FROM POLICY GRADIENT TO POLICY PROXIMAL POLICY OPTIMIZATION

Shuo Liu
Computer Science
Northeastern University
shuo.liu2@northeastern.edu

ABSTRACT

This article discusses PPO methods and the recent advances.

1 POLICY GRADIENT (PG)

Compared with value-based methods (Q-learning), Policy-based methods aim directly at learning the parameterized policy that can select actions without consulting a value function. PG methods seek to maximize a performance measure $J(\theta)$ with the policy's parameter θ, where the updates approximate gradient ascent in J,[1]

$$\theta^{(i+1)} \leftarrow \theta^{(i)} + \alpha \nabla J(\theta^{(i)}). \tag{1}$$

There are 2 main advantages of PG methods,

- Approximating policy can approach a deterministic policy, whereas ϵ-greedy always has probability of selecting a random action;
- With continuous policy parameterization, the action probabilities change smoothly as a function of the learned parameter, whereas ϵ-greedy may change dramatically for an arbitrarily small change in the estimated action values.

Since the major purpose of this article is to introduce PPO methods from PG, we omit some other important forms of PG here. Readers can find them in the Appendix.

1.1 PG THEOREM

An intuitive way to calculate Equation 1 is to replace $J(\theta)$ with $V^{\pi_\theta}(s_0)$.[2] However, the calculation is hard as it directly depends on both the action selection and indirectly the distribution of states following the target selection. PG theorem provides a nice reformulation of the derivative of the objective function to not involve the state distribution derivation.

Theorem 1. *Taking the state-value function as the optimizing target, the objective gradient follows,*

$$\nabla J(\theta) \propto \sum_s d^\pi(s) \sum_a Q^\pi(s,a) \nabla \pi(a|s), \tag{2}$$

where $d^\pi(s)$ is the stationary distribution of the policy π_θ.

[1] All methods following this schema are PG, whether or not they also learn an approximate value function.

[2] To simplify the notation, we omit θ in the subscripts, superscripts, and gradient operators, assuming π is a function of θ and all gradients are implicit with respect to θ, i.e., $V^\pi \equiv V^{\pi_\theta}$, $Q^\pi \equiv Q^{\pi_\theta}$ and $\nabla \equiv \nabla_\theta$.

To sample with expectation equals or approximates the expression Equ. 2,

$$\begin{aligned}\nabla J(\theta) &\propto \sum_s d^\pi(s) \sum_a Q^\pi(s,a) \nabla \pi(a|s) \\ &= \mathbb{E}_{d^\pi}\left[\sum_a Q^\pi(s,a) \nabla \pi(a|s)\right] \\ &= \mathbb{E}_{d^\pi}\left[\sum_a \pi(a|s) Q^\pi(s,a) \frac{\nabla \pi(a|s)}{\pi(a|s)}\right] \\ &= \mathbb{E}_\pi\left[Q^\pi(s,a) \frac{\nabla \pi(a|s)}{\pi(a|s)}\right] \\ &= \mathbb{E}_\pi\left[Q^\pi(s,a) \nabla \ln \pi(a|s)\right].\end{aligned} \tag{3}$$

The eligibility vector $\nabla \ln \pi(a|s)$ is the only place the policy parameterization appears, which can be omitted $L(\theta) = \mathbb{E}_\pi[Q^\pi(s,a)]$ since it will be automatically recovered when differentiating.

1.2 PG WITH BASELINE

Theorem 2. *PG theorem can be generalized to include a comparison of the action value to an arbitrary baseline $b(s)$, as long as $b(s)$ does not depend on a, and this will reduce the variance while keeping it unbiased.*

$$\begin{aligned}\nabla J(\theta) &\propto \sum_s d^\pi(s) \sum_a (Q^\pi(s,a) - b(s)) \nabla \pi(a|s) \\ &= \mathbb{E}_\pi\left[(Q^\pi(s,a) - b(s)) \nabla \ln \pi(a|s)\right].\end{aligned} \tag{4}$$

According to the Theorem 2, the expected return $Q(s,a)$ in Theorem 1 can be replaced by G (expected return of the full or following trajectory by Monte Carlo), A (advantage by Generalized Advantage Estimation or state-value prediction), and δ (TD-residual by critic prediction).

1.3 OFF-POLICY PG

Off-policy sampling reuses any past episodes, which has a higher efficiency and brings more exploration. To make PG off-policy, we adjust it with an importance weight $\frac{\pi(a|s)}{\beta(a|s)}$ to correct the mismatch between behavior and target policies.

$$\begin{aligned}\nabla J(\theta) &= \nabla\left(\sum_s d^\beta(s) V^\pi(s)\right) \\ &= \nabla\left(\sum_s d^\beta(s) \sum_a \pi(a|s) Q^\pi(s,a)\right) \\ &= \sum_s d^\beta(s) \sum_a (\nabla \pi(a|s) Q^\pi(s,a) + \pi(a|s) \nabla Q^\pi(s,a)) \\ &\overset{\text{(i)}}{\approx} \sum_s d^\beta(s) \sum_a Q^\pi(s,a) \nabla \pi(a|s) \\ &= \mathbb{E}_{d^\beta}\left[\sum_a \beta(a|s) \frac{\pi(a|s)}{\beta(a|s)} Q^\pi(s,a) \frac{\nabla \pi(a|s)}{\pi(a|s)}\right] \\ &= \mathbb{E}_\beta\left[\frac{\pi(a|s)}{\beta(a|s)} Q^\pi(s,a) \nabla \ln \pi(a|s)\right],\end{aligned} \tag{5}$$

where $d^\beta(s)$ is the stationary distribution of the behavior policy β, and Q^π is the Q-function estimated regard to the target policy π. Because of hard computation in reality (i), we ignore the approximation term $\nabla Q^\pi(s,a)$.

2 PROXIMAL POLICY OPTIMIZATION (PPO)

In this section, we introduce standard PPO and it variants in different domains.

2.1 CLIP-PPO

Schulman et al. (2017) proposed the standard PPO that uses a clipped surrogate objective to ensure the policy updates are small and controlled (proximal). Since the advantage under current policy is intangible, we can use Generalized Advantage Estimation (GAE) of the last policy to estimate $\hat{A}^{\pi_{\theta_{\text{old}}}}$ to reduce the variance of policy gradient methods and maintain low bias Schulman et al. (2015),

$$J^{\text{CLIP}}(\theta) = \mathbb{E}_{\pi_{\theta_{\text{old}}}}\left[\min\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_{\text{old}}}(a|s)}\hat{A}^{\pi_{\theta_{\text{old}}}}(s,a), \text{clip}(\frac{\pi_\theta(a|s)}{\pi_{\theta_{\text{old}}}(a|s)}, 1-\epsilon, 1+\epsilon)\hat{A}^{\pi_{\theta_{\text{old}}}}(s,a)\right)\right], \quad (6)$$

where $\hat{A}_t^{\text{GAE}} = \sum_{l=0}^{\infty}(\gamma\lambda)^l\delta_{t+l}$, δ is the TD error, and λ is a hyperparameter controlling the trade-off between bias and variance. Note that the clipping could also occur in the value network to stabilize the training process.

The objective function can be augmented with an entropy term to encourage exploration,

$$J^{\text{CLIP+}}(\theta) = \mathbb{E}_{\pi_{\theta_{\text{old}}}}\left[J^{\text{CLIP}}(\theta) - c\sum_a \pi_\theta(a|s)\log\pi_\theta(a|s))\right]. \quad (7)$$

Algorithm 1 Proximal Policy Optimization (PPO)

Initialize: policy parameter θ for actor network π_θ, parameter w for critic network V_w, replay memory $\mathcal{D}$
for each iteration **do**
  Generate an episode following policy $\pi_{\theta_{\text{old}}}$ and store it into $\mathcal{D}$
  Estimate reward-to-go $\hat{R}$ and $\hat{A}^{\pi_{\theta_{\text{old}}}}$ using GAE
  for each mini-batch of N transitions $\{s_i, a_i, r_{i+1}, s_{i+1}\}$ **do**
    Compute $J^{\text{CLIP+}}(\theta)$ for all samples according to Equ. 7
    $w \leftarrow w + \alpha_w \frac{1}{N}\sum_i \nabla_w(V_w(s_i) - \hat{R}(s_i, a_i))^2$
    $\theta \leftarrow \theta + \alpha_\theta \frac{1}{N}\sum_i \nabla_\theta J^{\text{CLIP+H}}(\theta)$
  end for
  $\theta_{\text{old}} \leftarrow \theta$
end for

2.2 KL-PPO

Another formulation of PPO to improve training stability, so-called Trust Region Policy Optimization (TRPO), enforces a KL divergence constraint on the size of the policy update at each iteration Schulman et al. (2017).

$$J^{\text{KL}}(\theta) = \mathbb{E}_{\pi_{\theta_{\text{old}}}}\left[\frac{\pi_\theta(a|s)}{\pi_{\theta_{\text{old}}}(a|s)}\hat{A}^{\pi_{\theta_{\text{old}}}}(s,a) - c\mathcal{D}_{\text{KL}}(\pi_{\theta_{\text{old}}}\|\pi_\theta)\right], \quad (8)$$

where $\mathcal{D}_{\text{KL}}(\pi_{\theta_{\text{old}}}\|\pi_\theta) = \sum_a \pi_{\theta_{\text{old}}}(a|s)\log\frac{\pi_{\theta_{\text{old}}}(a|s)}{\pi_\theta(a|s)}$.

Sometimes, the KL-penalty can be combined with policy clipping to achieve better performance in practice.

Adaptive-KL-PPO Schulman et al. (2017) also mentioned Adaptive-KL-PPO, where the KL penalty coefficient is adjusted dynamically. If the policy update is too aggressive ($\mathcal{D}_{\text{KL}} \gg \mathcal{D}_{\text{threshold}}$), c is increased to penalize large updates; else if the update is too conservative ($\mathcal{D}_{\text{KL}} \ll \mathcal{D}_{\text{threshold}}$), c is decreased to allow larger updates.

2.3 MULTI-AGENT PPO

In the multi-agent setting, the PPO algorithm can be implemented independently (IPPO) or by a centralized critic (MAPPO). In IPPO, each agent has its own actor and critic and learns independently according to a joint reward de Witt et al. (2020). Like IPPO, MAPPO employs weight sharing between agents' critics, and the advantage in MAPPO is estimated through joint GAE Yu et al. (2022).

$$
\begin{aligned}
J^{\text{IPPO}}(\theta_i) &= \mathbb{E}_{\pi_{\theta_{i,\text{old}}}}\left[\min\left(\frac{\pi_{\theta_i}(a|s)}{\pi_{\theta_{i,\text{old}}}(a|s)}\hat{A}^{\pi_{\theta_{i,\text{old}}}}(s,a), \text{clip}(\frac{\pi_{\theta_i}(a|s)}{\pi_{\theta_{i,\text{old}}}(a|s)}, 1-\epsilon, 1+\epsilon)\hat{A}^{\pi_{\theta_{i,\text{old}}}}(s,a)\right)\right], \\
J^{\text{MAPPO}}(\theta_i) &= \mathbb{E}_{\pi_{\theta_{\text{old}}}}\left[\min\left(\frac{\pi_{\theta_i}(a|s)}{\pi_{\theta_{i,\text{old}}}(a|s)}\hat{\boldsymbol{A}}^{\pi_{\theta_{\text{old}}}}(s,a), \text{clip}(\frac{\pi_{\theta_i}(a|s)}{\pi_{\theta_{i,\text{old}}}(a|s)}, 1-\epsilon, 1+\epsilon)\hat{\boldsymbol{A}}^{\pi_{\theta_{\text{old}}}}(s,a)\right)\right].
\end{aligned} \tag{9}
$$

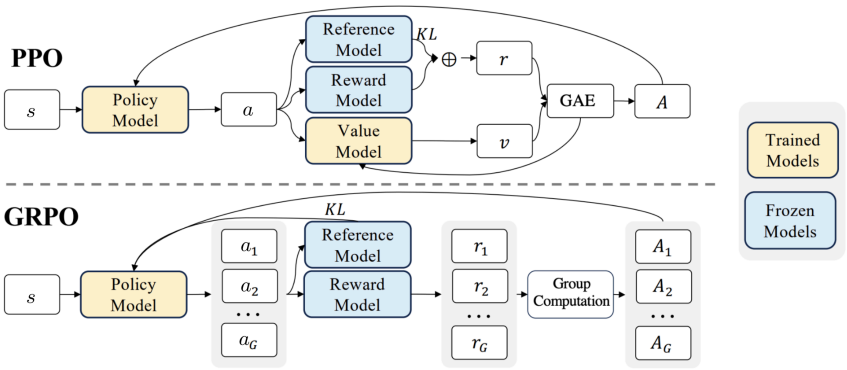
Note that there are some other instantiations of IPPO, but not all of them are vulnerable to non-convergence issues. The one with full actor critic parameter or information sharing can be regarded as a centralized method. Besides, for cases where a general solution is still intangible even with parameter sharing (e.g. the exclusive game), heterogeneous-agent PPO allows the agents to take turns learning by using others' information, which can work well with strong assumptions.[3]

2.4 GROUP RELATIVE POLICY OPTIMIZATION (GRPO)

As DeepSeek has made a splash in the LLM community, the RL method GRPO involved has received a lot of attention Zhihong Shao (2024). GRPO is a variant of PPO, where the advantage is estimated using group-relative comparisons rather than GAE. This approach eliminates the critic model, which improves the training efficiency and stability. The DeepSeek framework consists of: (i) a frozen *reference model*, which is a stable baseline for computing rewards; (ii) a given *reward model*, responsible for evaluating generated outputs and assigning scores; (iii) a *value model*, which estimates the expected return of a given state to aid in policy optimization; and (iv) a *policy model*, which generates $|\mathcal{G}|$ responses and is continuously updated to improve performance based on feedback from the other components. The learning objective for GRPO is,

$$
J^{\text{GRPO}}(\theta) = \mathbb{E}_{\pi_{\theta_{\text{old}}}, i\in\mathcal{G}}\left[\min\left(\frac{\pi_\theta(a_i|s,\vec{a}_i)}{\pi_{\theta_{\text{old}}}(a_i|s,\vec{a}_i)}\hat{A}^{\mathcal{G}}, \text{clip}(\frac{\pi_\theta(a_i|s,\vec{a}_i)}{\pi_{\theta_{\text{old}}}(a_i|s,\vec{a}_i)}, 1-\epsilon, 1+\epsilon)\hat{A}^{\mathcal{G}}\right) - c\mathcal{D}_{\text{KL}}(\pi_{\text{ref}}\|\pi_\theta)\right], \tag{10}
$$

where the advantage $\hat{A}_i^{\mathcal{G}} = \frac{r_i - \text{mean}(r)}{\text{std}(r)}$ is estimated by grouped actions produced at the same state. $\mathcal{D}_{\text{KL}}(\pi_{\text{ref}}\|\pi_\theta) = \frac{\pi_{\text{ref}}(a_i|s,\vec{a}_i)}{\pi_\theta(a_i|s,\vec{a}_i)} - \ln\frac{\pi_{\text{ref}}(a_i|s,\vec{a}_i)}{\pi_\theta(a_i|s,\vec{a}_i)} - 1$ is a positive unbiased estimator, which measures the difference between the policy of trained model and reference model (like direct policy optimization).



[3] A great example is PettingZoo's agent cycle and parallel environments.

REFERENCES

Max Brenner. Illustrated comparison of different distributed versions of ppo. *Medium*, February 28 2023. URL https://medium.com.

Christian Schroeder de Witt, Tarun Gupta, Denys Makoviichuk, Viktor Makoviychuk, Philip H. S. Torr, Mingfei Sun, and Shimon Whiteson. Is independent learning all you need in the starcraft multi-agent challenge?, 2020. URL https://arxiv.org/abs/2011.09533.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of ppo in cooperative, multi-agent games, 2022. URL https://arxiv.org/abs/2103.01955.

Qihao Zhu Runxin Xu Junxiao Song Mingchuan Zhang Y.K. Li Y. Wu Daya Guo Zhihong Shao, Peiyi Wang. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

A OMITTED PROOFS

This section shows the omitted (but non-trivial) proofs.

A.1 PROOF THEOREM 1

Proof. The gradient of V function can be written in terms of Q function,

$$\begin{aligned}
\nabla V^\pi(s) &= \nabla \left[\sum_a \pi(a|s) Q^\pi(s,a) \right] \\
&= \sum_a \left[\nabla \pi(a|s) Q^\pi(s,a) + \pi(a|s) \nabla Q^\pi(s,a) \right] \\
&= \sum_a \left[\nabla \pi(a|s) Q^\pi(s,a) + \pi(a|s) \nabla \sum_{s'} P(s'|s,a)(r + V^\pi(s')) \right] \\
&\overset{\text{(i)}}{=} \sum_a \left[\nabla \pi(a|s) Q^\pi(s,a) + \pi(a|s) \sum_{s'} P(s'|s,a) \nabla V^\pi(s') \right],
\end{aligned}$$

where we can have the derivation (i) since r only depends on the environment dynamics.

Let $\phi(s) = \sum_a \nabla \pi(a|s) Q^\pi(s,a)$. We use $\rho^\pi(s \to x, k)$ to represent the probability of transitioning from state s to x with policy π after k steps, i.e., $\rho^\pi(s \to s', 1) = \sum_a \pi(a|s) P(s'|s,a)$. Thus, we can unroll the recursive form as below,

$$\begin{aligned}
\nabla V^\pi(s) &= \phi(s) + \sum_a \pi(a|s) \sum_{s'} P(s'|s,a) \nabla V^\pi(s') \\
&= \phi(s) + \sum_{s'} \rho^\pi(s \to s', 1) \nabla V^\pi(s') \\
&= \phi(s) + \sum_{s'} \rho^\pi(s \to s', 1) \left[\phi(s') + \sum_{s''} \rho^\pi(s' \to s'', 1) \nabla V^\pi(s'') \right] \\
&= \phi(s) + \sum_{s'} \rho^\pi(s \to s', 1) \phi(s') + \sum_{s''} \rho^\pi(s \to s'', 2) \nabla V^\pi(s'') \\
&= \phi(s) + \sum_{s'} \rho^\pi(s \to s', 1) \phi(s') + \sum_{s''} \rho^\pi(s \to s'', 2) \phi(s'') + \sum_{s'''} \rho^\pi(s \to s''', 3) \nabla V^\pi(s''') \\
&\quad \vdots \\
&= \sum_{k=0}^{\infty} \sum_x \rho^\pi(s \to x, k) \phi(x)
\end{aligned}$$

We use $\eta(s)$ to represent the expected number of visits for s in a single episode (in episodic case $\sum_s \eta(s)$ is the averaged length of an episode; in continuous case $\sum_s \eta(s) = 1$). By plugging it into the object function J,

$$\begin{aligned}
\nabla J(\theta) &= \nabla V^\pi(s_0) \\
&= \sum_s \left(\sum_{k=0}^{\infty} \rho^\pi(s_0 \to s, k) \right) \sum_a \nabla \pi(a|s) Q^\pi(s,a) \\
&= \sum_s \eta(s) \sum_a \nabla \pi(a|s) Q^\pi(s,a) \\
&\overset{\text{norm}}{=} \left(\sum_s \eta(s) \right) \left(\sum_s \frac{\eta(s)}{\sum_s \eta(s)} \right) \sum_a \nabla \pi(a|s) Q^\pi(s,a) \\
&\propto \sum_s d^\pi(s) \sum_a \nabla \pi(a|s) Q^\pi(s,a)
\end{aligned}$$

□

A.2 PROOF OF THEOREM 2

Proof. We first prove REINFORCE with baseline is unbiased,

$$\begin{aligned}
&\mathbb{E}_{d^\pi}\left[\sum_a (Q^\pi(s,a) - b(s))\nabla \ln \pi(a|s)\right] \\
&= \mathbb{E}_{d^\pi}\left[\sum_a Q^\pi(s,a)\nabla \ln \pi(a|s)\right] - \mathbb{E}_{d^\pi}\left[\sum_a b(s)\nabla \ln \pi(a|s)\right] \\
&= \mathbb{E}_{d^\pi}\left[\sum_a Q^\pi(s,a)\nabla \ln \pi(a|s)\right] - \mathbb{E}_{d^\pi}\left[b(s)\nabla \sum_a \ln \pi(a|s)\right] \\
&= \mathbb{E}_{d^\pi}\left[\sum_a Q^\pi(s,a)\nabla \ln \pi(a|s)\right] - \mathbb{E}_{d^\pi}\left[b(s)\nabla 1\right] \\
&= \mathbb{E}_{d^\pi}\left[\sum_a Q^\pi(s,a)\nabla \ln \pi(a|s)\right],
\end{aligned}$$

and the variance of REINFORCE with baseline is,

$$\begin{aligned}
&\mathbb{V}_{d^\pi}\left[\sum_a (Q^\pi(s,a) - b(s))\nabla \ln \pi(a|s)\right] \\
&\overset{\text{(i)}}{\approxeq} \sum_a \mathbb{E}_{d^\pi}\left[((Q^\pi(s,a) - b(s))\nabla \ln \pi(a|s))^2\right] - \left(\mathbb{E}_{d^\pi}\left[\sum_a (Q^\pi(s,a) - b(s))\nabla \ln \pi(a|s)\right]\right)^2 \\
&\overset{\text{(ii)}}{\approxeq} \sum_a \mathbb{E}_{d^\pi}\left[(Q^\pi(s,a) - b(s))^2\right]\mathbb{E}_{d^\pi}\left[(\nabla \ln \pi(a|s))^2\right] - \left(\mathbb{E}_{d^\pi}\left[\sum_a Q^\pi(s,a)\nabla \ln \pi(a|s)\right]\right)^2 \\
&< \sum_a \mathbb{E}_{d^\pi}\left[(Q^\pi(s,a)\nabla \ln \pi(a|s))^2\right] - \left(\mathbb{E}_{d^\pi}\left[\sum_a Q^\pi(s,a)\nabla \ln \pi(a|s)\right]\right)^2 \\
&\overset{\text{(iii)}}{\approxeq} \mathbb{E}_{d^\pi}\left[\left(\sum_a Q^\pi(s,a)\nabla \ln \pi(a|s)\right)^2\right] - \left(\mathbb{E}_{d^\pi}\left[\sum_a Q^\pi(s,a)\nabla \ln \pi(a|s)\right]\right)^2 \\
&= \mathbb{V}_{d^\pi}\left[\sum_a Q^\pi(s,a)\nabla \ln \pi(a|s)\right].
\end{aligned}$$

In approximations (i) and (iii), we only keep the quadratic term and omit the products, but this won't affect the property of the inequality because the deduction loss caused by $\prod_a (Q^\pi(s,a) - b(s))\nabla \ln \pi(a|s)$ is less than the increase we compensate for $\prod_a Q^\pi(s,a)\nabla \ln \pi(a|s)$. In approximation (ii), we assume independence among the values involved in the expectation for factorization. This reveals that REINFORCE has less variance when using a baseline; and when $b(s) \approx V^\pi(s)$, the variance reaches optimal. □

B OTHER FORMS OF PG

Since the major purpose of this article is to introduce PPO methods from PG, we omit some other important forms of PG in the main body. Here are the complements.

B.1 DETERMINISTIC PG

Sometimes we hope the policy function to be deterministic to reduce the gradient estimation variance and improve the exploration efficiency for continuous action space [4] (i.e., a decision $a = \mu_\theta(s)$). PG for a deterministic policy in continuous action space is,

$$\begin{aligned}\nabla_\theta J(\theta) &= \nabla_\theta \left(\int_s d^\mu(s) V^\mu(s) ds \right) \\ &= \nabla_\theta \left(\int_s d^\mu(s) Q^\mu(s,a)|_{a=\mu_\theta(s)} ds \right) \\ &\overset{\text{(i)}}{=} \int_s d^\mu(s) \nabla_\theta \mu_\theta(s) \nabla_a Q^\mu(s,a)|_{a=\mu_\theta(s)} ds \\ &= \mathbb{E}_{d^\mu}[\nabla_\theta \mu_\theta(s) \nabla_a Q^\mu(s,a)|_{a=\mu_\theta(s)}],\end{aligned} \tag{11}$$

The derivation (i) the state distribution is non-differentiable w.r.t. θ (i.e., derivation (i))[5], To guarantee enough exploration of determinant PG, We can either add noise into the policy

$$\mu'(s) = \mu_\theta(s) + \mathcal{N}, \tag{12}$$

or learn it off-policy-ly by following a different stochastic behavior $\beta(a|s)$ policy to collect samples,

$$\begin{aligned}\nabla_\theta J(\theta) &= \nabla_\theta \left(\int_s d^\beta(s) Q^\mu(s,a)|_{a=\mu_\theta(s)} ds \right) \\ &= \mathbb{E}_{d^\beta}[\nabla_\theta \mu_\theta(s) \nabla_a Q^\mu(s,a)|_{a=\mu_\theta(s)}],\end{aligned} \tag{13}$$

B.2 DISTRIBUTED PG

Due to the efficiency of the GPU-cluster in training, some workers (machines or processes) are employed in a distributed manner to generate rollouts and compute policy gradients in PG methods Brenner (2023). The distributed advancement can also be extended to any PG extension, like Actor-Critic (AC), PPO, and deterministic PG methods.

Centralization These workers can either share a central parameter server or update their own weights in a decentralized manner, where aggregation techniques such as AllReduce may be utilized. Rather than merely collecting rollouts and calculating the gradient according to its replay buffer, the workers can be further decentralized into *agents* with their parameters, which is closely related to PG in multi-agent setting.

Synchrony In the centralized paradigm, weight updates can be conducted synchronously, where gradients from all workers are aggregated (typically through summation or averaging) before updating the model parameters. This ensures a globally consistent update but may introduce inefficiencies due to synchronization delays. Alternatively, asynchronous updating allows each worker to update the global parameters independently, without waiting for all gradients to be collected. This method can improve computational throughput but may lead to stale gradients and slower convergence. The difference between these 2 approaches is exemplified in Advantage Actor-Critic (A2C) and Asynchronous Advantage Actor-Critic (A3C).

[4] The deterministic PG is a special case of the stochastic PG, with $\sigma = 0$ in the re-parameterization $\pi_{\mu_\theta,\sigma}$.

[5] A small change in θ can cause a substantial change in the trajectory, and the state visitation distribution can exhibit non-smooth behavior as a function of θ.