Efficient Planning under Partial Observability with Unnormalized Q Functions and Spectral Learning (2020)

By T. Li, B. Mazoure, D. Precup, and G. Rabusseau

Reinforcement Learning

- "*Reinforcement learning is learning what to do (how to map situations to actions) so as to optimize a numerical reward signal*" - Richard Sutton, 1992

- Formalized using Markov decision processes (MDPs)
  - Typical to learn an action-value function q(s, a)

POMDPs and PSRs

- States are hidden in some cases, only certain observations are seen
- By adding a level of abstraction over the state space, we can represent the current state by a set of predictions about future observations based on a history of observations.

$$h \xrightarrow{PSR} s_{psr} \xrightarrow{Q-learning} Q(s_{psr}, a)$$

Unnormalized Q function

- By adding another level of abstraction over the state space, directly compute the action-value function since state representations are not needed in finding optimal policies
- Model-based vs model-free RL

$$h \xrightarrow{PSR} s_{psr} \xrightarrow{Q-learning} Q(s_{psr}, a)$$

$$h \xrightarrow{UQF} \widetilde{Q}(h, a)$$

Preliminaries

Reinforcement learning

MDPs are characterized by:

$$(\mathcal{T}, \mathbf{r}, \mathcal{A}, \mathcal{S}, \mu, \gamma, \Pi)$$

$$\mathcal{T} \in [0,1]^{\mathcal{S}\times\mathcal{A}\times\mathcal{S}}$$
$$\mathbf{r} \in \mathbb{R}^{\mathcal{S}}$$
$$\mu \in [0,1]^{\mathcal{S}}$$
$$\gamma \in [0,1)$$
$$\Pi \in [0,1]^{\mathcal{S}\times\mathcal{A}}$$

POMDPs are similarly:

$$(\mathcal{T}, \mathcal{O}, \mathbf{r}, \mathcal{A}, O, \mathcal{S}, \mu, \gamma, \Pi)$$

$$\mathcal{T} \in [0,1]^{\mathcal{S}\times\mathcal{A}\times\mathcal{S}}$$
$$\mathbf{r} \in \mathbb{R}^{\mathcal{S}}$$
$$\mu \in [0,1]^{\mathcal{S}}$$
$$\gamma \in [0,1)$$
$$\Pi \in [0,1]^{\mathcal{S}\times\mathcal{A}}$$
$$O \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}\times\mathcal{O}}$$

Reinforcement learning

- Goal: Find $\prod^*$which maximizes the long term reward
- Common approaches involve **General policy iteration**:
  - Repeat
    - **Policy evaluation**: estimate $Q^{\Pi_t}(s,a)$
    - **Policy improvement**: $\Pi_{t+1}(s) = \text{argmax}_{a \in \mathcal{A}} Q^{\Pi_t}(s,a)$

Weighted Finite Automata

- The tuple $(\alpha_0, \{\mathbf{A}_\sigma\}_{\sigma\in\Sigma}, \alpha_\infty, \Sigma)$ is a WFA of the function $f : \Sigma^* \to \mathbb{R}$ if $f(x) = \alpha_0^T \mathbf{A}_{x_1}\mathbf{A}_{x_2}\dots\mathbf{A}_{x_n}\alpha_\infty$ where $x = x_1x_2\dots x_n \in \Sigma^*$

Spectral algorithm for learning WFA

- To find $(\alpha_0, \{\mathbf{A}_\sigma\}_{\sigma\in\Sigma}, \alpha_\infty, \Sigma)$ using samples of $f : \Sigma^* \to \mathbb{R}$
  - Build $(\mathbf{H}_f)_{u,v} = f(uv)$ for all $u, v \in \Sigma^*$
  - Factorize $\mathbf{H}_f = \mathbf{P}\mathbf{S}$
  - Let $(\mathbf{H}_\sigma)_{u,v} = f(u\sigma v)$
  - Solution: $\alpha_0^T = \mathbf{P}_{\lambda,:}$, $\alpha_\infty = \mathbf{S}_{:,\lambda}$, $\mathbf{A}_\sigma = \mathbf{P}^\dagger \mathbf{H}_\sigma \mathbf{S}^\dagger$

Predictive State Representation

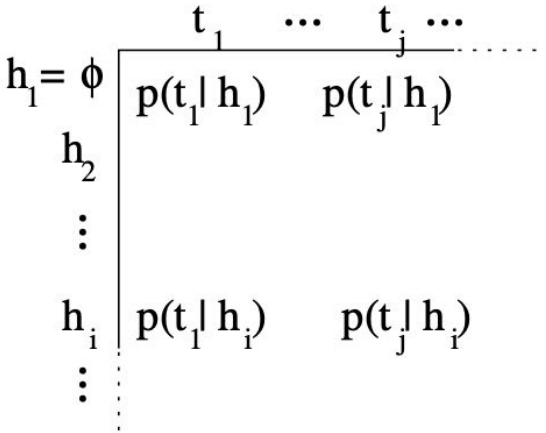
- Lossless representation of any POMDP
- Let $h = a_1 o_1 \dots a_n o_n \in (\mathcal{A} \times \mathcal{O})^*$ denote a possible history or action observation pairs
- Let $t = a_1 o_1 \dots a_n o_n \in (\mathcal{A} \times \mathcal{O})^*$ denote a possible future of action observation pairs. Referred to as tests from here on out.

Predictive State Representation

- Then there exists a set of tests $R = \{t_1, t_2, \ldots, t_k\}$ such that $.P(t|h) = P(R|h)^T m_t$ for all t and h and k is smaller or equal to the number of states in the POMDP. ($m_t$is a weight vector for each t mapping $P(t|h) = P(R|h)^T m_t$)
- In theory, maintaining the prediction vector $.P(h) = [P(t_1|h), P(t_1|h), \ldots, P(t_k|h)]$gives sufficient and complete information of the current state of the POMDP

Predictive State Representation

- Finding the prediction vector is analogous to building and solving the WFA associated with the following Hankel matrix



Predictive State Representation

- At a high level, planning using PSRs consists of learning the state representation for histories, and performing some planning algorithm developed in RL

$$h \xrightarrow{PSR} s_{psr} \xrightarrow{Q-learning} Q(s_{psr}, a)$$

Unnormalized Q function

Defining the UQF

- Definition: the action-value function of a trajectory h:

$$Q^{\Pi}(h,a) = \mathbb{E}^{\Pi}[r_t + \gamma r_{t+1} + \ldots + \gamma^i r_{t+i} + \ldots | ha]$$

- Definition: the reward function of a trajectory h: $\widetilde{R}(h) = \sum_{s \in \mathcal{S}} r_s \mathbb{P}(s|h)$

$$Q^{\Pi}(h,a) = \sum_{z \in \Sigma^*} \sum_{o \in O} \gamma^{|z|} \widetilde{R}(haoz) \mathbb{P}^{\Pi}(haoz|ha)$$

$$Q^{\Pi}(h,a) = \frac{\sum_{z \in \Sigma^*} \sum_{o \in O} \gamma^{|z|} \widetilde{R}(haoz) \mathbb{P}^{\Pi}(haoz)/\Pi(a|h)}{\mathbb{P}^{\Pi}(h)}$$

$$Q^{\Pi}(h,a) = \frac{\widetilde{Q}^{\Pi}(h,a)}{\mathbb{P}(h)}$$

Defining the UQF

- We have $Q^{\Pi}(h,.)\ \alpha\ \widetilde{Q}^{\Pi}(h,.)$ so

  $\text{argmax}_{a\in\mathcal{A}} Q^{\Pi}(h,a) = \text{argmax}_{a\in\mathcal{A}} \widetilde{Q}^{\Pi}(h,a)$
- And $\widetilde{Q}^{\Pi}(h,a) = \sum_{z\in\Sigma^*}\sum_{o\in O} \gamma^{|z|}\widetilde{R}(haoz)\mathbb{P}^{\Pi}(haoz)/\Pi(a|h)$

Theorem

- $\sum_{z \in \Sigma^*} \gamma^{|z|} \widetilde{R}(haoz) \mathbb{P}^{\Pi}(haoz)$ can be computed with a WFA

Lemma

- $g(h) = \widetilde{R}(h)\mathbb{P}^{\Pi}(h)$ can be computed by a WFA $B = (\beta^T, \{\mathbf{B}_\sigma\}_{\sigma\in\Sigma}, \tau)$

- Remember an HMM can be modeled by a WFA with

$$\mathbb{P}(h = x_1 x_2 x_3 \dots) = \alpha_0^T \mathbf{A}^{x_1} \mathbf{A}^{x_2} \dots \mathbf{A}^{x_k} \mathbf{1}$$

Theorem

- $\sum_{z \in \Sigma^*} \gamma^{|z|} \widetilde{R}(haoz) \mathbb{P}^{\Pi}(haoz)$ can be computed with a WFA

UQF algorithm

- Estimate $\mathbf{H}_g$ corresponding to $g(h) = \widetilde{R}(h)\mathbb{P}^{\Pi}(h)$
- Recover $B = (\beta^T, \{\mathbf{B}_\sigma\}_{\sigma\in\Sigma}, \tau)$
- Build $\widetilde{Q}^{\Pi}(h,a) = \sum_{z\in\Sigma^*}\sum_{o\in O}\gamma^{|z|}\widetilde{R}(haoz)\mathbb{P}^{\Pi}(haoz)/\Pi(a|h)$ using B

Estimating the Hankel matrix

- Let $\mathcal{D} = \text{Dataset of trajectories}$

  $\mathbf{y} = \text{Immediate rewards of trajectories}$

  $\mathbb{I}_{uv}(\mathcal{D}_i) = 1 \text{ if } \mathcal{D}_i = uv \ , \ 0 \text{ otherwise}$
- Then

$$(\hat{\mathbf{H}})_{uv} \approx \widetilde{R}(uv)\mathbb{P}(uv) \ = \frac{\sum_{i=0}^{|\mathcal{D}} \mathbb{I}_{uv}(\mathcal{D}_i)\mathbf{y}_i}{|\mathcal{D}|}$$

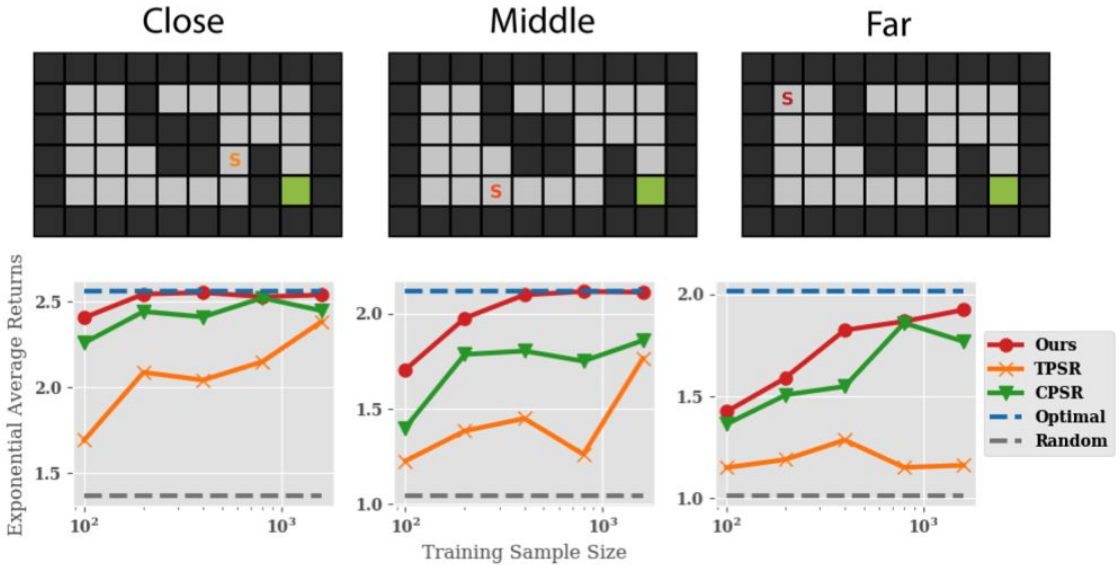
Algorithms summary and comparison

Step	TPSR + fitted-Q	UQF
1	Sample N trajectories using Π_s	Sample N trajectories using Π_s
2	Build $(\hat{\mathbf{H}}_f)_{uv}$ for $f(uv) = \mathbb{P}(uv)$	Build $(\hat{\mathbf{H}}_f)_{uv}$ for $f(uv) = \widetilde{R}(uv)\mathbb{P}(uv)$
3	Perform SVD on $\hat{\mathbf{H}}_f$ to estimate the WFA associated with f, thus giving the PSR of any h, denoted s_{psr}	**Policy evaluation**: perform SVD on $\hat{\mathbf{H}}_f$ to estimate the WFA associated with f, thus mapping $h \to \widetilde{Q}^{\Pi_s}(h, a)$
4	**Policy evaluation**: iteratively solve for $Q^{\Pi_s}(s_{psr}, a)$ using fitted-Q	
5	**Policy improvement**: $\Pi_{new} = \text{argmax}_{a \in \mathcal{A}} Q^{\Pi_s}(s_{psr}, a)$	**Policy improvement**: $\Pi_{new} = \text{argmax}_{a \in \mathcal{A}} \widetilde{Q}^{\Pi_s}(h, a)$
6	$\Pi_s \leftarrow \Pi_{new}$, and repeat from step 1	$\Pi_s \leftarrow \Pi_{new}$, and repeat from step 1

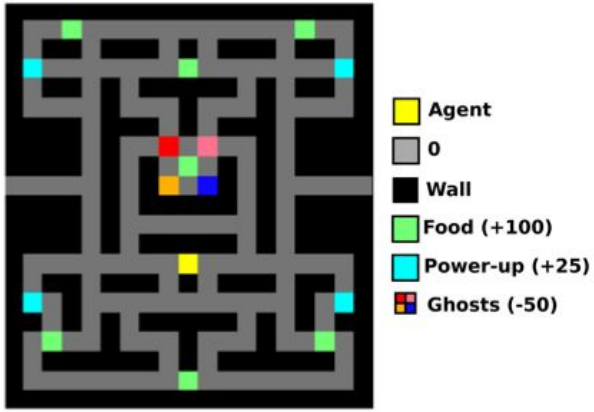
Experiments

Grid world

- 100x speed-up



S-PocMan



Method	Fitted-Q Iterations	Time (s)	Returns
UQF	-	**2**	**-92**
CPSR	400	489	-101
	100	116	-109
	50	60	-150
	10	15	-200

Conclusion

- Novel method for learning and planning in a POMDP environment
  - More sample efficient
  - Model-free version of traditional methods
- Experimental results show better performance and ~ 200 times faster run times for UQF compared to state of the art PSR models
- As in model-free vs model-based RL, model-free algorithms may lack interpretability
- (Scalability issues common to PSR and UQF is similarly addressed here)

References

- UQF (2020), https://arxiv.org/abs/1911.05010
- PSR (2002), https://web.eecs.umich.edu/~baveja/Papers/psr.pdf
- TSPR with Spectral learning (2011), http://www.cs.cmu.edu/~ggordon/boots-siddiqi-gordon-closing-loop-psrs.pdf
- PSR - more theory (2004), https://arxiv.org/abs/1207.4167
- CPSR (2013), http://proceedings.mlr.press/v28/hamilton13.html