A Generalized Algorithm for Multi-Objective Reinforcement Learning and Policy Adaptation

NEURAL INFORMATION PROCESSING SYSTEMS

"Tony" Runzhe Yang (runzhey@princeton.edu), Xingyuan Sun, Karthik Narasimhan
Computer Science Department, Princeton University

PRINCETON UNIVERSITY

Research Highlights:

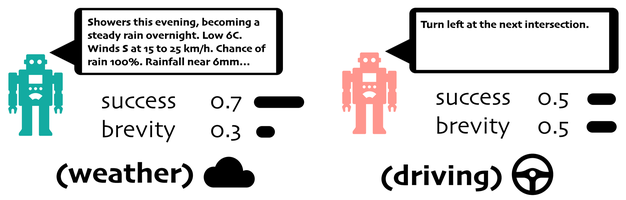
- **[Algorithmic]:** We propose an ***envelope MOQ-learning algorithm*** for *Multi-Objective Reinforce Learning (**MORL**)* with *linear preferences*, with the goal of enabling ***few-shot adaptation*** to new tasks.
- **[Theoretical]:** A ***theoretical framework*** for designing and analyzing *value-based MORL algorithms* and ***convergence analyses*** of our envelope MOQ-learning algorithm are provided.
- **[Empirical]:** We introduce new *evaluation* ***metrics*** and ***benchmark*** *environments*, test our algorithm on *a wide variety of domains*, including ***task-oriented dialogue*** and ***SuperMario***.

1. Motivation

Advantages of MORL with *Vectorized Reward*

(1) **Less dependence on reward design** to combine different objectives, which is both a tedious manual task and can lead to *unintended consequences* [1]

(2) **Dynamic adaptation or transfer** to related tasks through inferring their underlying preferences.



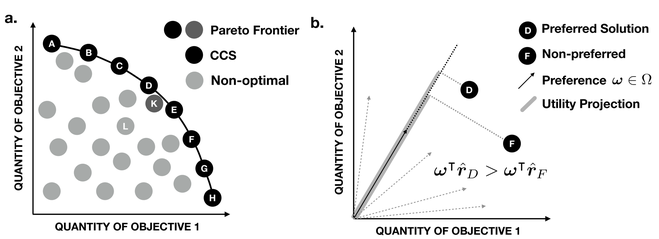
E.g., in **task-oriented dialogue systems,** users may expect either **briefer** or **more informative** dialogue.

Why challenging?

Previous methods convert a single-objective setting [16, 17], or approximate the Pareto front by learning several individual policies [8, 18], which are not **adaptable** and **scalable.**

2. MORL Under the Linear Preference Scenario

Linear Preference: $f_{\boldsymbol{\omega}}(\boldsymbol{r}(s_t, a_t)) = \boldsymbol{\omega}^\intercal \boldsymbol{r}(s_t, a_t)$



(a) Pareto frontier | CCS **(b)** linear preferences

(1) **Learning Phase** - learn *all optimal policies* corresponding to the *entire convex coverage set,* without being given any specific preference.

(2) **Adaptation Phase** - Infer the *underlying preference* of a new task in *few-shot*, and perform *the optimal policy*.

3. Value-Based MORL Algorithm

Envelope Deep Multi-Objective Q-Learning

1) Value Space: all the bounded functions in $\mathcal{Q} = (\Omega \to \mathbb{R}^m)^{\mathcal{S}\times\mathcal{A}}$

2) Bellman Operator: $(\mathcal{T}\boldsymbol{Q})(s,a,\boldsymbol{\omega}) := \boldsymbol{r}(s,a) + \gamma \mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}(\mathcal{H}\boldsymbol{Q})(s',\boldsymbol{\omega})$

is a ***generalized contraction*** with a ***fixed-point class*** $[Q^*]$

Optimality Filter $(\mathcal{H}\boldsymbol{Q})(s,\boldsymbol{\omega}) := \arg_{\boldsymbol{Q}} \sup_{a',\boldsymbol{\omega}'} \boldsymbol{\omega}^\intercal \boldsymbol{Q}(s,a',\boldsymbol{\omega}')$

3) Loss Functions: 1. $L_k^{\text{A}}(\theta) = \mathbb{E}_{s,a,\boldsymbol{\omega}}\left[\|\boldsymbol{y}_k - \boldsymbol{Q}(s,a,\boldsymbol{\omega};\theta)\|_2^2\right]$

2. $L_k^{\text{B}}(\theta) = \mathbb{E}_{s,a,\boldsymbol{\omega}}[|\boldsymbol{\omega}^\intercal \boldsymbol{y}_k - \boldsymbol{\omega}^\intercal \boldsymbol{Q}(s,a,\boldsymbol{\omega};\theta)|]$

Algorithm 1: Envelope MOQ-Learning

Input: a preference sampling distribution $\mathcal{D}_\omega$, path p_λ for the balance weight λ increasing from 0 to 1.
Initialize replay buffer $\mathcal{D}_\tau$, network $\boldsymbol{Q}_\theta$, and $\lambda = 0$.
for *episode* $= 1, \dots, M$ **do**
  Sample a linear preference $\boldsymbol{\omega} \sim \mathcal{D}_\omega$.
  for $t = 0, \dots, N$ **do**
    Exploration
    Observe state s_t.
    Sample an action ϵ-greedily:

$$a_t = \begin{cases} \text{random action in } \mathcal{A}, & \text{w.p. } \epsilon; \\ \max_{a\in\mathcal{A}} \boldsymbol{\omega}^\intercal \boldsymbol{Q}(s_t,a,\boldsymbol{\omega};\theta), & \text{w.p } 1-\epsilon. \end{cases}$$

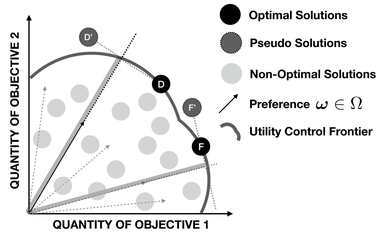
    Receive a vectorized reward $\boldsymbol{r}_t$ and observe s_{t+1}.
    Store transition $(s_t, a_t, \boldsymbol{r}_t, s_{t+1})$ in $\mathcal{D}_\tau$.
    if *update* **then**
      Envelope Q-update
      Sample N_τ transitions $(s_j, a_j, \boldsymbol{r}_j, s_{j+1}) \sim \mathcal{D}_\tau$.
      Sample N_ω preferences $W = \{\boldsymbol{\omega}_i \sim \mathcal{D}_\omega\}$.
      Compute $y_{ij} = (\mathcal{T}\boldsymbol{Q})_{ij} =$

$$\begin{cases} \boldsymbol{r}_j, & \text{for terminal } s_{j+1}; \\ \boldsymbol{r}_j + \gamma \arg_Q \max_{\substack{a\in\mathcal{A}, \\ \boldsymbol{\omega}'\in W}} \boldsymbol{\omega}_i^\intercal \boldsymbol{Q}(s_{j+1},a,\boldsymbol{\omega}';\theta), \text{o.w.} \end{cases}$$

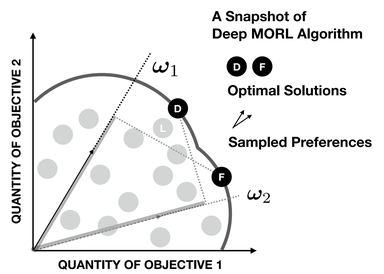
      for all $1 \le i \le N_\omega$ and $1 \le j \le N_\tau$.
      Update Q_θ by descending its stochastic gradient according to equations 1. and 2.:

$$\nabla_\theta L(\theta) = (1-\lambda)\cdot\nabla_\theta L^{\text{A}}(\theta) + \lambda\cdot\nabla_\theta L^{\text{B}}(\theta).$$

      Increase λ along the path p_λ.



Degeneracy problem: unable to recover reward from utility



Efficiency problem: misalignment between preferences and rewards

4. Preference Elicitation

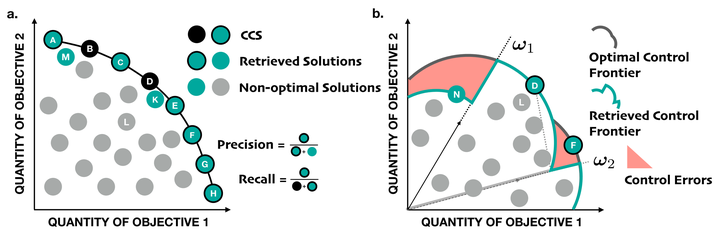
After learning phase, we can use the obtained model to ***infer the hidden preference*** on a specific task where *only scalar rewards are available*.

Use policy gradient (e.g., REINFORCE) and stochastic search to find:

$$\arg\max_{\mu_1,\dots,\mu_m} \mathbb{E}_{\boldsymbol{\omega}\sim\mathcal{D}_\omega^m}\left[\mathbb{E}_{\tau\sim(\mathcal{P},\Pi_{\mathcal{L}}(\boldsymbol{\omega}))}\left[\sum_{t=0}^{\infty}\gamma^t r_t(s_t,a_t)\right]\right]$$

with only few episodes.

5. Evaluation Metrics

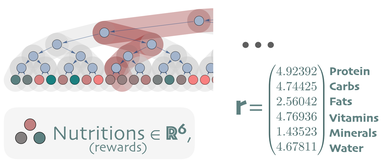


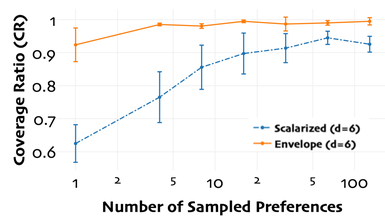
(a) Coverage Ratio (CR) **(b) Adaptation Error (AE)**

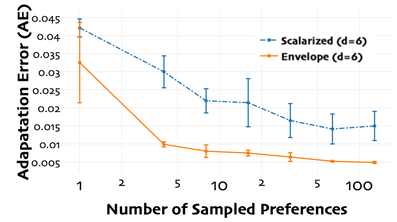
6. Experiments & Results

I. Fruit Tree Navigation:

Envelope MOQ-Learning has better Sample Efficiency & Scalability

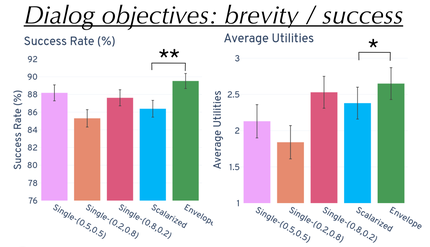






II. Task-Oriented Dialogue Policy Learning:

Dialog objectives: brevity / success



III. Multi-Objective SuperMario Games:

Ground Truth Preference

	x-pos	time	life	coin	enemy
g1	1.000	0.000	0.000	0.000	0.000
g2	0.000	1.000	0.000	0.000	0.000
g3	0.000	0.000	1.000	0.000	0.000
g4	0.000	0.000	0.000	1.000	0.000
g5	0.000	0.000	0.000	0.000	1.000

Δ Preference = Envelope - Scalarized

	x-pos	time	life	coin	enemy
g1	+0.2973	-0.0799	-0.1850	-0.0052	-0.0271
g2	+0.0385	+0.0829	-0.0337	-0.0533	-0.0348
g3	+0.0331	-0.0541	+0.2667	-0.1967	-0.0490
g4	-0.1039	-0.0658	-0.3311	+0.5720	-0.0715
g5	-0.1663	-0.0635	+0.0249	+0.0490	+0.1555

Inferred preferences of the **envelope multi-objective A3C** algorithm in different game variants *with 100 episodes*.