Constrained Policy Optimization

Joshua Achiam, David Held, Aviv Tamar, Pieter Abbeel

Notation

주요 기호 및 의미

- π: **정책 함수 (Policy Function)** – 상태에서의 행동 분포를 나타내는 함수. $\pi(a|s)$는 상태 s에서 행동 a를 선택할 확률.
- π_k: **k번째 반복에서의 정책** – 학습 과정에서 현재 정책을 나타냄. 매 반복마다 정책이 업데이트되어 새로운 정책 π_{k+1}로 변경.
- S: **상태 집합 (State Space)** – 에이전트가 탐색할 수 있는 모든 상태의 집합.
- A: **행동 집합 (Action Space)** – 에이전트가 선택할 수 있는 모든 행동의 집합.
- $R(s, a, s')$: **보상 함수 (Reward Function)** – 상태 s에서 행동 a를 취하고 다음 상태 s'로 전이할 때 얻는 보상.
- $P(s'|s, a)$: **상태 전이 확률 (Transition Probability Function)** – 상태 s에서 행동 a를 취할 때 다음 상태가 s'일 확률.
- μ: **초기 상태 분포 (Starting State Distribution)** – 에이전트가 학습을 시작할 때 상태 s가 선택될 확률 분포.
- τ: **경로 (Trajectory)** – 상태 및 행동의 순서인 $(s_0, a_0, s_1, a_1, \ldots)$로 구성된 에이전트의 전체 이동 경로.
- γ: **감쇠 인자 (Discount Factor)** – 미래 보상에 대한 가중치를 나타내는 파라미터로, 0과 1 사이의 값을 가지며 미래 보상에 대한 중요도를 결정.

Notation

가치 함수 및 반환 관련

- $J(\pi)$: **정책의 성능 지표 (Performance Measure)** – 정책 π에 따른 기대 총 보상.

$$J(\pi) = \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) \right]$$

- $R(\tau)$: **경로의 총 보상 (Total Return of a Trajectory)** – 경로 τ를 따라 수집된 모든 감쇠된 보상의 합.
- $V^\pi(s)$: **상태 가치 함수 (State Value Function)** – 주어진 상태 s에서 정책 π를 따를 때 기대되는 총 보상.

$$V^\pi(s) = \mathbb{E}_{\tau \sim \pi}[R(\tau) \mid s_0 = s]$$

- $Q^\pi(s, a)$: **상태-행동 가치 함수 (Action-Value Function)** – 주어진 상태 s와 행동 a에서 정책 π를 따를 때 기대되는 총 보상.

$$Q^\pi(s, a) = \mathbb{E}_{\tau \sim \pi}[R(\tau) \mid s_0 = s, a_0 = a]$$

- $A^\pi(s, a)$: **이점 함수 (Advantage Function)** – 상태-행동 가치 함수와 상태 가치 함수의 차이를 나타내며, 특정 행동이 얼마나 더 나은지를 나타냄.

$$A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$$

Notation

상태 분포 및 제약 관련

- $d^\pi(s)$: **할인된 미래 상태 분포 (Discounted Future State Distribution)** – 정책 π를 따를 때, 시간에 따른 감쇠 효과를 고려하여 상태 s에 있을 확률 분포.

$$d^\pi(s) = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t P(s_t = s \mid \pi)$$

- C_i: **제약 비용 함수 (Constraint Cost Function)** – 보상 함수와 유사하게 특정 제약 조건에 대해 비용을 나타냄.
- $J_{C_i}(\pi)$: **제약 비용 반환 (Constraint Return)** – 정책 π에 따른 제약 비용 함수의 기대 총 비용.

$$J_{C_i}(\pi) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t C_i(s_t, a_t, s_{t+1})\right]$$

- d_i: **제약 비용 상한 (Constraint Cost Limit)** – 각 제약 비용 함수 C_i에 대해 만족시켜야 하는 최대 값.
- Π_C: **제약 조건을 만족하는 정책 집합 (Feasible Policy Set for CMDP)** – 모든 제약을 만족하는 정책들의 집합.

$$\Pi_C = \{\pi \in \Pi : \forall i, J_{C_i}(\pi) \le d_i\}$$

Notation

최적화 및 근사 관련

- g: **정책의 기울기 벡터 (Policy Gradient Vector)** – 보상을 최대화하는 방향을 나타내는 벡터.
- b_i: **제약 조건의 기울기 벡터 (Constraint Gradient Vector)** – 제약 조건에 대한 기울기 벡터.
- H: **피셔 정보 행렬 (Fisher Information Matrix)** – 정책의 파라미터화된 분포의 기울기를 나타내는 정보 행렬로, 근사 이차식 제약에 사용됨.
- δ: **정책 업데이트의 신뢰 영역 크기 (Trust Region Size)** – 정책이 업데이트될 때 새로운 정책과 기존 정책 간의 변화량을 제한하는 파라미터.
- $D_{KL}(\pi \parallel \pi_k)$: **KL-divergence (쿨백-라이블러 발산)** – 새로운 정책 π와 기존 정책 π_k 간의 차이를 측정하는 지표로, 두 확률 분포 간의 차이.

Introduction

- In reinforcement learning (RL), agents learn to act by trial and error, **gradually improving their performance** at the task as learning progresses.
- In many realistic domains, however, it may be **unacceptable to give an agent complete freedom**.
    - e.g. industrial robot arm learning to assemble a new product in a factory
- In domains like this, **safe exploration for RL agents** is important

robot arm in a factory

Introduction

- To deal with these problem, **Constrained Markov Decision Process (CMDP)** is used.

$$\mathbf{MDP} = (S, A, P, R, \mu)$$

- S: 상태 집합 (State Space)
- A: 행동 집합 (Action Space)
- P(s′ l s,a): 상태 전이 확률 (Transition Probability)
- R(s,a,s′): 보상 함수 (Reward Function)
- μ: 초기 상태 분포 (Initial State Distribution)

Adding constraint cost functions $C_i(s, a, s')$ and constraint limits d_i

$$\mathbf{CMDP} = (S, A, P, R, \mu, C, d)$$

- C: 제약 비용 함수들 (Constraint Cost Functions)
- d: 제약 상한 값들 (Constraint Limits)

Purpose of MDP: 보상 최대화

$$J(\pi) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t R(s_t, a_t, s_{t+1})\right]$$

- π: 정책
- τ: 경로 (trajectory)
- ɣ: 할인 인자 (Discount Factor)

Incorporating constraints into optimization: $J_{C_i}(\pi) \le d_i$

Purpose of CMDP: 보상 최대화 및 제약 조건 만족

$$\max_{\pi} J(\pi) \quad \text{s.t.} \quad J_{C_i}(\pi) \le d_i, \quad \forall i \in \{1, \ldots, m\}$$

$$J_{C_i}(\pi) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t C_i(s_t, a_t, s_{t+1})\right]$$

Introduction

- Although optimal policies for finite CMDPs with known models can be obtained by linear programming, methods for **high-dimensional control are lacking**.
- Therefore, This paper shows the method that solve **CMDP for high-dimensional problem**.
- Proposed method provides **bounds on the difference in rewards or costs** between two policies π and π'.
  - Guarantees **reward increase** and **constraint satisfaction**.

Preliminaries

MDP 정의:

- An MDP is a tuple:

$$(S, A, R, P, \mu)$$

- S: Set of states
- A: Set of actions
- $R : S \times A \times S \to \mathbb{R}$: Reward function
- $P(s'|s,a) : S \times A \times S \to [0,1]$: Transition probability
- μ: Starting state distribution

Value Functions:

- **On-policy Value Function**:

$$V^\pi(s) = \mathbb{E}_{\tau\sim\pi}[R(\tau) \mid s_0 = s]$$

- **On-policy Action-Value Function**:

$$Q^\pi(s,a) = \mathbb{E}_{\tau\sim\pi}[R(\tau) \mid s_0 = s, a_0 = a]$$

- **Advantage Function**:

$$A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$$

Stationary Policy:

- A policy $\pi : S \to P(A)$ maps states to probability distributions over actions.
- $\pi(a|s)$: Probability of selecting action a in state s.
- **Goal**: Find a policy π that **maximizes** the performance:

$$J(\pi) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1})\right]$$

Discounted Future State Distribution:

- $d^\pi(s)$: Probability of visiting state s under policy π:

$$d^\pi(s) = (1-\gamma)\sum_{t=0}^{\infty} \gamma^t P(s_t = s \mid \pi)$$

두 정책 간의 성능 차이:

- Difference in performance between policies π' and π:

$$J(\pi') - J(\pi) = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\pi'}, a\sim\pi'}[A^\pi(s,a)]$$

- Where:
  - $a \sim \pi'$: Action sampled from policy π' in state s.

Kakade, Sham and Langford, John. Approximately Optimal Approximate Reinforcement Learning. Proceedings of the 19th International Conference on Machine Learning, pp. 267-274, 2002.

Method

- TRPO로부터 영감을 받아, Surrogate 함수로써 CPO를 표현함
  - TRPO의 정책 업데이트 함수: KL-divergence 제한을 통해 새로운 정책이 기존 정책에서 크게 변하지 않도록 제약을 둠

$$\pi_{k+1} = \arg\max_{\pi \in \Pi_\theta} \mathbb{E}_{s \sim d^{\pi_k}, a \sim \pi}[A^{\pi_k}(s,a)] \quad \text{s.t.} \quad \bar{D}_{KL}(\pi || \pi_k) \leq \delta$$

  - CPO의 정책 업데이트 함수: TRPO의 신뢰 영역 최적화 방법을 제약 조건이 있는 상황(CMDP)에서 적용한 방법

$$\pi_{k+1} = \arg\max_{\pi \in \Pi_\theta} \mathbb{E}_{s \sim d^{\pi_k}, a \sim \pi}[A^{\pi_k}(s,a)] \quad \text{s.t.} \quad \underline{J_{C_i}(\pi_k) + \frac{1}{1-\gamma}\mathbb{E}_{s \sim d^{\pi_k}, a \sim \pi}[A^{\pi_k}_{C_i}(s,a)] \leq d_i, \ \forall i,} \quad \bar{D}_{KL}(\pi || \pi_k) \leq \delta$$