function $Q_\pi^\sharp(s, a_{1:n}; f)$ are defined as*

$$V_\pi^\sharp(s; f) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \sum_{i=1}^{n^t} \gamma^t f(s_t^{(i)}, (a_t)_{1:n}^{(i)}) \Big| s_0 = s \right], \tag{3}$$

$$Q_\pi^\sharp(s, a_{1:n}; f) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \sum_{i=1}^{n^t} \gamma^t f(s_t^{(i)}, (a_t)_{1:n}^{(i)}) \Big|_{(a_0)_{1:n} = a_{1:n}}^{s_0 = s,} \right] \tag{4}$$

Here, we assume that the discount factor $\gamma \in (0, \frac{1}{n})$ to ensure values remain bounded - the range is smaller than in standard RL since we add the expected sum of rewards from n actions stemming out of each state. Intuitively, $V_\pi^\sharp(s)$ is the expected discounted return of the entire state tree rooted at s, as illustrated in fig. 1, where the reward at each node is given by the set objective. In contrast, $Q_\pi^\sharp(s, a_{1:n})$ evaluates the expected return of the tree that begins at s with the specific action set $a_{1:n}$.

With these definitions, we first see that the set RL problem can be written as:

$$\max_\theta \mathbb{E}_{s_0 \sim \rho}\left[V_{\pi_\theta}^\sharp(s_0; f)\right].$$

In other words, in set reinforcement learning, we want to learn the policy that maximizes the expected return from a *tree* generated using our policy is maximized. In contrast, in standard reinforcement learning, we want to learn the policy such that the expected return from a *trajectory* is maximized.

These constructions allow us to translate several well-known ideas and algorithms from standard reinforcement learning to the set reinforcement learning framework. As an example, we have the following result, which is the set RL analogue of the performance-difference lemma from [KL02b]:

Lemma 2.2. *Given any two policies π_θ and π_β and a fixed initial state s_0, under any set objective function f,*

$$V_{\pi_\theta}^\sharp(s_0; f) - V_{\pi_\beta}^\sharp(s_0; f) = \frac{1}{1 - \gamma n} \mathbb{E}_{s \sim d_{\pi_\theta}^\sharp(\cdot), a_{1:n} \sim \pi_\theta(\cdot|s)} \left[A_{\pi_\beta}^\sharp(s, a_{1:n}; f) \right].$$

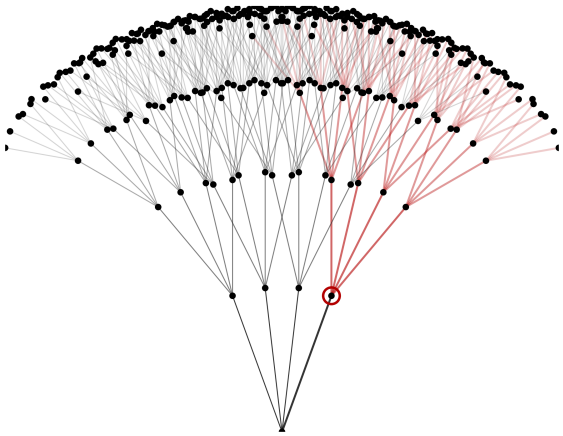

Figure 1: The set value of a state (circled) is the expected discounted return of the subtree (highlighted) rooted in this state.

Here $d^\sharp_\pi(s)$ is the stationary state-visitation distribution induced by π_θ. Similar to the standard reinforcement learning setting, this result says that if we update the policy π_β to π_θ such that, at all states visited by π_θ, the advantage $A^\sharp_{\pi_\beta}(s, a_{1:n}; f) = Q^\sharp_{\pi_\beta}(s, a_{1:n}; f) - V^\sharp_{\pi_\beta}(s)$ of a set of actions taken by our new policy π_θ is positive, then we will get a policy that has strictly higher performance (as measured by its value). This suggests that many of the principles underlying methods, like PPO from [SWD+17], can be extended to the set reinforcement learning paradigm as well for policy improvement. Similarly, algorithms such as Q-learning can also be adapted provided that we sample rollouts in the form of state-visitation trees.

The sequential formulation is conceptually useful, but it is computationally impractical in most settings because the number of sampled states grows exponentially with depth. However, sampling complexity aside, this formulation does indeed capture the goal of several problems. As an example, when training an agent to play a game such as chess, we want the model to sample the optimal *tree* of states visited where the tree can be scored by the sum of rewards received at its leaf nodes.

We will now look at the significantly more practical formulation of the problem.

2.2 Single-turn Formulation

In many practical settings, we do not want to sample state-visitation trees like we did in the sequential framework because of the large sample complexity and other practical challenges. For example, in the language model (LM) post-training setting, we sample a batch of prompts $x_1, \cdots, x_{|\mathcal{B}|} \sim \mathcal{D}$ where $\mathcal{D}$ is our training set, and then sample N generations from the LM conditioned on each prompt, i.e. $y_{1:n} := (y_1, \cdots, y_N) \sim \pi_\theta(\cdot \mid x_i)$ for each x_i.

In this setup, set RL aims to solve:

$$\max_\theta \mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y_{1:n}\sim\pi_\theta(\cdot|x)} \Big[f(x, y_{1:n}) \Big]. \tag{5}$$

Again, we require that the set objective function f is such that we cannot write, for each x, $\mathbb{E}_{y_{1:n}}[f(x, y_{1:n})]$ as $\mathbb{E}_y[g(x, y)]$ where g is independent of θ for all policy parameters $\theta \in \Theta$. This requirement is crucial because, otherwise, under some objectives (for e.g., $f(x, y_1, \cdots, y_n) = \frac{1}{n}\sum_{i=1}^n r(x, y_i)$) the set RL formulation in eq. (5) becomes equivalent to a standard RL formulation using some non-differentiable reward function.

We now look at some examples of objectives that satisfy the set reinforcement learning framework.

Example 2.3. Let $f(x, y_{1:n}) = \max_i r(x, y_i)$. This objective corresponds to pass@n optimization [WK25].

Example 2.4. More generally, one can consider objectives that represent the usage of test-time compute. Let $\texttt{Alg}(x, y_1, \cdots, y_n) \in \mathcal{Y}$, i.e. the algorithm returns the final generation to return. Then, one can consider $f(x, y_{1:n}) = r(x, \texttt{Alg}(x, y_{1:n}))$ which is an objective that rewards the generation chosen by our test-time compute. Alternatively, if the algorithm returns a subset of the generations, i.e. $\texttt{Alg}(x, y_{1:n}) \in \mathcal{Y}^m$ for $m < n$, then we can set $f(x, y_{1:n}) = \mu(\{r\left(x, \texttt{Alg}(x, y_{1:n})\right)_j\}_{j=1}^m)$ which is the average reward in the set of generations selected by the algorithm.

Example 2.5. Let $d(x, y_{1:n}) \in [0, 1]$ be a measure of the diversity in the generations $(y_1, \cdots, y_n)$. Then, the following objective also satisfies set RL requirements: $f(x, y_{1:n}) = \frac{1}{n}\sum_{i=1}^n r(x, y_i) \cdot d(x, y_{1:n})$. This is the *polychromic objective*, designed to encourage the model to balance exploration and exploitation, which we will look at in detail later.

To consolidate our understanding of how set RL optimizes differently from standard RL, we look at the policy gradient. After applying the score-function identity to the joint density, we get the following policy gradient:

$$\begin{aligned}
&\nabla_\theta \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y_{1:n}\sim\pi_\theta(\cdot|x)}\Big[f(x, y_{1:n})\Big] \\
&= \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y_{1:n}\sim\pi_\theta(\cdot|x)}\left[\big(f(x, y_{1:n}) - \hat{f}(x)\big)\sum_{i=1}^n \nabla_\theta \log \pi_\theta(y_i \mid x)\right]
\end{aligned} \tag{6}$$

where $\hat{f}(x)$ is a set-level baseline. In other words, $\hat{f}(x)$ is independent of $y_{1:n}$ or any strict subset of $y_{1:n}$. The defining feature of (6) is that all generations in the sampled set $y_{1:n}$ share the same scalar learning signal, $f(x, y_{1:n}) - \hat{f}(x)$. In contrast, in the standard RL setting, we get:

$$\nabla_\theta \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\Big[r(x, y)\Big] = \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[\nabla_\theta \log \pi_\theta(y \mid x) \cdot \big(r(x, y) - b\big)\right]. \tag{7}$$

As such, in standard RL, each generation receives its own individual advantage. In set RL, the objective f couples the samples in the set, and the gradient assigns equal credit to all elements in the set through a shared set-level signal. For this reason, admissible baselines must be independent of the sampled set $y_{1:n}$ or any subset of it.

The notion of coupling will be a recurring theme in this note. So far we have seen that in set RL, the objectives within a set are coupled in the sense that they all receive the same score. We will soon see that, depending on the choice of the function f, multiple *objectives* can be "coupled", i.e. sets can be optimized to balance several goals for our learning agent.

A naive algorithm for training an agent in this set RL framework would be to sample multiple sets of size n per prompt $x \sim \mathcal{D}$ to get a Monte Carlo estimate of the gradient in eq. (6). However, the sample complexity of this naive algorithm could become quite large. In particular, if the baseline, $\hat{f}(x)$ is also a Monte Carlo estimate, one would need to sample several sets of size n for there to be a meaningful reduction in variance. We will now look at a more practical, general recipe for training a policy using set reinforcement learning.

2.3 General Algorithm

We now discuss a general recipe for optimizing a model using set reinforcement learning, as introduced in [OHR+26]. This will be strongly motivated by the language model (LM) post-training setting.

Suppose, that our set objective, $f(x, y_1, \cdots, y_n)$, is defined over sets of size n. Moreover, we assume that f is symmetric in the arguments $y_{1:n}$. If we wanted to use an empirical estimate (e.g., a Monte Carlo estimate) of the gradient in (6), then we would need to sample several sets of size n conditioned on a single prompt which can be computationally expensive. In this section, we consider how we can instantiate set reinforcement learning using a computationally feasible gradient estimator.

For a fixed prompt x, we first draw N i.i.d. generations $y_1, \ldots, y_N \sim \pi_\theta(\cdot \mid x)$. We require that the number of samples, N, is greater than the size of sets, n, in the set RL framework. Next, from these samples, we construct K sets, $G_1, \cdots, G_K$, of size $n < N$, in a combinatorial fashion without replacement:

$$G_j = \{y_{(j_1)}, \ldots, y_{(j_n)}\}, \quad j_1 < \cdots < j_n, \{j_1, \ldots, j_n\} \subseteq \{1, \ldots, N\}.$$

The maximum number of sets that can be constructed in this manner is $K = \binom{N}{n}$. Using all $\binom{N}{n}$ sets can reduce the variance of the resulting gradient estimator. When evaluating f is computationally expensive, scoring all $\binom{N}{n}$ sets may be impractical. In this case, we instead sample K sets uniformly (without replacement) from the collection of all $\binom{N}{n}$ possible sets. Either of these choices leads us to an unbiased estimator of the set RL gradient as we will see later.

Now that we have constructed K sets, for each constructed set G_j, we compute the set-level score under the set objective function:

$$f(x, G_j) = f\big(x, y_{(j_1)}, \ldots, y_{(j_n)}\big).$$

We construct a variance-reduction baseline using the sampled sets:

$$\hat{f}(x) := \frac{1}{K} \sum_{j=1}^{K} f(x, G_j).$$

In other words, the baseline is simply an average of the scores of all sets under our objective function f. Given the baseline, for each set G_j, we can now construct a set advantage function which is simply

$$A^\sharp(x, G_j; f) = f(x, G_j) - \hat{f}(x). \tag{8}$$

So far, we have computed everything at the set-level, i.e. we constructed sets, scored them and computed each set's advantage. We now derive an advantage function at the level of a single generation.

Marginal set advantage of a single generation. Intuitively, we will set the advantage of each generation to be simply the sum of the advantages of all sets containing the generation. Let $\mathcal{G}(y)$ be the collection of all sets that contain the fixed generation y. Note that $\mathcal{G}(y) = \{G \in G_{1:K} \mid y \in G\} \subset G_{1:K}$. We define the marginal set advantage of the generation y to be:

$$\widehat{A^\sharp_{\text{marg}}}(x, y; f) := \sum_{G \in \mathcal{G}(y)} A^\sharp(x, G; f). \tag{9}$$

Final Estimator. Using this, we use the following set RL gradient estimator:

$$\nabla_\theta \widehat{\mathbb{E}}_{x \sim \mathcal{D}, y_{1:n} \sim \pi_\theta(\cdot|x)} \left[f(x, y_{1:n})\right] = \widehat{\mathbb{E}}_{x \sim \mathcal{D}, y_1, \cdots, y_N \sim \pi_\theta(\cdot|x)} \left[\sum_{i=1}^{N} \nabla_\theta \log \pi_\theta(y_i \mid x) \widehat{A^\sharp_{\text{marg}}}(x, y_i; f)\right] \tag{10}$$

Now, we have all the ingredients required for optimizing an LM with respect to the set RL framework. Since the marginal set advantage gives us an advantage function over single generations, we can use existing *standard* RL algorithms, such as PPO [SWD+17] or GRPO [SWZ+24], to optimize the set objective by replacing their standard advantage, $A(x, y_i)$, with $\widehat{A^\sharp_{\text{marg}}}(x, y_i)$. The pseudocode for our general recipe is provided in algorithm 1.

Note that our estimator is an unbiased estimator of the true policy gradient of set RL (up to a scaling factor). This is true for both the case where we construct all $\binom{N}{n}$ sets and the case where we sample K sets uniformly without replacement. We show this in the following proposition:

Algorithm 1 Gradient of On-Policy Implementation of Set RL

Require: Set objective f, Policy π_θ, batch of inputs B, number of rollouts N, set size n

for each input $x \in B$ **do**
  Sample $y_1, \ldots, y_N \sim \pi_\theta(\cdot \mid x)$
  // Construct sets either by enumerating all $\binom{N}{n}$ subsets or by uniformly sampling K subsets from them
  Construct sets $G_1, \cdots, G_K$ from $\{y_{1:N}\}$ without replacement
  Compute score of each set $\{f(x, G_l)\}_{l=1}^K$ and baseline $b = \frac{1}{K}\sum_{l=1}^K f(x, G_l)$
  Compute set advantage of each set $\widehat{A^\sharp}(x, G_l; f) = f(x, G_l) - b$ for $l = 1, \cdots, K$
  // Let $\mathcal{G}(y)$ be all the sets in $G_1, \cdots, G_K$ that contain y
  Compute marginal set advantage $\widehat{A^\sharp_{\text{marg}}}(x, y_j; f) := \sum_{G \in \mathcal{G}(y_j)} \widehat{A^\sharp}(x, \mathcal{G}(y_j))$ for $j = 1, \cdots, N$
  $\hat{g}(x) \leftarrow \sum_{j=1}^N \nabla_\theta \log \pi_\theta(y_j \mid x) \cdot \widehat{A^\sharp_{\text{marg}}}(x, y_j; f)$
end for
$\hat{g} \leftarrow \frac{1}{|B|}\sum_{x \in B} \hat{g}(x)$
return $\hat{g}$

Proposition 2.6. *Fix a prompt x, and let $y_1, \cdots, y_N \overset{\text{i.i.d.}}{\sim} \pi_\theta(\cdot \mid x)$ be our independently sampled N generations and let $f : \mathcal{X} \times \mathcal{Y}^{\oplus n} \to \mathbb{R}$ be our set objective. Then,*

$$\mathbb{E}[\sum_{i=1}^N \nabla_\theta \log \pi_\theta(y_i \mid x)\, \widehat{A^\sharp_{\text{marg}}}(x, y_i; f)] = M \nabla_\theta \mathbb{E}_{y_{1:n} \sim \pi_\theta(\cdot|x)}\big[f(x, y_{1:n})\big],$$

where $M \in \mathbb{R}_{>0}$ is a constant scaling factor that depends on the number of sets we construct. In particular, when we construct all $\binom{N}{n}$ sets without replacement, we have that

$$M = \binom{N}{n} - \binom{N-1}{n-1}$$

and when we sample, uniformly, K sets without replacement from $K_{\text{all}} := \binom{N}{n}$ sets, we have that

$$M = \frac{1}{\binom{K_{\text{all}}}{K}} \frac{K_{\text{all}}}{K} \binom{K_{\text{all}} - 2}{K - 2} \left(\binom{N}{n} - \binom{N-1}{n-1} \right).$$

Consequently, after also taking expectation over $x \sim \mathcal{D}$ and scaling the learning rate, the estimator is an unbiased estimator of the set RL gradient.

The proof for this can be found in [OHR+26]. Note that the proof shows that constructing all $\binom{N}{n}$ sets allows us to get the complete U-statistic estimator which, provably, has the lowest variance.

Finally, we consider another example where we address the following question: can we use the set RL framework where the set size itself is not fixed? Suppose we have an objective that can be defined on sets of variable size. How can we optimize this objective over sets of varying size?

Example 2.7. Suppose, we sample N generations in total and we seek to optimize some objective over sets of size $n_1, n_2, \cdots, n_C \in \mathbb{Z}_{\geq 1}$. Here, we assume that the objective function f could be defined on any of these sets (with some abuse of notation). For example, the objective $f(x, y_{1:n}) = \max_i r(x, y_i)$ does not directly depend on the size of the set insofar as the set has at least one generation. We could generalize the ideas we have presented so far to address this problem setting. First, we construct sets of size n_1 and denote the full collection of sets of size n_1 by $\mathcal{G}(n_1)$. For example, we could compute all possible sets of size n_1 combinatorially from the N generations. Similarly, we get a collection of sets of size n_2 called $\mathcal{G}(n_2)$, and so on. Then, we can optimize using set RL by optimizing the following:

$$\mathbb{E}_{y_1, \cdots, y_N \sim \pi_\theta(\cdot|x)} \left[\frac{1}{\binom{N}{n_1}} \sum_{Y \in \mathcal{G}(n_1)} f(x, Y) + \cdots + \frac{1}{\binom{N}{n_C}} \sum_{Y \in \mathcal{G}(n_C)} f(x, Y) \right]$$

3 Scaffold and Marginal Advantage

In this section, we analyse the learning dynamics of set reinforcement learning. In particular, we are interested in understanding the characteristics of generations that are encouraged versus discouraged in the set RL framework. We start by defining the following useful construction:

Definition 3.1 (Scaffold Value)**.** *The scaffold value of a set of generations, $y_{1:n}$, under a policy π and a set-RL objective $f : \mathcal{X} \times \mathcal{Y}^n \to \mathbb{R}$ is defined to be*

$$\Lambda_f(y_{1:n}; \pi, x) := \mathrm{Cov}_{y'_{1:n} \sim \pi(\cdot|x)}(f(x, y'_{1:n}), \frac{1}{I(a_{1:n})} \sum_{i,j=1}^{n} 1\{a'_i = a_j\}) \tag{11}$$

where $I(a_{1:n})$ is the maximum size of the intersection of $y_{1:n}$ with any other set $y'_{1:n}$. We refer to the space $\Lambda_f(\pi) = \{(a_{1:n}, \Lambda_f(a_{1:n}; \pi) \mid a_{1:n} \in \mathcal{A}^n\}$ as the scaffold of the policy π.

In particular, when the set is homogeneous, i.e. $y_{1:n} = (y, \cdots, y) =: y^{\oplus n}$, the scaffold value can be, equivalently, written as:

$$\Lambda_f(y^{\oplus n}; \pi, x) = \pi_\theta(y \mid x) \left(\mathbb{E}_{y_{2:n} \sim \pi_\theta(\cdot|x)} \left[f(x, y, y_{2:n}) \right] - \mathbb{E}_{y_{1:n} \sim \pi_\theta(\cdot|x)} \left[f(x, y_{1:n}) \right] \right).$$

We also look at the closely related marginal set advantage of a generation:

Definition 3.2 (Marginal Set Advantage)**.** *The marginal set advantage of a generation $y \in \mathcal{Y}$ under a policy π and a set-RL objective f is defined as*

$$A^{\sharp}_{\mathrm{marg}}(x, y; \pi) = \mathbb{E}_{y_{2:n} \sim \pi_\theta(\cdot|x)} \left[f(x, y, y_{2:n}) \right] - \mathbb{E}_{y_{1:n} \sim \pi_\theta(\cdot|x)} \left[f(x, y_{1:n}) \right]. \tag{12}$$

Observation. *Note that $\Lambda_f(y^{\oplus n}; \pi, x) = \pi(y \mid x) A^{\sharp}_{\mathrm{marg}}(x, y; \pi)$. For a given generation y, the scaffold value is often a more useful object to study. For example, if $\pi(y \mid x) = 0$, even if the marginal set advantage of y could be large, the policy would not be learning to upweight that generation since it fails to sample it in the first place. As such, the scaffold value is often a more useful indication of how well a policy might learn an action. We formalize this analysis next.*

The marginal advantage directly determines which generations' log likelihood increases in a set RL algorithm. Suppose, our policy is parametrized as a softmax distribution:

$$\pi_\theta(y \mid x) = \frac{\exp(z_{xy})}{\sum_{y'} \exp(z_{xy'})}$$

where z_{xy} is the output logit of y from x. Then, we have the following result determining how the logit changes for a specific generation.

Lemma 3.3 (Logit shift in set RL)**.** *In set reinforcement learning, assuming the objective function f is symmetric in its arguments and the learning rate is α, the shift in logit after one step gradient update from π^k_θ to π^{k+1}_θ can be written as:*

$$z^{k+1}_{xy} - z^k_{xy} = \alpha \pi_{\theta^k}(y \mid x) \left[\mathbb{E}_{y_{2:n} \sim \pi^k_\theta(\cdot|x)} \left[f(x, y, y_{2:n}) \right] - \mathbb{E}_{y_{1:n} \sim \pi^k_\theta(\cdot|x)} \left[f(x, y_{1:n}) \right] \right]. \tag{13}$$

We now compute the marginal set advantages under some objectives to better understand their learning dynamics.

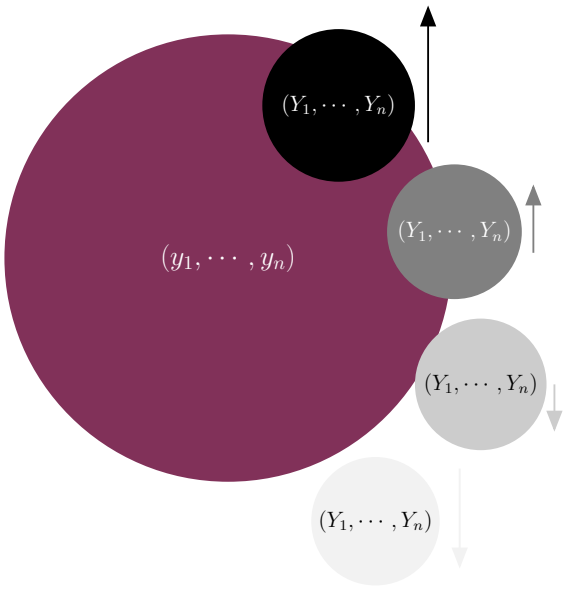

Figure 2: Visualization of a scaffold value of a fixed set of generations, $(y_1, \cdots, y_n)$. Here $(Y_1, \cdots, Y_n) \sim \pi_\theta(\cdot \mid x)$ are sets of generations being sampled and the arrows indicate the value of $f(x, Y_{1:n})$. In this visualization, larger intersection between $y_{1:n}$ and the sampled $Y_{1:n}$ tends to correlate with higher scores under the objective f. As such, the scaffold value of $y_{1:n}$ is positive.

Example 3.4. Let $f(x, y_{1:n}) = \max_{i \in \{1, \cdots, n\}} r(x, y_i)$. Suppose that the probability of sampling a successful generation is p. Then, one can easily show that for $f(x, y_{1:n}) = \max_{i \in \{1, \cdots, n\}} r(x, y_i)$ and for a fixed *incorrect generation* y (i.e. $r(x, y) = 0$),

$$\begin{aligned} A^\sharp_{\text{marg}}(x, y) &= \mathbb{E}_{y_{2:n} \sim \pi_\theta(\cdot|x)}\left[f(x, y, y_{2:n})\right] - \mathbb{E}_{y_{1:n} \sim \pi_\theta(\cdot|x)}\left[f(x, y_{1:n})\right] \\ &= (1 - (1-p)^{n-1}) - (1 - (1-p)^n) \\ &= (1-p)^n - (1-p)^{n-1}, \end{aligned}$$

which is non-positive for any $p \in [0, 1]$. As such, under this objective, incorrect generations' log likelihood is never increased by our set RL algorithm.

Example 3.5. We now consider majority voting where the objective is defined by $f(x, y_{1:n}) = r(x, \text{maj}(y_{1:n}))$. Once again, one can show that the marginal set advantage of any incorrect generation is not only always less than or equal to that of a correct generation, it is actually always non-positive.

Example 3.6. Let $f(x, y_{1:n}) = 1\{\exists y_i \neq y_j \mid y_i, y_j \in y_{1:n}\}$. In other words, f is a function that measures pairwise similarity and returns 1 if it finds at least one pair that is not similar. Then, under a uniform policy, $\Lambda_f(y^{\oplus n}; x, \pi)$ is 0 for all y. However, for any non-uniform policy, this is not 0. In particular, for a non-uniform policy, the marginal set advantage is larger for actions that have smaller probability of being sampled from the current distribution. However, the scaffold value multiplies the marginal advantage with the probability of being sampled which counteracts.

Furthermore, the notion of scaffold values helps us understand how entropy changes during set RL gradient updates as well.

Proposition 3.7. *Consider the set RL setup on $x \in \mathcal{X}$. After one update to the policy, the change in entropy, $\Delta = \mathcal{H}\left(\pi_\theta^{k+1} \mid x\right) - \mathcal{H}\left(\pi_\theta^k \mid x\right)$, is given by*

$$\Delta \approx -\alpha \text{Cov}_{y_{1:n}}\left(\frac{1}{n}\sum_{i=1}^n \log \pi_\theta^k(y_i \mid x), \text{Cov}_{y'_{1:n}}\left(f(x, y'_{1:n}), \sum_{i,j=1}^n 1\{y_i = y'_j\}\right)\right),$$

where both covariances are taken with respect to $\pi_\theta^k(\cdot \mid s)$ and α is the learning rate.

4 Polychromic Objectives

Exploration is a critical component in learning from experience. Firstly, it is necessary if you want to train an agent in a complex problem setting where one must explore to discover at least one successful solution; note that if the agent fails to discover any behaviour that gets some reward signal, then policy gradient methods will simply not be able to optimize at all since the advantage of all actions will be 0. Secondly, often it is quite important for the agent to discover several, diverse successful behaviours. An agent that is learning to play chess should ideally master multiple opening lines rather than relying on a single strategy that may prove ineffective against a different opponent at test time. Thirdly, exploration is necessary for an agent to stress test what it has learned and its internal beliefs. Otherwise, the model overfit on spurious correlations between the reward signal and specific behaviours.

The exploration-exploitation tradeoff in reinforcement learning is a well-known problem. This problem becomes especially pronounced in policy gradient methods. One must train the agent to learn what gets high rewards (i.e. exploit) while also attempting novel strategies (i.e. explore). This requires a form of *optimism under uncertainty*: the agent must attempt strategies during data collection that are exploratory even if they do not immediately get high rewards. Since the data is sampled from the policy itself, the learning signal itself must encourage allocating probability mass to strategies that have not yet demonstrated high reward. For example, when attempting to solve an algorithmic problem, an agent exploring a dynamic programming formulation should continue refining that strategy even if its initial implementations fail unit tests, rather than immediately reverting to a familiar but less scalable exhaustive search approach.

However, the learning signal cannot simply encourage attempting novel strategies. It must also encourage the policy to exploit by re-attempting what has already receive high rewards. As such, the algorithm must address the *trade-off* between exploration and exploitation. We seek an optimization objective for reinforcement learning that satisfies the following desiderata. First, the objective should encourage *optimistic exploration* by assigning positive learning signal to trajectories that attempt novel reasoning strategies, even when those strategies have not yet yielded high reward. Second, the objective should *intrinsically encode the exploration–exploitation trade-off* and incentivize balancing these two goals. This allows the model itself to learn how to balance these competing objectives through optimization.

Polychromic objectives are a family of objectives that are designed to do that. We first look at a practical example of a polychromic objective in the same framework as section 2.2:

$$f_{\text{poly}}(x, y_1, \cdots, y_n) = \frac{1}{n} \sum_{i=1}^{n} r(x, y_i) \cdot d(x, y_1, \cdots, y_n), \tag{14}$$

where $d(x, y_1, \cdots, y_n)$ is a function that measures the diversity of the set of generations, $y_{1:n}$. Here, we assume that $r(x, y) \in [0, 1]$ and $d(x, y_{1:n}) \in [0, 1]$. Because the set-RL gradient uses a shared advantage for all trajectories in a set, this objective increases the likelihood of successful behaviours *and* diverse exploratory trajectories. The shared advantage term amplifies exploratory trajectories that do not (yet) yield high rewards, pushing the policy to discover diverse strategies.

Before we move on to analysis, let us consolidate our understanding of how set reinforcement learning couples the generations in a set under this objective. One way to do this would be to rewrite the objective in the set RL framework as follows:

$$
\begin{aligned}
&\mathbb{E}_{y_{1:n}\sim\pi_\theta(\cdot|x)}\left[\frac{1}{n}\sum_{i=1}^{n} r(x,y_i)d(x,y_{1:n})\right] \\
&= \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{y_{1:n}\sim\pi_\theta(\cdot|x)}\left[r(x,y_i)\cdot d(x,y_{1:n})\right] \\
&= \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{y_i\sim\pi_\theta(\cdot|x)}\left[r(x,y_i)\mathbb{E}_{y_{1:i-1},y_{i+1:n}\sim\pi_\theta(\cdot|x)}\left[d(x,y_{1:n})\mid y_i\right]\right] \\
&= \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[r(x,y)\mathbb{E}_{y_{1:n-1}\sim\pi_\theta(\cdot|x)}\left[d(x,y,y_{1:n-1})\right]\right]
\end{aligned}
$$

Note that, even though the outer expectation is over a single generation, the final expression is not equivalent to $\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[g(x,y)\right]$ for any g that is independent of θ. Consequently, taking the gradient, with respect to θ would not only contribute a learning signal for y sampled in the outer expectation, but also for $y_{1:n-1}$ in the inner expectation. This is simply another way of saying that the set score acts as the same learning signal for all generations in the set.

In contrast, [LZY+25] proposed optimizing $\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[r(x,y)\cdot d(x,y\mid y_{1:n-1})\right]$, but $y_{1:n-1}$ are treated as constants which turns $d(x,y\mid y_{1:n-1})$ into a θ-independent reward function. Consequently, optimizing this does not lead to a set-level learning signal being shared by all constituents whereas set reinforcement learning causes the polychromic objective's signal to be shared by all generations in a set.

Now we turn our attention to analysing the objective in greater detail. First, we analyse the marginal set advantage of a generation (see eq. (12)) under the polychromic objective. Fix a generation $y\in\mathcal{Y}$ and assume that $\pi_\theta(y\mid x)>0$. Recall that in set reinforcement learning, a generation y receives increased probability mass (i.e., $\log\pi_\theta^{k+1}(y\mid x)-\log\pi_\theta^{k}(y\mid x)>0$) if and only if it has a positive marginal set advantage and non-zero probability under the rollout policy. The marginal set advantage of y can be decomposed as follows:

$$
\begin{aligned}
&A^{\sharp}_{\text{marg}}(x,y;f_{\text{poly}},\theta^k) \\
&= \underbrace{\left(\frac{1}{n}r(x,y)+\frac{n-1}{n}\mathbb{E}_Y[r(x,Y)]\right)\mathbb{E}_{Y_{2:n}}[d(x,y,Y_{2:n})]}_{\text{Term 1: Mean reward of sets containing } y \,\times\, \text{Mean diversity of sets containing } y} \quad - \quad \mathbb{E}_Y[r(x,Y)]\,\mathbb{E}_{Y_{1:n}}[d(x,Y_{1:n})] \\
&\quad \underbrace{+\,\text{Cov}_{Y_{2:n}}\left(\frac{1}{n}r(x,y)+\frac{1}{n}\sum_{i=2}^{n}r(x,Y_i),\, d(x,y,Y_{2:n})\right)}_{\text{Term 2: Exploration–Exploitation synergy in sets containing } y} \quad - \quad \text{Cov}_{Y_{1:n}}\left(r(x,Y_1),d(x,Y_{1:n})\right). \qquad (15)
\end{aligned}
$$

We now interpret this advantage:

- *Term 1* reflects the expected contribution of generation y to the product of reward and diversity in sets that contain it. Importantly, this term depends on both the reward and diversity contributions of other generations in the set. Consequently, even if y itself does not contribute reward (or diversity), this term can still be positive provided other sampled generations contribute reward (or diversity) in expectation.
- *Term 2* captures how the presence of generation y affects the covariance between reward and diversity across sets containing y. Intuitively, this term favors generations that enable sets to simultaneously achieve high reward and high diversity. Thus, generations that help align exploration with exploitation receive positive learning signal.

Therefore, even when $r(x,y)=0$, a generation can still receive increased probability mass if it contributes diversity while other generations contribute reward, or if it helps produce sets in which reward

and diversity are positively aligned. This directly reflects the two desiderata that motivate our method. First, the objective supports *optimistic exploration*: trajectories that attempt novel reasoning strategies can receive positive learning signal even before they become individually successful. Second, the objective *intrinsically encodes the exploration–exploitation trade-off*: the advantage assigned to a trajectory depends in part on whether it helps the model form sets that jointly achieve high reward and high diversity, rather than on a manually tuned weighting coefficient. In this sense, the policy update itself internalizes the balance between exploration and exploitation.

This has a few features.

- In set reinforcement learning with polychromic objectives, as a model's accuracy improves, exploratory generations can still receive positive learning signal even when incorrect. Because the set objective shares credit among all generations in a set, incorrect but diverse generations can benefit from the presence of correct generations in the same set. Consequently, the incentive for exploration does not vanish as success rate improves.
- Furthermore, the algorithm naturally assigns advantage based on how a generation contributes to aligning exploration and exploitation across sets. This behavior arises from two components: the set RL framework and the multiplicative structure of the polychromic objective - both of which are necessary for the covariance term to emerge in the marginal set advantage.

Note that this analysis can be extended to any objective of the form $f(x, y_{1:n}) = \varphi(x, y_{1:n})\psi(x, y_{1:n})$. Insofar as these are valid objectives for set reinforcement learning that do not collapse into standard reinforcement learning, the algorithmic framework will encourage generations that satisfy both (even if they do not strictly satisfy the other, as long as other generations can compensate) while also strongly encouraging generations that help the model achieve a positive covariance between φ and ψ. One can imaging objectives of this form where $\varphi(x, y_{1:n}) = \frac{1}{n}\sum_{i=1}^{n} r_A(x, y_i)$ and $\psi(x, y_{1:n}) = \frac{1}{n}\sum_{i=1}^{n} r_B(x, y_i)$, where r_A and r_B are two different reward functions that we want our model to satisfy in tandem.

We now come to the generalized definition of the polychromic objective. The motivation for the polychromic objective comes from understanding the entropy change in set reinforcement learning as shown in Proposition 3.7. We see that entropy collapses when there is a positive covariance between the log probabilities of generations in a set and the scaffold value of the generations. Note that if this covariance is always positive, we would get entropy collapse. The idea of a polychromic objective is to ensure that the entropy is never allowed to collapse on a set of generations that is a singleton. In other words, the policy can never collapse onto sampling a single generation.

Definition 4.1. *A polychromic objective is a function*

$$\varphi : \mathcal{X} \times \mathcal{Y}^n \to \mathbb{R}$$

that factors as

$$\varphi(x, y_{1:n}) = \varphi^{(R)}(x, y_{1:n})\varphi^{(d)}(x, y_{1:n}),$$

where $\varphi^{(R)}$ and $\varphi^{(d)}$ are scalar-valued functions satisfying:

1. $\mathrm{Cov}_{y_{1:n}\sim\pi_\theta(\cdot|x)}(\varphi^{(R)}(x, y_{1:n}), \sum_{i=1}^{n} r(x, y_i)) > 0$,
2. $\mathrm{Cov}_{y_{1:n}\sim\pi_\theta(\cdot|x)}\left(\varphi^{(d)}(x, y_{1:n}), \sum_{i=1}^{n} 1\{y_i = y\}\right) < 0$ *for any y, and*
3. $\varphi^{(R)}(x, \cdot)$ *and* $\varphi^{(d)}(x, \cdot)$ *share the same range, i.e.,*

$$\inf \varphi^{(R)}(x, \cdot) = \inf \varphi^{(d)}(x, \cdot), \quad \sup \varphi^{(R)}(x, \cdot) = \sup \varphi^{(d)}(x, \cdot),$$

where the infimum and supremum are taken over all sets $y_{1:n}$.

The practical polychromic objective we have seen before aims to ensure the second requirement using a diversity function since it is, often, intractable to find an objective that satisfies the requirement for all $y \in \mathcal{Y}$.

We confirm now, through a preliminary analysis, that polychromic objectives indeed satisfy the motivation behind it.

Proposition 4.2. *For any homogeneous set $y_{1:n} = \{y\}$ where $r(x, y) = 1$, there exists $\epsilon \in (0, 1)$ such that $\Lambda_{f_{\text{poly}}}(y^{\oplus n}; \pi, x) < 0$ when $\pi_\theta(y \mid x) > \epsilon$. Furthermore, the scaffold values of these homogeneous sets satisfy the bound $\Lambda_{f_{\text{poly}}}(y; \pi, x) \leq \sqrt{\frac{p(1-p)}{n}}$.*

This result shows that once a successful generation y accumulates sufficient probability mass, the polychromic objective automatically prevents further entropy collapse onto sets that only contain this action and, when we use larger sets in the set RL framework, the upper bound on the scaffold value of a homogeneous set decreases. This is desirable since it suggests that our policy learns to generate this action without incessantly collapsing probability mass on such homogeneous sets.

Proposition 4.3. *Suppose $y_{1:n}$ is heterogeneous where each y_i is unique with probability $p \in (0, \frac{1}{n})$. Suppose exactly q of the n actions satisfy $r(x, y_i) = 1$, and that any other action $y' \notin y_{1:n}$ with $\pi_\theta(y' \mid x) > 0$ yields $r(x, y') = 0$. Then, the scaffold value of $y_{1:n}$ satisfies $\Lambda_{f_{\text{poly}}}(y_{1:n}; \pi, x) > \frac{qp^n(1-p)}{n}$.*

Note that the set in this proposition includes unsuccessful generations as well that contribute diversity. As such, there are, likely, several such heterogeneous sets (provided that the size of sets n is large enough) with positive scaffold values that attract more probability mass than homogeneous sets. Furthermore, the lower bound guarantee increases as the number of successful actions in the set increases. The polychromic objective therefore channels entropy collapse toward those sets that balance success and exploration, rather than permitting collapse onto homogeneous behaviours.

We will now generalize some of the tools used in proving properties of the polychromic objective. In particular, in proving 4.2 and 4.3, we observe that there is a particularly interesting way in which the scaffold value of an action depends on the model's ability to sample other actions. This is yet another indication of how set RL couples generations within a set. The construction of specular sets aims to make this analysis more rigorous.

5 Specular Actions

In this section, we develop preliminary tools to better understand how set reinforcement learning can couple different actions (or generations). Throughout this section, we will use the words actions and generations interchangeably. The intuition we aim to consolidate is that, in set RL, for an action to receive a positive learning signal, it requires other actions to support it by behaving in particular manners. To analyse which actions are necessary for a fixed action y to receive this learning signal, we will simply analyse what happens when we remove certain actions from the support of the policy. We call this *diminishing* actions:

Definition 5.1 (Diminishing Actions)**.** *Given an action $y \in \mathcal{Y}$ with $p_y = \pi_\theta(y \mid x)$, diminishing the action y is the mapping $\pi_\theta \to \pi_{\theta'}$ by simply sending the probability of y to be 0 and distributing p_y over all other actions in $\mathcal{Y}$ where $\pi_\theta(\cdot \mid x) > 0$. In other words, $\pi_{\theta'}(y \mid x) = 0$ and, with $\mathcal{G} = \{y' \in \mathcal{Y} \mid \pi_\theta(y' \mid x) > 0, y' \neq y\}$, we have that $\pi_{\theta'}(y' \mid x) = \pi_\theta(y' \mid x) + \frac{p_y}{|\mathcal{G}|}$ for all $y' \in \mathcal{G}$, assuming $|\mathcal{G}| > 0$. Note that $\mathcal{G}$ is the new support of the policy.*

This can be generalized to diminishing a set. Given a set of actions X and $\mathcal{G} := \{y' \in \mathcal{Y} \mid \pi_\theta(y' \mid x) > 0, y \notin X\}$, diminishing X leads to a new distribution such that $\pi_{\theta'}(y \mid x) = 0$ for all $y \in X$ and $\pi_{\theta'}(\tilde{y} \mid x) = \pi_\theta(\tilde{y} \mid x) + \frac{p_X}{|\mathcal{G}|}$ for $\tilde{y} \in \mathcal{G}$ and $p_X = \sum_{y \in X} \pi_\theta(y \mid x)$.

Note that diminishing is a valid mapping between probability distributions. Suppressing the dependency on θ' and let $g = |\mathcal{G}|$. Then, the sum of the probability of all actions under the new distribution is 1 since $\sum_{y' \neq y} (\pi(y' \mid x)) + \pi(y \mid x) = \sum_{y' \neq y, \pi_\theta(y') > 0} (\pi(y' \mid x)) + \pi(y \mid x) = \sum_{y' \neq y, \pi_\theta(y') > 0} (\pi_\theta(y' \mid x) + \frac{p_y}{g}) + 0 = 1$. Similar logic applies to the generalized case where we diminish a set of actions instead.

Now, we ask the following question: given coupling under set reinforcement learning, for a fixed action y, can we identify which other actions enable y to receive a positive learning signal? In other words, what other generations $y' \in \mathcal{Y}, y' \neq 0$ exist without which y would not receive a positive learning signal? To make this question tractable without requiring us being able to write down π_θ, we answer this question using the diminishing map.

Definition 5.2 (Λ-specular Set)**.** *Let y be a fixed action such that $\Lambda_f(y; x, \pi_\theta) > 0$. Let X be a set of actions from $\mathcal{Y}$ such that $\pi_\theta(y' \mid x) > 0$ for all $y' \in X$. We say X is a $\Lambda_{f,\theta}$-specular set to y if diminishing X causes $\Lambda_f(y; x, \pi_{\theta'})$ to be negative. In particular, we say X is strong specular if it is the smallest set for which this holds true.*

When clear from context, we will simply call X a specular (or strong specular) set. First, let us show that this construction is not completely useless. In particular, we want to show that under any set objective function, all of the action space apart from y itself cannot be its specular. In other words, f always partitions the action space such that only a non-trivial set of actions is the specular of any y.

Note that the specular set of an action y cannot contain y since diminishing X would cause $\pi_\theta(y \mid x)$ to become zero and the scaffold value of such an ation is always zero.

Proposition 5.3. *Given $\mathcal{Y}$ is a finite space, then, for any y, the entirety of $\mathcal{Y} - \{y\}$ cannot be Λ_f-specular set of y under any objective f.*

Proof. If $\mathcal{Y} - \{y\}$ was the specular of y, then collapsing the policy to a deterministic one that always outputs y would result in y getting a negative scaffold. But the scaffold value becomes exactly equal to $(1-1)d(x, y^{\oplus n}) = 0$ which is not negative. □

We look at an example of an objective that, at first sight, seems to couple almost all pairs of actions. in other words, f is an objective that rewards any pairwise dissimilarity. We show that even under such an objective, the speculars are non-trivial.

Example 5.4. Given $f(x, y_{1:n}) = 1\{\exists y_i \neq y_j : y_i, y_j \in y_{1:n}\}$, under a uniform policy, no action gets a positive scaffold value in and, therefore, there are no speculars.

For a non-uniform policy, the scaffold value of any action with probability p of being sampled can be written as $p \cdot (\sum_y \pi(y \mid x)^n - p^{n-1})$. The scaffold value has a turning point at $p^* = \left(\sum_y \pi(y \mid x)^n\right)^{1/(n-1)}$. If $p > p^*$, then it is negative; if $0 < p < p^*$, then it is positive. Suppose p is positive. Then, what are the speculars of y? This is simply any set of actions such that diminishing the set causes $\pi_{\theta'}(y \mid x)$ to be in the negative scaffold range. How large is that set of actions? Let X be the set we diminish and let $\mathcal{G} = \{y \in \mathcal{Y} \mid \pi_\theta(y \mid x) > 0, y \notin X\}$. Then, we require,

$$\pi_{\theta'}(y \mid x) = \pi_\theta(y \mid x) + \varphi > (\sum_{y' \in \mathcal{G}} (\pi_\theta(y' \mid x) + \varphi)^n)^{1/(n-1)} = (\sum_{y'} \pi_{\theta'}(y' \mid x)^n)^{1/(n-1)}$$

where $\varphi = \frac{\sum_{y' \in \mathcal{G}} \pi_\theta(y' \mid x)}{|\mathcal{G}|}$. In particular, X can *not* be all actions apart from y (confirming our previous proposition). However, we can make much stronger statements now:

1. X must be a set such that, after diminishing it, the probability of $\pi_{\theta'}(y \mid x)$ must be larger than $\frac{1}{g}$ where $g = |\text{Support of } \pi_{\theta'}(\cdot \mid x)|$.

2. The specular set X can be made smaller by keeping the highest likelihood actions from the support of π_θ that are not equal to y.

We now understand that, under any objective function and any policy, the specular set of a fixed y can be at most $\mathcal{Y} - \{y, y'\}$ for some y'. Our previous example used an objective where we intentionally tested how large we can make the specular set to be. However, this objective does not represent any learning signal that we may want to use in a training algorithm. Next, we consider the following objective:

$$f_{\text{srl}}\left(x, y_{1:n}\right) = \max_i \left(\frac{r(x, y_i)}{N(x, y_i \mid y_{1:n})} \right),$$

where $N(x, y_i \mid y_{1:n})$ is the number of times the generation y_i appears in $y_{1:n}$. Now the marginal advantages under this objective can be written as follows:

$$A^\sharp_{\text{marg}}(x, y_{\text{correct}}; f_{\text{srl}}) = \sum_{k=0}^{n-1} \binom{n-1}{k} p_\theta^k (1-p_\theta)^{n-1-k} \frac{1}{k+1} - \sum_{k=1}^{n} \binom{n}{k} p_\theta^k (1-p_\theta)^{n-k} \frac{1}{k}$$

$$A^\sharp_{\text{marg}}(x, y_{\text{incorrect}}; f_{\text{srl}}) = \sum_{k=1}^{n-1} \binom{n-1}{k} p_\theta^k (1-p_\theta)^{n-1-k} \frac{1}{k} - \sum_{k=1}^{n} \binom{n}{k} p_\theta^k (1-p_\theta)^{n-k} \frac{1}{k}.$$

In particular, note that $A^\sharp_{\text{marg}}(x, y_{\text{incorrect}}; f_{\text{srl}}) > A^\sharp_{\text{marg}}(x, y_{\text{correct}}; f_{\text{srl}})$ when p_θ reaches a certain threshold τ i.e., when $p_\theta > \tau$. Furthermore, note that the marginal set advantage of a correct generation becomes negative only in the following range:

$$A^\sharp_{\text{marg}}(x, y_{\text{correct}}; f_{\text{srl}}) < 0 \quad \iff \quad \sum_{m=1}^{n-1} \binom{n-1}{m} \frac{p_\theta^m (1-p_\theta)^{\,n-1-m}}{m(m+1)} > (1-p_\theta)^{\,n-1}.$$

Therefore, the specular set of y_{correct} is a set such that diminishing it causes p_θ to be satisfy this bound.

6 Algorithms for Set RL

In this section, we present some examples of algorithms in set reinforcement learning.

6.1 Poly-PPO

Polychromic Proximal Policy Optimisation (Poly-PPO) is an algorithm that aims to do a mix of set reinforcement learning (as in the idealized formulation presented in §2.1) and standard reinforcement learning. Recall that in the idealised formulation, we sample a set of n actions at every state visited.

1. If the initial state of the environment is fixed, we can sample a set of trajectories from the initial state and optimize the policy at the initial state using set reinforcement learning since we will naturally get n actions sampled at the initial state.

2. If we can reset our environment to any desired state, then we can select a set of states from the trajectories we have sampled so far and generate $n-1$ more trajectories from those states to be able to do set reinforcement learning there. This is called *vine sampling* and was studied by [SLM+17] and [KAP+25] amongst others.

The algorithm Poly-PPO operates under the assumption that we can reset our environment to any state. At a high-level, the algorithm first samples trajectories using vine sampling. Then, at any state where we have sampled a set (larger than a singleton) of trajectories, the algorithm optimises the policy using set reinforcement learning. At all other states, i.e. states from which we did not sample a set of trajectories, the algorithm optimises the policy using standard reinforcement learning. The pseudocode is provided in algorithm 2. We refer the reader to [HOX+26] for implementation details. Note that:

1. The algorithm induces a balance between exploration and exploitation in two different ways. Firstly, at all the rollout states where we can use set reinforcement learning, we encourage a balance between exploration and exploitation by optimising the polychromic objective. Secondly, at all other states, when we apply just the GAE advantage, we are encouraging pure exploitation since the advantage estimator is in the standard RL framework using the standard return maximization objective. For long horizon tasks, since the number of rollout states is much smaller than the non-rollout state, we use a window hyperparameter T such that, at each rollout state s_t, we apply the set RL advantage to all states and actions from s_t up to s_{t+T}.

2. The mix of set reinforcement learning and standard reinforcement learning allows the model to explore-then-exploit *within* each generation. For classical RL environments that do not involve foundation models, this was a useful inductive bias. When training large language models with the polychromic objective, this inductive bias is not required as the model, by itself, learns to explore and exploit both across and within generations. Regardless, vine sampling is a powerful technique that allows us to get closer to the sequential set RL formulation we discussed in §2 where we sample a set of actions at every state visited.

This algorithm was proposed in [HOX+26] and the original motivation was to optimise the polychromic objective in §4, however the recipe can be adapted for optimising other set RL objectives as well.

Algorithm 2 Polychromic PPO

for iteration $= 1, 2, \ldots$ **do**
  Collect trajectories under π_β; rollout vines $\tau_{1:N}$ from rollout states
  if s_t is a rollout state **then**
    Construct sets, $G_1, \ldots, G_K$ of n trajectories from s_t
    // Given s, let $\mathcal{G}(a)_{1:p}$ be all sets containing a trajectory starting with a
    Set $\hat{A}(s_t, a_t) = \frac{1}{p}\sum_{i=1}^{p} f_{\text{poly}}(s_t, \mathcal{G}(a_t)_i) - \frac{1}{K}\sum_{j=1}^{K} f_{\text{poly}}(s_t, G_j)$
  else
    Compute $\hat{A}(s_t, a_t)$ via GAE
  end if
  Update π_θ for K epochs on minibatches $\mathcal{B}$ by maximizing the PPO objective
  Set $\pi_\beta \leftarrow \pi_\theta$
end for

6.2 Poly-EPO

Polychromic Exploratory Policy Optimisation (Poly-EPO) is an algorithm that leverages the general recipe in algorithm 1 for optimizing the practical polychromic objective proposed in [OHR+26].

We first discuss the implementation of the diversity function. There are a few considerations to keep in mind. Firstly, we require the computation of the diversity of each set, i.e. $d(x, y_1, \cdots, y_n)$, to be computationally practical. This is because our set RL recipe constructs all $K = \binom{N}{n}$ sets of size n from the N generations per prompt; if scoring each set under the polychromic objective is computationally expensive, then the runtime complexity of our algorithm will increase rapidly as we scale up the algorithm to a setting where we sample more generations per prompt. Secondly, we much be able to able to do this computation for a wide variety of problem settings. Language models are post-trained on problems in mathematics, coding, instruction-following, etc., and the training set for each of these settings can be quite large. We require a method for measuring diversity of sets of generations that can be easily transferred to a wide variety of settings.

The main ingredient we use is an language model judge (LM-judge) that is used to cluster the responses sampled from our policy. Since LM-judges are already incorporated in post-training methods in a wide variety of settings such as proof-writing and non-verifiable domains, this allows Poly-EPO to benefit from the post-training infrastructure that is already widely adopted. Furthermore, clustering responses based on semantic similarity is a substantially simpler task than solving the problem, which allows us to leverage the generator-verifier gap in designing our method.

Recall that, for each prompt x, we first sample N independent generations and, then, construct sets of size n from them. We first take all N generations conditioned on the same prompts and ask an LM-judge to cluster them based on their semantic similarity in their reasoning strategy. Let $\mathcal{C}(y_i) \in \{1, \cdots, N\}$ be the cluster to which generation y_i belongs. Then, we define the diversity function for each set $\{y_{(j_1)}, \cdots, y_{(j_n)}\}$ to be:

$$d\big(x, y_{(j_1)}, \ldots, y_{(j_n)}\big) = \frac{\text{No. of clusters in } (y_{(j_1)}, \ldots, y_{(j_n)})}{n} = \frac{\left|\left\{\mathcal{C}\big(y_{(j_1)}\big), \ldots, \mathcal{C}\big(y_{(j_n)}\big)\right\}\right|}{n}$$

Note that $d(x, y_{(j_1)}, \cdots, y_{(j_n)}) \in [0, 1]$, which is required for the polychromic objective. To increase the reliability of the judge, we use in-context learning to steer the judge to cluster them based on the strategy (both high-level and low-level) used in each generation, as opposed to differences in tone, style, etc. Now, we have all the ingredients for computing the score of each set under the polychromic objective. Since scoring each set is computationally straightforward, we can construct all $K = \binom{N}{n}$ sets in our set RL recipe from algorithm 1.

References

[HOX+26] Jubayer Ibn Hamid, Ifdita Hasan Orney, Ellen Xu, Chelsea Finn, and Dorsa Sadigh. Polychromic objectives for reinforcement learning, 2026.

[KAP+25] Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. Vineppo: Refining credit assignment in rl training of llms, 2025.

[KL02a] Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, ICML '02, page 267–274, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

[KL02b] Sham M. Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *International Conference on Machine Learning*, 2002.

[LZY+25] Tianjian Li, Yiming Zhang, Ping Yu, Swarnadeep Saha, Daniel Khashabi, Jason Weston, Jack Lanchantin, and Tianlu Wang. Jointly reinforcing diversity and quality in language model generations, 2025.

[OHR+26] Ifdita Hasan Orney, Jubayer Ibn Hamid, Shreya S Ramanujam, Shirley Wu, Hengyuan Hu, Noah Goodman, Dorsa Sadigh, and Chelsea Finn. Poly-epo: Training exploratory reasoning models, 2026.

[SLM+17] John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust region policy optimization, 2017.

[SWD+17] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[SWZ+24] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024.

[WK25] Christian Walder and Deep Karkhanis. Pass@k policy optimization: Solving harder reinforcement learning problems, 2025.