DeepTOP: Deep Threshold-Optimal Policy For MDPs and RMABs.

Khaled Nakhleh, I-Hong Hou

Overview

- **Objective:** learn the optimal threshold policy for control problems. Control problems are formed as Markov Decision Processes (MDPs) and Restless Multi-Armed Bandits (RMABs).
- Finding the optimal policy can be reduced to finding the appropriate threshold given the system's state (e.g. current room temperature for an AC).
- Actions of threshold policies are monotone. Given a certain threshold, the optimal action for a state with an assigned value is also optimal for all states with a higher assigned value.
- We design DeepTOP: model-free, off-policy deep reinforcement learning algorithms for MDPs and RMABs.
- Simulation results on MDP and RMAB problems show that DeepTOP outperforms the state-of-the-art baselines.

Threshold Policies

- For an MDP, we define a threshold function as $\mu : \mathcal{V} \to \mathbb{R}$.
- MDP is defined as a tuple $\mathcal{E} = \{\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma\}$. The state space $\mathcal{S} = \mathbb{R} \times \mathcal{V}$ consists of a scalar state $\lambda \in \mathbb{R}$ and a discrete set of vectors $v \in \mathcal{V}$. Binary action space $\mathcal{A} = \{0,1\}$.
- The optimal threshold policy deterministically picks the action $a_t = 1\{\mu(v_t) > \lambda_t\}$.

Examples

- Charging an Electric Vehicle (EV) given changing electricity prices.
- AC system deciding whether to cool a building or not.
- Central bank deciding whether to raise interest rate given the current inflation rate or not.

Threshold policy gradient for MDPs

- Action-value function under threshold function μ:

$$Q_u\big(\lambda, v, 1(\mu(v) > \lambda)\big) = \sum_{v \in \mathcal{V}} \int_{\lambda'=-M}^{\lambda'=+M} \rho_\mu(\lambda', v', \lambda, v)\, \bar{r}\left(\lambda', v', 1(\mu(v') > \lambda')\right).$$

- Goal is to maximize the objective function

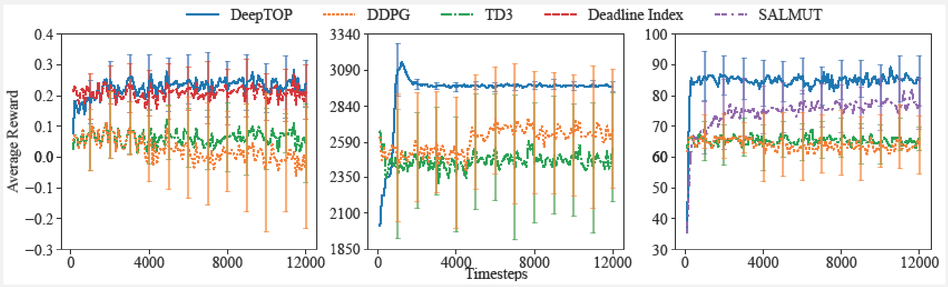
$$K(\mu^\phi) := \int_{\lambda=-M}^{\lambda=+M} \sum_{v \in V} Q_{\mu^\phi}(\lambda, v, 1(\mu^\phi(v) > \lambda))\, d\lambda.$$

Theorem 1. Given the parameter vector ϕ, let $\bar{\rho}(\lambda, v)$ be the discounted state distribution when the initial state is chosen uniformly at random under the threshold policy. If all vector states $v \in V$ have distinct values of $\mu^\phi(v)$, then

$$\nabla_\phi K(\mu^\phi) = 2M|V| \sum_{v \in V} \bar{\rho}\left(\mu^\phi(v), v\right)\Big(Q_{\mu^\phi}\big(\mu^\phi(v), v, 1\big) - Q_{\mu^\phi}\big(\mu^\phi(v), v, 0\big)\Big)\nabla_\phi \mu^\phi(v).$$

MDP Results

- **EV charging:** station decides whether it is optimal to charge the EV or not based on its state at time t.
- **Inventory management:** manager decides if its optimal to buy additional goods based on the season's fluctuations and in-lead times in orders.
- **Make-to-stock:** system that produces m items with W demand classes and buffer size s. system determines if it will accept class orders or not.



(a) EV charging. **(b) Inventory management.** **(c) Make-to-stock production.**

Threshold policy gradient extension to RMABs

- Threshold policy theorem and DeepTOP can be extended to the RMAB framework where each arm environment is an MDP.
- Define a threshold function $\mu_i(s_{i,t})$ for arm i in state $s_{i,t}$ and an *activation cost* λ. The threshold policy activates the arm (i.e. $a_{i,t} = 1$) if $\mu_i(s_{i,t}) > \lambda$.
- The Whittle index policy can be viewed as the optimal threshold function.
- **Goal:** obtain the optimal control policy that activates the largest V-valued arms out of N arms.

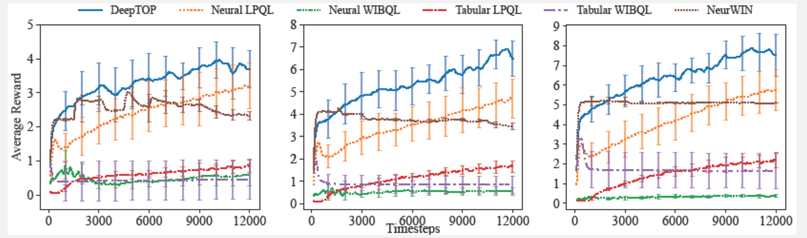
- **Theorem 2:** Given the parameter vector ϕ_i, let $\bar{\rho}_\lambda(s_i)$ be the discounted state distribution when the initial state is chosen uniformly at random and the activation cost is λ If all states $s_i \in S_i$ have distinct values of $\mu_i^{\phi_i}(s_i)$, then,

$$\nabla_{\phi_i} K_i\left(\mu_i^{\phi_i}\right) = |S_i| \sum_{s_i \in S_i} \bar{\rho}_{\mu_i^{\phi_i}(s_i)}(s_i)\left(Q_{i,\mu_i^{\phi_i}(s_i)}(s_i, 1) - Q_{i,\mu_i^{\phi_i}(s_i)}(s_i, 0)\right)\nabla_{\phi_i}\mu_i^{\phi_i}(s\_i).$$

RMAB Results

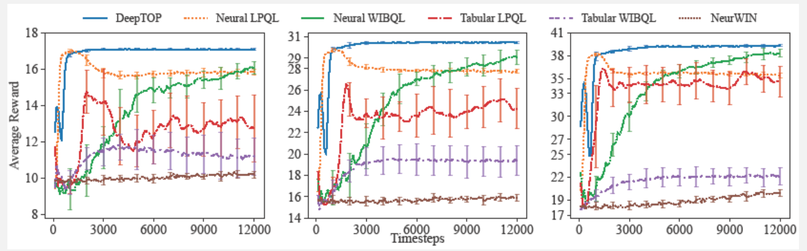
- **One-dimensional bandits:** Each arm has 100 states with the reward depending on the current state of arm i as $r_{i,t} = 1 - \left(\frac{s_{i,t}-99}{99}\right)^2$. If arm is activated, next state is $\min\{s_{i,t} + 1, 99\}$ with probability p_i. Otherwise, next state is $\max\{s_{i,t} - 1, 0\}$ with probability q_i.
- **Recovering bandits:** RMAB that studies the varying behavior of customers on advertisement links. If an arm is activated, the state $s_{i,t}$ is reset to state 1. Otherwise, the state increases by 1.

One-dimensional bandits



(a) N = 10. V = 3. **(b) N = 20. V = 5.** **(c) N = 30. V = 6.**

Recovering bandits



(a) N = 10. V = 3. **(b) N = 20. V = 5.** **(c) N = 30. V = 6.**