Planning (and learning) in multi-agent teams

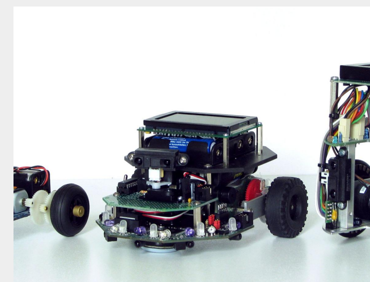
Aditya Mahajan
McGill University

Mila RL Workshop
3 June 2023

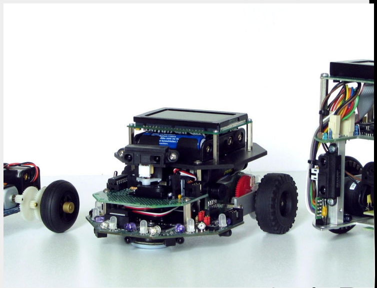
- email: aditya.mahajan@mcgill.ca
- web: http://cim.mcgill.ca/~adityam


Common theme: multi-stage multi-agent
decision making under uncertainty

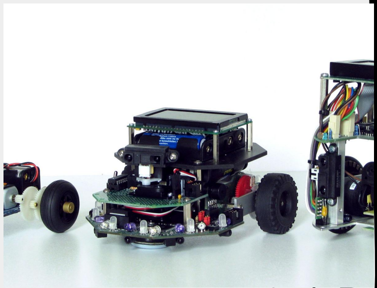
Networked control systems



Networked control systems



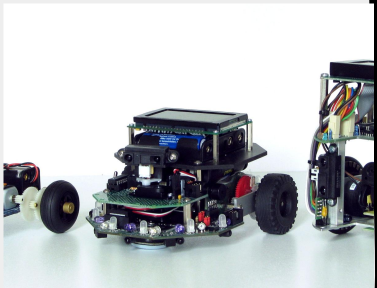
Networked control systems



Challenges

- Signals sent over wireless channels (packet drops)

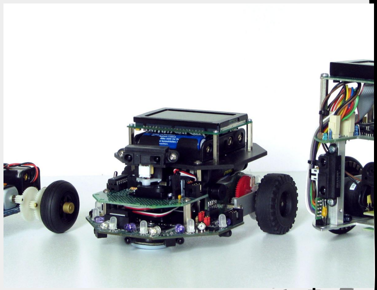
Networked control systems



Challenges

- Signals sent over wireless channels (packet drops)
- Different vehicles have different information

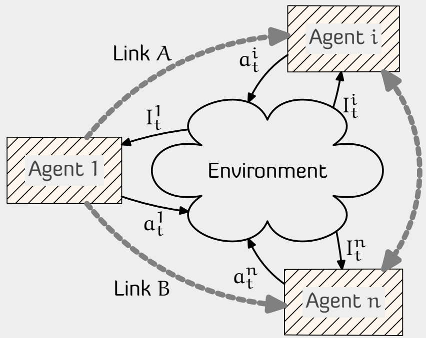
Networked control systems



Challenges

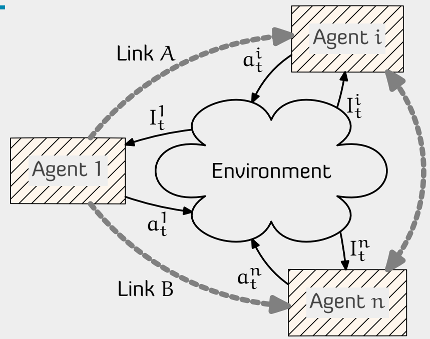
- Signals sent over wireless channels (packet drops)
- Different vehicles have different information
  - Decentralized control
  - Decentralized estimation
  - Decentralized learning

Salient Features



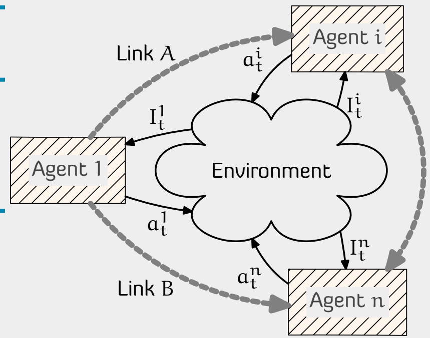
Salient Features

Multiple agents | Agents have different information and operate in stochastic dynamic environments



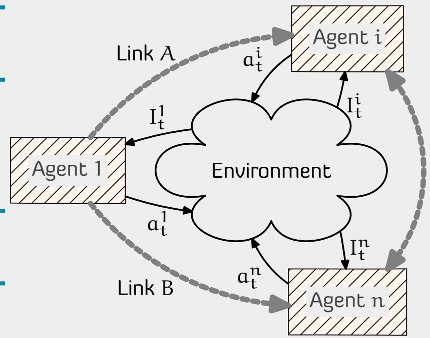
Salient Features

Multiple agents	Agents have different information and operate in stochastic dynamic environments
Decentralized Coordination	All agents must coordinate to achieve a system-wide objective



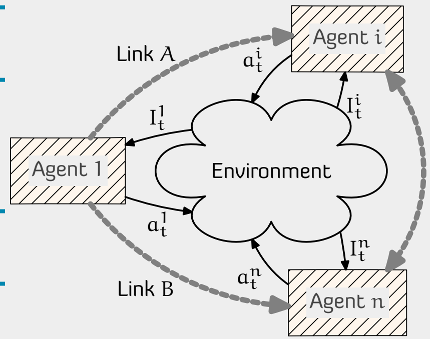
Salient Features

Multiple agents	Agents have different information and operate in stochastic dynamic environments
Decentralized Coordination	All agents must coordinate to achieve a system-wide objective
Communication & Signaling	Possible to explicitly or implicitly communicate information



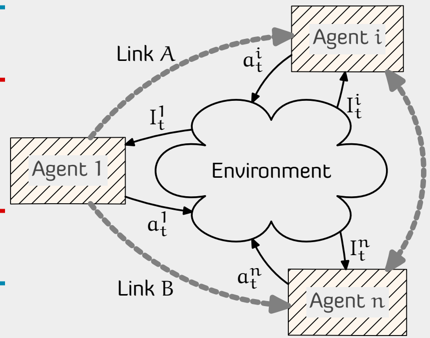
Salient Features

Multiple agents	Agents have different information and operate in stochastic dynamic environments
Decentralized Coordination	All agents must coordinate to achieve a system-wide objective
Communication & Signaling	Possible to explicitly or implicitly communicate information
Decentralized Learning	System model may not be completely known or may change over time



Salient Features

Multiple agents	Agents have different information and operate in stochastic dynamic environments
Decentralized Coordination	All agents must coordinate to achieve a system-wide objective
Communication & Signaling	Possible to explicitly or implicitly communicate information
Decentralized Learning	System model may not be completely known or may change over time



Teams versus Games

Teams vs Games

Teams

- All agents have **common objective**
- Agents **cooperate** to minimize team cost
- Agents are **not strategic**
- Solution concepts: person-by-person optimality, global optimality . . .

Games

- Each agent has **individual objective**
- Agents **compete** to minimize individual cost
- Agents are **strategic**
- Solution concepts: Nash equilibrium, Bayesian Nash, Subgame perfect equilibrium, Markov perfect equilibrium, Bayesian perfect equilibrium, . . .

Teams vs Games

Teams

- All agents have **common objective**
- Agents **cooperate** to minimize team cost
- Agents are **not strategic**
- Solution concepts: person-by-person

Games

- Each agent has **individual objective**
- Agents **compete** to minimize individual cost
- Agents are **strategic**

In many engineering problems, game theory is used as an algorithmic toolbox to provide distributed solutions to **static** problems.

We are interested in finding **globally optimal** solution to problems where **agents have decentralized information**.

Teams have a reputation of being notoriously difficult . . .

Some historical context

Some historical context

S&C until the 1960s

- About 300 years of knowledge in designing **LTI systems**
- Good "intuitive" understanding of **frequency domain methods**
  - Root locus
  - Bode plots
  - Nyquist plots
  - Loop shaping

Some historical context

S&C until the 1960s

- About 300 years of knowledge in designing **LTI systems**
- Good "intuitive" understanding of **frequency domain methods**
  - Root locus
  - Bode plots
  - Nyquist plots
  - Loop shaping

Advances in 1960s

- Emergence of **state space methods** for filtering and control
- Could be implemented in digital computers (of that time!)

Some historical context

S&C until the 1960s

- About 300 years of knowledge in designing **LTI systems**
- Good "intuitive" understanding of **frequency domain methods**
  - Root locus • Bode plots • Nyquist plots • Loop shaping

Advances in 1960s

- Emergence of **state space methods** for filtering and control
- Could be implemented in digital computers (of that time!)

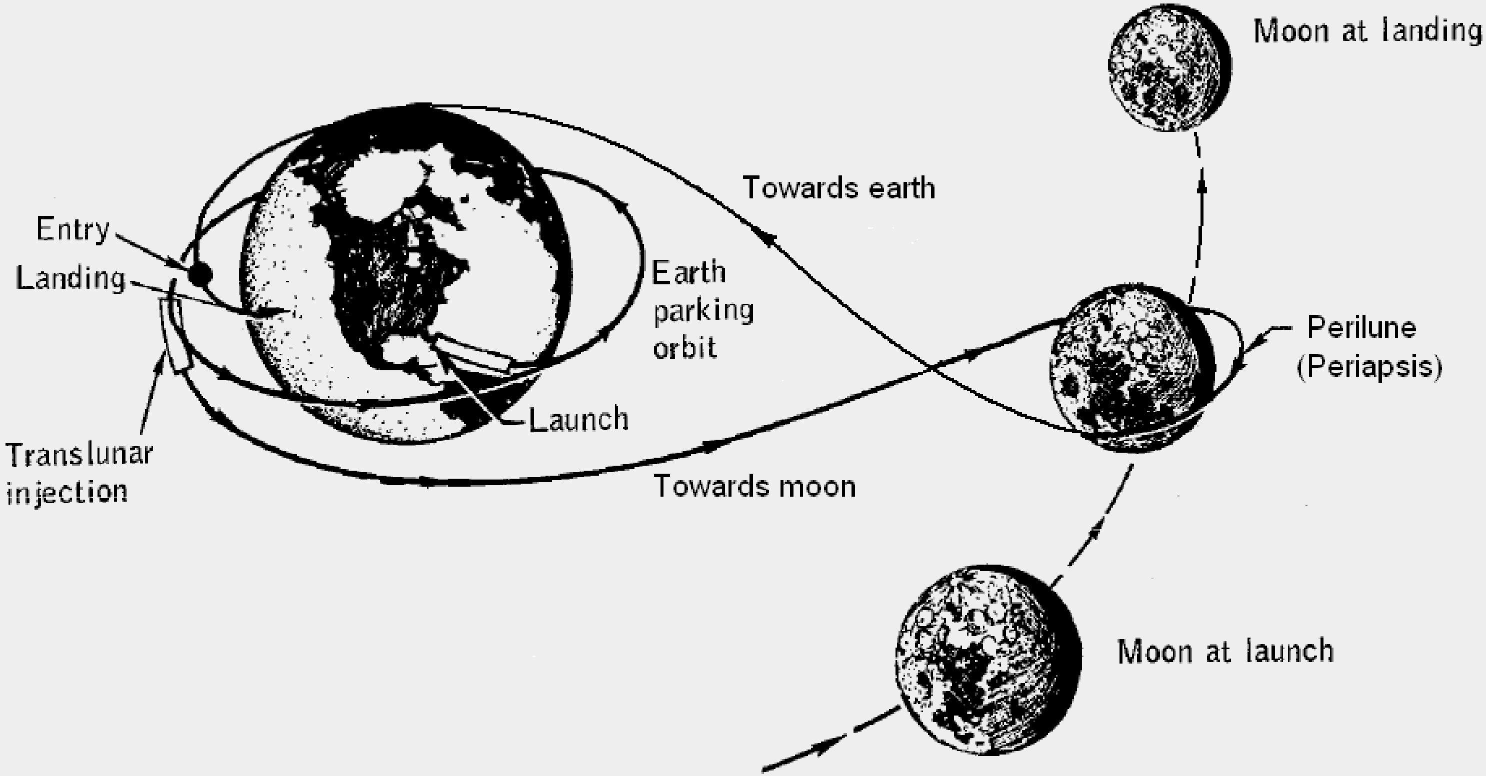
State Space Deisgn

- Linearize the system dynamics
- Design **optimal control** assuming full state feedback (LQR)

  control action(t) = −gain(t) · state(t)
- Estimate the state using noisy measurements (Kalman filtering)

  state estimate(t) = Function(estimate(t-1), measurement(t).
- **Optimal controller**:

  control action(t) = −gain(t) · state estimate(t)


Moon at landing
Towards earth
Entry
Landing
Earth
parking
orbit
Perilune
(Periapsis)
Launch
Translunar
injection
Towards moon
Moon at launch

Conceptual difficulties in team problems

Witsenhausen Counterexample

- A two step dynamical system with two controllers
- Linear dynamics, quadratic cost, and Gaussian disturbance
- Non-linear controllers outperform linear control strategies . . .

  . . . cannot use Kalman filtering + Riccati equations

Witsenhausen, "A counterexample in stochastic optimum control," SICON 1968.

Whittle and Rudge, "The optimal linear solution of a symmetric team control problem," App. Prob. 1974.

Bernstein, et al, "The complexity of decentralized control of Markov decision processes," MOR 2002.

Conceptual difficulties in team problems

Witsenhausen Counterexample

- A two step dynamical system with two controllers
- Linear dynamics, quadratic cost, and Gaussian disturbance
- Non-linear controllers outperform linear control strategies . . .

  . . . cannot use Kalman filtering + Riccati equations

Whittle and Rudge Example

- Infinite horizon dynamical system with two symmetric controllers
- Linear dynamics, quadratic cost, and Gaussian disturbance
- *A priori* restrict attention to linear controllers
- Best linear controllers **don't** have finite dimensional representation

---

Witsenhausen, "A counterexample in stochastic optimum control," SICON 1968.

Whittle and Rudge, "The optimal linear solution of a symmetric team control problem," App. Prob. 1974.

Bernstein, et al, "The complexity of decentralized control of Markov decision processes," MOR 2002.

Conceptual difficulties in team problems

Witsenhausen Counterexample

- A two step dynamical system with two controllers
- Linear dynamics, quadratic cost, and Gaussian disturbance
- Non-linear controllers outperform linear control strategies . . .

. . . cannot use Kalman filtering + Riccati equations

Whittle and Rudge Example

- Infinite horizon dynamical system with two symmetric controllers
- Linear dynamics, quadratic cost, and Gaussian disturbance
- A priori restrict attention to linear controllers
- Best linear controllers **don't** have finite dimensional representation

Complexity analysis

- All random variables are finite valued
- Finite horizon setup
- The problem of finding the best control strategy is in **NEXP**

Witsenhausen, "A counterexample in stochastic optimum control," SICON 1968.

Whittle and Rudge, "The optimal linear solution of a symmetric team control problem," App. Prob. 1974.

Bernstein, et al, "The complexity of decentralized control of Markov decision processes," MOR 2002.

Why are team problems hard?

Why are team problems hard?

Why are single agent problems easy?

Static stochastic optimization problems

$$\min_{\pi:\mathcal{Y}\to\mathcal{A}} \mathbb{E}[c(S, \pi(Y))]$$

	$S = 0$	$S = 1$	$S = 2$	$S = 3$
$A = 0$	0.5	0.2	1.2	0.5
$A = 1$	1.2	0.5	0.2	0.3

$Y = 0$	$Y = 1$

Static stochastic optimization problems

$$\min_{\pi:\mathcal{Y}\to\mathcal{A}} \mathbb{E}[c(S,\pi(Y))]$$

	$S=0$	$S=1$	$S=2$	$S=3$
$A=0$	0.5	0.2	1.2	0.5
$A=1$	1.2	0.5	0.2	0.3

$Y=0$	$Y=1$


Static stochastic optimization problems

$$\min_{\pi:\mathcal{Y}\to\mathcal{A}} \mathbb{E}[c(S,\pi(Y))]$$

	$S=0$	$S=1$	$S=2$	$S=3$
$A=0$	0.5	0.2	1.2	0.5
$A=1$	1.2	0.5	0.2	0.3

$Y=0$	$Y=1$

- This is a **functional optimization** problem.
- Search complexity $|\mathcal{A}|^{|\mathcal{Y}|}$.

Static stochastic optimization problems

$$\min_{\pi:\mathcal{Y}\to\mathcal{A}} \mathbb{E}[c(S,\pi(Y))]$$

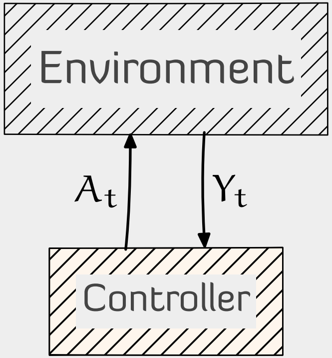
	$S=0$	$S=1$	$S=2$	$S=3$
$A=0$	0.5	0.2	1.2	0.5
$A=1$	1.2	0.5	0.2	0.3
	$Y=0$		$Y=1$	

- This is a **functional optimization** problem.
- Search complexity $|\mathcal{A}|^{|\mathcal{Y}|}$.

$$\text{for each } y, \quad \min_{a\in\mathcal{A}} \mathbb{E}[c(S,a) \mid Y=y]$$

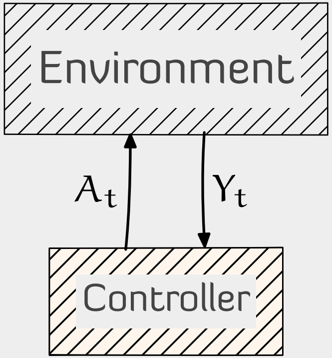
- Each sub-problem is a **parameter optimization** problem.
- Search complexity $|\mathcal{A}| \cdot |\mathcal{Y}|$.

Dynamic stochastic optimization problems



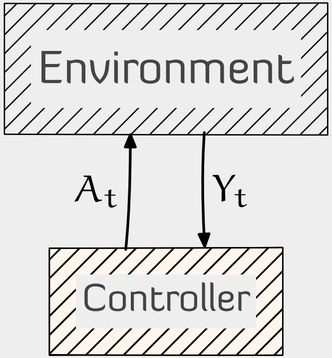
Dyanmics	$S_{t+1} = f_t(S_t, A_t, W_t)$
Observations	$Y_t = h^i_t(S_t, N_t)$
Control law	$A_t = \pi_t(Y_{1:t}, A_{1:t-1})$

Dynamic stochastic optimization problems



Dyanmics	$S_{t+1} = f_t(S_t, A_t, W_t)$
Observations	$Y_t = h^i_t(S_t, N_t)$
Control law	$A_t = \pi_t(Y_{1:t}, A_{1:t-1})$
Objective	Choose control strategy $\pi = (\pi_1, \dots, \pi_T)$ to minimize $$J(\pi) = \mathbb{E}\Big[\sum_{t=1}^{T} c_t(S_t, A_t)\Big]$$

Dynamic stochastic optimization problems



Dyanmics	$S_{t+1} = f_t(S_t, A_t, W_t)$
Observations	$Y_t = h^i_t(S_t, N_t)$
Control law	$A_t = \pi_t(Y_{1:t}, A_{1:t-1})$
	Choose control strategy $\pi =$

Dynamic programming solution

- Define **belief state** $b_t = P(S_t | Y_{1:t}, A_{1:t-1})$.
- Write a DP in terms of the belief state b_t.
- Solution complexity: $T \cdot |\mathcal{A}| \cdot |\mathcal{Z}|$.

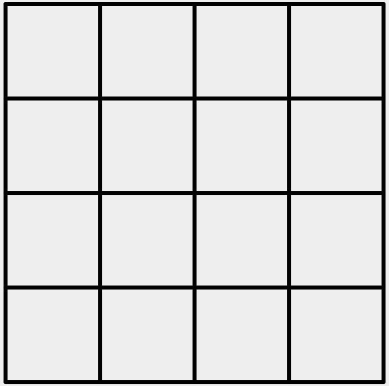
Why don't these simplifications work for teams?

Static team problem

$$\min_{\pi^1,\pi^2} \mathbb{E}[c(S,\pi^1(Y^1),\pi^2(Y^2))]$$

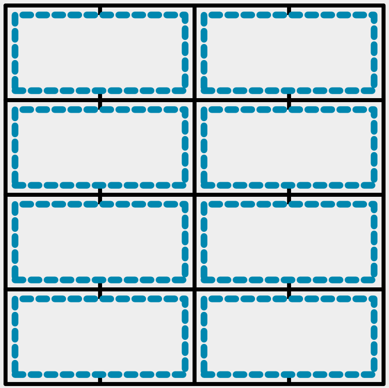
Static team problem

$$\min_{\pi^1, \pi^2} \mathbb{E}[c(S, \pi^1(Y^1), \pi^2(Y^2))]$$



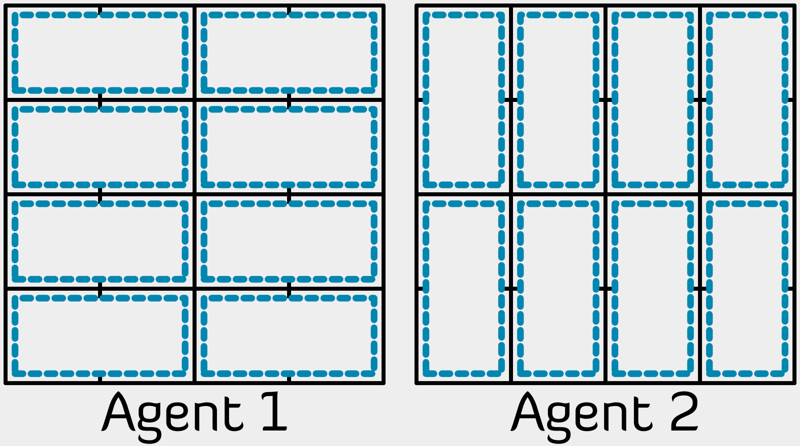
Static team problem

$$\min_{\pi^1, \pi^2} \mathbb{E}[c(S, \pi^1(Y^1), \pi^2(Y^2))]$$



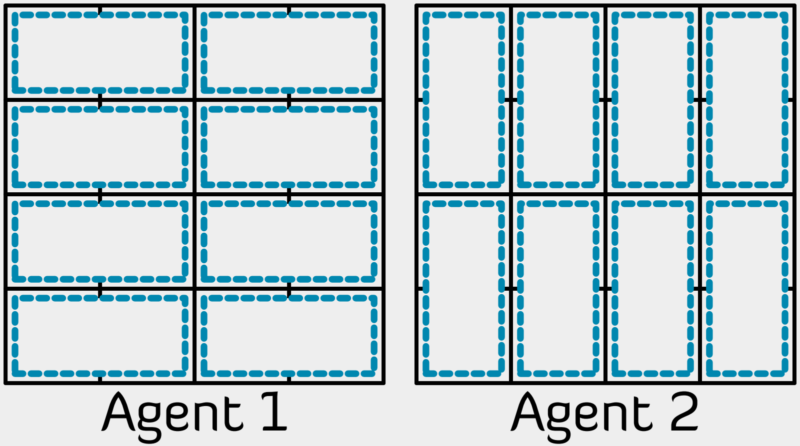
Static team problem

$$\min_{\pi^1, \pi^2} \mathbb{E}[c(S, \pi^1(Y^1), \pi^2(Y^2))]$$



Static team problem

$$\min_{\pi^1, \pi^2} \mathbb{E}[c(S, \pi^1(Y^1), \pi^2(Y^2))]$$



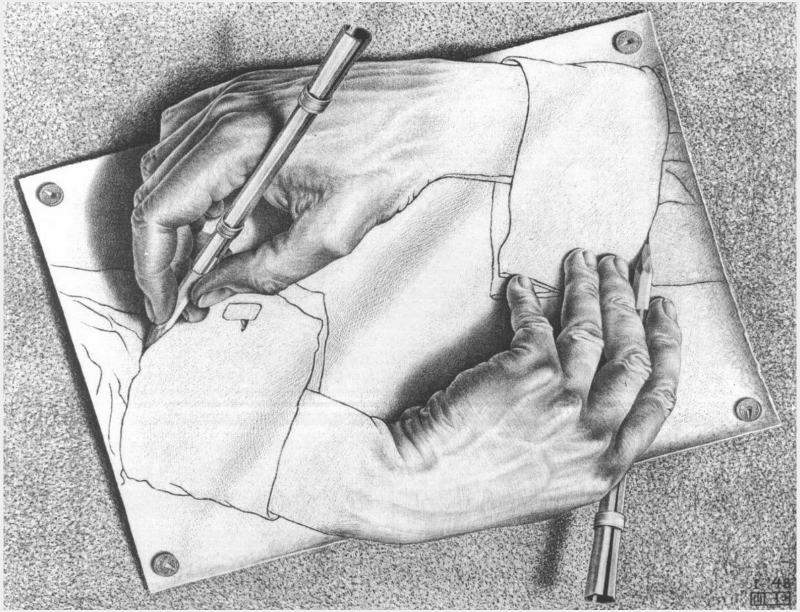
Previous idea of

$$\text{for all } y^1, \quad \min_{a^1} \mathbb{E}[c(S, a^1, \pi^2(Y^2)) \mid Y^1 = y^1]$$

leads to person-by-person optimal solution (not globally opt.)

Static team problem

$$\min_{\pi^1, \pi^2} \mathbb{E}[c(S, \pi^1(Y^1), \pi^2(Y^2))]$$



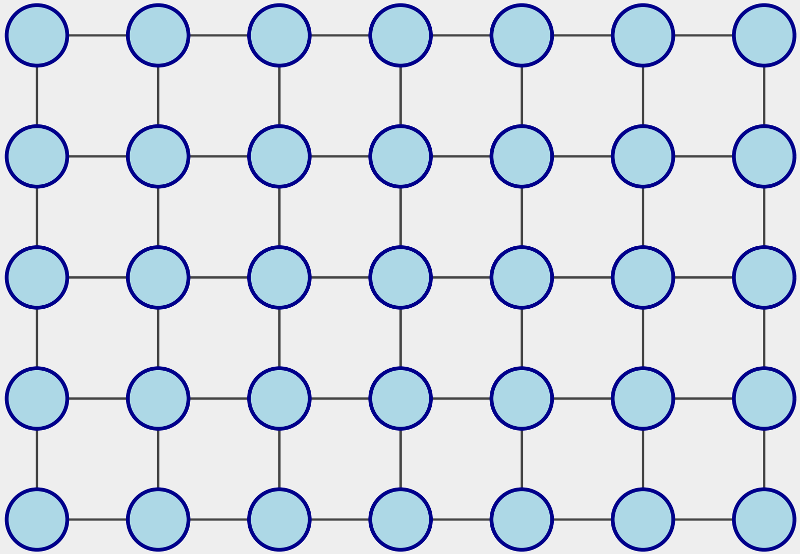
Previous idea of

$$\text{for all } y^1, \quad \min_{a^1} \mathbb{E}[c(S, a^1, \pi^2(Y^2)) \mid Y^1 = y^1]$$

leads to person-by-person optimal solution (not globally opt.)

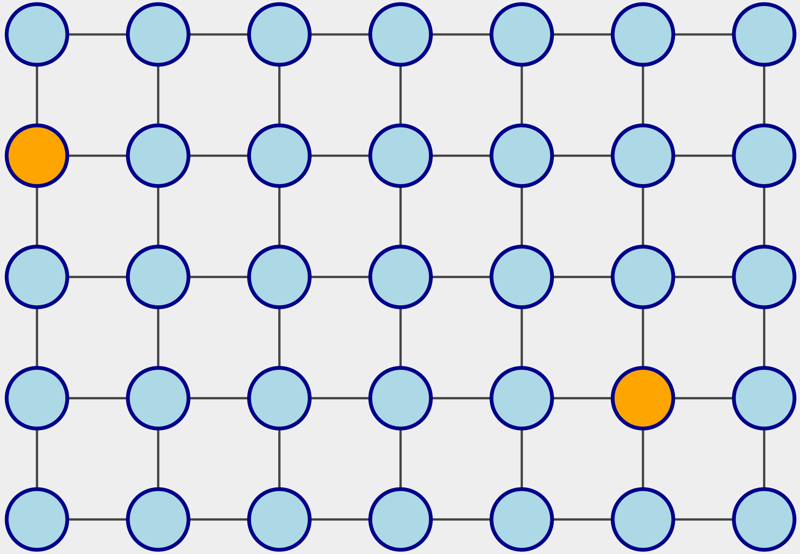
k-step delayed sharing information structure

- Consider a network with coupled dynamics.
- Information exchange between nodes with unit delay.



k-step delayed sharing information structure

- Consider a network with coupled dynamics.
- Information exchange between nodes with unit delay.

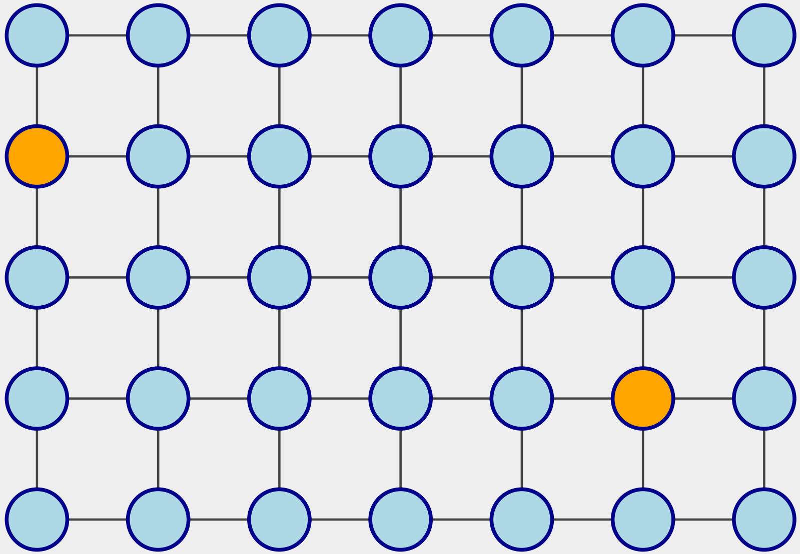


- Fix the strategy of all but two subsystems which are k-hop apart. What is the best response strategy at these two nodes?

Witsenhausen, “Separation of Esitmation and Control for Discrete-Time Systems,” Proc. IEEE, 1971.

k-step delayed sharing information structure

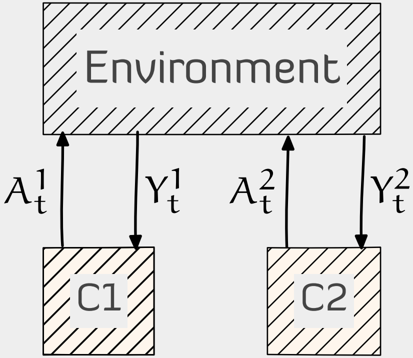
- Consider a network with coupled dynamics.
- Information exchange between nodes with unit delay.



- Fix the strategy of all but two subsystems which are k-hop apart. What is the best response strategy at these two nodes?
- Proposed by Witsenhausen in a seminal paper.
- Allows to smoothly transition between centralized ($k = 0$) and completely decentralized ($k = \infty$).

Witsenhausen, "Separation of Esitmation and Control for Discrete-Time Systems," Proc. IEEE, 1971.

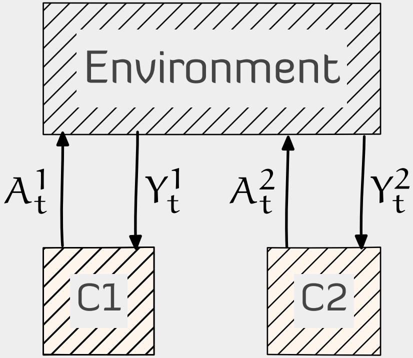
System Model



Dyanmics $S_{t+1} = f_t(S_t, A_t^1, A_t^2, W_t)$

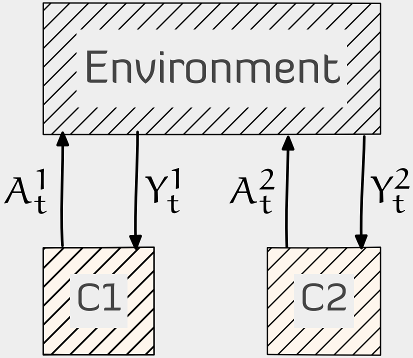
Observations $Y_t^i = h_t^i(S_t, N_t^i)$

System Model



Dyanmics	$S_{t+1} = f_t(S_t, A_t^1, A_t^2, W_t)$
Observations	$Y_t^i = h_t^i(S_t, N_t^i)$
Information Structure	$I_t^i = \{Y_{1:t}^i, A_{1:t-1}^i Y_{1:t-k}^{-i}, A_{1:t-k}^{-i}\}$
Control law	$A_t^i = \pi_t^i(I_t^i)$

System Model



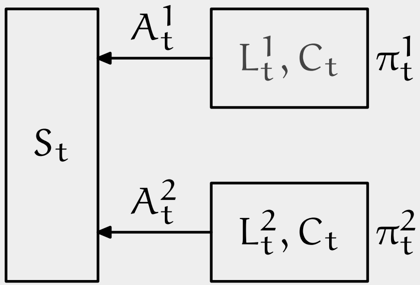
Dyanmics	$S_{t+1} = f_t(S_t, A_t^1, A_t^2, W_t)$
Observations	$Y_t^i = h_t^i(S_t, N_t^i)$
Information Structure	$I_t^i = \{Y^i_{1:t}, A^i_{1:t-1} Y^{-i}_{1:t-k}, A^{-i}_{1:t-k}\}$
Control law	$A_t^i = \pi_t^i(I_t^i)$
Objective	Choose control strategies (π_1, π_2) to minimize $$J(\pi_1, \pi_2) = \mathbb{E}\Big[\sum_{t=1}^{T} c_t(S_t, A_t^1, A_t^2)\Big]$$

Conceptual difficulty

The data I_t^i available at each controller is increasing with time.
How to find a sufficient statistic or an information state?

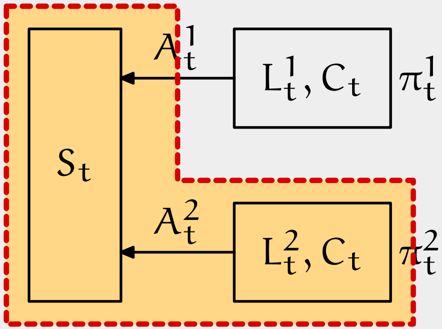
Conceptual difficulty

The data I_t^i available at each controller is increasing with time.
How to find a sufficient statistic or an information state?



Conceptual difficulty

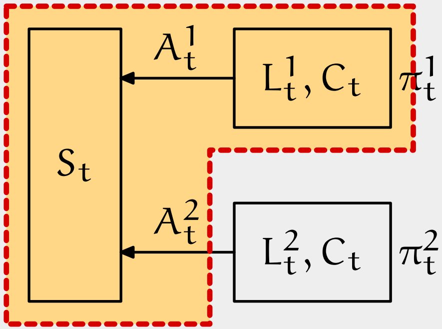
The data I_t^i available at each controller is increasing with time.
How to find a sufficient statistic or an information state?



- Unobserved state from the p.o.v. of ctrl 1: S_t, L_t^2, C_t.
  Information state $\pi_t^1 = \mathbb{P}(S_t, L_t^2, C_t | L_t^1, C_t)$.

Conceptual difficulty

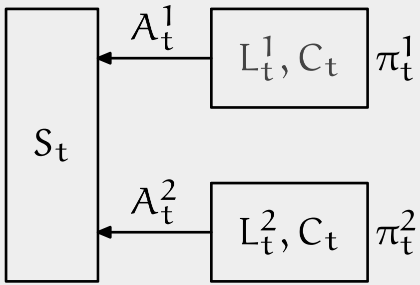
The data I_t^i available at each controller is increasing with time.
How to find a sufficient statistic or an information state?



- Unobserved state from the p.o.v. of ctrl 1: S_t, L_t^2, C_t.
  Information state $\pi_t^1 = \mathbb{P}(S_t, L_t^2, C_t | L_t^1, C_t)$.
- Unobserved state from the p.o.v. of ctrl 2: S_t, π_t^1.
  Information state $\pi_t^2 = \mathbb{P}(S_t, \pi_t^1 | L_t^2, C_t)$.

Conceptual difficulty

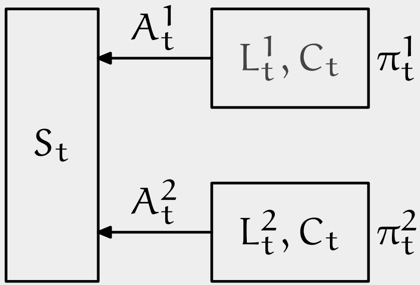
The data I_t^i available at each controller is increasing with time.
How to find a sufficient statistic or an information state?

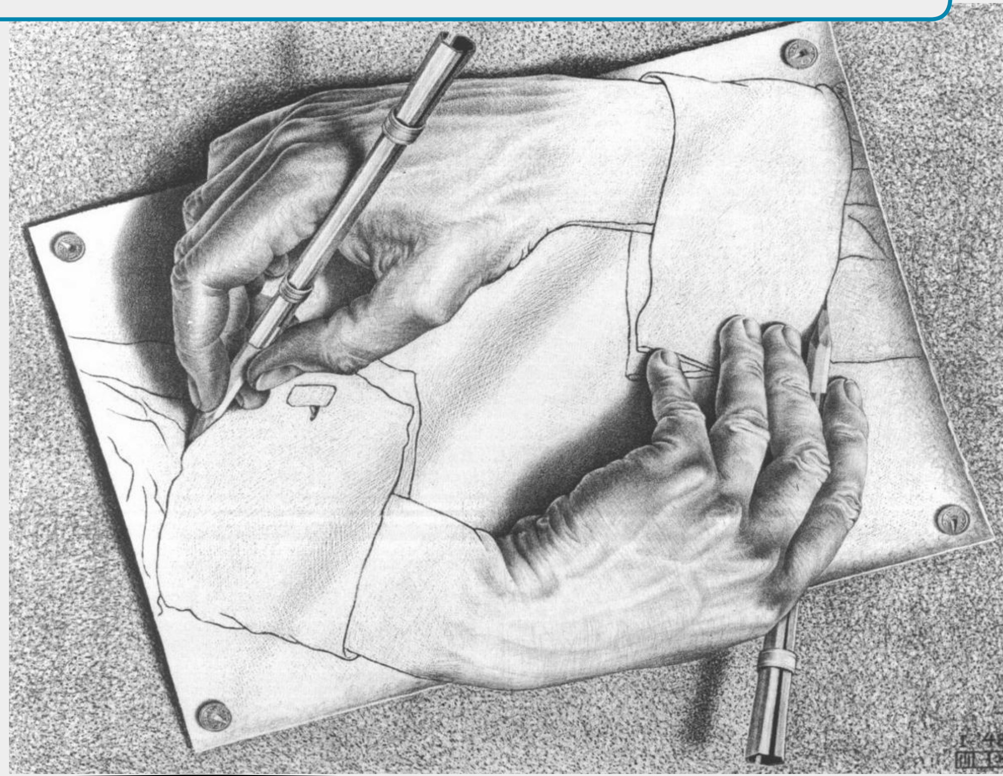


- Unobserved state from the p.o.v. of ctrl 1: S_t, L_t^2, C_t.
  Information state $\pi_t^1 = \mathbb{P}(S_t, L_t^2, C_t | L_t^1, C_t)$.
- Unobserved state from the p.o.v. of ctrl 2: S_t, π_t^1.
  Information state $\pi_t^2 = \mathbb{P}(S_t, \pi_t^1 | L_t^2, C_t)$.
- Unobserved state from the p.o.v. of ctrl 1: S_t, π_t^2.
  Information state $\pi_t^{1,2} = \mathbb{P}(S_t, \pi_t^2 | L_t^2, C_t)$.
- . . . infinite regress . . .

Conceptual difficulty

The data I_t^i available at each controller is increasing with time.
How to find a sufficient statistic or an information state?





History of the problem

Witsenhausen's Assertion

Let $C_t = \{Y_{1:t-k}, A_{1:t-k}\}$ and $L^i_t = \{Y^i_{t-k+1:t}, A^i_{t-k+1:t-1}\}$.

Then $\mathbb{P}(S_{t-k} \mid C_t)$ is a sufficient statistic for C_t.

Rationale: $\mathbb{P}(S_{t-k} | Y_{1:t-k}, A_{1:t-k})$ is policy independent.

Witsenhausen, "Separation of Esitmation and Control for Discrete-Time Systems," Proc. IEEE, 1971.

History of the problem

Witsenhausen's Assertion

Let $C_t = \{Y_{1:t-k}, A_{1:t-k}\}$ and $L_t^i = \{Y^i_{t-k+1:t}, A^i_{t-k+1:t-1}\}$.

Then $\mathbb{P}(S_{t-k} \mid C_t)$ is a sufficient statistic for C_t.

Rationale: $\mathbb{P}(S_{t-k} | Y_{1:t-k}, A_{1:t-k})$ is policy independent.

Follow-up Literature

- **Assertion true for $k = 1$**
  [Sandell, Athans, 1974], [Kurtaran, 1976]
- **Assertion false for $k > 1$**
  [Varaiya, Walrand 1979], [Yoshikawa, Kobayashi, 1978]
- **No subsequent positive result!**

Witsenhausen, "Separation of Esitmation and Control for Discrete-Time Systems," Proc. IEEE, 1971.

History of the problem

Witsenhausen's Assertion

Let $C_t = \{Y_{1:t-k}, A_{1:t-k}\}$ and $L^i_t = \{Y^i_{t-k+1:t}, A^i_{t-k+1:t-1}\}$.
Then $\mathbb{P}(S_{t-k} \mid C_t)$ is a sufficient statistic for C_t.

Rationale: $\mathbb{P}(S_{t-k}|Y_{1:t-k}, A_{1:t-k})$ is policy independent.

Follow-up Literature

- Assertion true for $k = 1$
  [Sandell, Athans, 1974], [Kurtaran, 1976]
- Assertion false for $k > 1$
  [Varaiya, Walrand 1979], [Yoshikawa, Kobayashi, 1978]
- No subsequent positive result!

Are there sufficient statistics or information states for C_t?

Witsenhausen, "Separation of Esitmation and Control for Discrete-Time Systems," Proc. IEEE, 1971.

Importance of the problem

Applications (of one-step delay sharing)

- **Power systems**: Altman et al, 2009
- **Queueing theory**: Kuri and Kumar, 1995
- **Communication networks**: Grizzle et al, 1982
- **Stochastic games**: Papavassilopoulos, 1982; Chang and Cruz, '83
- **Economics**: Li and Wu, 1991.

Importance of the problem

Applications (of one-step delay sharing)

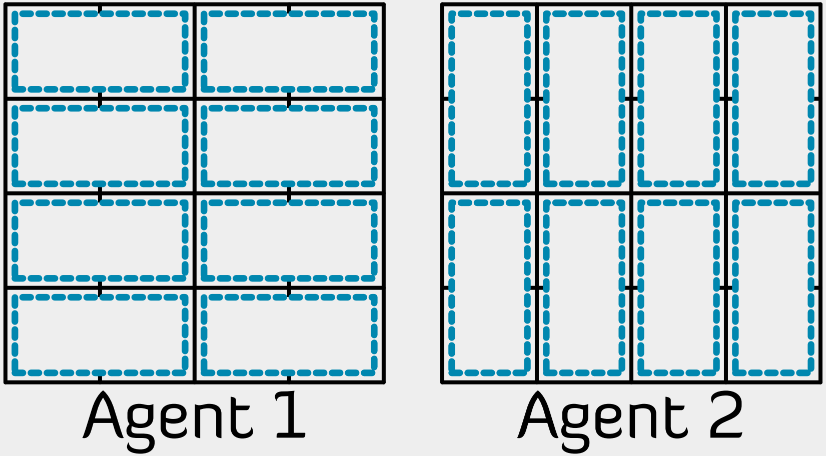
- **Power systems**: Altman et al, 2009
- **Queueing theory**: Kuri and Kumar, 1995
- **Communication networks**: Grizzle et al, 1982
- **Stochastic games**: Papavassilopoulos, 1982; Chang and Cruz, '83
- **Economics**: Li and Wu, 1991.

Conceptual Significance

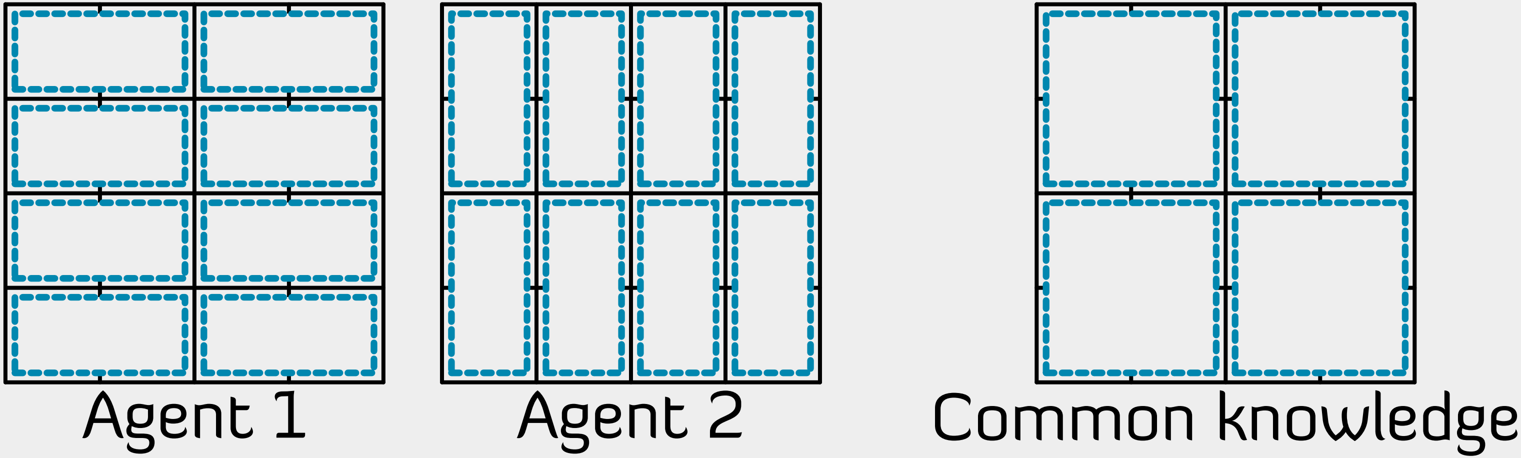
- Understanding the **design of networked control systems**
- **Bridge** between centralized and decentralized systems
- **Insights** for the design of general decentralized systems.

Common information approach for teams
[Nayyar, Mahajan, Teneketzis (2011, 2013)]

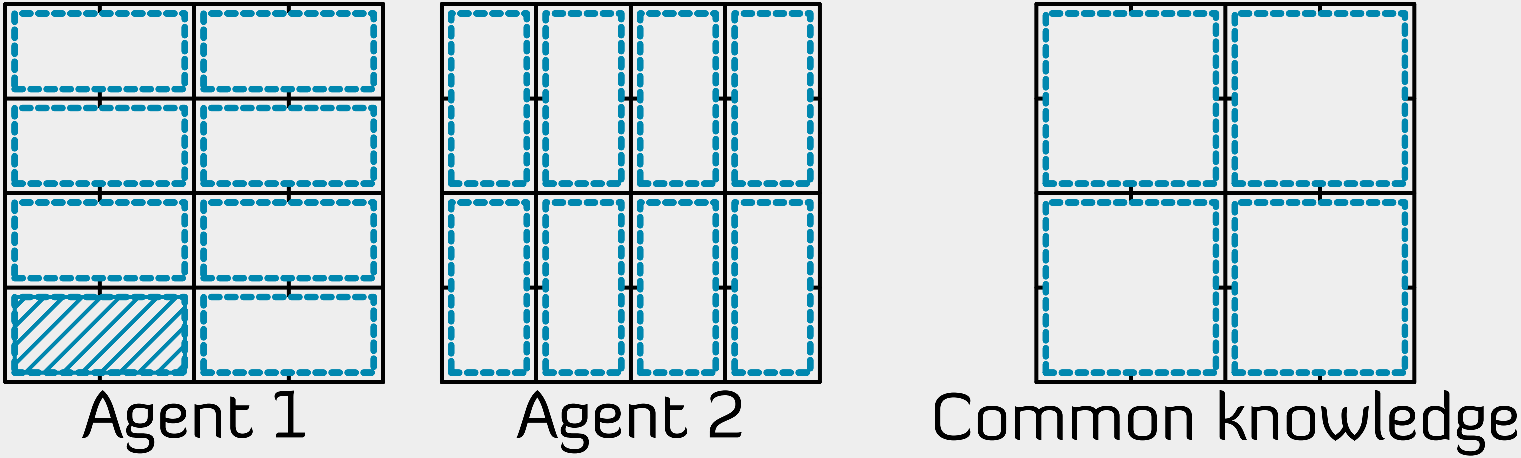
Key idea: exploit common knowledge



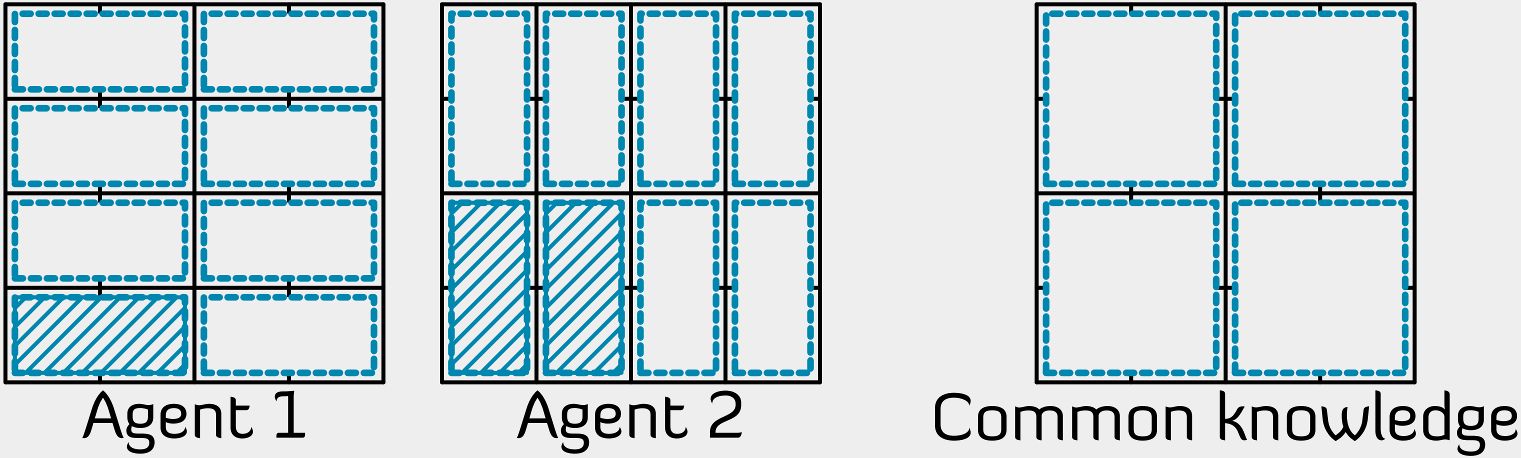
Key idea: exploit common knowledge



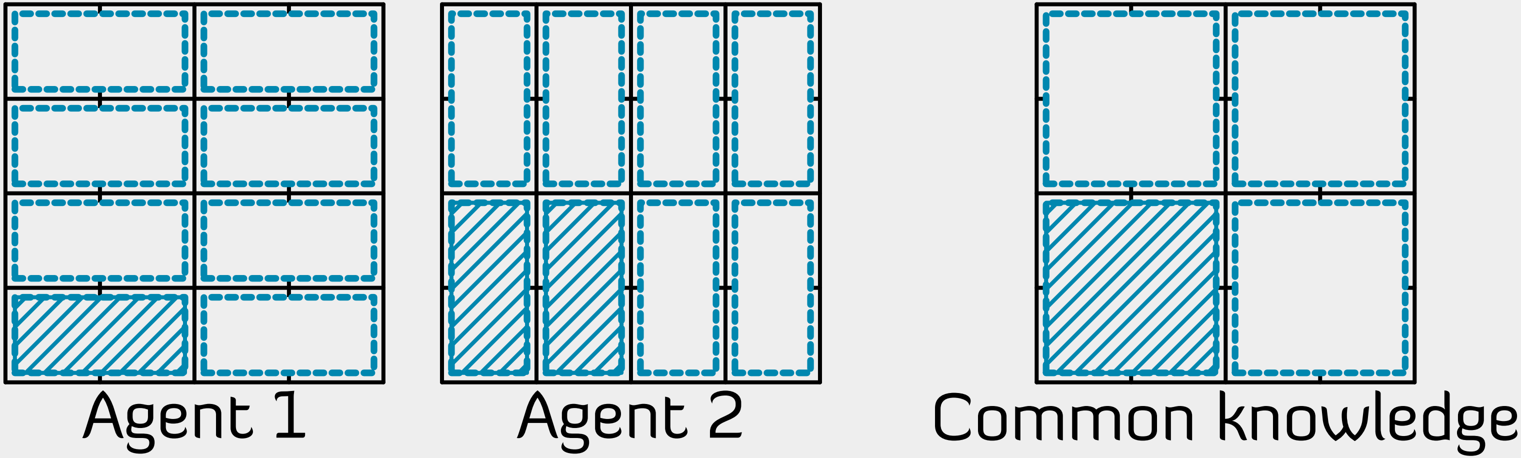
Key idea: exploit common knowledge



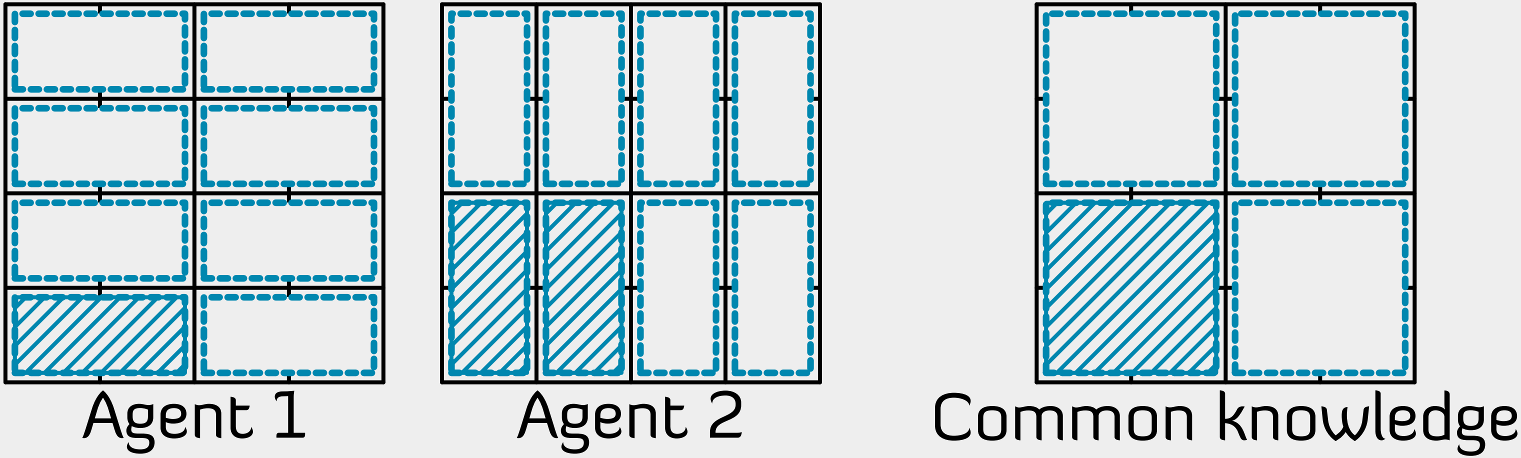
Key idea: exploit common knowledge



Key idea: exploit common knowledge



Key idea: exploit common knowledge



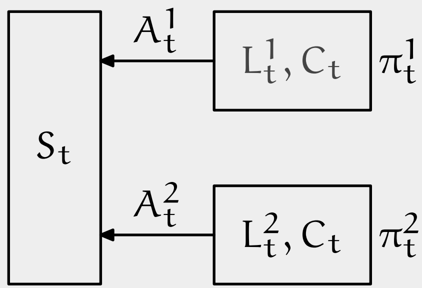
Split $Y^1 = (L^1, C)$ and $Y^2 = (L^2, C)$.

for all c, $\min_{\gamma^1, \gamma^2} \mathbb{E}[c(S, \gamma^1(L^1), \gamma^2(L^2))) \mid C = c]$

Reduction in complexity: $|\mathcal{A}|^8 \cdot |\mathcal{A}|^8$ to $4|\mathcal{A}|^2 \cdot |\mathcal{A}|^2$

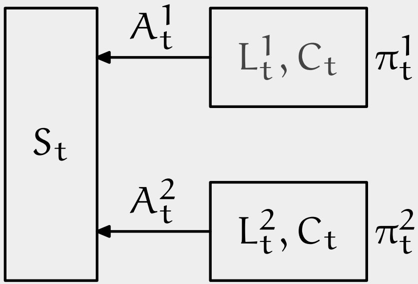
Common-info approach for k-step delay sharing

Original System

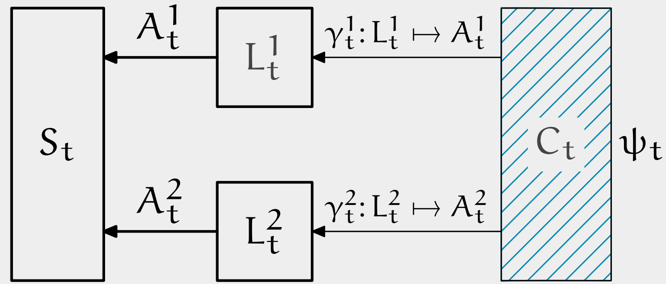


Common-info approach for k-step delay sharing

Original System

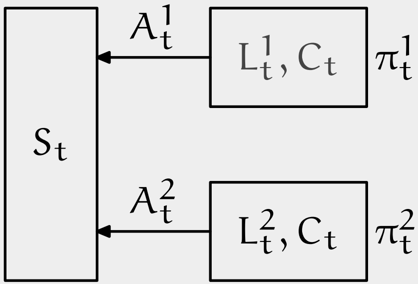


Virtual Coordinated System

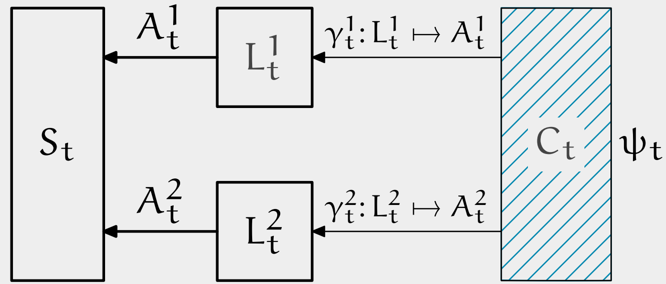


Common-info approach for k-step delay sharing

Original System



Virtual Coordinated System



Information split

- Common information: $C_t = I_t^1 \cap I_t^2 = \{Y_{1:t-k}, A_{1:t-k}\}$
- Local information: $L_t^i = I_t^i \setminus C_t = \{Y_{t-k+1:t}^i, A_{t-k+1:t-1}^i\}$.
- Prescription: $\gamma_t^i : L_t^i \mapsto A_t^i$.

Common-info approach for k-step delay sharing

Main Result

- The virtual coordinator is a single agent stochastic ctrl problem.
- Information state: for C_t: $b_t = \mathbb{P}(S_t, L^1_t, L^2_t \mid C_t, \gamma^1_{1:t-1}, \gamma^2_{1:t-1})$.
- Dynamic program: $V_{T+1}(b) = 0$ and

$$V_t(b_t) = \min_{\gamma^1_t, \gamma^2_t} \{\mathbb{E}[c_t(S_t, A^1_t, A^2_t) + V_{t+1}(B_+) \mid b_t, \gamma^1_t, \grave{\gamma}^2_t]\}.$$

- Each step of the DP is a functional optimization problem.

Common-info approach for k-step delay sharing

Main Result

- The virtual coordinator is a single agent stochastic ctrl problem.
- Information state: for C_t: $b_t = \mathbb{P}(S_t, L_t^1, L_t^2 \mid C_t, \gamma_{1:t-1}^1, \gamma_{1:t-1}^2)$.
- Dynamic program: $V_{T+1}(b) = 0$ and

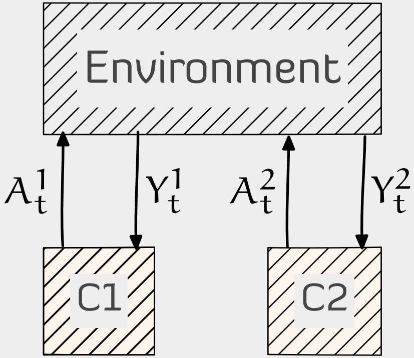
$$V_t(b_t) = \min_{\gamma_t^1, \gamma_t^2} \{\mathbb{E}[c_t(S_t, A_t^1, A_t^2) + V_{t+1}(B_+) \mid b_t, \gamma_t^1, `\gamma_t^2]\}.$$

- Each step of the DP is a functional optimization problem.

Salient Features

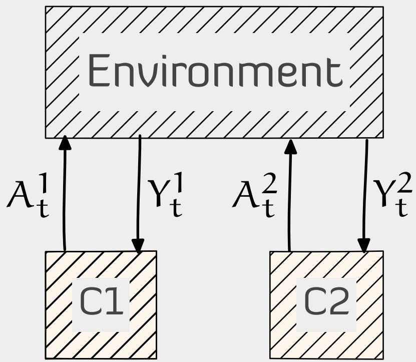
- The virtual coordinator is purely for conceptual clarity as it allows us to view the original problem from the p.o.v. of a "higher authority". The presence of the coordinator is not necessary.
- The common information is known to both controllers and therefore both of them can carry out the calculations to solve the DP on their own.

The general common-info approach



- n controllers with general info structure $\{I_t^i\}_{i=1}^n$.

The general common-info approach

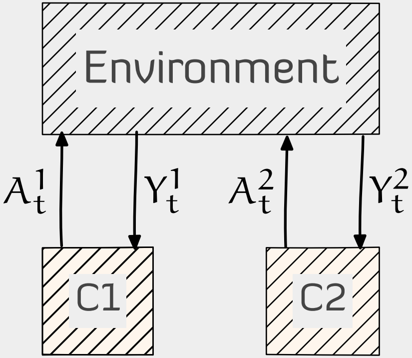


- n controllers with general info structure $\{I^i_t\}_{i=1}^n$.

Information Split

- **Common information**: $C_t = \bigcap_{s \geq t} \bigcap_{i=1}^{n} I^i_s$.
- **Local information**: $L^i_t = I^i_t \setminus C_t$.

The general common-info approach

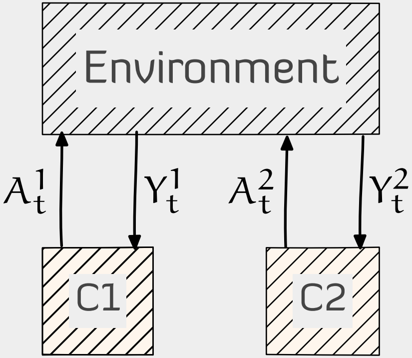


- n controllers with general info structure $\{I_t^i\}_{i=1}^n$.

Information Split	
	Common information: $C_t = \bigcap_{s \geq t} \bigcap_{i=1}^{n} I_s^i$. **Local information**: $L_t^i = I_t^i \setminus C_t$.

Partial history sharing	
	$\lvert L_t^i \rvert$ is uniformly bounded. $\mathbb{P}^\psi(C_{t+1} \setminus C_t \mid C_t, \gamma_t^1, \gamma_t^2)$ doesn't depend on ψ.

The general common-info approach



- n controllers with general info structure $\{I_t^i\}_{i=1}^n$.

Information Split

- Common information: $C_t = \bigcap_{s \geq t} \bigcap_{i=1}^{n} I_s^i$.
- Local information: $L_t^i = I_t^i \setminus C_t$.

Partial history sharing

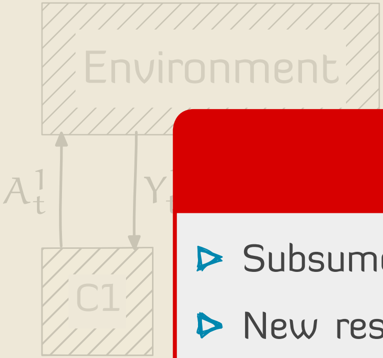
- $|L_t^i|$ is uniformly bounded.
- $\mathbb{P}^{\psi}(C_{t+1} \setminus C_t \mid C_t, \gamma_t^1, \gamma_t^2)$

Main Result

- Information state: for C_t: $b_t = \mathbb{P}(S_t, L_t^1, L_t^2 \mid C_t, \gamma_{1:t-1}^1, \gamma_{1:t-1}^2)$.
- Dynamic program: $V_{T+1}(\pi) = 0$ and

$$V_t(b) = \min_{\gamma_t^1, \gamma_t^2} \{\mathbb{E}[c_t(S_t, A_t^1, A_t^2) + V_{t+1}(B_+) \mid b_t, \gamma_t^1, \gamma_t^2]\}.$$

The general common-info approach



- n controllers with general info structure $\{I_t^i\}_{i=1}^n$.

Implications and impact

- Subsumes many existing results (. . .)
- New results on sufficient statistics and DP for specific models (control sharing, mean-field sharing, NCS, and others)
- Common-information based refinements of Nash equilibrium in dynamic games with asymmetric information

Main Result

- **Information state**: for C_t: $b_t = \mathbb{P}(S_t, L_t^1, L_t^2 \mid C_t, \gamma^1_{1:t-1}, \gamma^2_{1:t-1})$.
- **Dynamic program**: $V_{T+1}(\pi) = 0$ and

$$V_t(b) = \min_{\gamma_t^1, \gamma_t^2} \{\mathbb{E}[c_t(S_t, A_t^1, A_t^2) + V_{t+1}(B_+) \mid b_t, \gamma_t^1, \gamma_t^2]\}.$$

Common information resolves conceptual difficulties in decentralized control.

Common information resolves conceptual difficulties in decentralized control.

Hey, but this is an RL workshop!

Learning in dynamic teams

Implications of common-info approach

- Converts planning in multi-agent teams to a POMDP
- In the learning setting, use your favorite RL algo for POMDP at the coordinator (offline traning) or each agent's local copy of the coordinator (online training)
- Beautiful theory ... doesn't work in practice.
- Too complicated. The action space is too large.

Learning in dynamic teams

Implications of common-info approach

- Converts planning in multi-agent teams to a POMDP
- In the learning setting, use your favorite RL algo for POMDP at the coordinator (offline traning) or each agent's local copy of the coordinator (online training)
- Beautiful theory … doesn't work in practice.
- Too complicated. The action space is too large.

Practical MARL algorithms

- Many SOTA MARL algos build on the common-info approach

  BAD (Bayesian action decoder), SOTA on Hannabi
  CAPI (cooperative approximate policy iteration), SOTA on Tiny-Bridge
  . . .

Learning in dynamic teams

Implications of common-info approach

- Converts planning in multi-agent teams to a POMDP
- In the learning setting, use your favorite RL algo for POMDP at the coordinator (offline traning) or each agent's local copy of the coordinator (online training)
- Beautiful theory … doesn't work in practice.
- Too complicated. The action space is too large.

Practical MARL algorithms

- Many SOTA MARL algos build on the common-info approach

  BAD (Bayesian action decoder), SOTA on Hannabi

  CAPI (cooperative approximate policy iteration), SOTA on Tiny-Bridge

  …

But no theory! How do we develop RL theory MARL?

Tentative Roadmap for MARL Theory

Step 1
RL for POMDPs

- Simplest "MARL" environment. Theory still lacking.
- We have recent results that resolve key conceptual challenges
- Could generalize to MARL using common-info approach

Tentative Roadmap for MARL Theory

Step 1
RL for POMDPs

- Simplest "MARL" environment. Theory still lacking.
- We have recent results that resolve key conceptual challenges
- Could generalize to MARL using common-info approach

Step 2
Centralized vs decentralized training

- Most MARL algos use centralized training.
- Some recent preliminary results for analysis of centralized training.
- Some empirical results on decentralized training.

Tentative Roadmap for MARL Theory

Step 1
RL for POMDPs

- Simplest "MARL" environment. Theory still lacking.
- We have recent results that resolve key conceptual challenges
- Could generalize to MARL using common-info approach

Step 2
Centralized vs decentralized training

- Most MARL algos use centralized training.
- Some recent preliminary results for analysis of centralized training.
- Some empirical results on decentralized training.

Next Steps

- Credit assignment (among agents)
- Agents helping each other to learning

How are we doing on time?

Approximate Information States for POMDPs

Key solution concept: Information state

Informally, an information state is a compression of information which is sufficient for performance evaluation and predicting itself.

Key solution concept: Information state

> Informally, an information state is a compression of information which is sufficient for performance evaluation and predicting itself.

Historical overview

- **Old concept**. May be viewed as as generalization of the notion of state (Nerode, 1958).
- Informal definitions given in Kwakernaak (1965), Bohlin (1970), Davis and Varaiya (1972), Kumar and Varaiya (1986) but no formal analysis.
- Related to but different from concepts such bisimulation, predictive state representations (PSR), and ε-machines.

Information state: Definition

Given a state space $\mathcal{Z}$, an INFORMATION STATE GENERATOR is a tuple of

- history compression functions $\{\sigma_t \colon \mathcal{H}_t \to \mathcal{Z}\}_{t \geqslant 1}$
- reward function $\hat{r} : \mathcal{Z} \times \mathcal{A} \to \mathbb{R}$
- transition kernel $\hat{P} : \mathcal{Z} \times \mathcal{A} \to \Delta(\mathcal{Z})$

which satisfies two properties:

Information state: Definition

Given a state space $\mathcal{Z}$, an INFORMATION STATE GENERATOR is a tuple of

- history compression functions $\{\sigma_t : \mathcal{H}_t \to \mathcal{Z}\}_{t \geqslant 1}$
- reward function $\hat{r} : \mathcal{Z} \times \mathcal{A} \to \mathbb{R}$
- transition kernel $\widehat{P} : \mathcal{Z} \times \mathcal{A} \to \Delta(\mathcal{Z})$

which satisfies two properties:

(P1) The reward function $\hat{r}$ is sufficient for performance evaluation:

$$\mathbb{E}[R_t \mid H_t = h_t, A_t = a_t] = \hat{r}(\sigma_t(h_t), a_t).$$

Information state: Definition

Given a state space $\mathcal{Z}$, an INFORMATION STATE GENERATOR is a tuple of

- history compression functions $\{\sigma_t : \mathcal{H}_t \to \mathcal{Z}\}_{t \geqslant 1}$
- reward function $\hat{r} : \mathcal{Z} \times \mathcal{A} \to \mathbb{R}$
- transition kernel $\widehat{P} : \mathcal{Z} \times \mathcal{A} \to \Delta(\mathcal{Z})$

which satisfies two properties:

(P1) The reward function $\hat{r}$ is sufficient for performance evaluation:

$$\mathbb{E}[R_t \mid H_t = h_t, A_t = a_t] = \hat{r}(\sigma_t(h_t), a_t).$$

(P2) The transition kernel $\widehat{P}$ is sufficient for predicting the info state:

$$\mathbb{P}(Z_{t+1} \in B \mid H_t = h_t, A_t = a_t) = \widehat{P}(B \mid \sigma_t(h_t), a_t).$$

Information state: Key result

An information state **always** leads to a dynamic programming decomposition.

Information state: Key result

An information state **always** leads to a dynamic programming decomposition.

Let $\{Z_t\}_{t \geqslant 1}$ be any information state process. Let $\hat{V}$ be the fixed point of:

$$\hat{V}(z) = \max_{a \in \mathcal{A}} \Big\{ \hat{r}(z, a) + \gamma \int_{\mathcal{Z}} \hat{V}(z_+) \hat{P}(dz_+ | z, a) \Big\}$$

Let $\pi^*(z)$ denote the arg max of the RHS. **Then, the policy $\pi = (\pi_t)_{t \geqslant 1}$ given by $\pi_t = \pi^* \circ \sigma_t$ is optimal.**

Examples of information state

Markov decision processes (MDP)

Current state S_t is an info state

POMDP

Belief state is an info state

Examples of information state

Markov decision processes (MDP)

Current state S_t is an info state

MDP with delayed observations

$(S_{t-\delta+1}, A_{t-\delta+1:t-1})$ is an info state

POMDP

Belief state is an info state

POMDP with delayed observations

$(\mathbb{P}(S_{t-\delta} | Y_{1:t-\delta}, A_{1:t-\delta}), A_{t-\delta+1:t-1})$
is info state

Examples of information state

Markov decision processes (MDP)

Current state S_t is an info state

MDP with delayed observations

$(S_{t-\delta+1}, A_{t-\delta+1:t-1})$ is an info state

POMDP

Belief state is an info state

POMDP with delayed observations

$(\mathbb{P}(S_{t-\delta} | Y_{1:t-\delta}, A_{1:t-\delta}), A_{t-\delta+1:t-1})$
is info state

Linear Quadratic Gaussian (LQG)

The state estimate $\mathbb{E}[S_t | H_t]$ is an info state

Machine Maintenance

(τ, S^{+}_{τ}) is info state,
where τ is the time of last maintenance

And now to Approximate Information States . . .

Main idea

- Info state is defined in terms of two properties (P1) & (P2).
- An AIS is a process which satisfies these approximately

And now to Approximate Information States . . .

Main idea

- Info state is defined in terms of two properties (P1) & (P2).
- An AIS is a process which satisfies these approximately

- Show that AIS always leads to approx. DP
- Recover (and improve up on) many existing results

Approximate Information state: Definition

An (ε, δ)-APPROXIMATE INFORMATION STATE (AIS) generator is a tuple $(\sigma_t, \hat{r}, \widehat{P})$ which approximately satisfies (P1) and (P2):

Approximate Information state: Definition

An (ε, δ)-APPROXIMATE INFORMATION STATE (AIS) generator is a tuple $(\sigma_t, \hat{r}, \hat{P})$ which approximately satisfies (P1) and (P2):

(AP1) $\hat{r}$ is sufficient for approximate performance evaluation:

$$\left|\mathbb{E}[R_t \mid H_t = h_t, A_t = a_t] - \hat{r}(\sigma_t(h_t), a_t)\right| \leqslant \varepsilon$$

Approximate Information state: Definition

An (ε, δ)-APPROXIMATE INFORMATION STATE (AIS) generator is a tuple $(\sigma_t, \hat{r}, \hat{P})$ which approximately satisfies (P1) and (P2):

(AP1) $\hat{r}$ is sufficient for approximate **performance evaluation:**

$$\left| \mathbb{E}[R_t \mid H_t = h_t, A_t = a_t] - \hat{r}(\sigma_t(h_t), a_t) \right| \leqslant \varepsilon$$

(AP2) $\hat{P}$ is sufficient for approximately **predicting next AIS**:

$$d_{\mathfrak{F}}\Big(\mathbb{P}(Z_{t+1} = \cdot \mid H_t = h_t, A_t = a_t), \hat{P}(\cdot \,|\sigma_t(h_t), a_t)\Big) \leqslant \delta$$

Approximate Information state: Definition

An (ε, δ)-APPROXIMATE INFORMATION STATE (AIS) generator is a tuple $(\sigma_t, \hat{r}, \hat{P})$ which approximately satisfies (P1) and (P2):

(AP1) $\hat{r}$ is sufficient for approximate **performance evaluation:**

$$\left|\mathbb{E}[R_t \mid H_t = h_t, A_t = a_t] - \hat{r}(\sigma_t(h_t), a_t)\right| \leqslant \varepsilon$$

(AP2) $\hat{P}$ is sufficient for approximately **predicting next AIS:**

$$d_{\mathfrak{F}}\big(\mathbb{P}(Z_{t+1} = \cdot \mid H_t = h_t, A_t = a_t), \hat{P}(\cdot \mid \sigma_t(h_t), a_t)\big) \leqslant \delta$$

Results depend on the choice of **metric on probability spaces**

Examples of AIS

Example 1: Robustness to model mismatch in MDPs

Real-world model

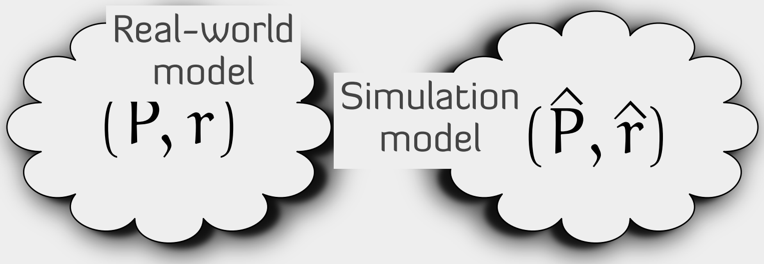
(P, r)

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

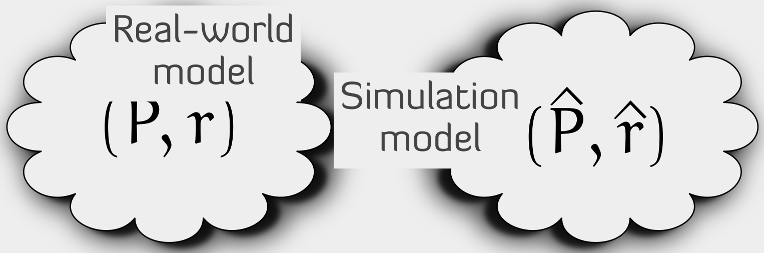


What is the loss in performance if we choose a policy using the simulation model and use it in the real world?

Model mismatch as an AIS

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

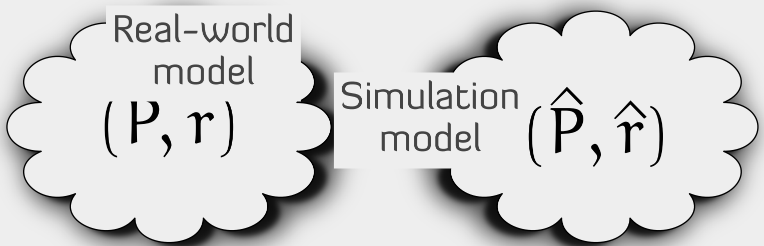
- $(\text{Identity}, \hat{P}, \hat{r})$ is an (ε, δ)-AIS with $\varepsilon = \sup_{s,a} \big| r(s,a) - \hat{r}(s,a) \big|$ and $\delta_{\mathfrak{F}} = \sup_{s,a} d_{\mathfrak{F}}(P(\cdot|s,a), \hat{P}(\cdot|s,a))$.

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

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

Recover bounds of Müller (1997).

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.
- Asadi, Misra, Littman, "Lipscitz continuity in model-based reinfocement learning," ICML 2018.

Model mismatch as an AIS

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

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

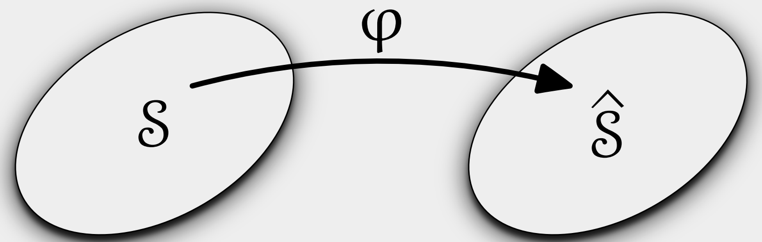
Recover bounds of Müller (1997).

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

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

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

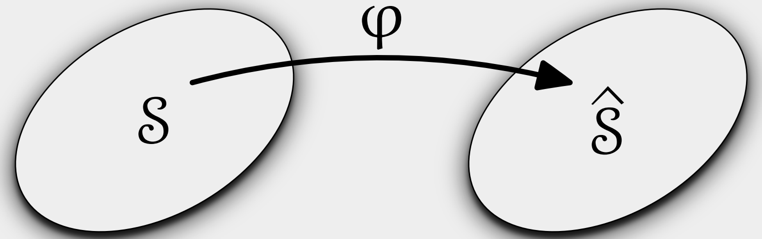
Example 2: Feature abstraction in MDPs



$(\widehat{P}, \widehat{r})$ 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



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

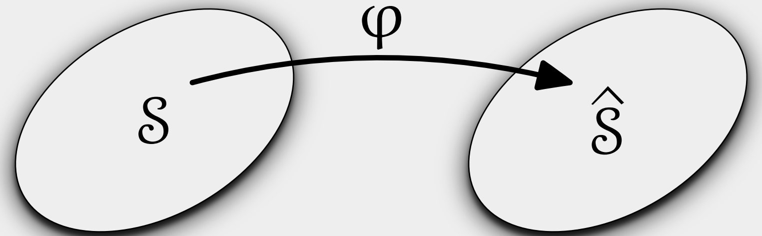
What is the loss in performance if we choose a policy using the abstract model and use it in the original model?

Feature abstraction as AIS

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

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

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 $\varepsilon = \sup_{s,a} \big| r(s,a) - \hat{r}(\varphi(s), a) \big|$

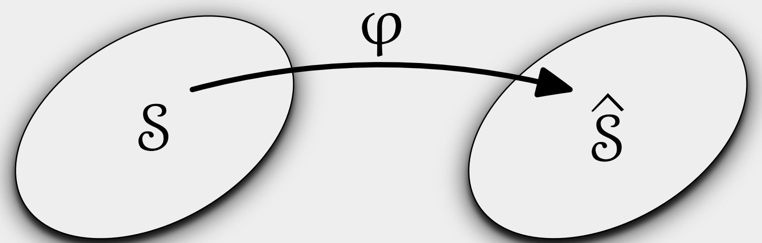
  and $\delta_{\mathfrak{F}} = \sup_{s,a} d_{\mathfrak{F}}(P(\varphi^{-1}(\cdot)|s,a), \hat{P}(\cdot|\varphi(s), a)$.

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

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

Improve bounds of Abel et al. (2016)

Example 2: Feature abstraction in MDPs



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

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.

Feature abstraction as AIS

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

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

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

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

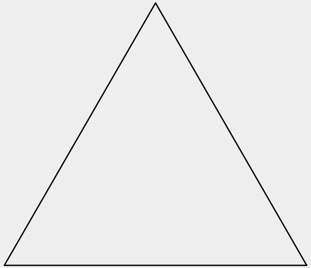
Improve bounds of Abel et al. (2016)

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

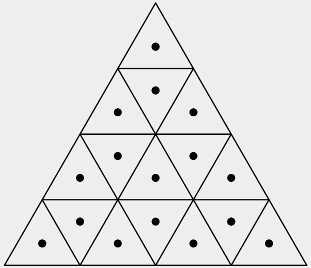
$$V(s) - V^{\pi}(s) \leqslant \frac{2\varepsilon}{1-\gamma} + \frac{2\gamma \delta_{\mathfrak{F}} \|\hat{V}\|_{\text{Lip}}}{(1-\gamma)^2}$$

Recover bounds of Gelada et al. (2019).

Example 3: Belief approximation in POMDPs



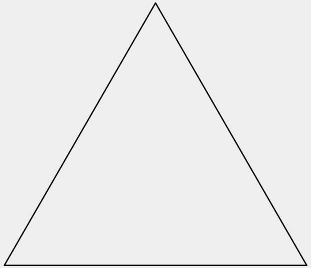
Belief space



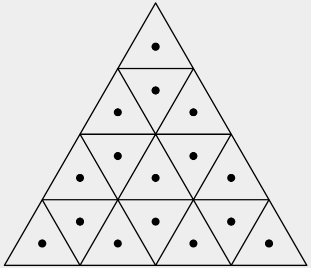
Quantized beliefs

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



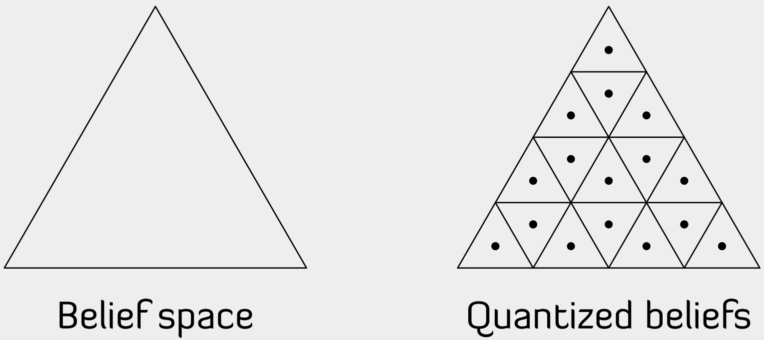
Quantized beliefs

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.

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

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

Thus, the notion of AIS unifies many of the approximation results in the literature, both for MDPs and POMDPs.

Hey, this is an RL workshop remember

From approximation bounds to reinforcement learning . . .

Main idea

- AIS is defined in terms of two losses ε and δ.
- Minimizing ε and δ will minimize the AIS approximation loss.

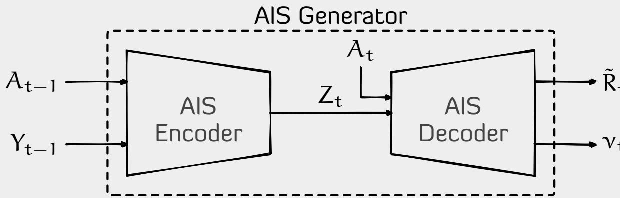
From approximation bounds to reinforcement learning . . .

Main idea

- AIS is defined in terms of two losses ε and δ.
- Minimizing ε and δ will minimize the AIS approximation loss.

- Use $\lambda\varepsilon^2 + (1-\lambda)\delta^2$ as surrogate loss for the AIS generator
- . . . and combine it with standard actor-critic algorithm using multi-timescale stochastic approximation.

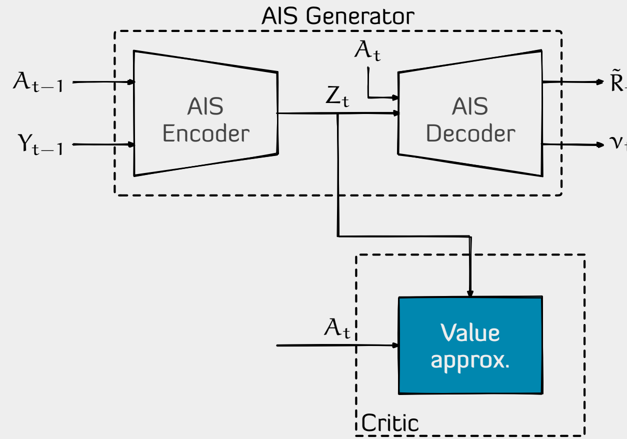
Reinforcement learning setup



AIS Generator

- Use LSTM for $\sigma_t : \mathcal{H}_t \to \mathcal{Z}$ and a NN for functions $\hat{r}$ and $\widehat{P}$.
- Use $\lambda(\tilde{R}_t - R_t)^2 + (1-\lambda) d_{\mathfrak{F}}(\mu_t, \nu_t)^2$ as surrogate loss.
- We show that $\nabla d_{\mathfrak{F}}(\mu_t, \nu_t)^2$ can be computed efficiently for Wasserstein distance and MMD.

Reinforcement learning setup



AIS Generator

- Use LSTM for $\sigma_t \colon \mathcal{H}_t \to \mathcal{Z}$ and a NN for functions $\hat{r}$ and $\hat{P}$.
- Use $\lambda(\tilde{R}_t - R_t)^2 + (1-\lambda) d_{\mathfrak{F}}(\mu_t, \nu_t)^2$ as surrogate loss.
- We show that $\nabla d_{\mathfrak{F}}(\mu_t, \nu_t)^2$ can be computed efficiently for Wasserstein distance and MMD.

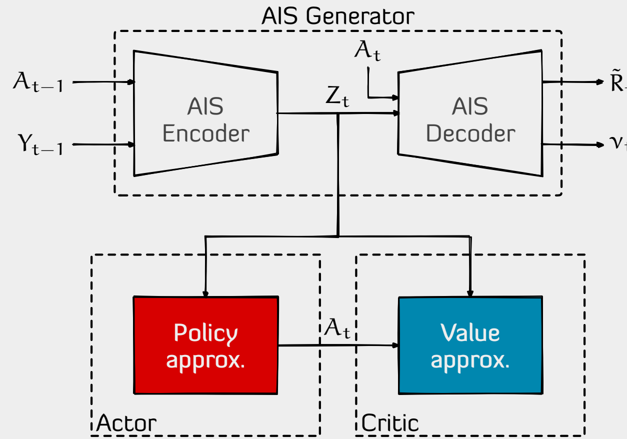
Value approximator

- Use a NN to approx. action-value function $Q \colon \mathcal{Z} \times \mathcal{A} \to \mathbb{R}$.
- Update the parameters to minimize temporal difference loss

Reinforcement learning setup

AIS Generator

- Use LSTM for $\sigma_t : \mathcal{H}_t \to \mathcal{Z}$ and a NN for functions $\hat{r}$ and $\hat{P}$.
- Use $\lambda(\tilde{R}_t - R_t)^2 + (1-\lambda) d_{\mathfrak{F}}(\mu_t, \nu_t)^2$ as surrogate loss.
- We show that $\nabla d_{\mathfrak{F}}(\mu_t, \nu_t)^2$ can be computed efficiently for Wasserstein distance and MMD.



Policy approximator

- Use a NN to approx. policy $\pi : \mathcal{Z} \to \Delta(\mathcal{A})$.
- Use policy gradient theorem to efficiently compute $\nabla J(\pi)$.

Value approximator

- Use a NN to approx. action-value function $Q : \mathcal{Z} \times \mathcal{A} \to \mathbb{R}$.
- Update the parameters to minimize temporal difference loss