NEURAL INFORMATION PROCESSING SYSTEMS

Weakly Coupled Deep Q-Networks

Ibrahim El Shar[1,2], Daniel Jiang[1,3]

[1]University of Pittsburgh
[2]Hitachi America, Ltd.
[3]Meta

Introduction

- Reinforcement Learning (RL) has contributed to a range of sequential decision-making and control problems: games (Silver et al., 2016), robotic manipulation (Lee et al., 2020), chemical reactions (Zhou et al., 2017), efficient and targeted COVID-19 border testing via RL (Bastani et al. 2021) Nature, ChatGPT (https://openai.com/blog/chatgpt/)
- Despite notable successes, **practical implementation of RL remains challenging**
- Real-world settings pose unique challenges due to **costly interactions** (Dulac-Arnold et al. 2021, Google & DeepMind).
- In a widely-read blog post, Mannor & Tamar, 2023 suggest that RL community focus on "solving concrete real-world problems" (as opposed to, e.g., Atari benchmarks) & the "**deployability**" of RL.
- **How can we make RL more efficient and build deployable RL systems and approaches?**
    - One potential approach to improving sample efficiency is to incorporate additional **structural information** about the problem into the learning process
    - **Examples**: Factored decompositions, Latent or contextual MDPs, Block MDPs, Linear MDPs , Shape-constrained value and/or policy functions, MDPs adhering to closure under policy improvement, Multi-timescale or hierarchical MDPs.

Weakly Coupled MDPs (WCMDPs)

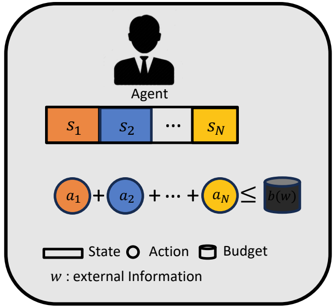
A broad class of sequential decision-making problems. We leverage their inherent structure through a tailored RL approach.

- Multiple independent subproblems: $\boldsymbol{s} = (s_1, \ldots, s_N)$ where $s_i = (x_i, w)$ is the state of subproblem $i \in \{1, \ldots, N\}$, $P(\boldsymbol{s}'|\boldsymbol{s}, a) = \Pi_{i=1}^{N} P(s_i'|s_i, a_i)$ and $q(w'|w)$, $r(\boldsymbol{s}, \boldsymbol{a}) = \sum_{i=1}^{N} r_i(s_i, a_i)$
- Coupling constraint on action space $\mathcal{A}$. Feasible actions:

$$\mathcal{A}(\boldsymbol{s}) = \left\{\boldsymbol{a} \in \mathcal{A}: \textstyle\sum_{i=1}^{N} \boldsymbol{d}_i(s_i, a_i) \le \boldsymbol{b}(w)\right\} \text{ where } \boldsymbol{d}_i(s_i, a_i), \boldsymbol{b}(\mathrm{w}) \in \mathbb{R}^d$$

Bellman equation

$$Q^*(s, a) = \mathbb{E}_{s' \sim P(.|s,a)}[r + \gamma \max_{a' \in \mathcal{A}(s')} Q^*(s', a')|s, a]$$



Real-world applications: supply chain management, recommender systems, EV charging, online advertising, revenue management, stochastic job scheduling, etc.

Challenges:

- Exponential growth of state and action spaces
- Intractability with naive RL algorithms

Lagrangian Relaxation

An approximation technique that **decomposes** WCMDPs

- by **relaxing the linking constraints** to obtain separate subproblems
- these separate problems are much **easier to solve** when considered individually

For any $\boldsymbol{\lambda} \in \mathbb{R}_+^d$, let

$$Q^{\lambda}(\boldsymbol{s},\boldsymbol{a}) = r(\boldsymbol{s},\boldsymbol{a}) + \boldsymbol{\lambda}^T\left(\boldsymbol{b}(w) - \sum_{i=1}^{N}\boldsymbol{d}_i(s_i,a_i)\right) + \gamma\mathbf{E}\left[\max_{\boldsymbol{a}'\in\mathcal{A}} Q^{\lambda}(\boldsymbol{s}',\boldsymbol{a}')\middle|(\boldsymbol{s},\boldsymbol{a})\right]$$

Proposition

- (Weak Duality). $\mathrm{Q}^*(\boldsymbol{s},\boldsymbol{a}) \le Q^{\lambda}(\boldsymbol{s},\boldsymbol{a})$ for any, $\boldsymbol{\lambda} \in \mathbb{R}_+^{\mathrm{d}}, \boldsymbol{a} \in \mathcal{A}(\boldsymbol{s})$
- (Decomposition) $\boxed{Q^{\lambda}(\boldsymbol{s},\boldsymbol{a}) = \boldsymbol{\lambda}^T\boldsymbol{B}(w) + \sum_{i=1}^{N} Q_i^{\lambda}(s_i,a_i),}$ where

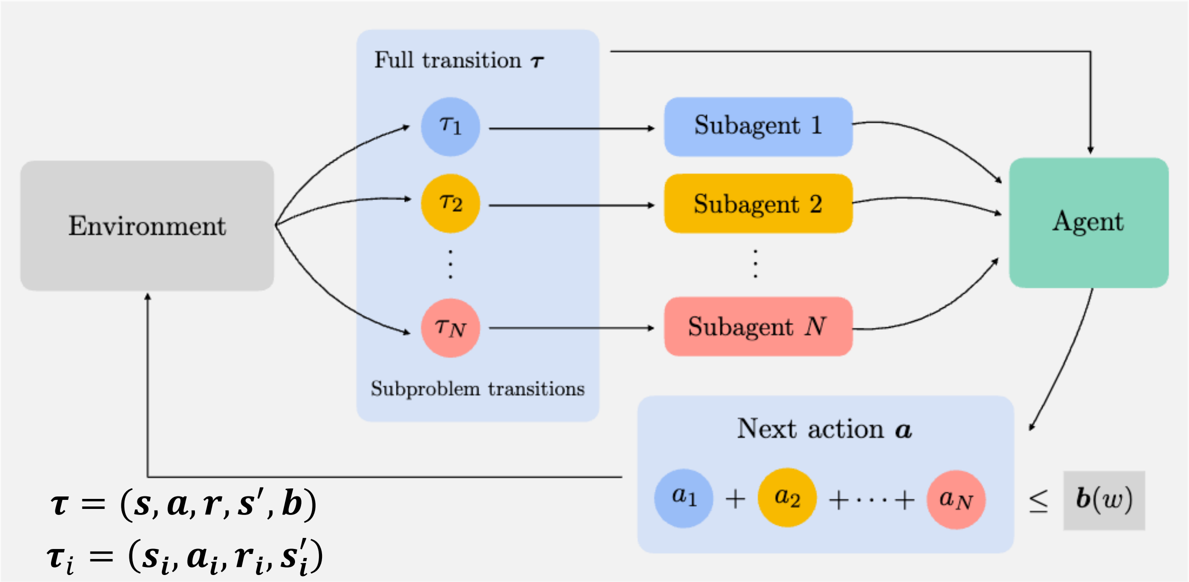
$$Q_i^{\lambda}(s_i,a_i) = r_i(s_i,a_i) - \boldsymbol{\lambda}^T\boldsymbol{d}_i(s_i,a_i) + \gamma\mathbf{E}\left[\max_{a_i'\in\mathcal{A}_i} Q_i^{\lambda}(s_i',a_i')\right] \text{ and } \boldsymbol{B}(w) = \boldsymbol{b}(w) + \gamma\, E[\boldsymbol{B}(w')]$$

Lagrangian dual problem:

$$\mathrm{Q}^{\boldsymbol{\lambda}^*}(\mathbf{s},\mathbf{a}) = \min_{\boldsymbol{\lambda}} \mathrm{Q}^{\boldsymbol{\lambda}}(\mathbf{s},\mathbf{a})$$

Weakly Coupled Q-learning (WCQL)

Main idea is to use the collected experience $\boldsymbol{\tau}$ efficiently by learning from the full problem experience using a main agent and at the same time from the subproblems experience $\boldsymbol{\tau}_i$ using subagents to generate an upper bound on Q^*



Weakly Coupled Q-learning (WCQL)

WCQL algorithm comprises three main steps

Step 1: Subproblems and Subagents

$$Q_{i,n+1}^{\lambda}(s_i, a_i) = Q_{i,n}^{\lambda}(s_i, a_i) + \beta_n(s_i, a_i)[r_i(s_i, a_i) - \lambda^T \boldsymbol{d}(s_i, a_i) + \gamma \mathrm{max}_{a_i'} Q_{i,n}^{\lambda}(s_i, a_i)]$$

Step 2: Learning the Lagrangian Bounds

$$\boldsymbol{B}_{n+1}(w) = \boldsymbol{B}_n(w) + \eta_n(w)[\boldsymbol{b}(w) + \gamma \boldsymbol{B}_n(w') - \boldsymbol{B}_n(w)]$$

$$Q_{n+1}^{\lambda}(\boldsymbol{s}, \boldsymbol{a}) = \lambda^T \boldsymbol{B}_{n+1}(w) + \sum_{i=1}^{N} Q_{i,n+1}^{\lambda}(s_i, a_i)$$

$$Q_{n+1}^{\lambda^*}(\boldsymbol{s}, \boldsymbol{a}) = \min_{\lambda \in \Lambda} Q_{n+1}^{\lambda}(\boldsymbol{s}, \boldsymbol{a})$$

Step 3: Q-Learning Guided by Lagrangian Bounds

$$Q_{n+1}(\boldsymbol{s}, \boldsymbol{a}) = Q_n'(\boldsymbol{s}, \boldsymbol{a}) + \alpha_n(\boldsymbol{s}, \boldsymbol{a}) \left[r(\boldsymbol{s}, \boldsymbol{a}) + \gamma \max_{\boldsymbol{a}' \in \mathcal{A}(\boldsymbol{s}')} Q_n'(\boldsymbol{s}', \boldsymbol{a}') \right]$$

$$Q_n'(\boldsymbol{s}, \boldsymbol{a}) = \min(Q_{n+1}^{\lambda^*}(\boldsymbol{s}, \boldsymbol{a}),\ Q_{n+1}(\boldsymbol{s}, \boldsymbol{a}))$$

Convergence Guarantees

Theorem (Convergence of WCQL). Under typical assumptions on the learning rates and the state visit, the following hold with probability 1:

1. For each i and $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}$, $Q_{i,n}^{\boldsymbol{\lambda}}(s_i, a_i)$ converges to $Q_i^{\boldsymbol{\lambda},*}(s_i, a_i)$ for all $(s_i, a_i) \in \mathcal{S}_i \times \mathcal{A}_i$
2. For each $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}$, $Q_n^{\boldsymbol{\lambda}}(\boldsymbol{s}, \boldsymbol{a}) \geq Q^*(\boldsymbol{s}, \boldsymbol{a})$ as $n \to \infty$ for all $(\boldsymbol{s}, \boldsymbol{a}) \in \mathcal{S} \times \mathcal{A}$
3. $Q'_n(\boldsymbol{s}, \boldsymbol{a})$ converges to $Q^*(\boldsymbol{s}, \boldsymbol{a})$ for all $(\boldsymbol{s}, \boldsymbol{a}) \in \mathcal{S} \times \mathcal{A}$

Weakly Coupled Deep Q-Networks

Initialize main Q-network, subproblems network, and their target networks

$a \sim \epsilon$-greedy(Q)
Store transition in buffer

Sample a minibatch and λs sample

Update the subproblems Q-network using stochastic gradient descent (SGD)

Combine the subproblems to obtain the Lagrangian upper bounds and find the best upper bound target

Compute main target and do a stochastic gradient descent on the soft constrained objective

Algorithm 2 Weakly Coupled DQN

Input: Initialized replay buffer $\mathcal{D}$, Lagrangian multiplier set $\boldsymbol{\Lambda}$, Q-network weights θ, θ^-, subproblem $Q_i^{\boldsymbol{\lambda}}$-network weights θ_U, θ_U^-, and penalty coefficient κ_U.
Output: Approximation $\{Q_n\}$
for $n = 0, 1, 2, \ldots$ **do**
Choose $a_n \sim \epsilon$-greedy(Q_n)), update $\mathbf{B}_{n+1}(w_n)$, and store transition $\boldsymbol{\tau}_n$ in $\mathcal{D}$. Sample minibatch of transitions $\boldsymbol{\tau}$ from $\mathcal{D}$ along with random sample of $\boldsymbol{\lambda}$.
Update subproblem network:
for $i = 1, \ldots, N$ **do**
Compute targets $y_i^{\boldsymbol{\lambda}}$

$$y_i^{\boldsymbol{\lambda}} = r_i(s_i, a_i) - \boldsymbol{\lambda}^T \mathbf{d}_i(s_i, a_i) + \gamma\, \mathbb{E}\big[\max_{a_i'} Q_i^{\boldsymbol{\lambda}}(s_i', a_i'; \theta_U^-)\big],$$

Perform a gradient descent step on

$$l_U(\theta_U) = \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\rho,\boldsymbol{\lambda}\sim\mu}\Big[\textstyle\sum_{i=1}^N \big(y_i^{\boldsymbol{\lambda}} - Q_i^{\boldsymbol{\lambda}}(s_i, a_i; \theta_U)\big)^2\Big],$$

end for
Find the best upper bound:
For $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}$ and $\mathbf{a} \in \mathcal{A}(\mathbf{s}_n)$ find $Q_n^{\boldsymbol{\lambda}}(\mathbf{s}_n, \mathbf{a})$

$$Q^{\boldsymbol{\lambda}}(\mathbf{s}, \mathbf{a}; \theta_U) = \boldsymbol{\lambda}^T \mathbf{B}(w) + \sum_{i=1}^N Q_i^{\boldsymbol{\lambda}}(s_i, a_i; \theta_U).$$

Set $Q_n^{\boldsymbol{\lambda}^*}(\mathbf{s}_n, \mathbf{a}) = \min_{\boldsymbol{\lambda}\in\boldsymbol{\Lambda}} Q_n^{\boldsymbol{\lambda}}(\mathbf{s}_n, \mathbf{a})$.
Compute target y_U

$$y_U = r(\mathbf{s}, \mathbf{a}) + \gamma\mathbb{E}\Big[\max_{\mathbf{a}'} \min_{\boldsymbol{\lambda}\in\boldsymbol{\Lambda}} Q^{\boldsymbol{\lambda}}(\mathbf{s}', \mathbf{a}'; \theta_U^-)\Big].$$

Update main network:
Compute target y

$$y = r(\mathbf{s}, \mathbf{a}) + \gamma\mathbb{E}\big[\max_{\mathbf{a}} Q'(\mathbf{s}', \mathbf{a}; \theta^-)\big]$$

Perform a gradient descent step on

$$l(\theta) = \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\rho,\boldsymbol{\lambda}\sim\mu}\Big[\big(y - Q'(\mathbf{s}, \mathbf{a}; \theta)\big)^2 + \kappa_U \big(Q'(\mathbf{s}, \mathbf{a}; \theta) - y_U\big)_+^2\Big],$$

end for

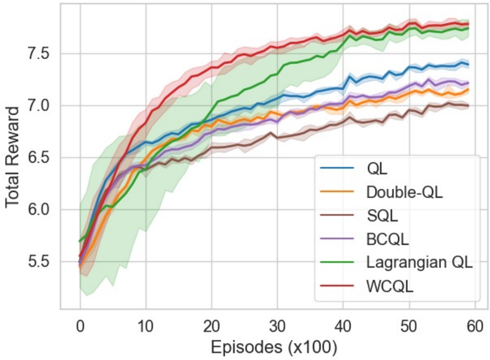
Remarks:

1. Knowledge of environment dynamics is not required
2. We use a **single network** to learn the all subproblems action-values by augmenting the subproblem state and the subproblem number
3. Since the **$\boldsymbol{\lambda}$s are decoupled from the transitions**, each environment transition can be used to learn Q^{λ} for a large sample of **$\boldsymbol{\lambda}$**s

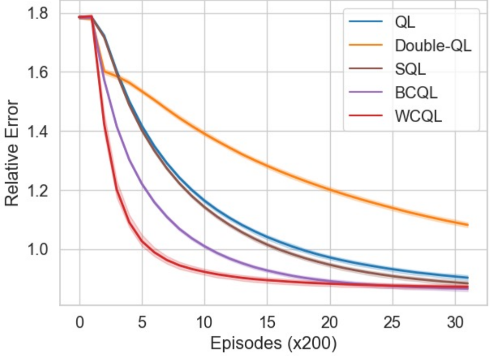
Numerical Experiments

Weakly Coupled Q-learning

- **EV Charging with Exogenous Electricity Cost** (Yu et al . 2018)
  - N=3 charging spots; available charging spots depends on electricity cost
  - Vehicles arrive with a random charging load and duration



Performance plot



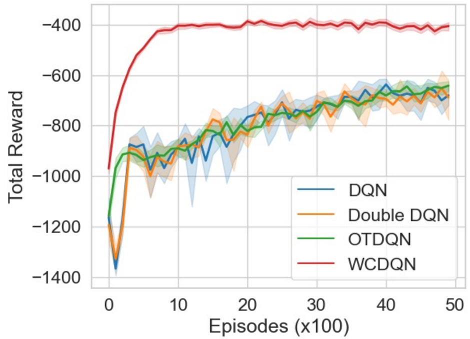
Relative Error
$\|V - V^*\|_2 / \|V^*\|_2$

Numerical Experiments

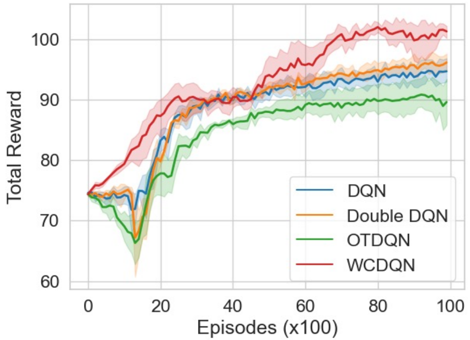
Weakly Coupled Deep Q-Networks

- **Multi-product Inventory Control with an Exogenous Production Rate** (Hodge and Glazebrook 2011)
    - Resource allocation for a facility that manufactures K=10 products
    - Production rates depend on resource allocation and exogenous factors
    - In total there are 3^{10} total actions and a continuous state space

- **Online Stochastic Ad Matching** (Feldman et al., 2009)
    - Matching N=6 advertisers to arriving impressions
    - Advertiser states represent the number of remaining ads to display
    - Rewards depend on the impression type



Multi-product Inventory Control



Online Stochastic Ad. Matching

References

- Silver, D., Huang, A., Maddison, C., Guez, A., Sifre, L., Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., & Hassabis, D. (2016). Mastering the game of go with deep neural networks and tree search. Nature, 529 (7587), 484–489.
- Lee, J., Hwangbo, J., Wellhausen, L., Koltun, V., & Hutter, M. (2020b). Learning quadrupedal locomotion over challenging terrain. Science in Robotics, 5
- Zhou, Z., Li, X., & Zare, R. (2017). Optimizing chemical reactions with deep reinforcement learning. ACS central science, 3 (12), 1337–1344.
- Bastani, Hamsa, et al. "Efficient and targeted COVID-19 border testing via reinforcement learning." Nature 599.7883 (2021): 108-113.
- Dulac-Arnold, Gabriel, Daniel Mankowitz, and Todd Hester. "Challenges of real-world reinforcement learning." arXiv preprint arXiv:1904.12901 (2019)
- Mannor, S., & Tamar, A. (2023). Towards deployable rl–what's broken with rl research and a potential fix. In arXiv preprint arXiv:2301.01320.
- Zhe Yu, Yunjian Xu, and Lang Tong. Deadline scheduling as restless bandits. IEEE Transactionson Automatic Control, 63(8):2343–2358, 2018
- David J Hodge and Kevin D Glazebrook. Dynamic resource allocation in a multi-product make-to-stock production system. Queueing Systems, 67(4):333–364, 2011
- Jon Feldman, Aranyak Mehta, Vahab Mirrokni, and Shan Muthukrishnan. Online stochastic matching: Beating 1-1/e. In 2009 50th Annual IEEE Symposium on Foundations of Computer Science, pages 117–126. IEEE, 2009