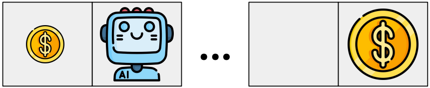
Beyond Optimism: Exploration With Partially Observable Rewards

Simone Parisi
Alireza Kazemipour
Michael Bowling

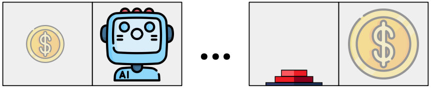
38th Conference on Neural Information Processing Systems (NeurIPS 2024)

Optimism Works When **Rewards Are Observable**



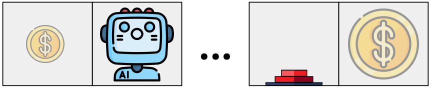
- Small and easy-to-find reward to the left.
- Large and hard-to-find reward to the far right.
- Optimistic algorithms (e.g., Q-Learning with optimistic initialization) would solve this problem easily.

But Fails If Rewards Are Partially Observable



- Coin rewards are observable only if the agent **pushes the button first**.
- But the agent will also receives **negative rewards** in the meantime.

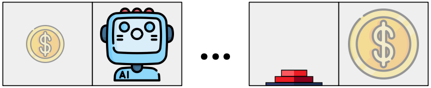
But Fails If Rewards Are Partially Observable



- Coin rewards are observable only if the agent **pushes the button first**.
- But the agent will also receives **negative rewards** in the meantime.

- Coin rewards **do exist** even when not observed, so the optimal policy is to ignore the button and collect the large coin ...
- ... but the agent must push the button first **(suboptimal action)** to learn the optimal policy!

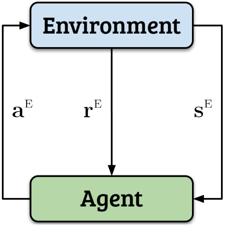
But **Fails** If Rewards Are Partially Observable



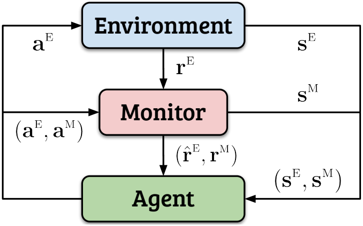
- Coin rewards are observable only if the agent **pushes the button first**.
- But the agent will also receives **negative rewards** in the meantime.

- Coin rewards **do exist** even when not observed, so the optimal policy is to ignore the button and collect the large coin ...
- ... but the agent must push the button first **(suboptimal action)** to learn the optimal policy!

Optimism $\neq$ ***Suboptimal Actions***



MDP



Monitored MDP

- The **monitor is a separate MDP** "on the top" of the environment.
- The goal is to maximize $\sum_{t=1}^{\infty} \gamma^t (r_t^{\mathrm{E}} + r_t^{\mathrm{M}})$ ...
- ... but the agent **observes proxy rewards** $\hat{r}_t^{\mathrm{E}} \sim \mathcal{M}(r_t^{\mathrm{E}}, s_t^{\mathrm{M}}, a_t^{\mathrm{M}})$ instead of true rewards r_t^{E} ...
- ... and **sometimes $\hat{r}_t^{\mathrm{E}} = \text{NaN}$!**

[*"Monitored Markov Decision Processes"*, AAMAS 2024]

Directed Reward-Free Exploration-Exploitation

$(s^{\mathrm{G}}, a^{\mathrm{G}}) = \arg\min_{s,a} N_t(s,a)$ `// goal: the least-visited state-action pair`

$\beta_t = {}^{\log t}/_{N_t(s^{\mathrm{G}}, a^{\mathrm{G}})}$ `// how much did we visit it?`

if $\beta_t > \bar{\beta}$ **then return** $\rho(a \mid s_t, s^{\mathrm{G}}, a^{\mathrm{G}})$ `// explore: follow goal-conditioned policy`

else return $\arg\max_a \widehat{Q}(s_t, a)$ `// exploit: follow the greedy policy`

Directed Reward-Free Exploration-Exploitation

$\boxed{(s^{\mathrm{G}}, a^{\mathrm{G}}) = \arg\min_{s,a} N_t(s,a)}$ // goal: the least-visited state-action pair
$\beta_t = {}^{\log t}/_{N_t(s^{\mathrm{G}}, a^{\mathrm{G}})}$ // how much did we visit it?
if $\beta_t > \bar{\beta}$ **then return** $\boxed{\rho(a \mid s_t, s^{\mathrm{G}}, a^{\mathrm{G}})}$ // explore: follow goal-conditioned policy
else return $\arg\max_a Q(s_t, a)$ // exploit: follow the greedy policy

Directed Reward-Free Exploration-Exploitation

$\boxed{(s^{G}, a^{G}) = \arg\min_{s,a} N_t(s, a)}$ `// goal: the least-visited state-action pair`
$\beta_t = {}^{\log t}/_{N_t(s^{G}, a^{G})}$ `// how much did we visit it?`
if $\beta_t > \bar{\beta}$ **then return** $\boxed{\rho(a \mid s_t, s^{G}, a^{G})}$ `// explore: follow goal-conditioned policy`
else return $\arg\max_a Q(s_t, a)$ `// exploit: follow the greedy policy`

$\rho(a \mid s_t, s^G, a^G)$ should ***reach any goal pair "as fast as possible"****. How?*

Directed Reward-Free Exploration-Exploitation

$\boxed{(s^{G}, a^{G}) = \arg\min_{s,a} N_t(s,a)}$ // goal: the least-visited state-action pair

$\beta_t = {}^{\log t}/_{N_t(s^{G}, a^{G})}$ // how much did we visit it?

if $\beta_t > \bar{\beta}$ **then return** $\boxed{\rho(a \,|\, s_t, s^{G}, a^{G})}$ // explore: follow goal-conditioned policy

else return $\arg\max_a Q(s_t, a)$ // exploit: follow the greedy policy

*$\rho(a \,|\, s_t, s^G, a^G)$ should **reach any goal pair "as fast as possible"**. How?*

It **maximizes the goal occurrences**, i.e.,

$$\rho^*(s \mid a, s_i, a_j) = \arg\max_{\rho} S^{\rho}_{s_i a_j}(s, a)$$

$$S^{\rho}_{s_i a_j}(s_t, a_t) = \mathbb{E}\left[\sum_{k=t}^{\infty} \gamma^{k-t} \mathbb{1}_{\{s_k = s_i, a_k = a_j\}}\right] \quad \Leftarrow \textbf{S-function}$$

$$\mathbb{1}_{\{s_k = s_i, a_k = a_j\}} = \begin{cases} 1 & \text{if } s_k = s_i \text{ and } a_k = a_j \\ 0 & \text{otherwise} \end{cases} \quad \Leftarrow \textbf{Successor features}$$

Directed Reward-Free Exploration-Exploitation

$\boxed{(s^{\mathrm{G}}, a^{\mathrm{G}}) = \arg\min_{s,a} N_t(s,a)}$ // goal: the least-visited state-action pair
$\beta_t = {}^{\log t}/_{N_t(s^{\mathrm{G}}, a^{\mathrm{G}})}$ // how much did we visit it?
if $\beta_t > \bar{\beta}$ **then return** $\boxed{\rho(a \mid s_t, s^{\mathrm{G}}, a^{\mathrm{G}})}$ // explore: follow goal-conditioned policy
else return $\arg\max_a Q(s_t, a)$ // exploit: follow the greedy policy

$\rho(a \mid s_t, s^G, a^G)$ should ***reach any goal pair "as fast as possible"****. How?*
It **maximizes the goal occurrences**, i.e.,

$$\rho^*(s \mid a, s_i, a_j) = \arg\max_{\rho} S^{\rho}_{s_i a_j}(s, a)$$

$$S^{\rho}_{s_i a_j}(s_t, a_t) = \mathbb{E}\left[\sum_{k=t}^{\infty} \gamma^{k-t} \mathbb{1}\{s_k = s_i, a_k = a_j\}\right]$$ ⇐ **S-function**

$$\mathbb{1}\{s_k = s_i, a_k = a_j\} = \begin{cases} 1 & \text{if } s_k = s_i \text{ and } a_k = a_j \\ 0 & \text{otherwise} \end{cases}$$ ⇐ **Successor features**

Exploration does not depend on the reward observability!

Results (Expected Return)