A Note on Approximately Optimal Approximate Reinforcement Learning

Chao Yu Huang
Department of Computer Science
National Chiao Tung University
b608390.cs08g@nctu.edu.tw

1 Introduction

In this paper [Kakade and Langford [2002]], the author present the *conservative policy iteration* algorithm, which finds an "approximately" optimal policy. To achieve this goal, two algorithms are introduced, first called *restart distribution*, and second called *"approximate" greedy policy chooser*.In following subsections, an overview of this paper will be provided.

- Main challenges of this paper

  For the problem of finding the "approximately" optimal policy, three questions are the author desire answers to:

  1. Is there some performance measure that is guaranteed to improve at every step?
  2. How difficult is it to verify if a particular update improves this measure?
  3. After a reasonable number of policy updates, what performance level is obtained?

- Finding the "Approximate" optimal policy

  The optimal policy is the policy which simultaneously maximizes values for all states. The policy can be updated in a iterative process, the problem this paper want to solve is to find a policy which is close to the optimal policy in the reasonable policy updates and evaluate the performance level of the policy found by the algorithm, this policy is called "approximate" optimal policy. In addition, the update process of this algorithm should guaranteed to improve the policy at every step.

- Main contributions

  As the author stated in the paper, two kinds of methods are used to find the optimal policy, *approximate value function methods* and *policy gradient methods*. But both of them can't give satisfactory answers to the three questions mentioned in "Main challenges" subsection. And the new algorithm presented in this paper can address these three questions.

  As the author claimed in the paper, the algorithm can:

  1. Improve the performance metric.
  2. Terminate in a small number of timesteps.
  3. Returns an "approximately" optimal policy.

- Personal perspective on the proposed method
  In reinforcement learning, sometimes we want to find an optimal solution, we find the "best" result based on the information we have in current time step (ex. state information), but

Preprint. Under review.

the solution we find currently may not really leads us to the true optimal value, because there may be some information we don't know in other states, this problem will effect the performance much especially when the state probability distribution is very nonuniform. So the "Exploration" will be important for a reinforcement learning algorithm. For example, in Deep Q network (DQN) [Mnih et al. [2013]], the ϵ-greedy algorithm is used to increase the exploration when the algorithm choosing action in each episodes and in Deep Deterministic Policy Gradient (DDPG) [Lillicrap et al. [2015]], a random noise will added to the choosing actions to help the actions get more information from the state that is rarely visited. In this paper, the author also want to improve this by applying a restart distribution method, this method make the algorithm more sensitive to the policy improvement in unlikely states. For the greedy policy chooser, the author proves it can guarantee a policy improvement at every step and the performance can also be measured after reasonable policy updates.

2 Problem Formulation

- Notations

Markov decision process(MDP)

Defined by the tuple $(S, D, A, \mathcal{R}, P(s'; s, a))$ where: S is a finite set of states, D is the starting state distribution, A is a finite set of actions, $\mathcal{R}$ is a reward function $\mathcal{R} : S \times A \to [0, R]$, and $P'(s'; s, a)$ are the transition probabilities, with $P'(s'; s, a)$ giving the next-state distribution upon taking action a in state s.

Value

Given $0 \leqslant \gamma < 1$, we define the value function for a given policy π as:

$$V_\pi(s) \equiv (1-\gamma)E\left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_t, a_t)|\pi, s\right]$$

$(1-\gamma)$ is for normalization, so $V_\pi(s) \in [0, R]$

State-action value

We define state-action value as:

$$Q_\pi(s,a) \equiv (1-\gamma)\mathcal{R}(s,a) + \gamma E_{s' \sim P(s';s,a)}[V_{pi}(s^{'})]$$

$Q_\pi(s,a) \in [0, R]$ due to normalization.

Advantage

$$A_\pi(s,a) \equiv Q_\pi(s,a) - V_\pi(s)$$

$A_\pi(s,a) \in [-R, R]$ due to normalization.

Restart distribution

Draws the next state from the distribution μ.

γ-discount future state distribution

γ-discount future state distribution for a starting state distribution μ:

$$d_{\pi,\mu}(s) \equiv (1-\gamma)\sum_{t=0}^{\infty} \gamma^t Pr(st = s; \pi, \mu)$$

- The optimization problem of interest

The goal of algorithm is to maximize the discounted reward from the start distribution D,

$$\eta_D \equiv E_{s\sim D}[V_\pi(s)]$$

The optimal policy exists which can maximize $V_\pi(s)$ for all states.

- The technical assumptions

Restart distribution
The author desire an algorithm which uses only the MDP M and assumes access to restart distribution μ.

Approximate greedy policy chooser
Crudely, the greedy policy chooser outputs a policy that usually chooses actions with largest state-action values of the current policy, ie it output an "approximate" optimal policy.

Goal of the agent
Only consider the case where maximize the γ-discounted average reward from the start distribution D.

3 Theoretical Analysis

- The problem of other methods
To get insight into the algorithm of this paper, the author first introduce the problems with current methods, *approximate value function methods* and *policy gradient methods*.

Approximate value function methods
As the author stated in the paper, approximate value function methods use approximate value in an exact method and suffer from a paucity of theoretical results on the performance of a policy based on the approximate values. So the method can't give satisfactory answers to the three questions mentioned in the first section.

Policy gradients methods
By following the gradient of future reward, the policy gradient method attempt to find a good policy.To compute the policy gradient, we can use the following method:

$$\nabla \eta_D = \sum_{s,a} d_{\pi,D}(s) \nabla \pi(a;s) Q_\pi(s,a)$$

This method answer the question 1 in the first section, because by applying this method, the measure of interest(η_D) is guaranteed to improve under gradient ascent.
Then the author provide two examples to address question 2 in the first section ("How difficult is it to verify if a particular update improves this measure?").

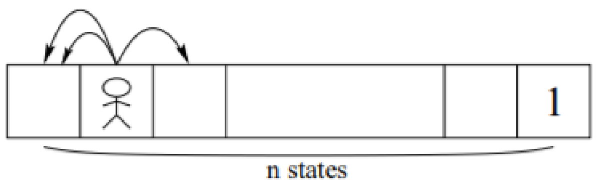

Figure 3.1: policy gradient example 1

The first example is the agent walking in the grid, and when it walk to the empty grid, it can get the reward 0. And if it walk to the grid with number 1 (rightmost grid), it can get reward 1. And in this environment, two actions will move agent to the left and one action move agent to the right. For these class of problems, the expected time to reach the the goal state (reward 1 state) using undirected exploration *ie* random walk exploration, is exponential in the size of the state space. And if the agent didn't reach the goal state in a trajectory, the estimate Q value will be 0, according to the policy gradient method, the estimate of the gradient will be 0, too. Which means that to obtain a non-zero estimate of gradient, it requires exponential time with on-policy samples.

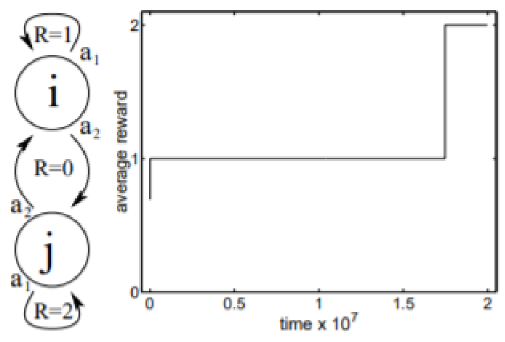

Figure 3.2: policy gradient example 2

And for the second example, there are two states state i and state j, And for each state there are two actions, a1 and a2. The initial distribution for this example will be:

$$\rho(i) = 0.8,\ \rho(j) = 0.2$$

And the policy of this example will be :

$$\pi(i, a1) = 0.8,\ \pi(i, a2) = 0.2,\ \pi(j, a1) = 0.8,\ \pi(j, a2) = 0.2$$

There is one assumption in this example, that the policy is using the common Gibbs table-lookup distributions:

$$\pi_\theta\ :\ \pi(a; s)\ \propto\ exp(\theta_{sa})$$

According to the discussion in [Agrawal [2019]], the $\theta_{s,a}$ can be updated by the following method:

$$\theta_{s,a} \leftarrow \theta_{s,a} + \alpha d^\pi(s)\pi_\theta(s, a)(Q_\pi(s, a) - V_\pi(s))$$

Then we know that for state i, because $Q_\pi(i, a1)$ is bigger than $Q_\pi(i, a2)$ and $\pi(i, a1) = 0.8$ is bigger than$\pi(i, a2) = 0.2$, too. So the $\theta_{i,a1}$ will increase faster than $\theta_{i,a2}$, which make the policy stay more times on state i. For state j, $\theta_{j,a1}$ increase more, but because of the initial distribution $\rho(i) = 0.8 > \rho(j) = 0.2$, the policy will favor update at state i but not state j. The analysis above shows that learning at state i reduces the learning at state j. In Figure 3.2, we can observe an flat plateau at 1 average reward.

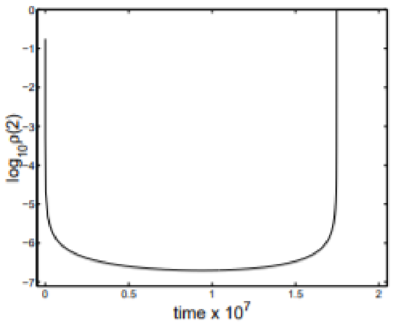

Figure 3.3

And Figure 3.3 shows the $\rho(j)$ become very low (10^{-7}) The author suggest that this example didn't give a good result to question 3. ("Which is concerned how fast such a policy can be found?") Because a gradient method could end up trapped at the plateaus where estimating the gradient has an unreasonably large sample complexity.

- Approximately optimal RL

Restart distribution

According to two examples in the previous paragraph, the author shows that the policy gradient method may not sensitive to some unlikely states, but these states may be important for finding the optimal policy. So the author present another performance measure that can weight the improvement from all states more uniformly:

$$\eta_\mu(\pi) \equiv E_{s\sim\mu}[V_\pi(s)]$$

Where μ is some "exploratory" *restart distribution*. First the author wants to maximize η_μ, but maximize η_μ may have poor performance on η_D, so the author want to ensure that when maximize η_μ results in good policy under η_D.

Conservative policy update rule

Then author present more conservative policy update rule:

$$\pi_{new}(a;s) = (1-\alpha)\pi(a;s) + \alpha\pi'(a;s)$$

for some π' and $\alpha \in 1$, π' is the policy that is updated by greedy policy iteration based on some approximate state-action values.
Following subsections are about the main theorems of this paper.

Policy improvement

In this subsection, the author show that by finding some π' that can make $\Delta\eta_\mu$ be positive, then the policy improvement occurs. And the author also present a theorem for the lower bound of the policy improvement.
First, **policy advantage** is defined as:

$$A_{\pi,\mu}(\pi') \equiv E_{s\sim d_{\pi,\mu}}[E_{a\sim\pi'(a;s)}[A_\pi(s,a)]]$$

And by $\frac{\rho\eta_\mu}{\rho\alpha}|_{\alpha=0} = \frac{1}{1-\gamma}A_{\pi,\mu}$, the author shows that:

$$\Delta\eta_\mu = \frac{\alpha}{1-\gamma}A_{\pi,\mu}(\pi') + O(\alpha^2)$$

So for the sufficiently small α, policy improvement occur if policy advantage is positive. By this observation, the author want to find the bound for the policy improvement, which is theorem 3.1:

Theorem 3.1 (Theorem 4.1 in original paper) *Let $\mathbb{A}$ be the policy advantage of π' with respect to π and μ. and let ϵ = $\max_s |E_{a\sim\pi'}(a;s)[A_\pi(s,a)]|$. For the conservative policy update rule and for all $\alpha \in [0,1]$:*

$$\eta_\mu(\pi_{new}) - \eta_\mu(\pi) \geq \frac{\alpha}{1-\gamma}(\mathbb{A} - \frac{2\alpha\gamma\epsilon}{1-\gamma(1-\alpha)})$$

And the following corollary shows that the greater the policy advantage, the greater the guaranteed performance increase.

Corollary 3.2 (Corollary 4.2 in original paper) *Let R be the maximal possible reward and $\mathbb{A}$ be the policy advantage of π' with respect to π and μ. If $\mathbb{A} \geq 0$, then using $\alpha = \frac{(1-\gamma)\mathbb{A}}{4R}$ guarantees the following policy improvement:*

$$\eta_\mu(\pi_{new}) - \eta_\mu(\pi) \geq \frac{A^2}{8R}$$

The author didn't give a formal prove of this corollary, so I provide a prove in this report based on the author's hint :

First, based on the author's hint, I want to show that the change is bound by $\frac{\alpha}{1-\gamma}(\mathbb{A} - \alpha\frac{2R}{1-\gamma})$:
By using the theorem 3.1, because $(1-\alpha) \leq 1$, we know that:

$$\begin{aligned}
&\eta_\mu(\pi_{new}) - \eta_\mu(\pi) \\
&\geq \frac{\alpha}{1-\gamma}(\mathbb{A} - \frac{2\alpha\gamma\epsilon}{1-\gamma(1-\alpha)}) \\
&\geq \frac{\alpha}{1-\gamma}(\mathbb{A} - \frac{2\alpha\gamma\epsilon}{1-\gamma})
\end{aligned}$$

Because $\gamma\epsilon \leq \epsilon \leq \max_s |E_{a\sim\pi'}(a;s)[A_\pi(s,a)]| \leq R$, we know:

$$\frac{\alpha}{1-\gamma}(\mathbb{A} - \frac{2\alpha\gamma\epsilon}{1-\gamma}) \geq \frac{\alpha}{1-\gamma}(\mathbb{A} - \alpha\frac{2R}{1-\gamma})$$

Then we prove that:

$$\eta_\mu(\pi_{new}) - \eta_\mu(\pi) \geq \frac{\alpha}{1-\gamma}(\mathbb{A} - \alpha\frac{2R}{1-\gamma})$$

And the author also state that we need to choose α which maximize the bound, first I arrange the right part to the quadratic form:

$$\frac{\alpha}{1-\gamma}(\mathbb{A} - \alpha\frac{2R}{1-\gamma}) = -\frac{2R}{(1-\gamma)^2}\alpha^2 + \frac{\mathbb{A}}{1-\gamma}\alpha$$

Now we can see that the right part is a quadratic function, and we can find out the max value when it's first derivative = 0:

$$-\frac{4R}{(1-\gamma)^2}\alpha + \frac{\mathbb{A}}{1-\gamma} = 0$$

Then solve for α, when $\mathbb{A} \geq 0$:

$$\alpha = \frac{(1-\gamma)\mathbb{A}}{4R}$$

- Answer question 3
  The author address question 3 by first addressing how fast the policy converge and find the bound for the quality of the policy. First, the author access to an ϵ -greedy policy chooser to solve the problem. The algorithm is called ϵ-good algorithm $G_\epsilon(\pi,\mu)$ which is defined as:

  Definition 3.3 An **ϵ-greedy policy chooser**, $G_\epsilon(\pi,\mu)$, is a function of a policy π and a state distribution μ which returns a policy π' such that $\mathbb{A}_{\pi,\mu}(\pi') \geq OPT(\mathbb{A}_{\pi,\mu}) - \epsilon$, where $OPT(\mathbb{A}_{\pi,\mu}) \equiv \max_{\pi'} \mathbb{A}_{\pi,\mu}(\pi')$.

  Then the author introduce an outline of the main algorithm of this paper, **Conservative Policy Iteration**:

  (1) Call $G_\epsilon(\pi,\mu)$ to obtain some π'.

  (2) Estimate the policy advantage $A_{\pi,\mu}(\pi')$

  (3) If the policy advantage is small (less than ϵ), STOP and return π.

  (4) Else, update the policy and go to (1).

Then the author start to solve question 3, "After a reasonable number of policy updates, what performance level is obtained?".
First, the author show that the algorithm satisfy the first condition of the question, "A reasonable number of policy updates" by introduce Theorem 4.4:

Theorem 3.4 (Theorem 4.4 in original paper) *With probability at least 1 - δ, conservative policy iteration: i) improves η_μ with every policy update, ii) ceases in at most $72\frac{R^2}{\epsilon^2}$ calls to $G_\epsilon(\pi, \mu)$, and iii) return a policy π such that $OPT(A_{\pi,\mu}) \leq 2\epsilon$.*

And to bounds the performance of interest, which means answering "What performance level is obtained?" in question 3, the author gives Corollary 3.5.

Corollary 3.5 (Corollary 4.5 in original paper) *Assume that for some policy π, $OPT(A_\pi \mu) < \epsilon$. Let $\pi*$ be an optimal policy. Then*

$$\eta_D(\pi*) - \eta_D(\pi) \leq \frac{\epsilon}{(1-\gamma)} \left\| \frac{d_{\pi*,D}}{d_\pi \mu} \right\|_\infty$$

$$\leq \frac{\epsilon}{(1-\gamma)^2} \left\| \frac{d_{\pi*,D}}{\mu} \right\|_\infty$$

By present theorem 3.4 and corollary 3.5, the author shows that the Conservative Policy Iteration can give a satisfactory answer to question 3.

And by corollary 3.2 and theorem 3.4, the author formally define the Conservative Policy Iteration as:

(1) Call $G_\epsilon(\pi, \mu)$ to obtain some π'

(2) Use $O(\frac{R^2}{\epsilon^2} log \frac{R^2}{\delta\epsilon^2})$ μ-restarts to obtain an $\frac{\epsilon}{3}$ -accurate estimate $\hat{\mathbb{A}}$ of $\mathbb{A}_{\pi,\mu}(\pi')$.

(3) If $\hat{\mathbb{A}}\frac{2\epsilon}{3}$, STOP and return π.

(4) If $\hat{\mathbb{A}} \geq \frac{2\epsilon}{3}$, then update policy π according to **Conservative policy update rule** using $\frac{(1-\gamma)(\hat{\mathbb{A}}-\frac{\epsilon}{3})}{4R}$ and return to step 1.

- Extension of the theorem 3.1
The theorem 3.1 proves the lower bound of the policy improvement, and I think it is also possible to give a upper bound to this improvement.
And to achieve this, I need a minor adjustment to the proof of theorem 3.1 from the original paper. Following is how I make this adjustment:
From the original proof, because $\Sigma_a \pi(a; s) A_\pi(s, a) = 0$ (proved in original paper), we know:

$$E_{s\sim P}(s_t; \pi_{new}) \left[\sum_a \pi_{new}(a; s) A_\pi(s, a) \right]$$

$$= \alpha E_{s\sim P}(s_t; \pi_{new}) \left[\sum_a \pi'(a; s) A_\pi(s, a) \right]$$

$$= \alpha(1-\rho_t) E_{s\sim P}(s_t | ct = 0; \pi_{new}) \left[\sum_a \pi'(a; s) A_\pi(s, a) \right]$$

$$+ \alpha\rho_t E_{s\sim P}(s_t | ct \geq 1; \pi_{new}) \left[\sum_a \pi'(a; s) A_\pi(s, a) \right]$$

And as the author stated in the paper, $Pr(ct = 0) = (1 - \alpha)^t$, and $\rho_t \equiv Pr(c_t \geq 1) = 1 - (1 - \alpha)^t$, we know that:

$$\alpha(1-\rho_t)E_{s\sim P}(s_t|ct=0;\pi_{new})\left[\sum_a \pi'(a;s)A_\pi(s,a)\right]$$

$$+\alpha\rho_t E_{s\sim P}(s_t|ct\geq 1;\pi_{new})\left[\sum_a \pi'(a;s)A_\pi(s,a)\right]$$

$$\leq \alpha E_{s\sim P}(s_t|ct=0;\pi_{new})\left[\sum_a \pi'(a;s)A_\pi(s,a)\right] + 2\alpha\rho_t\epsilon$$

$$= \alpha E_{s\sim P}(st;\pi)\left[\sum_a \pi'(a;s)A_\pi(s,a)\right] + 2\alpha\rho_t\epsilon$$

Here the only change from the original proof is to change $-2\alpha\rho_t\epsilon$ to $+2\alpha\rho_t\epsilon$ and the $\geq$ in third line to $\leq$. Because we know that $\epsilon = \max_s |E_{a\sim\pi'}(a;s)[A_\pi(s,a)]|$. So not only the lower bound can be defined by adding $-2\alpha\rho_t\epsilon$ to the $\alpha E_{s\sim P}(st;\pi)\left[\sum_a \pi'(a;s)A_\pi(s,a)\right]$, the upper bound can be found by adding $+2\alpha\rho_t$ to $\alpha E_{s\sim P}(st;\pi)\left[\sum_a \pi'(a;s)A_\pi(s,a)\right]$. And by changing $-2\alpha\rho_t\epsilon$ to $+2\alpha\rho_t\epsilon$ in the original proof and lemma 6.1 in original paper, I get a policy improvement upper bound, $\alpha \in [0,1]$ and $\epsilon = \max_s |E_{a\sim\pi'}(a;s)[A_\pi(s,a)]|$:

$$\eta_\mu(\pi_{new}) - \eta_\mu(\pi) \leq \frac{\alpha}{1-\gamma}(\mathbb{A} + \frac{2\alpha\gamma\epsilon}{1-\gamma(1-\alpha)})$$

Different from the original paper, this bound is represent the limitation for a policy improvement.

4 Conclusion

Please provide succinct concluding remarks for your report. You may discuss the following aspects:

- The potential future research directions
  This paper prove the Conservative Policy Iteration can find the optimal policy after reasonable update times, the future research direction can be applying the algorithm in the existing reinforcement learning environment (ex.gym in python) to evaluate the performance of the algorithm. It can be compared with the traditional value-iteration, policy iteration algorithm for solving the game like Taxi-v2, FrozenLake8x8, etc.
  And a MDP environment like Figure3.2 can be created, with this environment, Conservative Policy Iteration can compare its performance with policy gradient method to see if the problem of unbalance updating on two states can be reduced by the introducing of restart distribution and conservative policy update rule (by comparing $\rho(i)$ and $\rho(j)$).

- Technical limitations
  In this algorithm, in the step 2 of Conservative Policy Iteration, to estimate $A_{\pi,\mu}(\pi')$, we need to compute the expectation value of $A_\pi(s,a)$, but if the state number is very large or the action is continuous, the program will take unreasonable computation time and memory. This could be solved by applying "sampling" to estimate the true expectation value (ex. Monte Carlo method).

- Latest results on the problem of interest
  Trust Region Policy Optimization (TRPO) [Schulman et al. [2015]] is also an algorithm that can optimize the policy and which is effective for finding large nonlinear policies. And the theorem 4.1 of this paper is also a theoretical basis for the prove of TRPO. And compare to TRPO, Proximal Policy Optimization (PPO) [Schulman et al. [2017]] is another algorithm in the policy gradient family which is much simpler to implement, and enable the multiple epochs of minibatch multiple epochs of minibatch policy gradient updates.

References

Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. pages 267–274, 2002.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. 2015.

Shipra Agrawal. Lecture 7: Approximately optimal approximate rl, trpo. 2019.

John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust region policy optimization. *CoRR*, abs/1502.05477, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.