Agent-state based policies in POMDPs: Beyond belief-state MDPs

Aditya Mahajan
McGill University

Data Driven Learning and Control Seminar Series
Cornell University
23 Jan 2025

▷ email: aditya.mahajan@mcgill.ca
▷ web: https://adityam.github.io

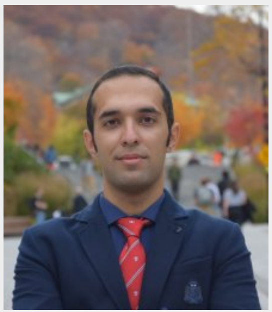
Acknowledgements

Jayakumar Subramanian

Amit Sinha

Mathieu Geist

Erfan SeyedSalehi



Nima Akbarzadeh

Tianwei Ni

Pierre Luc Bacon

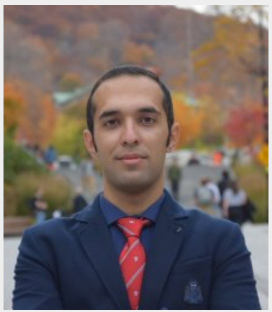
Acknowledgements

Jayakumar Subramanian

Amit Sinha

Mathieu Geist

Erfan SeyedSalehi



Nima Akbarzadeh

Tianwei Ni

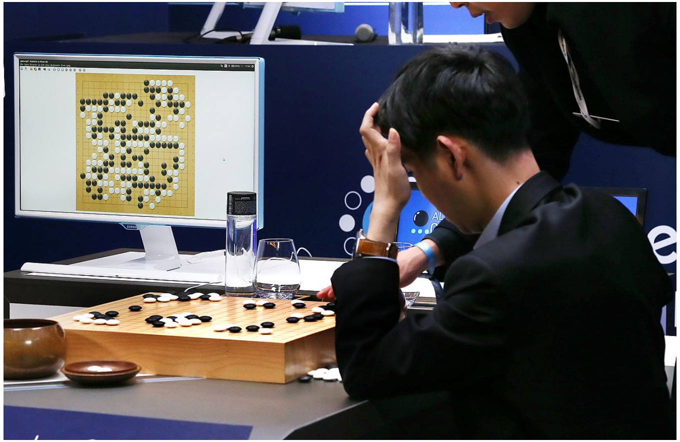
Pierre Luc Bacon

NSERC CRSNG

IDEaS INNOVATION FOR DEFENCE EXCELLENCE AND SECURITY

Recent successes of RL

Recent successes of RL

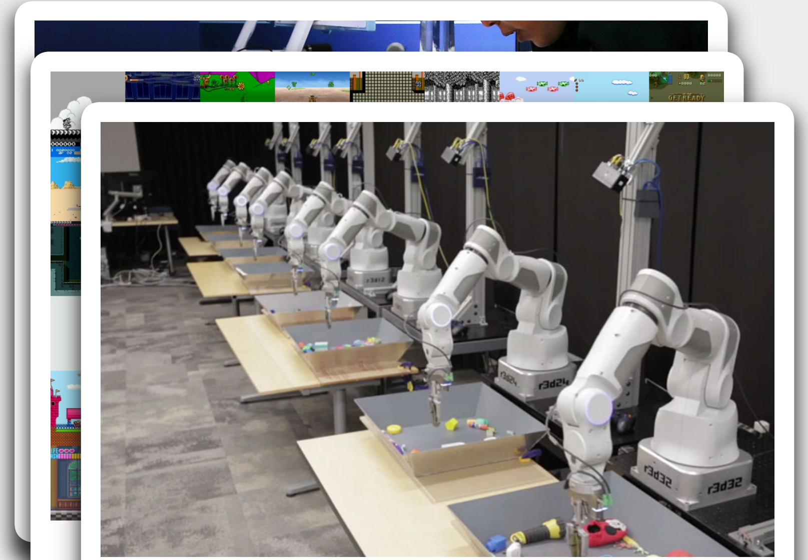


Alpha Go

Recent successes of RL

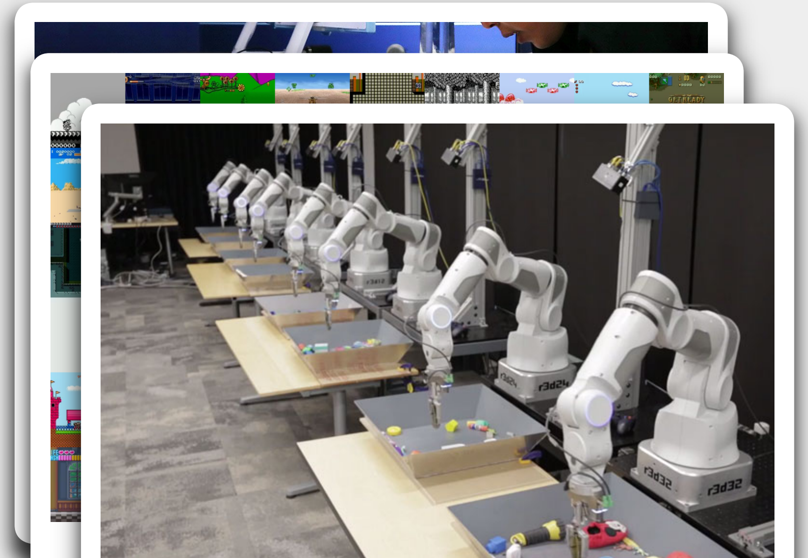
Arcade games

Recent successes of RL



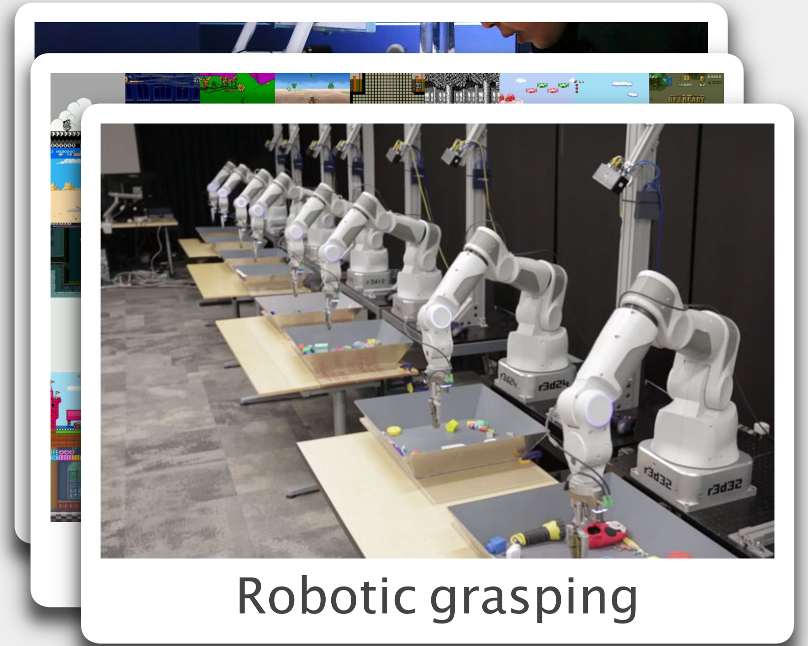
Robotic grasping

Recent successes of RL



Robotic grasping

- Algorithms based on comprehensive theory
- The theory is restricted almost exclusively to systems with perfect state observations



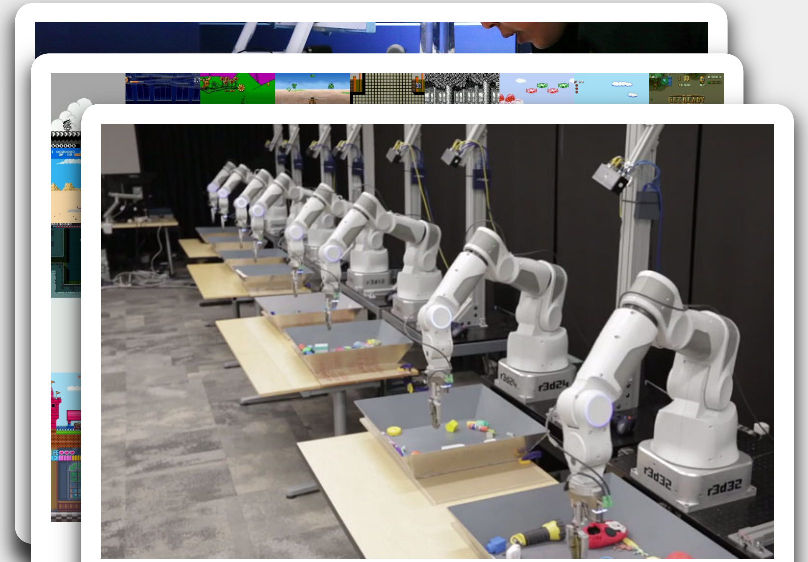
Robotic grasping

Recent successes of RL

- Algorithms based on comprehensive theory
- The theory is restricted almost exclusively to systems with perfect state observations

Many real-world applications are partially observed

- Healthcare
- Autonomous driving
- Finance (portfolio management)
- Retail and marketing



Robotic grasping

Recent successes of RL

- Algorithms based on comprehensive theory
- The theory is restricted almost exclusively to systems with perfect state observations

Many real-world applications are partially observed

- Healthcare
- Autonomous driving
- Finance (portfolio management)
- Retail and marketing

How do we develop a theory for RL for partially observed systems?

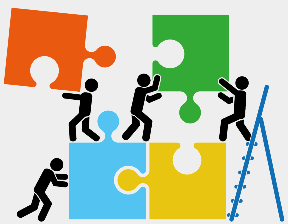
Outline

Background

- Review of MDPs and RL
- Review of POMDPs
- Why is RL for POMDPs difficult?

Agent-state based planning

- Agent state based policies
- Policy classes
- Planning for different policy classes



Agent-state based learning

- Agent state based Q-Learning
- Self-predictive representation learning
- Agent state based actor-critic

Outline

Background

- Review of MDPs and RL
- Review of POMDPs
- Why is RL for POMDPs difficult?

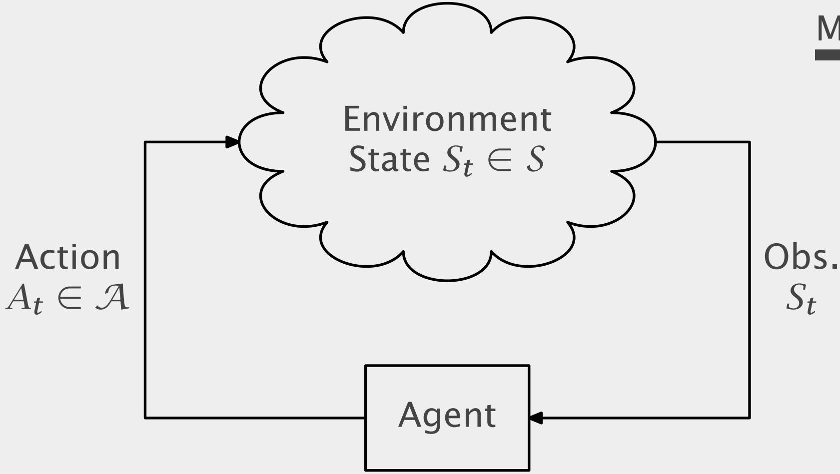
Agent-state based planning

- Agent state based policies
- Policy classes
- Planning for different policy classes

Agent-state based learning

- Agent state based Q-Learning
- Self-predictive representation learning
- Agent state based actor-critic

Review: Markov decision processes (MDPs)



MDP: MARKOV DECISION PROCESS

Dynamics: $\mathbb{P}(S_{t+1} \mid S_t, A_t)$

Observations: S_t

Reward $R_t = r(S_t, A_t)$.

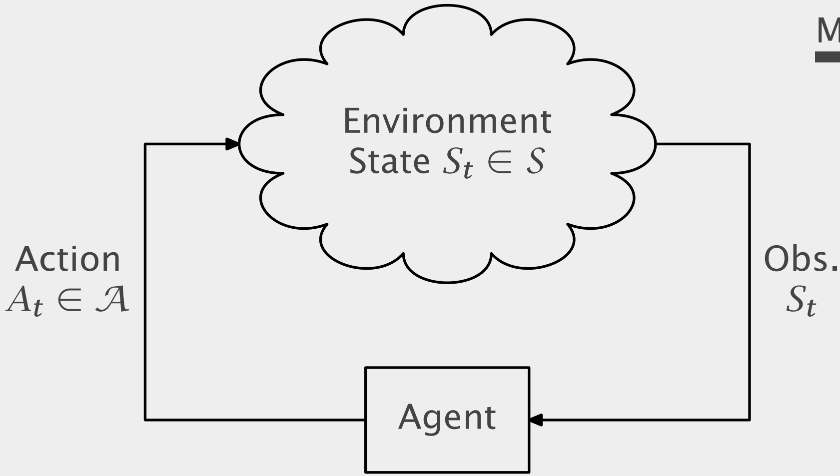
Action: $A_t \sim \pi_t(S_{1:t}, A_{1:t-1})$.

$\pi = (\pi_t)_{t \ge 1}$ is called a policy.

The objective is to choose a policy π to maximize:

$$J(\pi) \coloneqq \mathbb{E}^{\pi}\Big[\sum_{t=1}^{\infty} \gamma^{t-1} R_t \Big]$$

Review: Markov decision processes (MDPs)



MDP: MARKOV DECISION PROCESS

Dynamics: $\mathbb{P}(S_{t+1} \mid S_t, A_t)$

Observations: S_t

Reward $R_t = r(S_t, A_t)$.

Action: $A_t \sim \pi_t(S_{1:t}, A_{1:t-1})$.

$\pi = (\pi_t)_{t \geq 1}$ is called a policy.

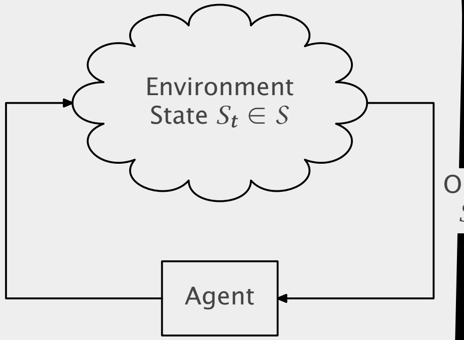
The objective is to choose a policy π to maximize:

$$J(\pi) \coloneqq \mathbb{E}^{\pi}\left[\sum_{t=1}^{\infty} \gamma^{t-1} R_t\right]$$

Conceptual challenge

- Brute force search has an exponential complexity in time horizon.
- How to efficiently search an optimal policy?

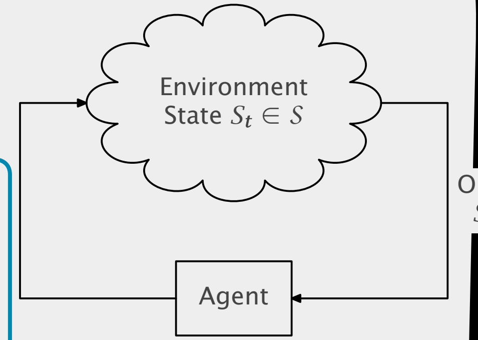
Review: Key simplifying ideas



Principle of Irrelevant information	There is no loss of optimality in choosing the action A_t as a function of the current state S_t

Blackwell, "Memoryless strategies in finite-stage dynamic prog.," Annals Math. Stats, 1964.

Review: Key simplifying ideas



Principle of Irrelevant information	There is no loss of optimality in choosing the action A_t as a function of the current state S_t

Blackwell, "Memoryless strategies in finite-stage dynamic prog.," Annals Math. Stats, 1964.

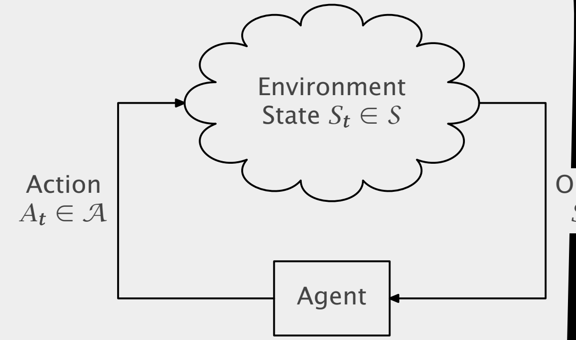
Principle of Optimality	The optimal control policy is given a DP with state S_t: $V(s) = \max_{a \in \mathcal{A}} \Big\{ r(s,a) + \gamma \int V(s') P(ds' \mid s, a) \Big\}$

Bellman, "Dynamic Programming," 1957.

Review: Reinforcement Learning (RL)

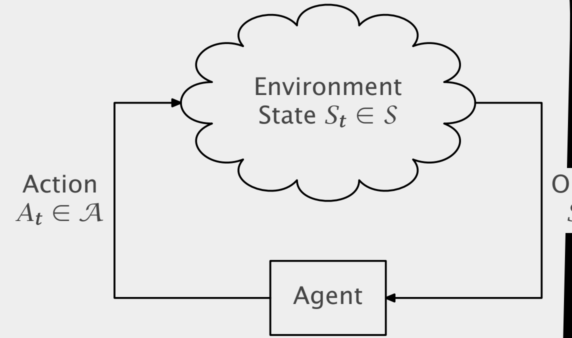
The (online) RL setting

- Dynamics and reward functions are unknown.
- Agent can interact with the environment and observe states and rewards.
- Design algorithm that asymptotically identify an optimal policy.



Review: Reinforcement Learning (RL)

The (online) RL setting



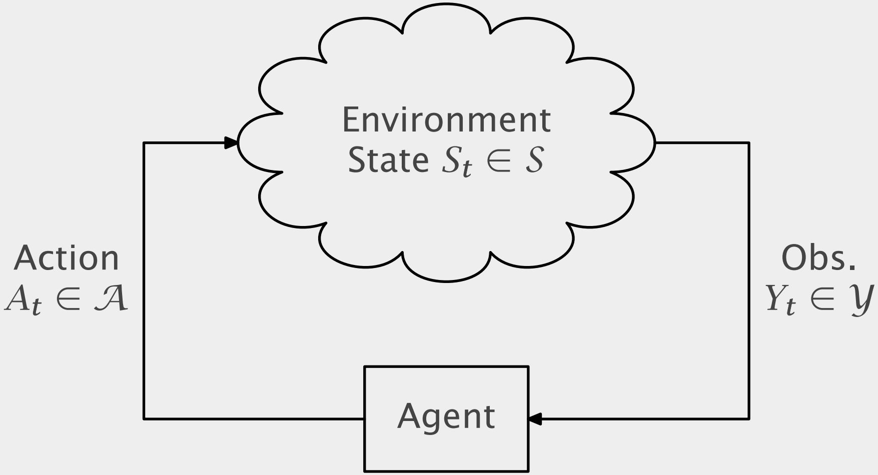
- Dynamics and reward functions are unknown.
- Agent can interact with the environment and observe states and rewards.
- Design algorithm that asymptotically identify an optimal policy.

Value based methods	Estimate the Q-function $Q(s,a) = r(s,a) + \gamma \int V(s')P(ds'\|s,a)$ using temporal difference learning (i.e., stochastic approximation). [Watkins and Dayan, 1992; Tsitsiklis, 1994]

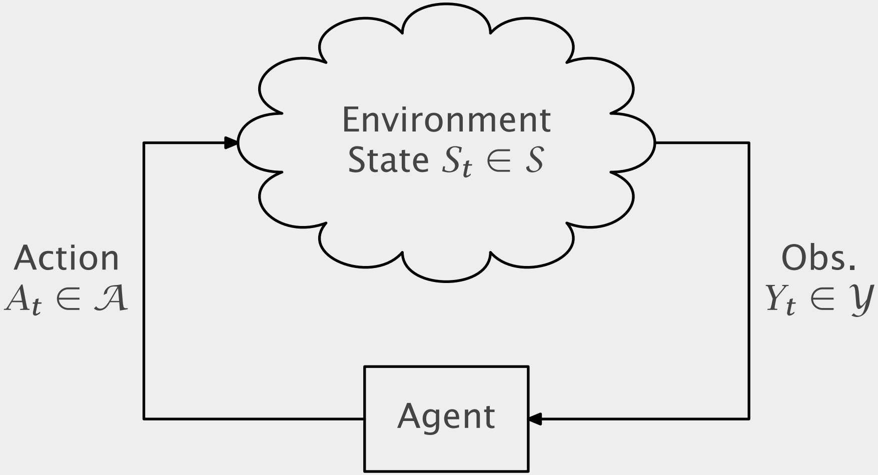
Policy-based methods	Use parameterized policies π_θ. Estimate $\nabla_\theta V_\theta(s)$ using single trajectory gradient estimates (i.e., infitesimal perturbation analysis). [Sutton 2000, Marback and Tsitsiklis 2001], [Cao, 1985; Ho, 1987]

Why is learning difficult in partially observable environments?

POMDPs: Partially observable Markov decision processes



POMDPs: Partially observable Markov decision processes



$$\mathbb{P}(S_{t+1}, Y_{t+1} | S_{1:t}, Y_{1:t}, A_{1:t}) = \mathbb{P}(S_{t+1}, Y_{t+1} | S_t, A_t)$$

Reward: $R_t = r(S_t, A_t)$.

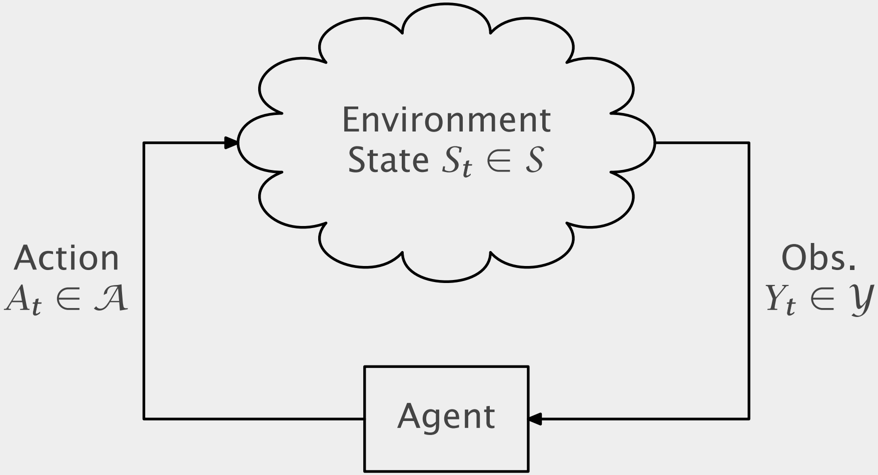
Policy: $\vec{\boldsymbol{\pi}} = (\vec{\pi}_1, \vec{\pi}_2, \dots)$ where

$$A_t \sim \vec{\pi}_t(Y_{1:t}, A_{1:t-1})$$

Performance:

$$J(\vec{\boldsymbol{\pi}}) \coloneqq \mathbb{E}^{\vec{\boldsymbol{\pi}}}\Big[\sum_{t=1}^{\infty} \gamma^{t-1} R_t \,\Big|\, S_1 \sim \xi_1\Big]$$

POMDPs: Partially observable Markov decision processes



$$\mathbb{P}(S_{t+1}, Y_{t+1} | S_{1:t}, Y_{1:t}, A_{1:t}) = \mathbb{P}(S_{t+1}, Y_{t+1} | S_t, A_t)$$

Reward: $R_t = r(S_t, A_t)$.

Policy: $\vec{\boldsymbol{\pi}} = (\vec{\pi}_1, \vec{\pi}_2, \dots)$ where

$$A_t \sim \vec{\pi}_t(Y_{1:t}, A_{1:t-1})$$

Performance:

$$J(\vec{\boldsymbol{\pi}}) \coloneqq \mathbb{E}^{\vec{\boldsymbol{\pi}}}\Bigg[\sum_{t=1}^{\infty} \gamma^{t-1} R_t \,\Bigg|\, S_1 \sim \xi_1 \Bigg]$$

Objective: Find the (history-dependent) policy $\vec{\boldsymbol{\pi}}$ that maximizes $J(\vec{\boldsymbol{\pi}})$

Review: Belief-state based planning

Key simplifying idea

Define **belief state** $B_t \in \Delta(\mathcal{S})$ as $B_t(s) = \mathbb{P}(S_t = s \mid Y_{1:t}, A_{1:t-1})$.

- Belief state updates in a state-like manner: $B_{t+1} = \text{function}(B_t, Y_{t+1}, A_t)$.
- Belief state is sufficient to evaluate rewards: $\mathbb{E}[R_t \mid Y_{1:t}, A_{1:t}] = \hat{r}(B_t, A_t)$.

Thus, $\{B_t\}_{t \geq 1}$ is a perfectly observed controlled Markov process.

Astrom, "Optimal control of Markov processes with incomplete information," JMAA 1965.
Stratonovich, "Conditional Markov Processes," TVP 1960.

Review: Belief-state based planning

Key simplifying idea

Define **belief state** $B_t \in \Delta(\mathcal{S})$ as $B_t(s) = \mathbb{P}(S_t = s \mid Y_{1:t}, A_{1:t-1})$.

- Belief state updates in a state-like manner: $B_{t+1} = \text{function}(B_t, Y_{t+1}, A_t)$.
- Belief state is sufficient to evaluate rewards: $\mathbb{E}[R_t \mid Y_{1:t}, A_{1:t}] = \hat{r}(B_t, A_t)$.

Thus, $\{B_t\}_{t \ge 1}$ is a perfectly observed controlled Markov process. Therefore:

Structure of optimal policy	There is no loss of optimality in choosing the action A_t as a function of the belief state B_t
Dynamic Program	The optimal control policy is given a DP with belief B_t as state.

Implications of the POMDP modeling framework

Implications for planning	
	▷ Allows the use of the MDP machinery for partially observed sys.
	▷ Various exact and approximate algorithms to efficiently solve DP.
	Exact: incremental pruning, witness algorithm, linear support algo
	Approximate: QMDP, point based methods, SARSOP, DESPOT, . . .

Implications of the POMDP modeling framework

- Allows the use of the MDP machinery for partially observed sys.

Implications for learning

- The construction of the belief state depends on the system model.
- So, when the system model is unknown, we cannot construct the belief state and therefore cannot use standard RL algorithms.

Implications of the POMDP modeling framework

- Allows the use of the MDP machinery for partially observed sys.

Implications for learning

- The construction of the belief state depends on the system model.
- So, when the system model is unknown, we cannot construct the belief state and therefore cannot use standard RL algorithms.
- **On the theoretical side:**
  - Propose alternative methods: PSRs (predictive state representations), bisimulation metrics, ...
  - Good theoretical guarantees, but difficult to scale.

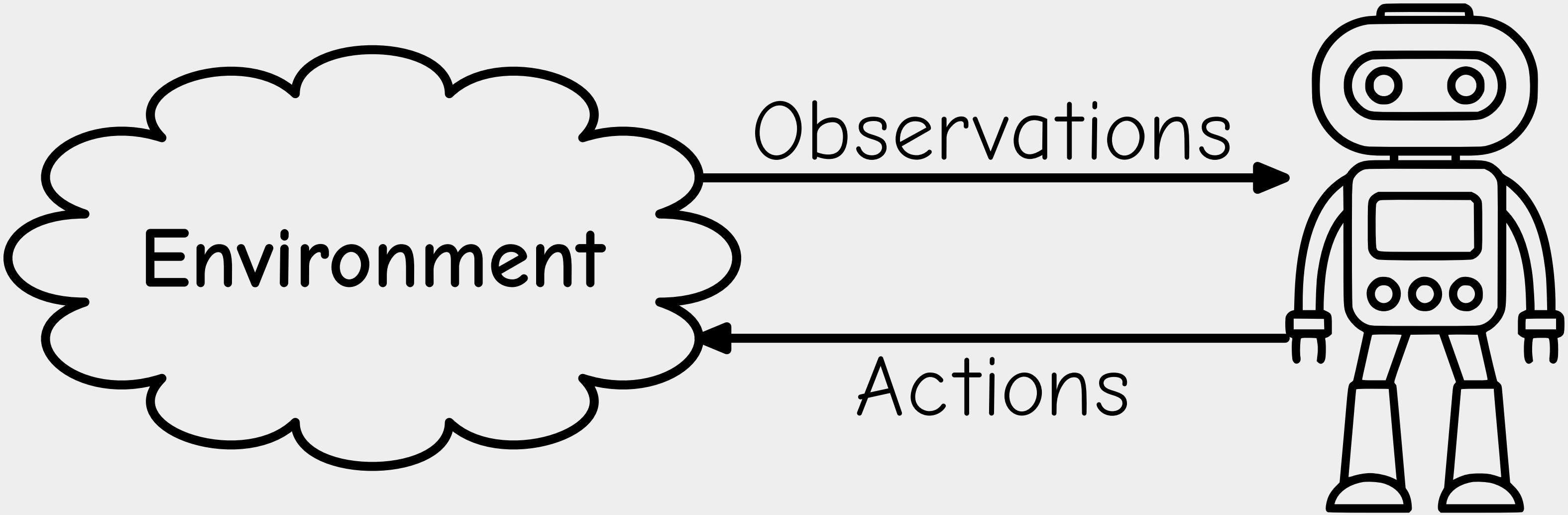
Implications of the POMDP modeling framework

- Allows the use of the MDP machinery for partially observed sys.

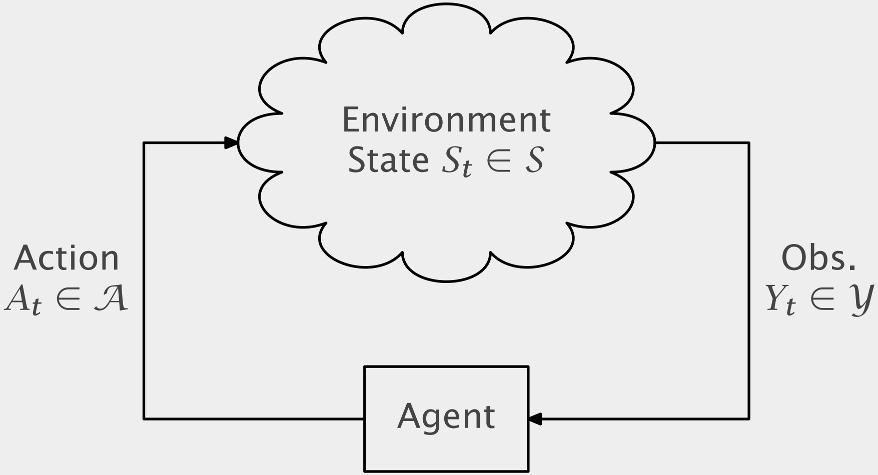
Implications for learning

- The construction of the belief state depends on the system model.
- So, when the system model is unknown, we cannot construct the belief state and therefore cannot use standard RL algorithms.
- **On the theoretical side:**
  - Propose alternative methods: PSRs (predictive state representations), bisimulation metrics, ...
  - Good theoretical guarantees, but difficult to scale.
- **On the practical side:**
  - Simply stack the previous k observations and treat it as a “state”.
  - Instead of a CNN, use an RNN to model policy and action-value fn.
  - Can be made to work but lose theoretical guarantees and insights.

Deep RL learns agent-state based policies



Abstract model of agent-state based policies



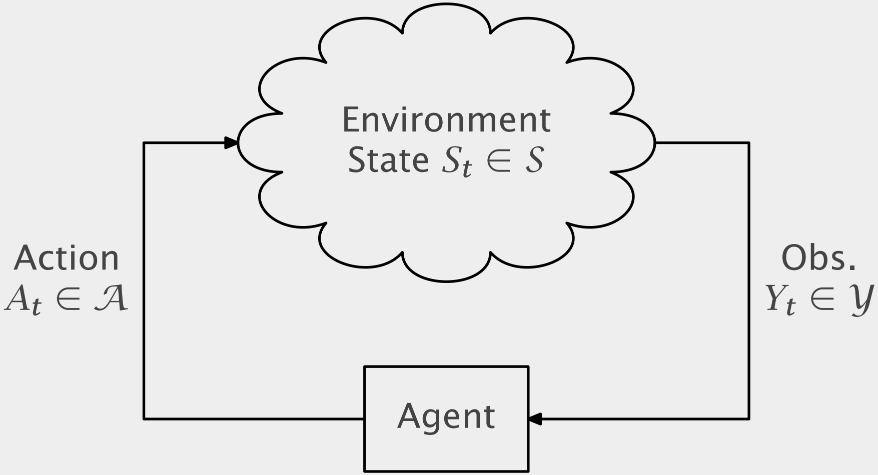
Agent state: $Z_t \in \mathcal{Z}$, where

$$Z_{t+1} = \phi(Z_t, Y_{t+1}, A_t)$$

Examples:

- $Z_t = (Y_{t-n:t}, A_{t-n:t-1})$
- Finite-state controllers
- Recurrent neural networks

Abstract model of agent-state based policies



Agent state: $Z_t \in \mathcal{Z}$, where

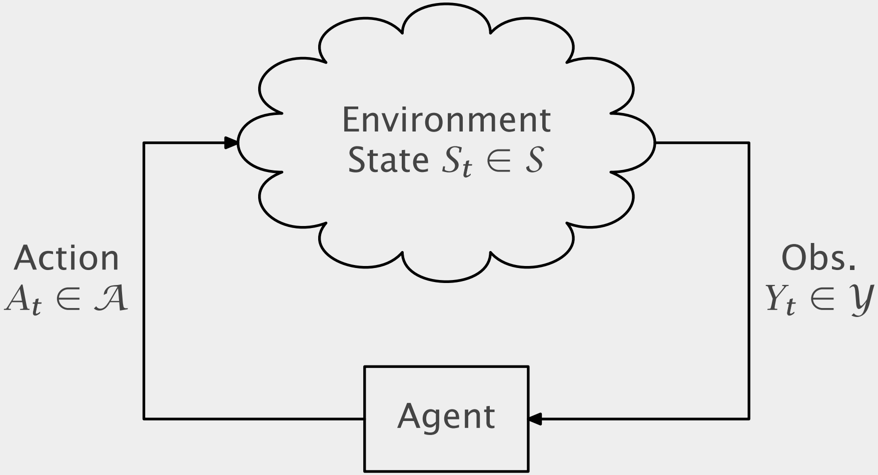
$$Z_{t+1} = \phi(Z_t, Y_{t+1}, A_t)$$

Examples:

- $Z_t = (Y_{t-n:t}, A_{t-n:t-1})$
- Finite-state controllers
- Recurrent neural networks

Notation: $H_t = (Y_{1:t}, A_{1:t-1})$
and $Z_t = \vec{\sigma}_t(H_t)$.

Abstract model of agent-state based policies



Agent state: $Z_t \in \mathcal{Z}$, where

$$Z_{t+1} = \phi(Z_t, Y_{t+1}, A_t)$$

Examples:

- $Z_t = (Y_{t-n:t}, A_{t-n:t-1})$
- Finite-state controllers
- Recurrent neural networks

Notation: $H_t = (Y_{1:t}, A_{1:t-1})$
and $Z_t = \vec{\sigma}_t(H_t)$.

Fundamental Questions

Q1. When is there no loss of optimality in restricting attention to agent state based policies?

Q2. For given $\mathcal{Z}$ and ϕ, find optimal agent-state based policy.

Q3. For given $\mathcal{Z}$, find optimal state update rule ϕ and optimal agent-state based policy.

Answer to Q1: Information states

Answer to Q1: Information states

Information State

Agent state is an information state if it satisfies:

(P1) Sufficient for performance evaluation $\exists r_{\mathrm{IS}} \colon \mathcal{Z} \times \mathcal{A} \to \mathbb{R}$ s.t.

$$\mathbb{E}[R_t \mid H_t, A_t] = r_{\mathrm{IS}}(\vec{\sigma}_t(H_t), A_t)$$

(P2) Sufficient for predicting itself $\exists P_{\mathrm{IS}} \colon \mathcal{Z} \times \mathcal{A} \to \Delta(\mathcal{Z})$ s.t.

$$\mathbb{P}(Z_{t+1} = \cdot \mid H_t, A_t) = P_{\mathrm{IS}}(Z_{t+1} = \cdot \mid \vec{\sigma}_t(H_t), A_t)$$

Answer to Q1: Information states

Information State

Agent state is an information state if it satisfies:

(P1) Sufficient for performance evaluation $\exists r_{\text{IS}} \colon \mathcal{Z} \times \mathcal{A} \to \mathbb{R}$ s.t.

$$\mathbb{E}[R_t \mid H_t, A_t] = r_{\text{IS}}(\vec{\sigma}_t(H_t), A_t)$$

(P2) Sufficient for predicting itself $\exists P_{\text{IS}} \colon \mathcal{Z} \times \mathcal{A} \to \Delta(\mathcal{Z})$ s.t.

$$\mathbb{P}(Z_{t+1} = \cdot \mid H_t, A_t) = P_{\text{IS}}(Z_{t+1} = \cdot \mid \vec{\sigma}_t(H_t), A_t)$$

Info state based DP

Consider the following DP:

$$Q^\star_{\text{IS}}(z, a) = r_{\text{IS}}(z, a) + \sum_{z' \in \mathcal{Z}} P_{\text{IS}}(z' | z, a) V^\star_{\text{IS}}(z, a)$$

$$V^\star_{\text{IS}}(z) = \max_{a \in \mathcal{A}} Q^\star_{\text{IS}}(z, a), \qquad \pi^\star_{\text{IS}}(z) = \arg\max_{a \in \mathcal{A}} Q^\star_{\text{IS}}(z, a).$$

Define $\vec{\pi}_{\text{IS},t}(h_t) \coloneqq \pi^\star_{\text{IS}}(\vec{\sigma}_t(h_t))$. Then the policy $\vec{\boldsymbol{\pi}}_{\text{IS}} = (\vec{\pi}_{\text{IS},1}, \vec{\pi}_{\text{IS},2}, \dots)$ **is optimal**, i.e., $\vec{J}(\vec{\boldsymbol{\pi}}_{\text{IS}}) = \vec{J}^\star_{\text{ND}}$.

More on information states

Examples of info states

- Current state in MDPs
- Belief state $B_t = \mathbb{P}(S_t = \cdot | H_t, A_t)$ in POMDPs
- Conditional mean in LQG models
- ...

More on information states

Examples of info states

- Current state in MDPs
- Belief state $B_t = \mathbb{P}(S_t = \cdot | H_t, A_t)$ in POMDPs
- Conditional mean in LQG models
- ...

Non-examples of info state

- Last observation in POMDPs
- Window of last obs. (frame stacking)
- Recurrent neural networks
- ...

More on information states

Examples of info states

- Current state in MDPs
- Belief state $B_t = \mathbb{P}(S_t = \cdot | H_t, A_t)$ in POMDPs
- Conditional mean in LQG models
- ...

Non-examples of info state

- Last observation in POMDPs
- Window of last obs. (frame stacking)
- Recurrent neural networks
- ...

Info states $\equiv$ DP info

What to do if agent state is not an information state?

Dynamic programming decomposition does not work

Dynamic programming decomposition does not work

General idea of DP

$$
\begin{aligned}
V_t(z_t) &= \min_{a_t \in \mathcal{A}} \mathbb{E}\Big[\text{current reward} + \quad \text{future reward} \qquad \Big|\, Z_t = z_t, A_t = a_t\Big] \\
&= \min_{a_t \in \mathcal{A}} \mathbb{E}\Big[\text{current reward} + \mathbb{E}[\text{future reward} \mid Z_{t+1}] \,\Big|\, Z_t = z_t, A_t = a_t\Big] \\
&= \min_{a_t \in \mathcal{A}} \mathbb{E}\Big[\text{current reward} + V_{t+1}(Z_{t+1}) \,\Big|\, Z_t = z_t, A_t = a_t\Big]
\end{aligned}
$$

Dynamic programming decomposition does not work

General idea of DP

$$\begin{aligned} V_t(z_t) &= \min_{a_t \in \mathcal{A}} \mathbb{E}\Big[\text{current reward} + \quad \text{future reward} \qquad \Big| Z_t = z_t, A_t = a_t\Big] \\ &= \min_{a_t \in \mathcal{A}} \mathbb{E}\Big[\text{current reward} + \mathbb{E}[\text{future reward} \mid Z_{t+1}] \Big| Z_t = z_t, A_t = a_t\Big] \\ &= \min_{a_t \in \mathcal{A}} \mathbb{E}[\Big[\text{current reward} + V_{t+1}(Z_{t+1}) \Big| Z_t = z_t, A_t = a_t\Big] \end{aligned}$$

When agent state is not info-state:

$\sigma(Z_t, A_t) \not\subset \sigma(Z_{t+1})$. Thus, cannot use smoothing property of conditional expectation and

$$\mathbb{E}\Big[\mathbb{E}[\text{future reward} \mid Z_{t+1}]\Big| Z_t, A_t\Big] \neq \mathbb{E}\Big[\text{cost-to-go} \Big| Z_t, A_t\Big]$$

Policy classes for history based policies

$\overrightarrow{\Pi}_{\mathsf{NS}}$: history-dependent Non-stationary Stochastic

$\overrightarrow{\Pi}_{\mathsf{ND}}$: history-dependent Non-stationary Deterministic

Policy classes for history based policies

$\vec{\Pi}_{\text{NS}}$: history-dependent Non-stationary Stochastic $\qquad \vec{J}^{\star}_{\text{NS}} = \sup_{\vec{\boldsymbol{\pi}} \in \vec{\Pi}_{\text{NS}}} J(\vec{\boldsymbol{\pi}}).$

$\vec{\Pi}_{\text{ND}}$: history-dependent Non-stationary Deterministic $\qquad \vec{J}^{\star}_{\text{ND}} = \sup_{\vec{\boldsymbol{\pi}} \in \vec{\Pi}_{\text{ND}}} J(\vec{\boldsymbol{\pi}}).$

Policy classes for history based policies

$\vec{\Pi}_{\text{NS}}$: history-dependent Non-stationary Stochastic $\qquad \vec{J}^{\star}_{\text{NS}} = \sup_{\vec{\boldsymbol{\pi}} \in \vec{\Pi}_{\text{NS}}} J(\vec{\boldsymbol{\pi}})$.

$\vec{\Pi}_{\text{ND}}$: history-dependent Non-stationary Deterministic $\qquad \vec{J}^{\star}_{\text{ND}} = \sup_{\vec{\boldsymbol{\pi}} \in \vec{\Pi}_{\text{ND}}} J(\vec{\boldsymbol{\pi}})$.

There is no loss of optimality in restricting attention to deterministic policies (follows from Kuhn's theorem in Game Theory)

$$\vec{J}^{\star}_{\text{ND}} = \vec{J}^{\star}_{\text{NS}}$$

Policy classes for agent state based policies

Π_{NS} : agent-state based Non-stationary Stochastic

$\boldsymbol{\pi} = (\pi_1, \pi_2, \dots),\ \pi_t \colon \mathcal{Z} \to \Delta(\mathcal{A})$

Policy classes for agent state based policies

Π_{NS} : agent-state based Non-stationary Stochastic

$$\boldsymbol{\pi} = (\pi_1, \pi_2, \dots),\ \pi_t \colon \mathcal{Z} \to \Delta(\mathcal{A})$$

Π_{ND} : agent-state based Non-stationary Deterministic

$$\boldsymbol{\pi} = (\pi_1, \pi_2, \dots),\ \pi_t \colon \mathcal{Z} \to \mathcal{A}$$

Policy classes for agent state based policies

Π_{NS} : agent-state based Non-stationary Stochastic

$$\boldsymbol{\pi} = (\pi_1, \pi_2, \dots),\ \pi_t \colon \mathcal{Z} \to \Delta(\mathcal{A})$$

Π_{ND} : agent-state based Non-stationary Deterministic

$$\boldsymbol{\pi} = (\pi_1, \pi_2, \dots),\ \pi_t \colon \mathcal{Z} \to \mathcal{A}$$

Π_{SS} : agent-state based Stationary Stochastic

$$\boldsymbol{\pi} = (\pi, \pi, \dots),\ \pi \colon \mathcal{Z} \to \Delta(\mathcal{A})$$

Policy classes for agent state based policies

Π_{NS} : agent-state based Non-stationary Stochastic

$$\boldsymbol{\pi} = (\pi_1, \pi_2, \dots),\ \pi_t \colon \mathcal{Z} \to \Delta(\mathcal{A})$$

Π_{ND} : agent-state based Non-stationary Deterministic

$$\boldsymbol{\pi} = (\pi_1, \pi_2, \dots),\ \pi_t \colon \mathcal{Z} \to \mathcal{A}$$

Π_{SS} : agent-state based Stationary Stochastic

$$\boldsymbol{\pi} = (\pi, \pi, \dots),\ \pi \colon \mathcal{Z} \to \Delta(\mathcal{A})$$

Π_{SD} : agent-state based Stationary Deterministic

$$\boldsymbol{\pi} = (\pi, \pi, \dots),\ \pi \colon \mathcal{Z} \to \mathcal{A}$$

Policy classes for agent state based policies

Π_{NS} : agent-state based Non-stationary Stochastic
$\boldsymbol{\pi} = (\pi_1, \pi_2, \dots),\ \pi_t \colon \mathcal{Z} \to \Delta(\mathcal{A})$

$$J^\star_{\text{NS}} = \sup_{\boldsymbol{\pi} \in \Pi_{\text{NS}}} J(\boldsymbol{\pi}).$$

Π_{ND} : agent-state based Non-stationary Deterministic
$\boldsymbol{\pi} = (\pi_1, \pi_2, \dots),\ \pi_t \colon \mathcal{Z} \to \mathcal{A}$

$$J^\star_{\text{ND}} = \sup_{\boldsymbol{\pi} \in \Pi_{\text{ND}}} J(\boldsymbol{\pi}).$$

Π_{SS} : agent-state based Stationary Stochastic
$\boldsymbol{\pi} = (\pi, \pi, \dots),\ \pi \colon \mathcal{Z} \to \Delta(\mathcal{A})$

$$J^\star_{\text{SS}} = \sup_{\boldsymbol{\pi} \in \Pi_{\text{SS}}} J(\boldsymbol{\pi}).$$

Π_{SD} : agent-state based Stationary Deterministic
$\boldsymbol{\pi} = (\pi, \pi, \dots),\ \pi \colon \mathcal{Z} \to \mathcal{A}$

$$J^\star_{\text{SD}} = \sup_{\boldsymbol{\pi} \in \Pi_{\text{SD}}} J(\boldsymbol{\pi}).$$

Relationship between different policy classes

$$
\begin{array}{c}
J^\star_{\mathsf{SD}} \\
\big\downarrow \\
J^\star_{\mathsf{SS}}
\end{array}
$$

Relationship between different policy classes

```tikzcd
J^\star_{\mathsf{SD}} \arrow[r] \arrow[d] & J^\star_{\mathsf{ND}} \arrow[d] \\
J^\star_{\mathsf{SS}} \arrow[r] & J^\star_{\mathsf{NS}}
```

Relationship between different policy classes

$$\begin{tikzcd}
J^\star_{\mathsf{SD}} \arrow[r] \arrow[d] & J^\star_{\mathsf{ND}} \arrow[r] \arrow[d] & \vec{J}^\star_{\mathsf{ND}} \arrow[d] \\
J^\star_{\mathsf{SS}} \arrow[r] & J^\star_{\mathsf{NS}} \arrow[r] & \vec{J}^\star_{\mathsf{NS}}
\end{tikzcd}$$

Relationship between different policy classes

$$
\begin{tikzcd}
J^\star_{\mathsf{SD}} \arrow[r] \arrow[d] & J^\star_{\mathsf{ND}} \arrow[r] \arrow[d] & \vec{J}^\star_{\mathsf{ND}} \arrow[d, leftrightarrow] \\
J^\star_{\mathsf{SS}} \arrow[r] & J^\star_{\mathsf{NS}} \arrow[r] & \vec{J}^\star_{\mathsf{NS}}
\end{tikzcd}
$$

History based policies

Relationship between different policy classes

$$
\begin{tikzcd}
J^\star_{\mathsf{SD}} \arrow[r] \arrow[d] & J^\star_{\mathsf{ND}} \arrow[r] \arrow[d] & \vec{J}^\star_{\mathsf{ND}} \arrow[d, leftrightarrow] \\
J^\star_{\mathsf{SS}} \arrow[r] & J^\star_{\mathsf{NS}} \arrow[r] & \vec{J}^\star_{\mathsf{NS}}
\end{tikzcd}
$$

Agent-state based policies

History based policies

Relationship between different policy classes

$$
\begin{tikzcd}
J^\star_{\mathsf{SD}} \arrow[r] \arrow[d] & J^\star_{\mathsf{ND}} \arrow[r] \arrow[d] & \vec{J}^\star_{\mathsf{ND}} \arrow[d, leftrightarrow] \\
J^\star_{\mathsf{SS}} \arrow[r] & J^\star_{\mathsf{NS}} \arrow[r] & \vec{J}^\star_{\mathsf{NS}}
\end{tikzcd}
$$

When agent state is an information state (e.g., belief state), all policy classes have the same performance.

Salient features of agent state-based policies

$$J^\star_{\mathsf{SD}} < J^\star_{\mathsf{SS}} < J^\star_{\mathsf{ND}}$$

Non-stationary policies can outperform stationary ones

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	. . . . . .

- Observation: Odd or Even
- In red states:
  - Action 0 gives reward 1 and moves to right.
  - Action 1 gives reward -1 and resets state to 1.
- In the non-red states: opposite behavior

Sinha, Geist, Mahajan, “Periodic agent-state based Q-learning for POMDPs”, NeurIPS 2024.

Non-stationary policies can outperform stationary ones

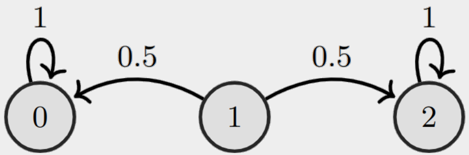
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	$\cdots\cdots$

- Observation: Odd or Even
- In red states:
  - Action 0 gives reward 1 and moves to right.
  - Action 1 gives reward -1 and resets state to 1.
- In the non-red states: opposite behavior

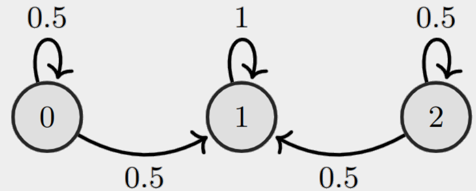
- $\vec{J}^{\star}_{\mathsf{ND}} = \dfrac{1}{1-\gamma}$
- $J^{\star}_{\mathsf{SD}} = \dfrac{1+\gamma-\gamma^2}{1-\gamma^3}$
- For all $\gamma \in (0,1)$, $J^{\star}_{\mathsf{SD}} < \vec{J}^{\star}_{\mathsf{ND}}$

Sinha, Geist, Mahajan, "Periodic agent-state based Q-learning for POMDPs", NeurIPS 2024.

Stochastic policies can outperform deterministic ones



(a) Dynamics under action 0



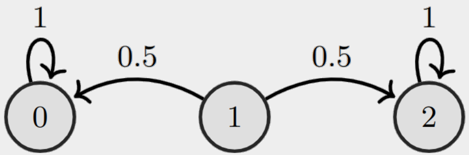
(b) Dynamics under action 1

$$r(\cdot, 0) = [-1, 0, 2]$$

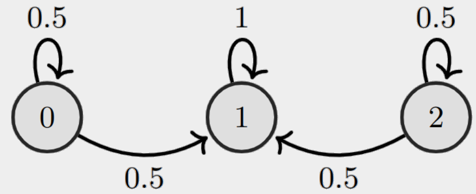
$$r(\cdot, 1) = [-0.5, -0.5, -0.5]$$

Sinha, Geist, Mahajan, “Periodic agent-state based Q-learning for POMDPs”, NeurIPS 2024.

Stochastic policies can outperform deterministic ones

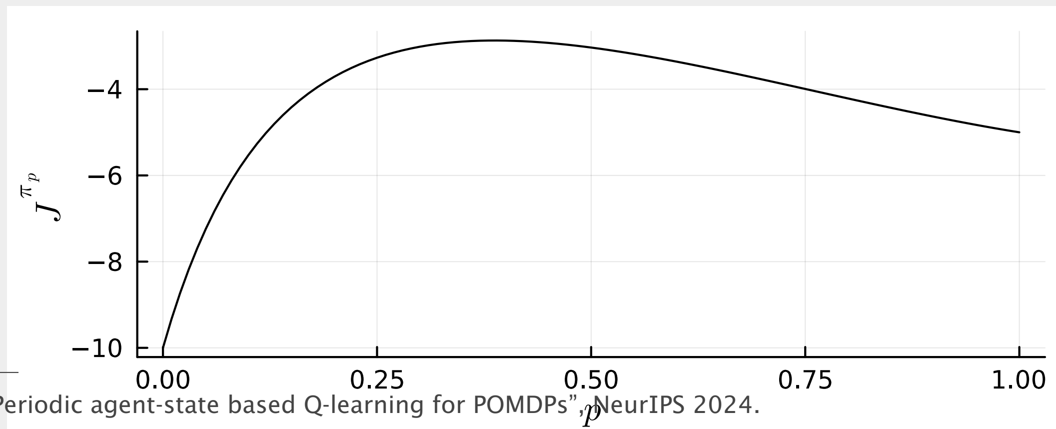


(a) Dynamics under action 0



(b) Dynamics under action 1

$$r(\cdot, 0) = [-1, 0, 2] \qquad r(\cdot, 1) = [-0.5, -0.5, -0.5]$$



Sinha, Geist, Mahajan, “Periodic agent-state based Q-learning for POMDPs”, NeurIPS 2024.

Outline

Background

- Review of MDPs and RL
- Review of POMDPs
- Why is RL for POMDPs difficult?

Agent-state based planning

- Agent state based policies
- Policy classes
- Planning for different policy classes

Agent-state based learning

- Agent state based Q-Learning
- Self-predictive representation learning
- Agent state based actor-critic

$$
\begin{tikzcd}
J^{\star}_{\mathsf{SD}} \arrow[r] \arrow[d] & J^{\star}_{\mathsf{ND}} \arrow[r] \arrow[d] & \vec{J}^{\star}_{\mathsf{ND}} \arrow[d, leftrightarrow] \\
J^{\star}_{\mathsf{SS}} \arrow[r] & J^{\star}_{\mathsf{NS}} \arrow[r] & \vec{J}^{\star}_{\mathsf{NS}}
\end{tikzcd}
$$

How to find optimal non-stationary agent-state based policies?

$$
\begin{tikzcd}
J^{\star}_{\mathsf{SD}} \arrow[r] \arrow[d] & J^{\star}_{\mathsf{ND}} \arrow[r] \arrow[d] & \vec{J}^{\star}_{\mathsf{ND}} \arrow[d, leftrightarrow] \\
J^{\star}_{\mathsf{SS}} \arrow[r] & J^{\star}_{\mathsf{NS}} \arrow[r] & \vec{J}^{\star}_{\mathsf{NS}}
\end{tikzcd}
$$

Finding best agent-state based policies

Key observation: Finding the best agent-state based policy is a decentralized control problem

Finding best agent-state based policies

Key observation: Finding the best agent-state based policy is a decentralized control problem

Why?

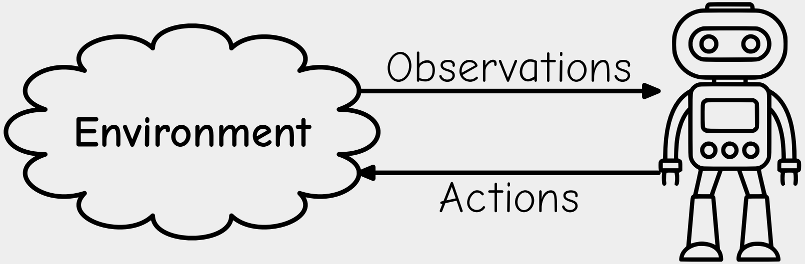
- Consider each "agent at time t" as separate decision maker.
- Let $\mathcal{I}_t$ denote the information sigma-algebra generated by Z_t.
- Information is non-nested: $\mathcal{I}_t \not\subset \mathcal{I}_{t+1}$.
- Thus, the problem is a decentralized control problem.

Finding best agent-state based policies

Key observation: Finding the best agent-state based policy is a decentralized control problem

So, we can use tools from decentralized control to find optimal agent-state based policies!

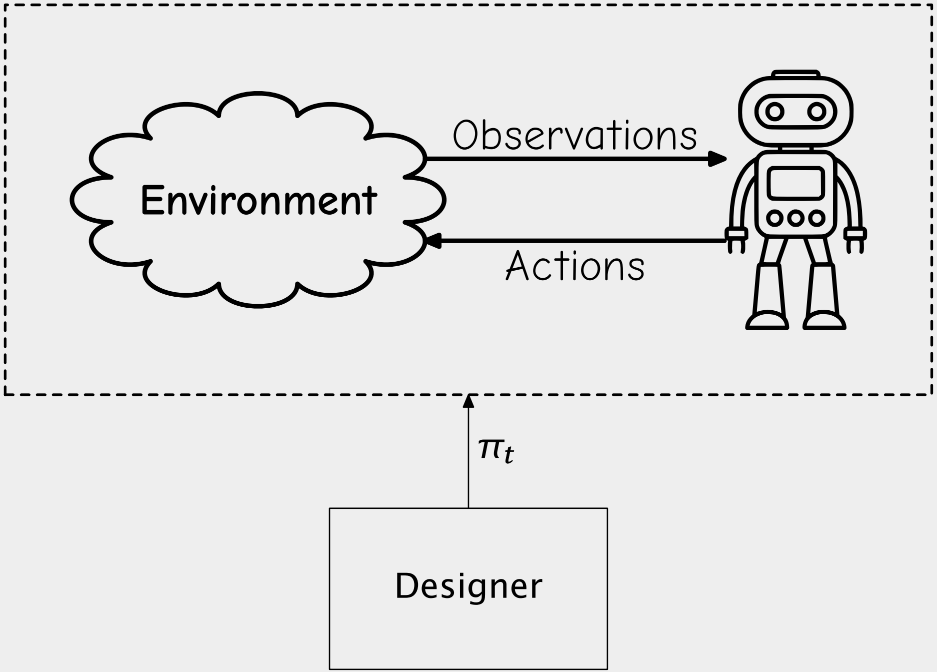
Designer's approach to find optimal policy in Π_{NS}



Witsenhausen, "A standard form for sequential stochastic control," Math. Systems Theory, 1973/

Mahajan, "Sequential decomposition of sequential teams", PhD thesis, 2008.

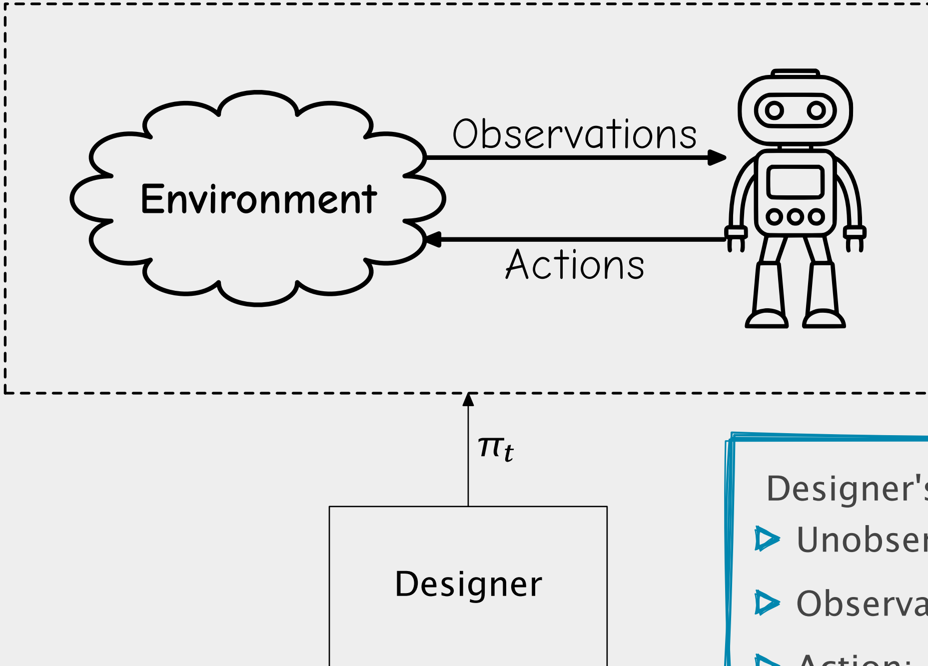
Designer's approach to find optimal policy in Π_{NS}



Witsenhausen, “A standard form for sequential stochastic control,” Math. Systems Theory, 1973/

Mahajan, “Sequential decomposition of sequential teams”, PhD thesis, 2008.

Designer's approach to find optimal policy in Π_{NS}



Designer's problem is a POMDP with:

- Unobserved state: (S_t, Z_t)
- Observations: $\varnothing$
- Action: π_t

Witsenhausen, “A standard form for sequential stochastic control,” Math. Systems Theory, 1973/
Mahajan, “Sequential decomposition of sequential teams”, PhD thesis, 2008.

Designer's approach to find optimal policy in Π_{NS}

Mahajan, "Sequential decomposition of sequential teams", PhD thesis, 2008.

Designer's approach to find optimal policy in Π_{NS}

Joint distribution of env and agent states	For any $\boldsymbol{\pi} \in \Pi_{\mathsf{NS}}$, define $\xi^{\boldsymbol{\pi}}_t(s, z) \coloneqq \mathbb{P}^{\boldsymbol{\pi}}(S_t = s, Z_t = z)$. Then: ▷ $\xi^{\boldsymbol{\pi}}_{t+1} = \phi_{\mathsf{DES}}(\pi_t, \xi^{\boldsymbol{\pi}}_t)$. ▷ $\mathbb{E}^{\boldsymbol{\pi}}[R_t] = r_{\mathsf{DES}}(\pi_t, \xi^{\boldsymbol{\pi}}_t)$.

Mahajan, "Sequential decomposition of sequential teams", PhD thesis, 2008.

Designer's approach to find optimal policy in Π_{NS}

Joint distribution of env and agent states

For any $\boldsymbol{\pi} \in \Pi_{\mathsf{NS}}$, define $\xi_t^{\boldsymbol{\pi}}(s, z) \coloneqq \mathbb{P}^{\boldsymbol{\pi}}(S_t = s, Z_t = z)$. Then:

- $\xi_{t+1}^{\boldsymbol{\pi}} = \phi_{\mathsf{DES}}(\pi_t, \xi_t^{\boldsymbol{\pi}})$.
- $\mathbb{E}^{\boldsymbol{\pi}}[R_t] = r_{\mathsf{DES}}(\pi_t, \xi_t^{\boldsymbol{\pi}})$.

DP using desginer's approach

Consider the following DP:

$$V_{\mathsf{DES}}(\xi) = \max_{\pi \colon \mathcal{Z} \to \Delta(\mathcal{A})} \{ r_{\mathsf{DES}}(\pi, \xi) + \gamma V_{\mathsf{DES}}(\phi_{\mathsf{DES}}(\pi, \xi)) \}.$$

Let $\psi_{\mathsf{DES}}(\xi)$ denote any arg max of the RHS. Let $\xi_1^\star = \xi_1$ and recursively define

$$\pi_t^\star = \psi_{\mathsf{DES}}(\xi_t^\star) \quad \text{and} \quad \xi_{t+1}^\star = \phi_{\mathsf{DES}}(\pi_t^\star, \xi_t^\star).$$

Then, the policy $\boldsymbol{\pi}^\star = (\pi_1^\star, \pi_2^\star, \dots) \in \Pi_{\mathsf{NS}}$ is optimal in Π_{NS}.

Mahajan, “Sequential decomposition of sequential teams”, PhD thesis, 2008.

Some comments

Historical review

- The idea goes back to Witsenhausen's standard form (1973).
- Used for POMDPs in Sandell (1974) and general finite state Dec-POMDPs in Mahajan (2008).
- Related to NO MDP approach of Dibangoye et al (2016).

Some comments

Historical review

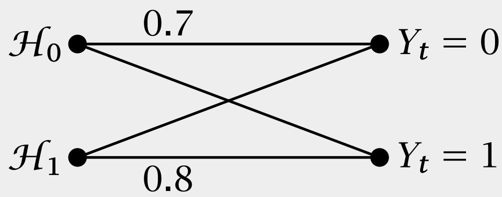
- The idea goes back to Witsenhausen's standard form (1973).
- Used for POMDPs in Sandell (1974) and general finite state Dec-POMDPs in Mahajan (2008).
- Related to NO MDP approach of Dibangoye et al (2016).

Implications

- Provides a DP to find best policy in Π_{ND} and Π_{NS}.
- Using properties of the DP can show that $J^\star_{\mathsf{NS}} = J^\star_{\mathsf{ND}}$.

$$
\begin{tikzcd}
J^\star_{\mathsf{SD}} \arrow[r] \arrow[d] & J^\star_{\mathsf{ND}} \arrow[r] \arrow[d, leftrightarrow] & \vec{J}^\star_{\mathsf{ND}} \arrow[d, leftrightarrow] \\
J^\star_{\mathsf{SS}} \arrow[r] & J^\star_{\mathsf{NS}} \arrow[r] & \vec{J}^\star_{\mathsf{NS}}
\end{tikzcd}
$$

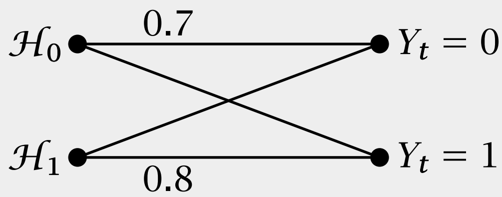
An example: Reactive hypothesis testing



$\mathbb{P}(H = \mathcal{H}_0) = 0.7$

Agent state: $Z_t = Y_t$ (last observation)

An example: Reactive hypothesis testing



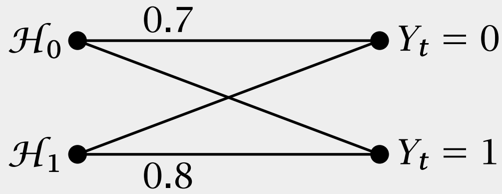
$\mathbb{P}(H = \mathcal{H}_0) = 0.7$

Agent state: $Z_t = Y_t$ (last observation)

Actions

- Stop and declare $\mathcal{H}_0$
- Stop and declare $\mathcal{H}_1$
- Continue and take another measurement

An example: Reactive hypothesis testing



$\mathbb{P}(H = \mathcal{H}_0) = 0.7$

Agent state: $Z_t = Y_t$ (last observation)

Actions

- Stop and declare $\mathcal{H}_0$
- Stop and declare $\mathcal{H}_1$
- Continue and take another measurement

Per-step reward

$$r(\mathcal{H}_0, \mathcal{H}_0) = 1, \qquad r(\mathcal{H}_0, \mathcal{H}_1) = -1,$$

$$r(\mathcal{H}_1, \mathcal{H}_1) = 2, \qquad r(\mathcal{H}_1, \mathcal{H}_0) = -2,$$

$$r(\cdot, c) = -0.01.$$

An example: Reactive hypothesis testing

Designer's state space

Global state: $(\text{Terminated}, \text{Hypothesis}, \text{Obs})$.

Designer's state: $\mathbb{P}\begin{pmatrix} (0,0,0) & (0,0,1) & (1,0,0) & (1,0,1) \\ (0,1,0) & (0,1,1) & (1,1,0) & (1,1,1) \end{pmatrix}$

An example: Reactive hypothesis testing

Designer's state space

Global state: $(\text{Terminated}, \text{Hypothesis}, \text{Obs})$.

Designer's state: $\mathbb{P}\begin{pmatrix} (0,0,0) & (0,0,1) & \boxed{\begin{matrix} (1,0,0) & (1,0,1) \\ (1,1,0) & (1,1,1) \end{matrix}} \\ (0,1,0) & (0,1,1) & \end{pmatrix} \in \Delta^5$

An example: Reactive hypothesis testing

Designer's state space

Global state: $(\text{Terminated}, \text{Hypothesis}, \text{Obs})$.

Designer's state: $\mathbb{P}\begin{pmatrix} (0,0,0) & (0,0,1) & \boxed{\begin{matrix} (1,0,0) & (1,0,1) \\ (1,1,0) & (1,1,1) \end{matrix}} \\ (0,1,0) & (0,1,1) & \end{pmatrix} \in \Delta^5$

Designer's Action space

Designer's action space: $\{0, 1\} \to \{\mathcal{H}_0, \mathcal{H}_1, c\}$. 9 possibilities

An example: Reactive hypothesis testing

Designer's state space

Global state: $(\text{Terminated}, \text{Hypothesis}, \text{Obs})$.

Designer's state: $\mathbb{P}\left(\begin{matrix} (0,0,0) & (0,0,1) \\ (0,1,0) & (0,1,1) \end{matrix} \quad \boxed{\begin{matrix} (1,0,0) & (1,0,1) \\ (1,1,0) & (1,1,1) \end{matrix}} \right) \in \Delta^5$

Designer's Action space

Designer's action space: $\{0,1\} \to \{\mathcal{H}_0, \mathcal{H}_1, c\}$. 9 possibilities

- No need to consider stochastic policies.

$$\begin{tikzcd}
J^\star_{\mathsf{SD}} \arrow[r] \arrow[d] & J^\star_{\mathsf{ND}} \arrow[r] \arrow[d, leftrightarrow] & \vec{J}^\star_{\mathsf{ND}} \arrow[d, leftrightarrow] \\
J^\star_{\mathsf{SS}} \arrow[r] & J^\star_{\mathsf{NS}} \arrow[r] & \vec{J}^\star_{\mathsf{NS}}
\end{tikzcd}$$

An example: Reactive hypothesis testing

Solution of the DP (for $\gamma = 0.95$)

	1	2	3	4	5	6
$Y_t = 0$	$\mathcal{H}_0$	c	$\mathcal{H}_0$	c	$\cdots$	$\cdots$
$Y_t = 1$	c	$\mathcal{H}_1$	c	$\mathcal{H}_1$	$\cdots$	$\cdots$

- $J^\star_{\mathsf{SD}} = 0.8093$
- In this case, optimal solution turns out to be **periodic**! Not a general result.

An example: Reactive hypothesis testing

Solution of the DP (for $\gamma = 0.95$)

	1	2	3	4	5	6
$Y_t = 0$	$\mathcal{H}_0$	c	$\mathcal{H}_0$	c	$\cdots$	$\cdots$
$Y_t = 1$	c	$\mathcal{H}_1$	c	$\mathcal{H}_1$	$\cdots$	$\cdots$

- $J^\star_{\mathsf{SD}} = 0.8093$
- In this case, optimal solution turns out to be **periodic**! Not a general result.

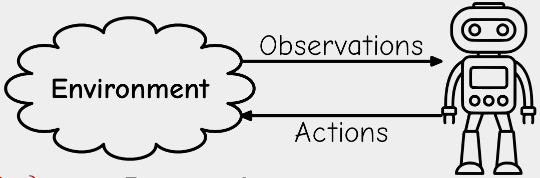
Key Takeaway

- Designer's DP provides optimal agent-state based policy.
- It is solvable for small models ... but still hard to solve for large models.

Are there methods which scale to large models

$$
\begin{tikzcd}
J^{\star}_{\mathsf{SD}} \arrow[r] \arrow[d] & J^{\star}_{\mathsf{ND}} \arrow[r] \arrow[d] & \vec{J}^{\star}_{\mathsf{ND}} \arrow[d, leftrightarrow] \\
J^{\star}_{\mathsf{SS}} \arrow[r] & J^{\star}_{\mathsf{NS}} \arrow[r] & \vec{J}^{\star}_{\mathsf{NS}}
\end{tikzcd}
$$

Policy evaluation for policies in Π_{SS}



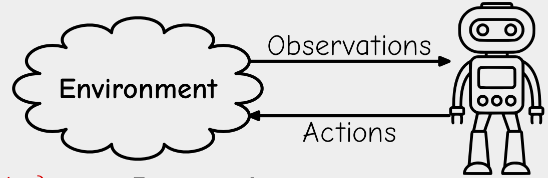
Joint env and agent state process

- $\{(S_t, Z_t)\}_{t\ge1}$ is a **controlled Markov process** controlled by $\{A_t\}_{t\ge1}$. In particuar,

$$P_{\mathsf{PROD}}(s', z' | s, z, a) = \sum_{y' \in \mathcal{Y}} P(s', y' | s, a)\, \mathbb{1}\{z' = \phi(z, y', a)\}$$

- We can use standard formulas to evaluate any policy in Π_{PROD}.

Policy evaluation for policies in Π_{SS}



Joint env and agent state process

- $\{(S_t, Z_t)\}_{t \ge 1}$ is a **controlled Markov process** controlled by $\{A_t\}_{t \ge 1}$. In particuar,

$$P_{\mathsf{PROD}}(s', z' | s, z, a) = \sum_{y' \in \mathcal{Y}} P(s', y' | s, a) \mathbb{1}\{z' = \phi(z, y', a)\}$$

- We can use standard formulas to evaluate any policy in Π_{PROD}.

- Any $\boldsymbol{\pi} \in \Pi_{\mathsf{SS}}$ also belongs to Π_{PROD} (the set of stationary stochastic policies on $\mathcal{S} \times \mathcal{Z}$). Thus:

$$J(\boldsymbol{\pi}) = \sum_{(s,z) \in \mathcal{S} \times \mathcal{Z}} \xi_1(s, z) V_{\mathsf{PROD}}(s, z)$$

where

$$V_{\mathsf{PROD}}(s, z) = \sum_{a \in \mathcal{A}} \pi(a|z) \Big[r(s, a) + \gamma \sum_{s', z' \in \mathcal{S} \times \mathcal{Z}} P_{\mathsf{PROD}}(s', z' | s, z, a) V_{\mathsf{PROD}}(s', z') \Big].$$

Some comments

Historical review

- The idea of policy evaluation on the product space goes back to Platzman (1977) and has been rediscovered multiple times: Littman (1996), Hauskrecht (1997), Cassandra (1998), Hansen (1998),

Some comments

Historical review

- The idea of policy evaluation on the product space goes back to Platzman (1977) and has been rediscovered multiple times: Littman (1996), Hauskrecht (1997), Cassandra (1998), Hansen (1998),

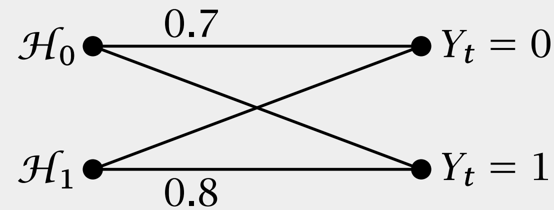
Implications

- Since we can do policy evaluation, we can do policy search!

  ...provided we have access to env state.

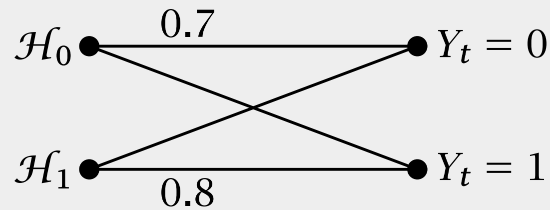
Back to example: Reactive hypothesis testing

Brute force search



- Stochastic policy: $\{0, 1\} \to \Delta(\{\mathcal{H}_0, \mathcal{H}_1, c\})$.
- Characterized by two PMFs: $\pi(\cdot \mid 0)$ and $\pi(\cdot \mid 1)$.
- Discretize each PMF to 50 bins (approximately 1.7×10^6 policies)
- Evaluate performance of each policy by previous formula to find the best policy.

Back to example: Reactive hypothesis testing



Brute force search

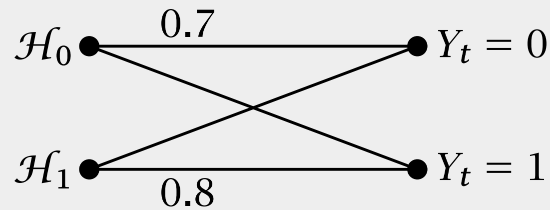
- Stochastic policy: $\{0, 1\} \to \Delta(\{\mathcal{H}_0, \mathcal{H}_1, c\})$.
- Characterized by two PMFs: $\pi(\cdot \mid 0)$ and $\pi(\cdot \mid 1)$.
- Discretize each PMF to 50 bins (approximately 1.7×10^6 policies)
- Evaluate performance of each policy by previous formula to find the best policy.

Best (quantized) stochastic policy

- $\pi(\cdot \mid 0) = [1, 0, 0]$ and $\pi(\cdot \mid 1) = [0, 0.72, 0.28]$
- $J^{\star}_{\mathsf{SS}} = 0.6532$ (21% worse than the best non-stationary policy)

Back to example: Reactive hypothesis testing

Brute force search



- Stochastic policy: $\{0, 1\} \to \Delta(\{\mathcal{H}_0, \mathcal{H}_1, c\})$.
- Characterized by two PMFs: $\pi(\cdot \mid 0)$ and $\pi(\cdot \mid 1)$.
- Discretize each PMF to 50 bins (approximately 1.7×10^6 policies)
- Evaluate performance of each policy by previous formula to find the best policy.

Best (quantized) stochastic policy

- $\pi(\cdot \mid 0) = [1, 0, 0]$ and $\pi(\cdot \mid 1) = [0, 0.72, 0.28]$
- $J^\star_{\mathsf{SS}} = 0.6532$ (21% worse than the best non-stationary policy)

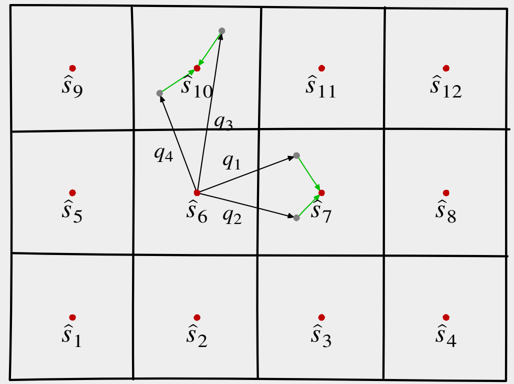
Main takeaway

- Possible to search for stationary stochastic policies

Another idea to search for stationary policies

State discretization (for cts state MDPs)

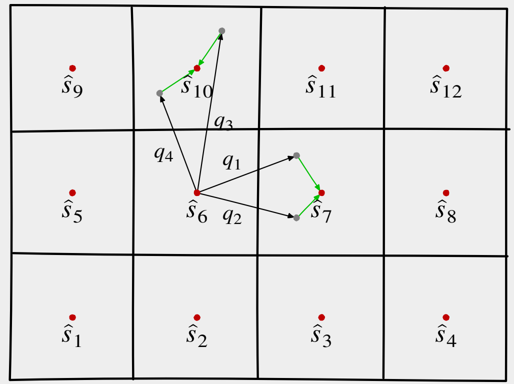
- Quantize the state space into disjoint cells
- Associate a grid point with each cell.
- Construct a model $(\hat{r}, \hat{P})$ for a discrete MDP with grid cells.
- Compute policy $\hat{\pi}$ for the discrete MDP



Another idea to search for stationary policies

State discretization (for cts state MDPs)

- Quantize the state space into disjoint cells
- Associate a grid point with each cell.
- Construct a model $(\hat{r}, \hat{P})$ for a discrete MDP with grid cells.
- Compute policy $\hat{\pi}$ for the discrete MDP



Observations

- The discretized model is a POMDP. But we treat it as an MDP!
- **Why does this work?** For fine discretization:
  - The constructed discrete MDP model is close enough to the discrete POMDP.
  - The discrete POMDP model is close to the cts MDP model.

Use the same idea for finding good policies in Π_{SD}

Intuition

- Any made-up model $(P_{\mathsf{AIS}}, r_{\mathsf{AIS}})$ where $P_{\mathsf{AIS}}\colon \mathcal{Z} \times \mathcal{A} \to \Delta(\mathcal{Z})$ and $r_{\mathsf{AIS}}\colon \mathcal{Z} \times \mathcal{A} \to \mathbb{R}$ gives rise to an feasible policy $\pi_{\mathsf{AIS}} \in \Pi_{\mathsf{SD}}$.
- If the model $(P_{\mathsf{AIS}}, r_{\mathsf{AIS}})$ is close to the “true” model, then policy π_{AIS} is approx. optimal.

Use the same idea for finding good policies in Π_{SD}

Intuition

- Any made-up model $(P_{\text{AIS}}, r_{\text{AIS}})$ where $P_{\text{AIS}}\colon \mathcal{Z} \times \mathcal{A} \to \Delta(\mathcal{Z})$ and $r_{\text{AIS}}\colon \mathcal{Z} \times \mathcal{A} \to \mathbb{R}$ gives rise to an feasible policy $\pi_{\text{AIS}} \in \Pi_{\text{SD}}$.
- If the model $(P_{\text{AIS}}, r_{\text{AIS}})$ is close to the "true" model, then policy π_{AIS} is approx. optimal.

How to make this precise?

- Need to measure "closeness" of models.
- Depends on the type of approximation guarantees we want (absolute error vs relative error)
- Large literature on approximation of MDPs. Need to extend it to POMDPs.

Quantifying model approximation

Subramanian, Sinha, Seraj, and Mahajan, “Approximate information state for . . . partially observed systems”, JMLR 2022.

Quantifying model approximation

Approximate info state

Agent state $\{Z_t\}_{t \ge 1}$ and a model $(P_{\mathsf{AIS}}, r_{\mathsf{AIS}})$ is said to be an $(\vec{\boldsymbol{\varepsilon}}, \vec{\boldsymbol{\delta}})$ **approximate information state (AIS)** if it is

(AP1) Approximately sufficient for performance evaluation

$$\left| \mathbb{E}[R_t \mid H_t, A_t] - r_{\mathsf{AIS}}(\vec{\sigma}_t(H_t), A_t) \right| \le \varepsilon_t$$

(AP2) Approximately sufficient for predicting itself

$$d_{\mathfrak{F}}(\mathbb{P}(Z_{t+1} = \cdot \mid H_t, A_t), P_{\mathsf{AIS}}(Z_{t+1} = \cdot \mid \vec{\sigma}_t(H_t), A_t)) \le \delta_t$$

Subramanian, Sinha, Seraj, and Mahajan, "Approximate information state for ... partially observed systems", JMLR 2022.

Quantifying model approximation

Approximate info state

Agent state $\{Z_t\}_{t \ge 1}$ and a model $(P_{\text{AIS}}, r_{\text{AIS}})$ is said to be an $(\vec{\boldsymbol{\varepsilon}}, \vec{\boldsymbol{\delta}})$ **approximate information state (AIS)** if it is

(AP1) Approximately sufficient for performance evaluation

$$\left| \mathbb{E}[R_t \mid H_t, A_t] - r_{\text{AIS}}(\vec{\sigma}_t(H_t), A_t) \right| \le \varepsilon_t$$

(AP2) Approximately sufficient for predicting itself

$$d_{\mathfrak{F}}(\mathbb{P}(Z_{t+1} = \cdot \mid H_t, A_t), P_{\text{AIS}}(Z_{t+1} = \cdot \mid \vec{\sigma}_t(H_t), A_t)) \le \delta_t$$

AIS based approx DP

Let π_{AIS} be the optimal policy for model $(P_{\text{AIS}}, r_{\text{AIS}})$. Define $\vec{\boldsymbol{\pi}}_{\text{AIS}} = (\vec{\pi}_{\text{AIS},1}, \vec{\pi}_{\text{AIS},2}, \dots)$ where

$$\vec{\boldsymbol{\pi}}_{\text{AIS},t}(h_t) = \pi_{\text{AIS}}(\vec{\sigma}_t(h_t))$$

Then, $\vec{J}^{\star}_{\text{ND}} - J(\vec{\boldsymbol{\pi}}_{\text{AIS}}) \le \dfrac{2}{1-\gamma}\left[\varepsilon + \gamma \delta \rho_{\mathfrak{F}}(V^{\star}_{\text{AIS}})\right]$

where $\varepsilon = (1-\gamma)\sum_{t=1}^{\infty} \gamma^{t-1}\varepsilon_t$, and $\delta = (1-\gamma)\sum_{t=1}^{\infty} \gamma^{t-1}\delta_t$.

Some remarks on AIS

- Two ways to interpret the results:
  - Given the information state space $\mathcal{Z}$, find the best compression $\sigma_t \colon \mathcal{H}_t \to \mathcal{Z}$
  - Given any compression function $\sigma_t \colon \mathcal{H}_t \to \mathcal{Z}$, find the approximation error.

Some remarks on AIS

- Two ways to interpret the results:
  - Given the information state space $\mathcal{Z}$, find the best compression $\sigma_t \colon \mathcal{H}_t \to \mathcal{Z}$
  - Given any compression function $\sigma_t \colon \mathcal{H}_t \to \mathcal{Z}$, find the approximation error.
- **Key obs:** the second interpretation allows us to develop AIS-based RL algorithms

Some remarks on AIS

- Two ways to interpret the results:
  - Given the information state space $\mathcal{Z}$, find the best compression $\sigma_t \colon \mathcal{H}_t \to \mathcal{Z}$
  - Given any compression function $\sigma_t \colon \mathcal{H}_t \to \mathcal{Z}$, find the approximation error.
- **Key obs:** the second interpretation allows us to develop AIS-based RL algorithms

- Results depend on the choice of metric on probability spaces.
- The bounds use what are known as integral probability metrics (IPM), which include many commonly used metrics:
  - Total variation
  - Wasserstein distance
  - Maximum mean discrepancy (MMD)

Example 1: Robustness to model mismatch in MDPs

Real-world model (P, r)

Simulation model $(\hat{P}, \hat{r})$

What is the loss in performance if we choose a policy using the simulation model and use it in the real world?

Example 1: Robustness to model mismatch in MDPs

Real-world model (P, r)

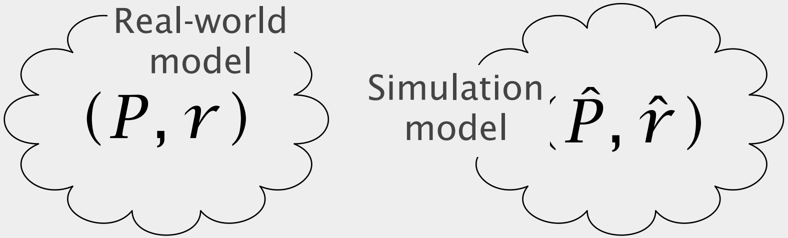
Simulation model $(\hat{P}, \hat{r})$

What is the loss in performance if we choose a policy using the simulation model and use it in the real world?

Model mismatch as an AIS

- $(\text{Identity}, \hat{P}, \hat{r})$ is an (ε, δ)-AIS with $\varepsilon = \sup_{s,a} |r(s,a) - \hat{r}(s,a)|$ and $\delta_{\mathfrak{F}} = \sup_{s,a} d_{\mathfrak{F}}(P(\cdot|s,a), \hat{P}(\cdot|s,a))$.

Example 1: Robustness to model mismatch in MDPs



Müller, "How does the value function of a Markov decision process depend on the transition probabilities?" MOR 1997.

Model mismatch as an AIS

- (Identity, $\hat{P}, \hat{r}$) is an (ε, δ)-AIS with $\varepsilon = \sup_{s,a} |r(s,a) - \hat{r}(s,a)|$ and $\delta_{\mathfrak{F}} = \sup_{s,a} d_{\mathfrak{F}}(P(\cdot|s,a), \hat{P}(\cdot|s,a))$.

$d_{\mathfrak{F}}$ is total variation

$$V(s) - V^{\pi}(s) \le \frac{2\varepsilon}{1-\gamma} + \frac{\gamma\delta\,\mathrm{span}(r)}{(1-\gamma)^2}$$

Recover bounds of Müller (1997).

Example 1: Robustness to model mismatch in MDPs

Real-world model (P, r)

Simulation model $(\hat{P}, \hat{r})$

- Müller, "How does the value function of a Markov decision process depend on the transition probabilities?" MOR 1997.
- Asadi, Misra, Littman, "Lipscitz continuity in model-based reinfocement learning," ICML 2018.

Model mismatch as an AIS

- (Identity, $\hat{P}, \hat{r}$) is an (ε, δ)-AIS with $\varepsilon = \sup_{s,a} |r(s,a) - \hat{r}(s,a)|$ and $\delta_{\mathfrak{F}} = \sup_{s,a} d_{\mathfrak{F}}(P(\cdot|s,a), \hat{P}(\cdot|s,a))$.

$d_{\mathfrak{F}}$ is total variation

$$V(s) - V^{\pi}(s) \le \frac{2\varepsilon}{1-\gamma} + \frac{\gamma\delta \operatorname{span}(r)}{(1-\gamma)^2}$$

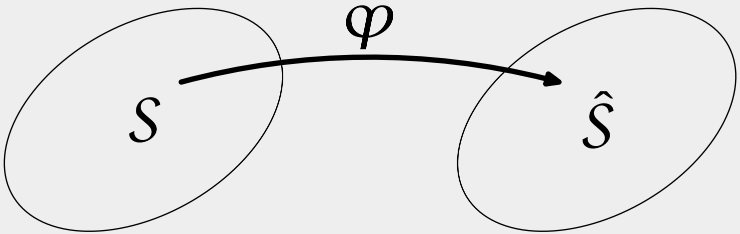
Recover bounds of Müller (1997).

$d_{\mathfrak{F}}$ is Wasserstein distance

$$V(s) - V^{\pi}(s) \le \frac{2\varepsilon}{1-\gamma} + \frac{2\gamma\delta L_r}{(1-\gamma)(1-\gamma L_p)}$$

Recover bounds of Asadi, Misra, Littman (2018).

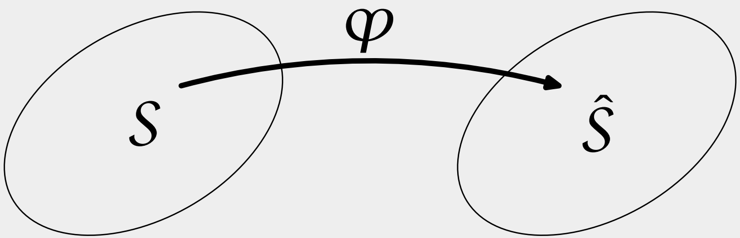
Example 2: Feature abstraction in MDPs



$(\hat{P}, \hat{r})$ is determined from (P, r) using φ

What is the loss in performance if we choose a policy using the abstract model and use it in the original model?

Example 2: Feature abstraction in MDPs



What is the loss in performance if we choose a policy using the abstract model and use it in the original model?

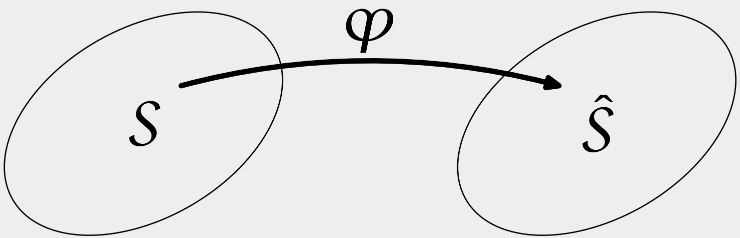
$(\hat{P}, \hat{r})$ is determined from (P, r) using φ

Feature abstraction as AIS

- $(\varphi, \hat{P}, \hat{r})$ is an (ε, δ)-AIS with $\varepsilon = \sup_{s,a} |r(s,a) - \hat{r}(\varphi(s), a)|$

  and $\delta_{\mathfrak{F}} = \sup_{s,a} d_{\mathfrak{F}}(P(\varphi^{-1}(\cdot) \,|\, s, a), \hat{P}(\cdot \,|\, \varphi(s), a)$.

Example 2: Feature abstraction in MDPs



Abel, Hershkowitz, Littman, "Near optimal behavior via approximate state abstraction," ICML 2016.

$(\hat{P}, \hat{r})$ is determined from (P, r) using φ

Feature abstraction as AIS

- $(\varphi, \hat{P}, \hat{r})$ is an (ε, δ)-AIS with $\varepsilon = \sup_{s,a} |r(s,a) - \hat{r}(\varphi(s), a)|$

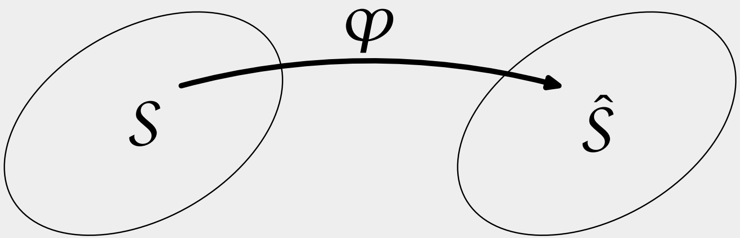
  and $\delta_{\mathfrak{F}} = \sup_{s,a} d_{\mathfrak{F}}(P(\varphi^{-1}(\cdot)\,|s,a), \hat{P}(\cdot\,|\varphi(s), a)$.

$d_{\mathfrak{F}}$ is total variation

$$V(s) - V^{\pi}(s) \le \frac{2\varepsilon}{1-\gamma} + \frac{\gamma \delta_{\mathfrak{F}}\, \mathrm{span}(r)}{(1-\gamma)^2}$$

Improve bounds of Abel et al. (2016)

Example 2: Feature abstraction in MDPs



Abel, Hershkowitz, Littman, "Near optimal behavior via approximate state abstraction," ICML 2016.

Gelada, Kumar, Buckman, Nachum, Bellemare, "DeepMDP: Learning continuous latent space models for representation learning," ICML 2019.

$(\hat{P}, \hat{r})$ is determined from (P, r) using φ

Feature abstraction as AIS

- $(\varphi, \hat{P}, \hat{r})$ is an (ε, δ)-AIS with $\varepsilon = \sup_{s,a} |r(s,a) - \hat{r}(\varphi(s), a)|$

  and $\delta_{\mathfrak{F}} = \sup_{s,a} d_{\mathfrak{F}}(P(\varphi^{-1}(\cdot)\,|s,a), \hat{P}(\cdot\,|\varphi(s), a)$.

$d_{\mathfrak{F}}$ is total variation

$$V(s) - V^{\pi}(s) \le \frac{2\varepsilon}{1-\gamma} + \frac{\gamma \delta_{\mathfrak{F}} \operatorname{span}(r)}{(1-\gamma)^2}$$

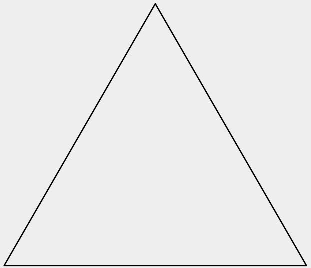
Improve bounds of Abel et al. (2016)

$d_{\mathfrak{F}}$ is Wasserstein distance

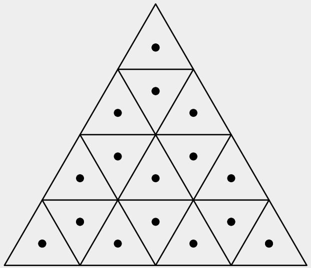
$$V(s) - V^{\pi}(s) \le \frac{2\varepsilon}{1-\gamma} + \frac{2\gamma \delta_{\mathfrak{F}} \|\hat{V}\|_{\text{Lip}}}{(1-\gamma)^2}$$

Recover bounds of Gelada et al. (2019).

Example 3: Belief approximation in POMDPs



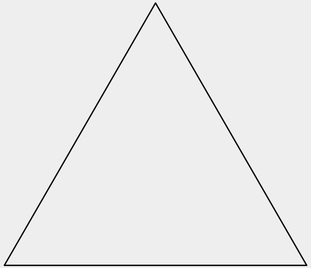
Belief space



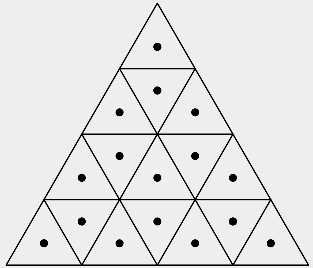
Quantized beliefs

What is the loss in performance if we choose a policy using the approximate beliefs and use it in the original model?

Example 3: Belief approximation in POMDPs



Belief space



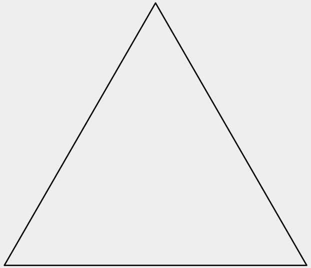
Quantized beliefs

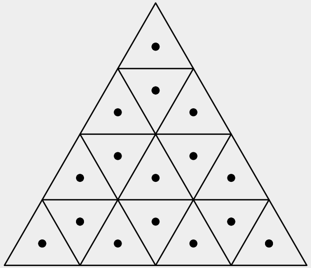
What is the loss in performance if we choose a policy using the approximate beliefs and use it in the original model?

Belief approximation in POMDPs

- Quantized cells of radius ε (in terms of total variation) are $(\varepsilon\|r\|_\infty, 3\varepsilon)$-AIS.

Example 3: Belief approximation in POMDPs





Francois-Lavet, Rabusseau, Pineau, Ernst, Fonteneau, "On overfitting and asymptotic bias in batch reinforcement learning with partial observability," JAIR 2019.

Belief approximation in POMDPs

- Quantized cells of radius ε (in terms of total variation) are $(\varepsilon \| r \|_\infty, 3\varepsilon)$-AIS.

$$V(s) - V^\pi(s) \le \frac{2\varepsilon \| r \|_\infty}{1-\gamma} + \frac{6\gamma\varepsilon \| r \|_\infty}{(1-\gamma)^2}$$

Improve bounds of Francois Lavet et al. (2019) by a factor of $1/(1-\gamma)$.

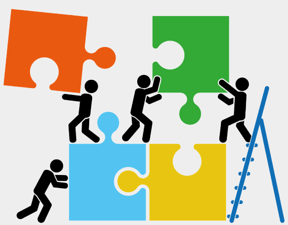
Outline

Background

- Review of MDPs and RL
- Review of POMDPs
- Why is RL for POMDPs difficult?

Agent-state based planning

- Agent state based policies
- Policy classes
- Planning for different policy classes



Agent-state based learning

- Agent state based Q-Learning
- Self-predictive representation learning
- Agent state based actor-critic

Agent-state based Q-learning (ASQL)

$$Q_{t+1}(z,a) = Q_t(z,a) + \alpha_t(z,a)\Big[R_t + \gamma \max_{a' \in \mathcal{A}} Q_t(Z_{t+1}, a') - Q_t(z,a)\Big]$$

Agent-state based Q-learning (ASQL)

$$Q_{t+1}(z,a) = Q_t(z,a) + \alpha_t(z,a)\Big[R_t + \gamma \max_{a' \in \mathcal{A}} Q_t(Z_{t+1}, a') - Q_t(z,a)\Big]$$

Key Questions

- Does this converge?
- To what?

Agent-state based Q-learning (ASQL)

$$Q_{t+1}(z,a) = Q_t(z,a) + \alpha_t(z,a)\left[R_t + \gamma \max_{a' \in \mathcal{A}} Q_t(Z_{t+1}, a') - Q_t(z,a)\right]$$

Key Questions

- Does this converge?
- To what?

Challenges

- $\{Z_t\}_{t \ge 1}$ is not a controlled Markov process.
- No DP to find optimal policy in Π_{SD}

Agent-state based Q-learning (ASQL)

$$Q_{t+1}(z,a) = Q_t(z,a) + \alpha_t(z,a)\Big[R_t + \gamma \max_{a' \in \mathcal{A}} Q_t(Z_{t+1}, a') - Q_t(z,a)\Big]$$

Key Questions

- Does this converge?
- To what?

Challenges

- $\{Z_t\}_{t \ge 1}$ is not a controlled Markov process.
- No DP to find optimal policy in Π_{SD}

Main result: Converges, under mild conditions, but not to optimal.
Characterize degree of sub-optimality and use it to improve algorithm.

Characterization of convergence

Sinha, Geist, Mahajan, “Periodic agent-state based Q-learning for POMDPs”, Neurips 2024.

Characterization of convergence

Assumptions

(A1) The behavior policy μ such that the MC $\{(S_t, Y_t, Z_t, A_t)\}_{t \ge 1}$ is irreducible and aperiodic with stationary distribution ζ^μ.

Moreover, each (z, a) is visited infinitely often.

(A2) The learning rate satisfies: $\alpha_t(z, a) = 0$ when $(z, a) \neq (Z_t, A_t)$ and for all (z, a):

$$\sum_{t \ge 1} \alpha_t(z, a) = \infty \quad \text{and} \quad \sum_{t \ge 1} \alpha_t^2(z, a) < \infty$$

Sinha, Geist, Mahajan, "Periodic agent-state based Q-learning for POMDPs", Neurips 2024.

Characterization of convergence

Assumptions

(A1) The behavior policy μ such that the MC $\{(S_t, Y_t, Z_t, A_t)\}_{t \geq 1}$ is irreducible and aperiodic with stationary distribution ζ^μ.

Moreover, each (z, a) is visited infinitely often.

(A2) The learning rate satisfies: $\alpha_t(z, a) = 0$ when $(z, a) \neq (Z_t, A_t)$ and for all (z, a):

$$\sum_{t \geq 1} \alpha_t(z, a) = \infty \quad \text{and} \quad \sum_{t \geq 1} \alpha_t^2(z, a) < \infty$$

Convergence guarantee

Under (A1) and (A2), **ASQL converges almost surely to Q^μ_{ASQL}** where Q^μ_{ASQL} is the Q-function for the model $(P^\mu_{\text{ASQL}}, r^\mu_{\text{ASQL}})$ given by

$$P^\mu_{\text{ASQL}}(z' | z, a) = \sum_{s', y' \in \mathcal{S} \times \mathcal{Y}} \mathbb{1}\{z' = \phi(z, y', a)\} P(y' | s, a) \zeta^\mu(s | z, a)$$

$$r^\mu_{\text{ASQL}}(z, a) = \sum_{s \in \mathcal{S}} \zeta^\mu(s | z, a) r(s, a)$$

Sinha, Geist, Mahajan, "Periodic agent-state based Q-learning for POMDPs", Neurips 2024.

But how good is the converged policy?

Salient features

- $\pi^{\mu}_{\mathsf{ASQL}} \in \Pi_{\mathsf{SD}}$. So, doesn't converge to best agent-state based policy since $J^{\star}_{\mathsf{SD}} \le \vec{J}^{\star}_{\mathsf{ND}}$.
- May not even converge to the optimal within Π_{SD}.
- In fact, the converged policy $\pi^{\mu}_{\mathsf{ASQL}}$ depends on the exploration policy!

But how good is the converged policy?

Salient features

- $\pi^{\mu}_{\mathsf{ASQL}} \in \Pi_{\mathsf{SD}}$. So, doesn't converge to best agent-state based policy since $J^{\star}_{\mathsf{SD}} \le \vec{J}^{\star}_{\mathsf{ND}}$.
- May not even converge to the optimal within Π_{SD}.
- In fact, the converged policy $\pi^{\mu}_{\mathsf{ASQL}}$ depends on the exploration policy!

Convergence guarantees

- $\pi^{\mu}_{\mathsf{ASQL}}$ is optimal policy of model $(P^{\mu}_{\mathsf{ASQL}}, r^{\mu}_{\mathsf{ASQL}})$.
- So, we can use AIS approximation bounds to get sub-optimality bounds.
- But give bounds between $\vec{J}^{\star}_{\mathsf{ND}} - J(\vec{\pi}^{\mu}_{\mathsf{ASQL}})$ rather than $J^{\star}_{\mathsf{SD}} - J(\vec{\pi}^{\mu}_{\mathsf{ASQL}})$.

Can we do better?

Q-learning will always learn policies in Π_{SD}. But that is the worst policy class!

$$\begin{tikzcd}
J^\star_{\mathsf{SD}} \arrow[r] \arrow[d] & J^\star_{\mathsf{ND}} \arrow[r] \arrow[d, leftrightarrow] & \vec{J}^\star_{\mathsf{ND}} \arrow[d, leftrightarrow] \\
J^\star_{\mathsf{SS}} \arrow[r] & J^\star_{\mathsf{NS}} \arrow[r] & \vec{J}^\star_{\mathsf{NS}}
\end{tikzcd}$$

Can we do better?

Q-learning will always learn policies in Π_{SD}. But that is the worst policy class!

$$
\begin{tikzcd}
J^{\star}_{\mathsf{SD}} \arrow[r] \arrow[d] & J^{\star}_{\mathsf{ND}} \arrow[r] \arrow[d, leftrightarrow] & \vec{J}^{\star}_{\mathsf{ND}} \arrow[d, leftrightarrow] \\
J^{\star}_{\mathsf{SS}} \arrow[r] & J^{\star}_{\mathsf{NS}} \arrow[r] & \vec{J}^{\star}_{\mathsf{NS}}
\end{tikzcd}
$$

Periodic policies

$$\pi = (\pi^{(1)}, \pi^{(2)}, \dots, \pi^{(L)}, \pi^{(1)}, \pi^{(2)}, \dots, \pi^{(L)}, \dots)$$

Periodic policies are a class of finitely parameterized non-stationary policies.

Periodic ASQL

$$Q^{\ell}_{t+1}(z,a) = Q^{\ell}_{t}(z,a) + \alpha^{\ell}_{t}(z,a)\Big[R_t + \gamma \max_{a' \in \mathcal{A}} Q^{[\![\ell+1]\!]}_{t}(Z_{t+1},a') - Q^{\ell}_{t}(z,a)\Big]$$

Sinha, Geist, Mahajan, "Periodic agent-state based Q-learning for POMDPs", Neurips 2024.

Periodic ASQL

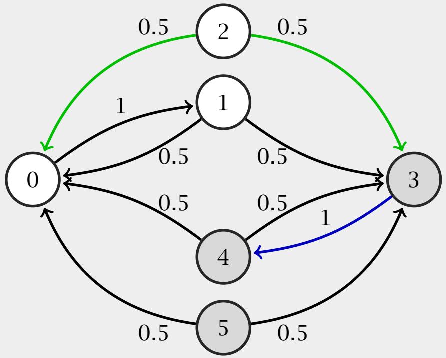
$$Q^{\ell}_{t+1}(z,a) = Q^{\ell}_{t}(z,a) + \alpha^{\ell}_{t}(z,a)\Big[R_t + \gamma \max_{a' \in \mathcal{A}} Q^{[\![\ell+1]\!]}_{t}(Z_{t+1},a') - Q^{\ell}_{t}(z,a)\Big]$$

Similar guarantees as before

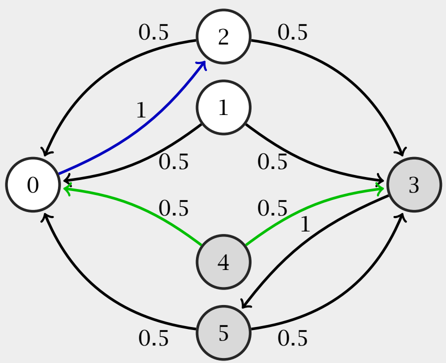
- Periodic ASQL converges almost surely to the solution of a periodic MDP.
- The converged periodic policy depends on the exploration policy.
- We can use AIS approxiation bounds to get sub-optimality bounds for the converged policy.

Sinha, Geist, Mahajan, "Periodic agent-state based Q-learning for POMDPs", Neurips 2024.

PAQSL may outperform ASQL

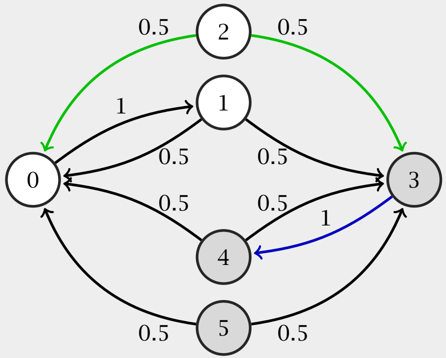


(a) Action 0

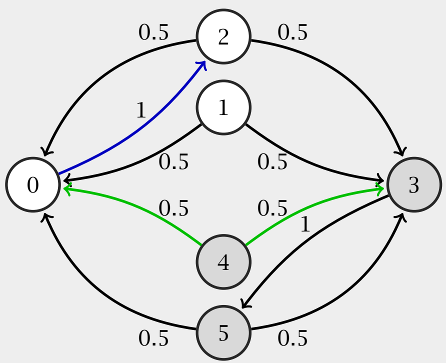


(b) Action 1

PAQSL may outperform ASQL



(a) Action 0

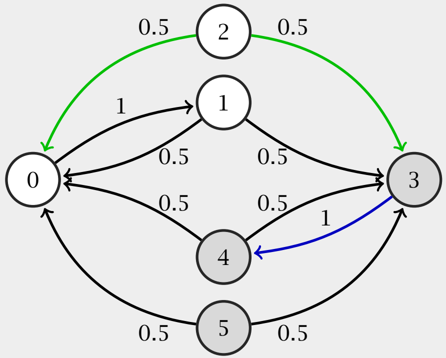


Search over stationary policies

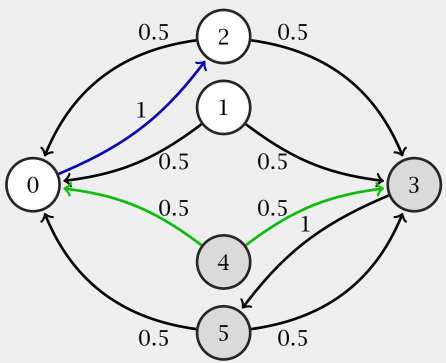
Consider three exploration policies

- $\mu_1 = [0.2; 0.8]$
- $\mu_2 = [0.5; 0.5]$
- $\mu = [0.8; 0.2]$

PAQSL may outperform ASQL



(a) Action 0

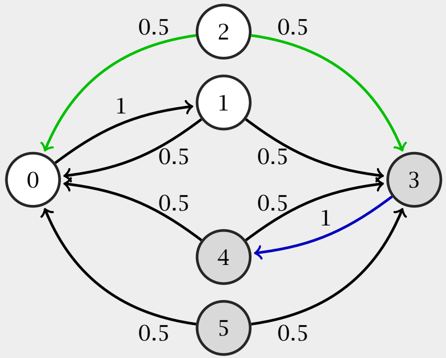


Search over stationary policies

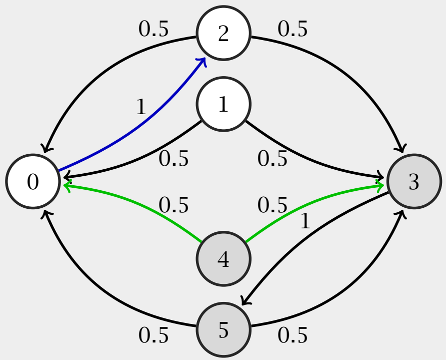
Consider three exploration policies

- $\mu_1 = [0.2; 0.8] \quad J^{\pi_{\mu_1}} = 0.0$
- $\mu_2 = [0.5; 0.5] \quad J^{\pi_{\mu_2}} = 1.064$
- $\mu = [0.8; 0.2] \quad J^{\pi_{\mu_3}} = 2.633$

PAQSL may outperform ASQL



(a) Action 0



Search over stationary policies

Consider three exploration policies

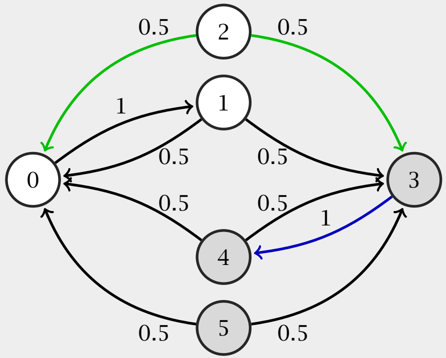
- $\mu_1 = [0.2; 0.8] \quad J^{\pi_{\mu_1}} = 0.0$
- $\mu_2 = [0.5; 0.5] \quad J^{\pi_{\mu_2}} = 1.064$
- $\mu = [0.8; 0.2] \quad J^{\pi_{\mu_3}} = 2.633$

Search over period $L = 2$ policies

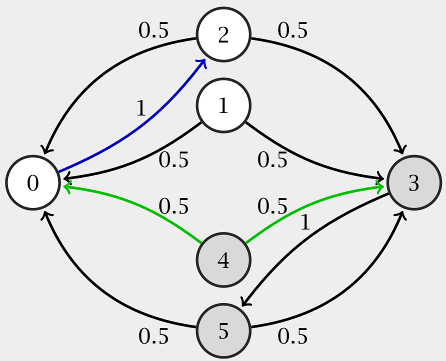
Consider three exploration policies

- $\mu_1 = [0.2, 0.8; 0.8, 0.2]$
- $\mu_2 = [0.5, 0.5; 0.5, 0.5]$
- $\mu_3 = [0.8, 0.2; 0.2, 0.8]$

PAQSL may outperform ASQL



(a) Action 0



Search over stationary policies

Consider three exploration policies

- $\mu_1 = [0.2; 0.8] \quad J^{\pi_{\mu_1}} = 0.0$
- $\mu_2 = [0.5; 0.5] \quad J^{\pi_{\mu_2}} = 1.064$
- $\mu = [0.8; 0.2] \quad J^{\pi_{\mu_3}} = 2.633$

Search over period $L = 2$ policies

Consider three exploration policies

- $\mu_1 = [0.2, 0.8; 0.8, 0.2] \quad J^{\pi_{\mu_1}} = 6.793$
- $\mu_2 = [0.5, 0.5; 0.5, 0.5] \quad J^{\pi_{\mu_2}} = 1.064$
- $\mu_3 = [0.8, 0.2; 0.2, 0.8] \quad J^{\pi_{\mu_3}} = 0.532$

Agent-state based actor-critic (ASAC)

Faster timescale:

$$Q^{\pi}_{t+1}(z,a) = Q^{\pi}_{t}(z,a) + \alpha_t(z,a)\Big[R_t + \gamma Q^{\pi}_{t}(Z_{t+1}, A_{t+1}) - Q_t(z,a)\Big]$$

Slower timescale: Use policy gradient to update π

Subramanian, Sinha, Seraj, and Mahajan, "Approximate information state for ... partially observed systems", JMLR 2022.

Agent-state based actor-critic (ASAC)

Faster timescale:

$$Q^{\pi}_{t+1}(z,a) = Q^{\pi}_{t}(z,a) + \alpha_t(z,a)\Big[R_t + \gamma Q^{\pi}_{t}(Z_{t+1},A_{t+1}) - Q_t(z,a)\Big]$$

Slower timescale: Use policy gradient to update π

Some comments

- Similar to ASQL, can show that $\{Q^{\pi}_t\}_{t\ge 1}$ converges to some Q^{π}_{ASAC} almost surely.
- Different ways to compute the policy gradient. Either converges to something related to Q^{π}_{ASAC} or leads to biased gradients. Difficult to characterize convergence.

Subramanian, Sinha, Seraj, and Mahajan, "Approximate information state for ... partially observed systems", JMLR 2022.

All this theory is good, but what does it mean in practice?

Adding representation learning losses help

ASQL

$$Q_{t+1}(z,a) = Q_t(z,a) + \alpha_t(z,a)\Big[R_t + \gamma \max_{a' \in \mathcal{A}} Q_t(Z_{t+1}, a') - Q_t(z,a)\Big]$$

Sub-optimality bound: $\vec{J}^{\star}_{\mathsf{ND}} - J(\vec{\boldsymbol{\pi}}^{\mu}_{\mathsf{ASQL}}) \le$ function(ε, δ) where

$$\varepsilon_t = \sup_{h_t, a_t} \big| \mathbb{E}[R_t | h_t, a_t] - r^{\mu}_{\mathsf{ASQL}}(\vec{\sigma}_t(h_t), a_t) \big|$$

$$\delta_t = \sup_{h_t, a_t} d_{\mathfrak{F}}\big(\mathbb{P}(Z_{t+1} | h_t, a_t), P^{\mu}_{\mathsf{ASQL}}(Z_{t+1} | \vec{\sigma}_t(h_t), a_t)\big)$$

Adding representation learning losses help

ASQL

$$Q_{t+1}(z,a) = Q_t(z,a) + \alpha_t(z,a)\Big[R_t + \gamma \max_{a' \in \mathcal{A}} Q_t(Z_{t+1}, a') - Q_t(z,a)\Big]$$

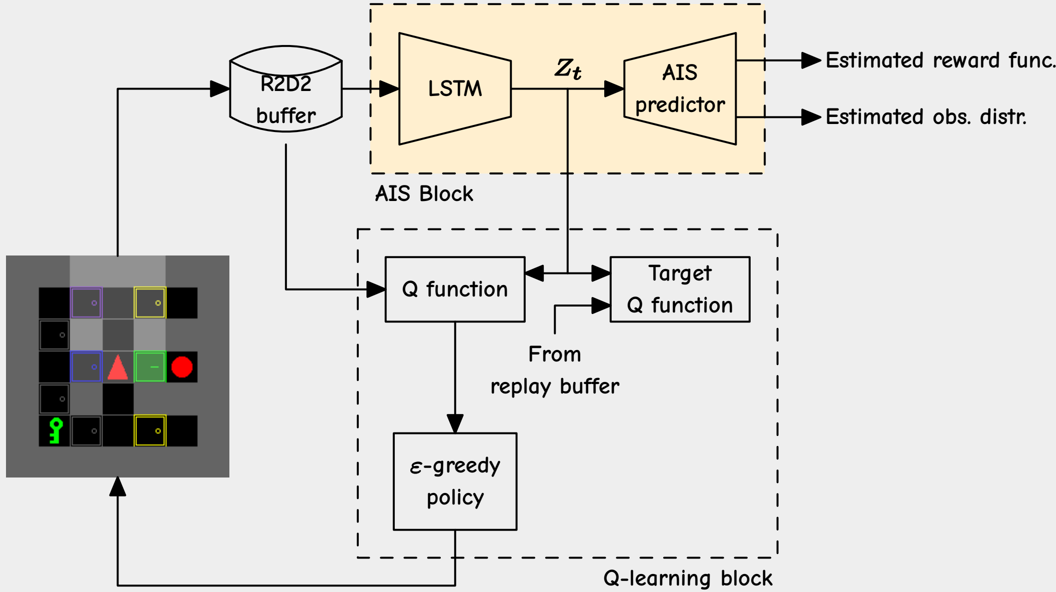
Sub-optimality bound: $\vec{J}^{\star}_{\mathsf{ND}} - J(\vec{\boldsymbol{\pi}}^{\mu}_{\mathsf{ASQL}}) \le$ function(ε, δ) where

$$\varepsilon_t = \sup_{h_t, a_t} \big| \mathbb{E}[R_t \mid h_t, a_t] - r^{\mu}_{\mathsf{ASQL}}(\vec{\sigma}_t(h_t), a_t) \big|$$

$$\delta_t = \sup_{h_t, a_t} d_{\mathfrak{F}}\big(\mathbb{P}(Z_{t+1} \mid h_t, a_t), P^{\mu}_{\mathsf{ASQL}}(Z_{t+1} \mid \vec{\sigma}_t(h_t), a_t)\big)$$

Main idea: Minimizing ε and δ will lead to better learning.

Adding AIS losses

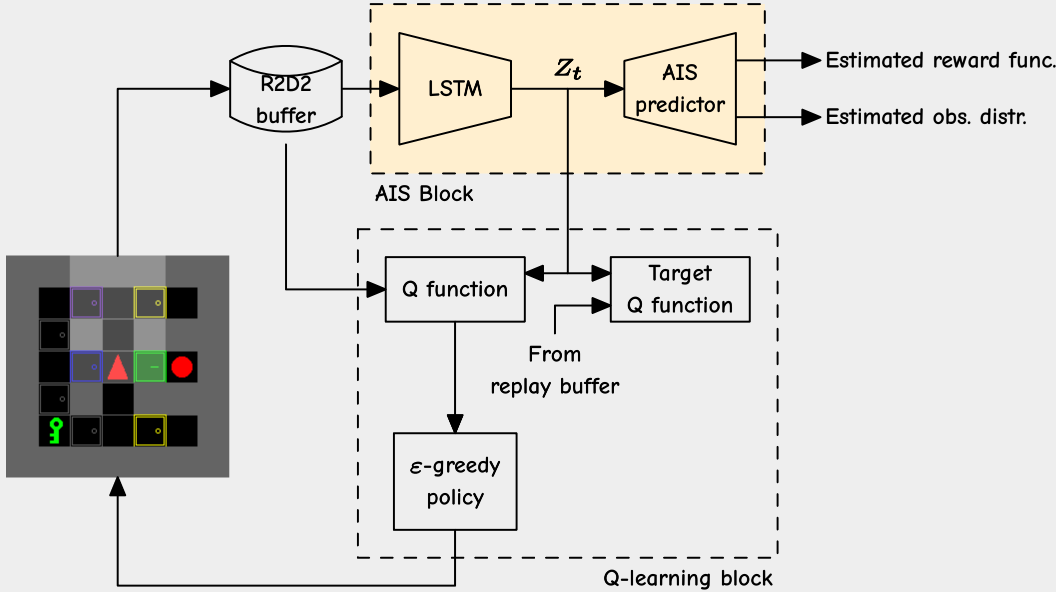


SeyedSalehi, Akbarzadeh, Sinha, Mahajan, "Approximate information state based convergence analysis of recurrent Q-learning", EWRL 2023.

Subramanian, Sinha, Seraj, and Mahajan, "Approximate information state for ... partially observed systems", JMLR 2022.

Ni, et al, "Briding State and History Representations: Understanding self-predictive RL", ICLR 2024.

Adding AIS losses



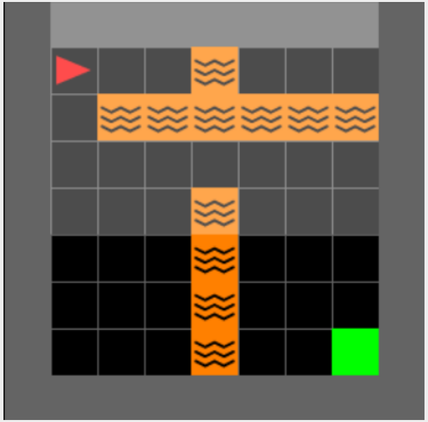
Same idea in actor-critic algorithms

SeyedSalehi, Akbarzadeh, Sinha, Mahajan, “Approximate information state based convergence analysis of recurrent Q-learning”, EWRL 2023.

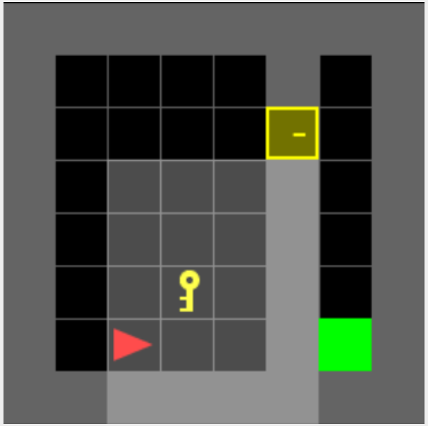
Subramanian, Sinha, Seraj, and Mahajan, “Approximate information state for ... partially observed systems”, JMLR 2022.

Ni, et al, “Briding State and History Representations: Understanding self-predictive RL”, ICLR 2024.

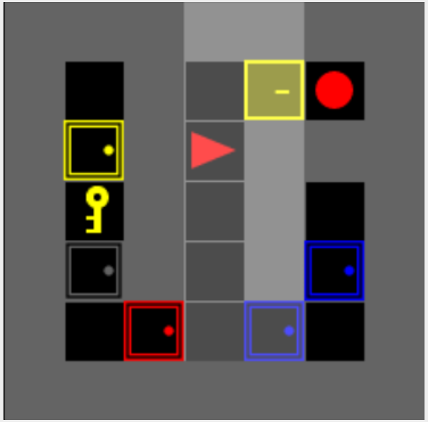
Minigrid test bench



Lava Crossing



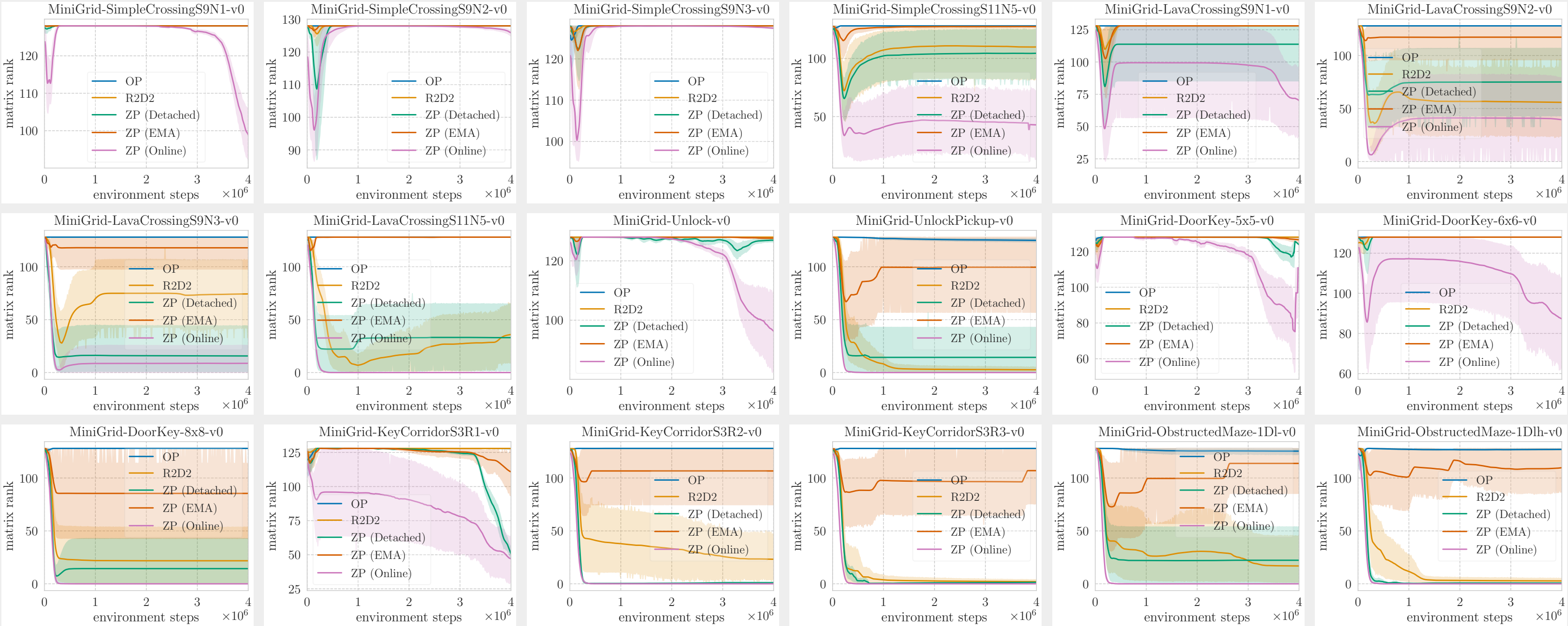
Door Key



Key Corridor

- Partially observable gridworlds with increasing complexity
- Compare several variations of QL+AIS with R2D2

Experimental results



Ni, et al, “Briding State and History Representations: Understanding self-predictive RL”, ICLR 2024.

Conclusion

Partial characterization of (approxmately) optimal agent-state based policies in different policy classes.

A general framework for analyzing and improving RL algorithms for POMDPs.

$$
\begin{tikzcd}
J^\star_{\mathsf{SD}} \arrow[r] \arrow[d] & J^\star_{\mathsf{ND}} \arrow[r] \arrow[d, leftrightarrow] & \vec{J}^\star_{\mathsf{ND}} \arrow[d, leftrightarrow] \\
J^\star_{\mathsf{SS}} \arrow[r] & J^\star_{\mathsf{NS}} \arrow[r] & \vec{J}^\star_{\mathsf{NS}}
\end{tikzcd}
$$

Conclusion

Partial characterization of (approxmately) optimal agent-state based policies in different policy classes.

A general framework for analyzing and improving RL algorithms for POMDPs.

$$
\begin{tikzcd}
J^\star_{\mathsf{SD}} \arrow[r] \arrow[d] & J^\star_{\mathsf{ND}} \arrow[r] \arrow[d, leftrightarrow] & \vec{J}^\star_{\mathsf{ND}} \arrow[d, leftrightarrow] \\
J^\star_{\mathsf{SS}} \arrow[r] & J^\star_{\mathsf{NS}} \arrow[r] & \vec{J}^\star_{\mathsf{NS}}
\end{tikzcd}
$$

Theory is still in its infancy. There are lots of interesting question to be answered.

- email: aditya.mahajan@mcgill.ca
- web: https://adityam.github.io

Thank you

- tutorial: Agent-state based policies on POMDPs
- paper: https://arxiv.org/abs/2409.15703