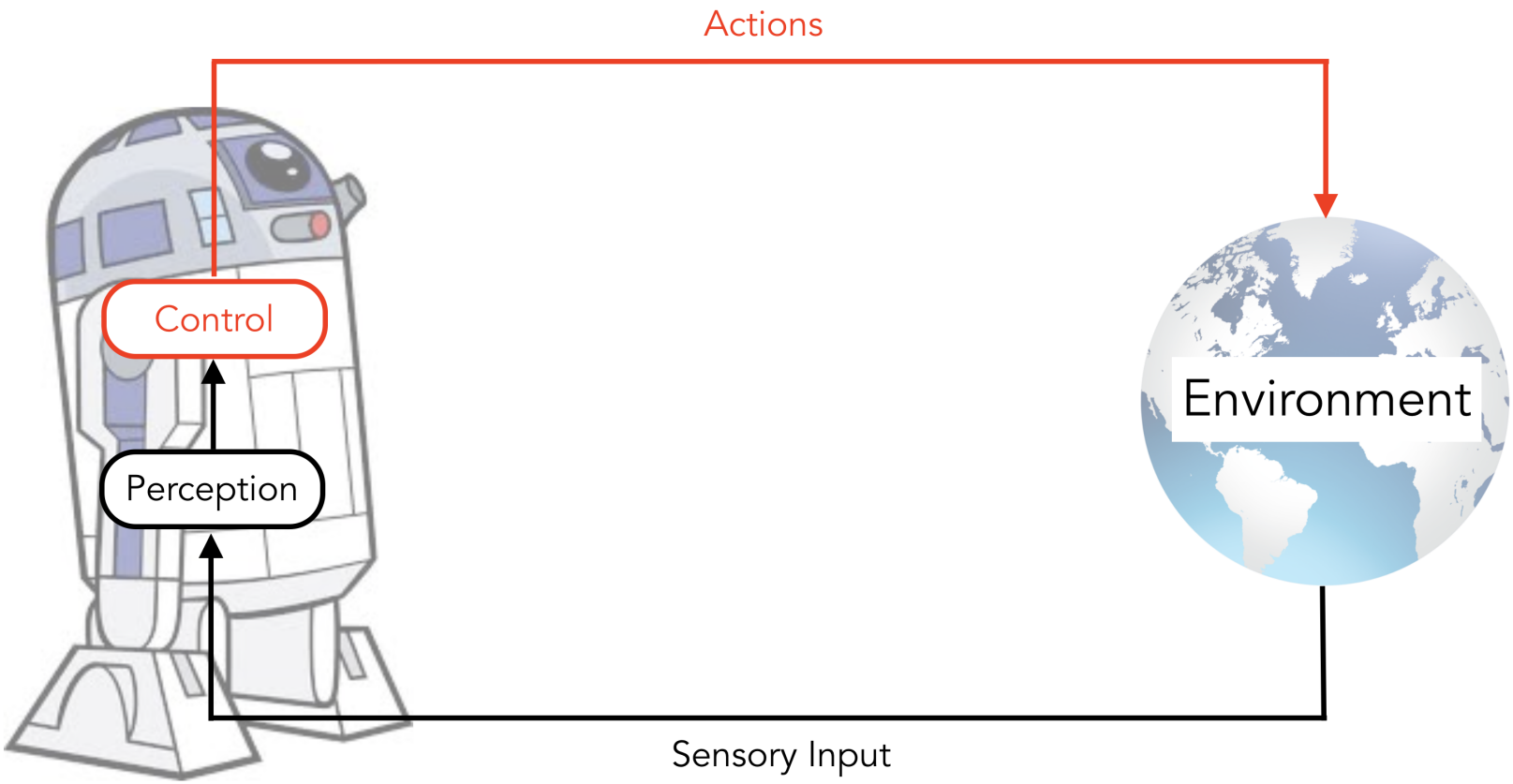
Factored Value Functions for Cooperative Multi-Agent Reinforcement Learning

Shimon Whiteson
Dept. of Computer Science
University of Oxford

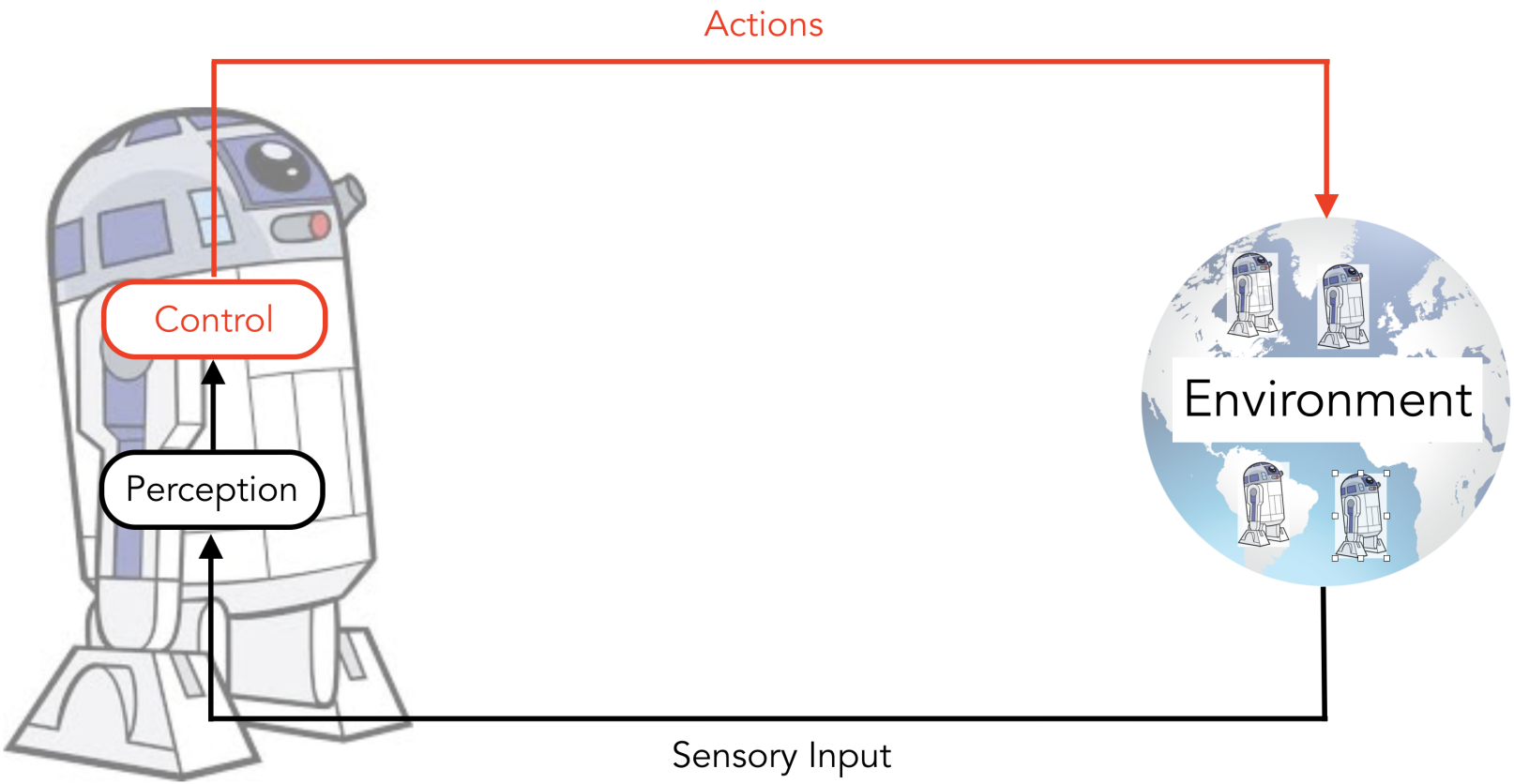
joint work with Jakob Foerster, Gregory Farquhar, Anuj Mahajan, Tabish Rashid, Mikayel Samvelyan, Christian Schroeder de Witt, Nantas Nardelli, Tim Rudner, Chia-Man Hung, and Phil Torr

October 23, 2020

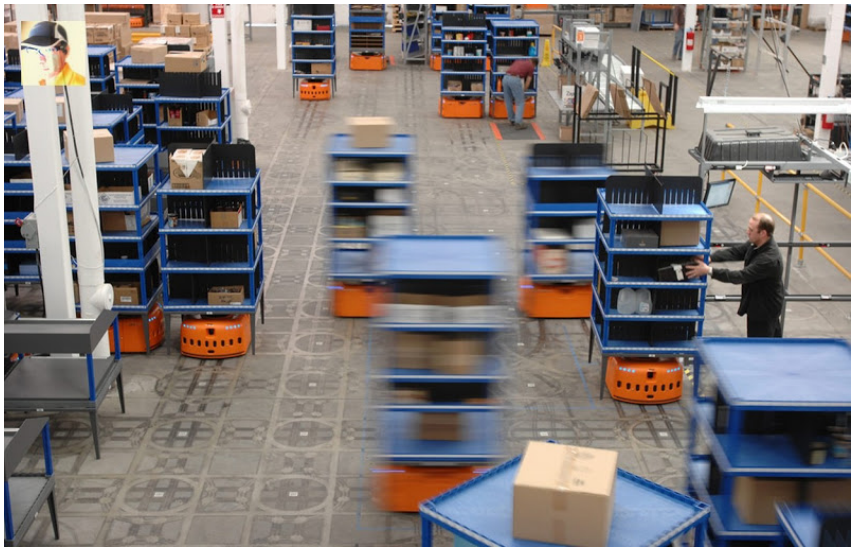
Single-Agent Paradigm



Multi-Agent Paradigm



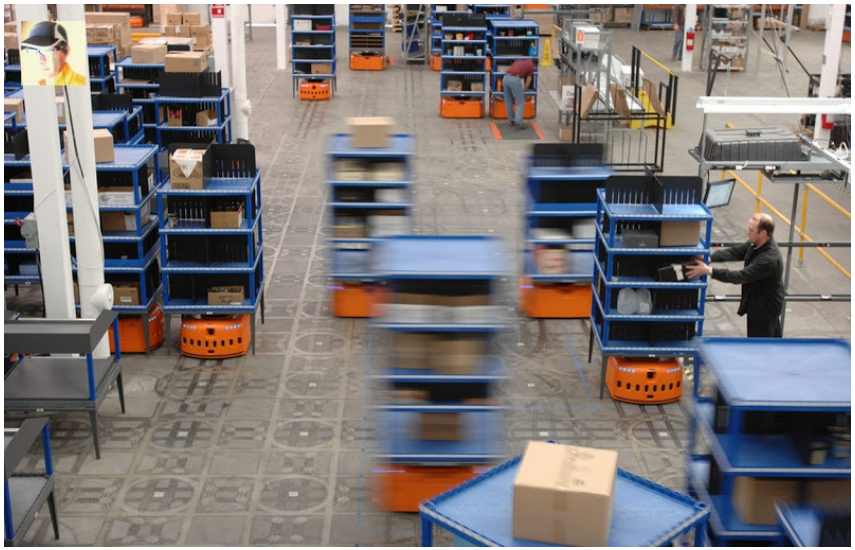
Multi-Agent Systems are Everywhere



Types of Multi-Agent Systems

- *Cooperative:*
  - Shared team reward
  - Coordination problem
- *Competitive:*
  - Zero-sum games
  - Individual opposing rewards
  - Minimax equilibria
- *Mixed:*
  - General-sum games
  - Nash equilibria
  - What is the question? [Shoham et al. 2007]

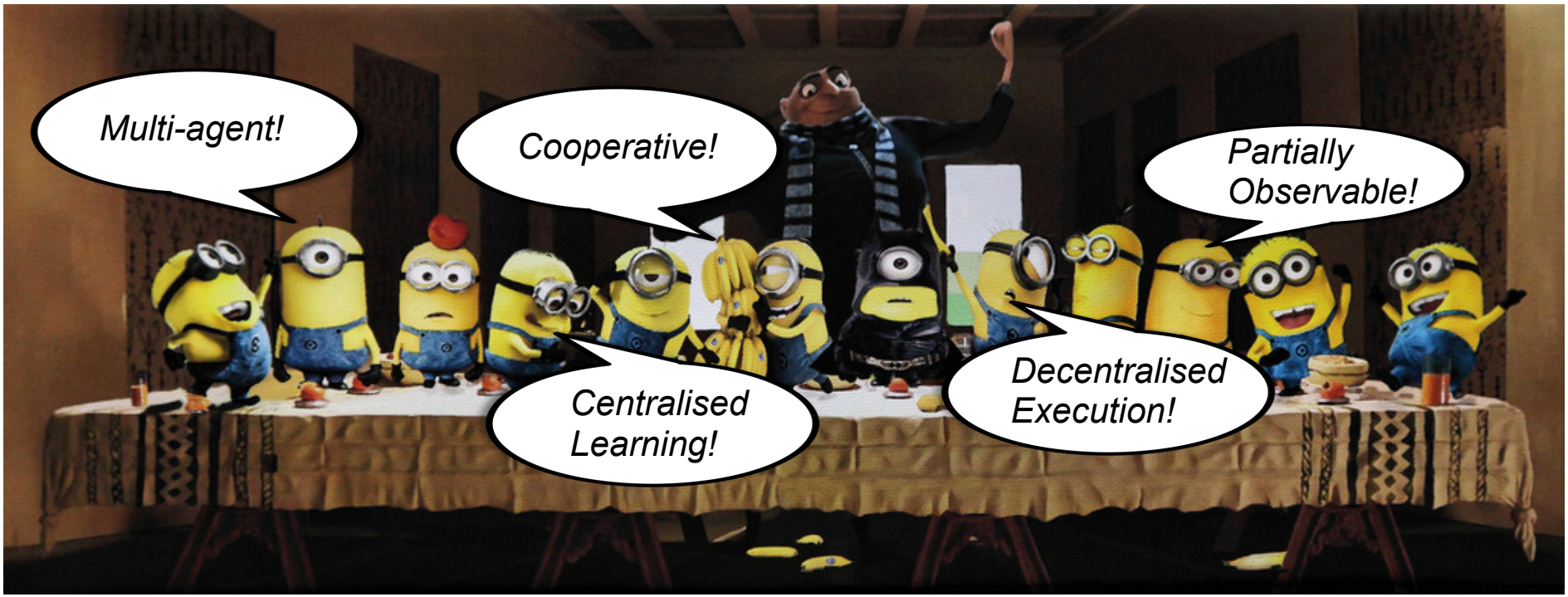
Coordination Problems are Everywhere



Multi-Agent RL Methods from WhiRL

- DIAL [Foerster et al. 2015]
- Multi-Agent Fingerprints [Foerster et al. 2017]
- COMA [Foerster et al. 2018]
- *QMIX [Rashid et al. 2018]*
- LOLA [Foerster et al. 2019]
- SOS [Letcher et al. 2019]
- MACKRL [Schroeder de Witt et al. 2019]
- MAVEN [Mahajan et al. 2019]
- WQMIX [Rashid et al. 2020]
- COMIX [Schroeder de Witt et al. 2020]

Setting



(Figure by Jakob Foerster)

Markov Decision Process

- Agent observes *state* $s \in S$ and selects an *action* $u \in U$
- State transitions: $P(s'|s,u) : S \times U \times S \rightarrow [0,1]$
- Receives *reward*: $r(s,u) : S \times U \rightarrow \mathbb{R}$
- Goal: maximise expected cumulative discounted *return*:

$$R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$$

- *Value functions* given policy $\pi(s,u)$:

$$V^{\pi}(s) = \mathbb{E}_{\pi}\left[R_t | s_t = s\right] \quad \text{and} \quad Q^{\pi}(s,u) = \mathbb{E}_{\pi}\left[R_t | s_t = s, u_t = u\right]$$

Multi-Agent MDP

- All agents see the global state s
- Individual actions: $u^a \in U$
- State transitions: $P(s'|s, \mathbf{u}) : S \times \mathbf{U} \times S \rightarrow [0, 1]$
- Shared team reward: $r(s, \mathbf{u}) : S \times \mathbf{U} \rightarrow \mathbb{R}$
- Equivalent to an MDP with a factored action space

Dec-POMDP

- Observation function: $O(s, a) : S \times A \rightarrow Z$
- Action-observation history: $\tau^a \in T \equiv (Z \times U)^*$
- Decentralised policies: $\pi^a(u^a|\tau^a) : T \times U \rightarrow [0, 1]$
- Natural decentralisation: communication and sensory constraints
- Artificial decentralisation: improve tractability
- Centralised learning of decentralised policies

Setting

CENTRALISED TRAINING OF DECENTRALISED CONTROL

The Predictability / Exploitation Dilemma

- Exploitation:
  - Maximising performance requires collecting reward
  - In a single-agent setting, this requires *exploiting* observations
- Predictability:
  - Dec-POMDP agents cannot explicitly communicate
  - Coordination requires *predictability*: "stick to the plan!"
  - Predictability can require ignoring private information

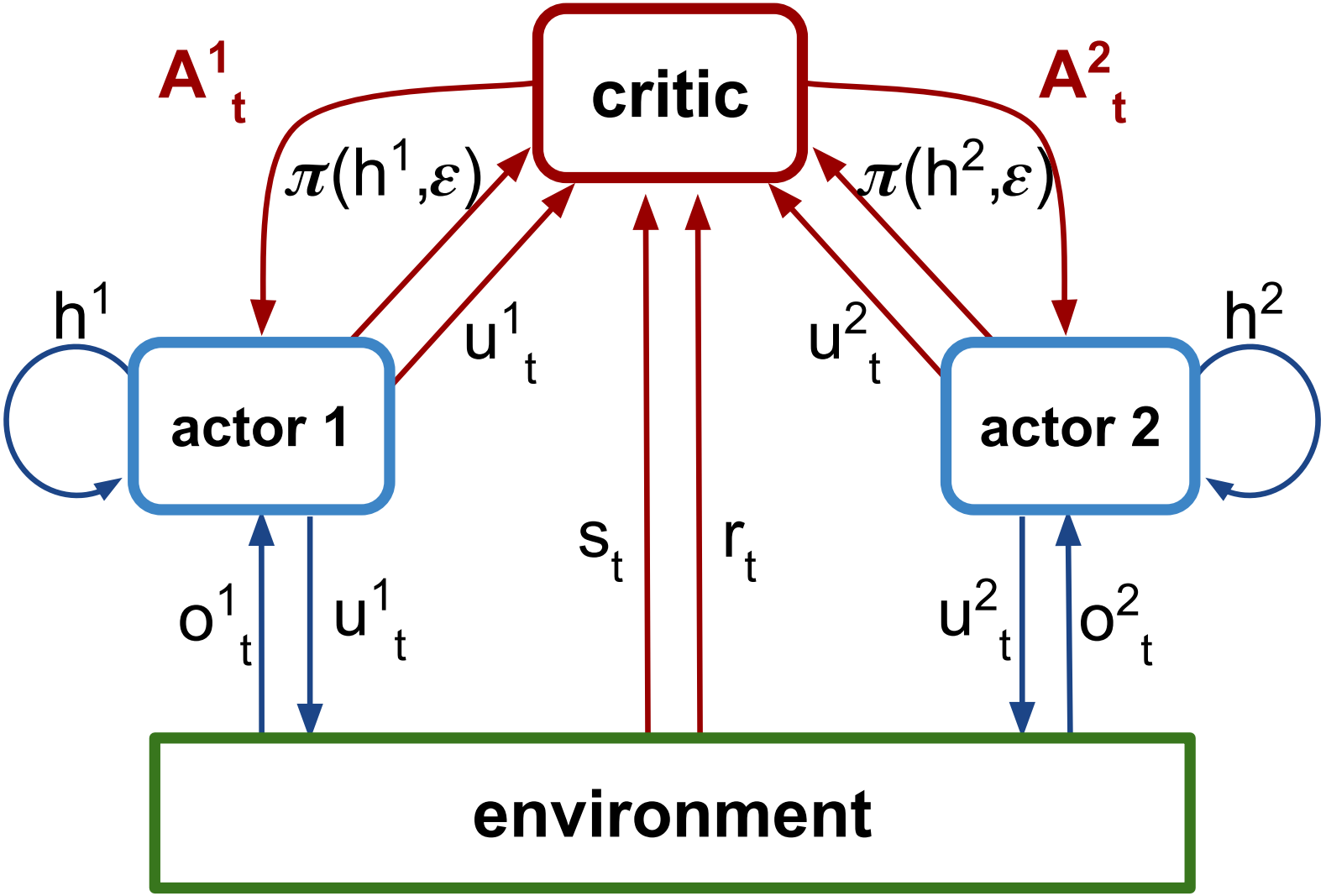
When does the benefit of exploiting private observations outweigh the cost in predictability?

Independent Learning

- Independent Q-learning [Tan 1993]
  - Each agent learns independently with its own Q-function
  - Treats other agents as part of the environment
- Independent actor-critic [Foerster et al. 2018]
  - Each agent learns independently with its own actor-critic
  - Treats other agents as part of the environment
- Speed learning with *parameter sharing*
  - Different inputs, including a, induce different behaviour
  - Still independent: value functions condition only on τ^a and u^a
- Limitations:
  - Nonstationary learning
  - Hard to learn to coordinate

Centralised Critics [Lowe et al. 2017; Foerster et al. 2018]

Centralised $V(s, \tau)$ or $Q(s, \tau, \mathbf{u}) \rightarrow$ hard greedification $\rightarrow$ actor-critic

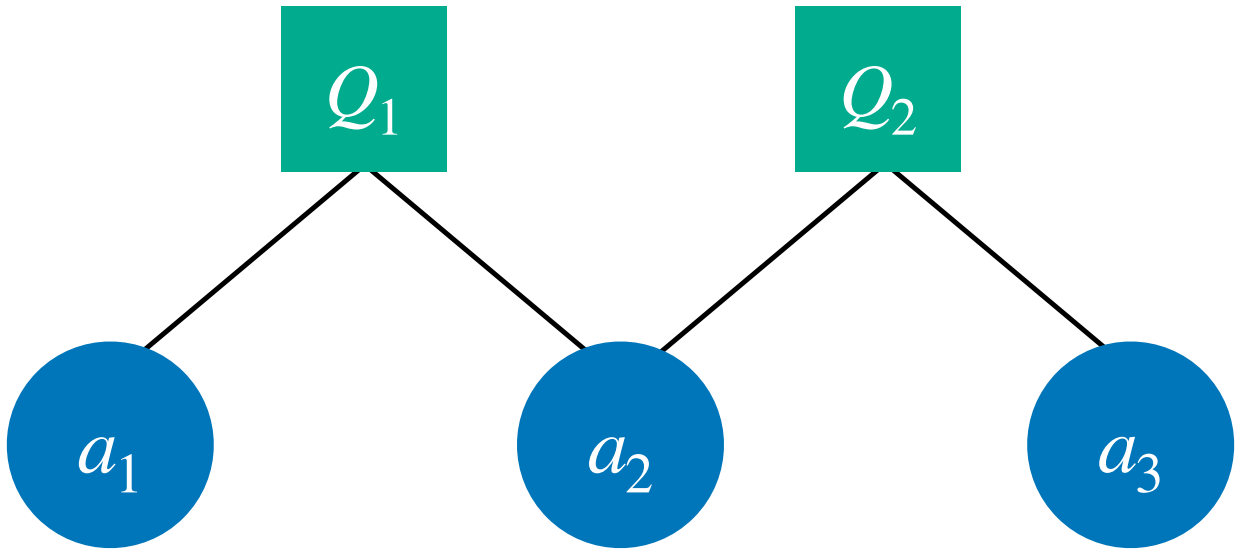


Factored Joint Value Functions

- *Factored value functions* [Guestrin et al. 2003] can improve scalability:

$$Q_{tot}(\boldsymbol{\tau}, \mathbf{u}; \boldsymbol{\theta}) = \sum_{e=1}^{E} Q_e(\tau^e, \mathbf{u}^e; \theta^e)$$

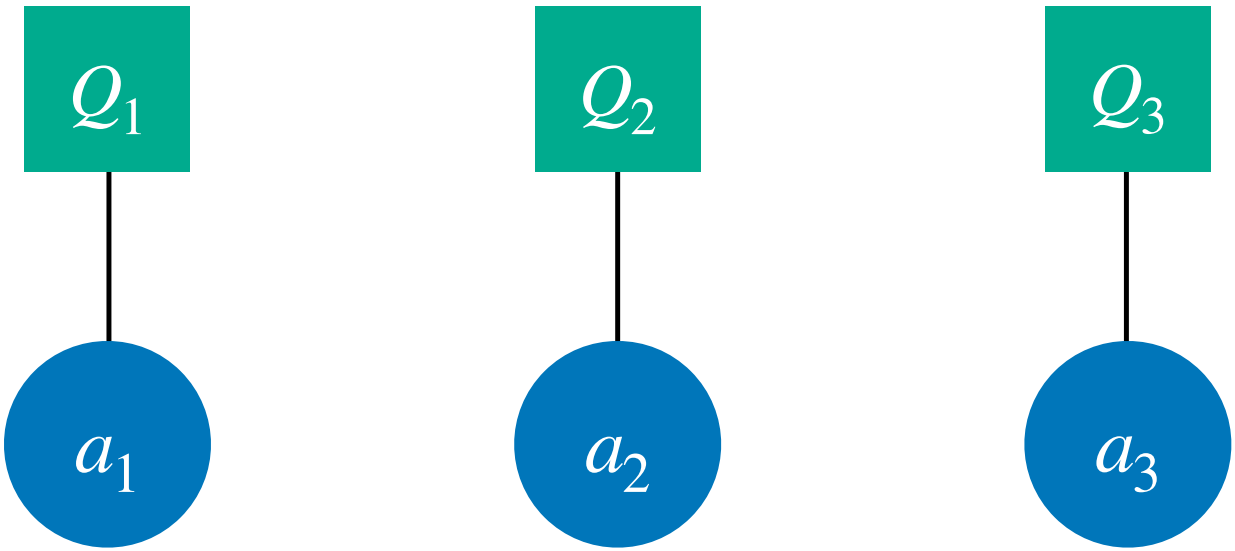
where each e indicates a subset of the agents



Value Decomposition Networks [Sunehag et al., 2017]

- Most extreme factorisation: one per agent:

$$Q_{tot}(\boldsymbol{\tau}, \mathbf{u}; \boldsymbol{\theta}) = \sum_{a=1}^{N} Q_a(\tau^a, u^a; \theta^a)$$



Decentralisability

- Added benefit of decentralising the max and arg max:

$$\max_{\mathbf{u}} Q_{tot}(\boldsymbol{\tau}, \mathbf{u}; \boldsymbol{\theta}) = \sum \max_{u^a} Q_a(\tau^a, u^a; \theta^a)$$

$$\arg\max_{\mathbf{u}} Q_{tot}(\boldsymbol{\tau}, \mathbf{u}; \boldsymbol{\theta}) = \begin{pmatrix} \arg\max_{u^1} Q_1(\tau^1, u^1; \theta^1) \\ \vdots \\ \arg\max_{u^n} Q_n(\tau^n, u^n; \theta^n) \end{pmatrix}$$

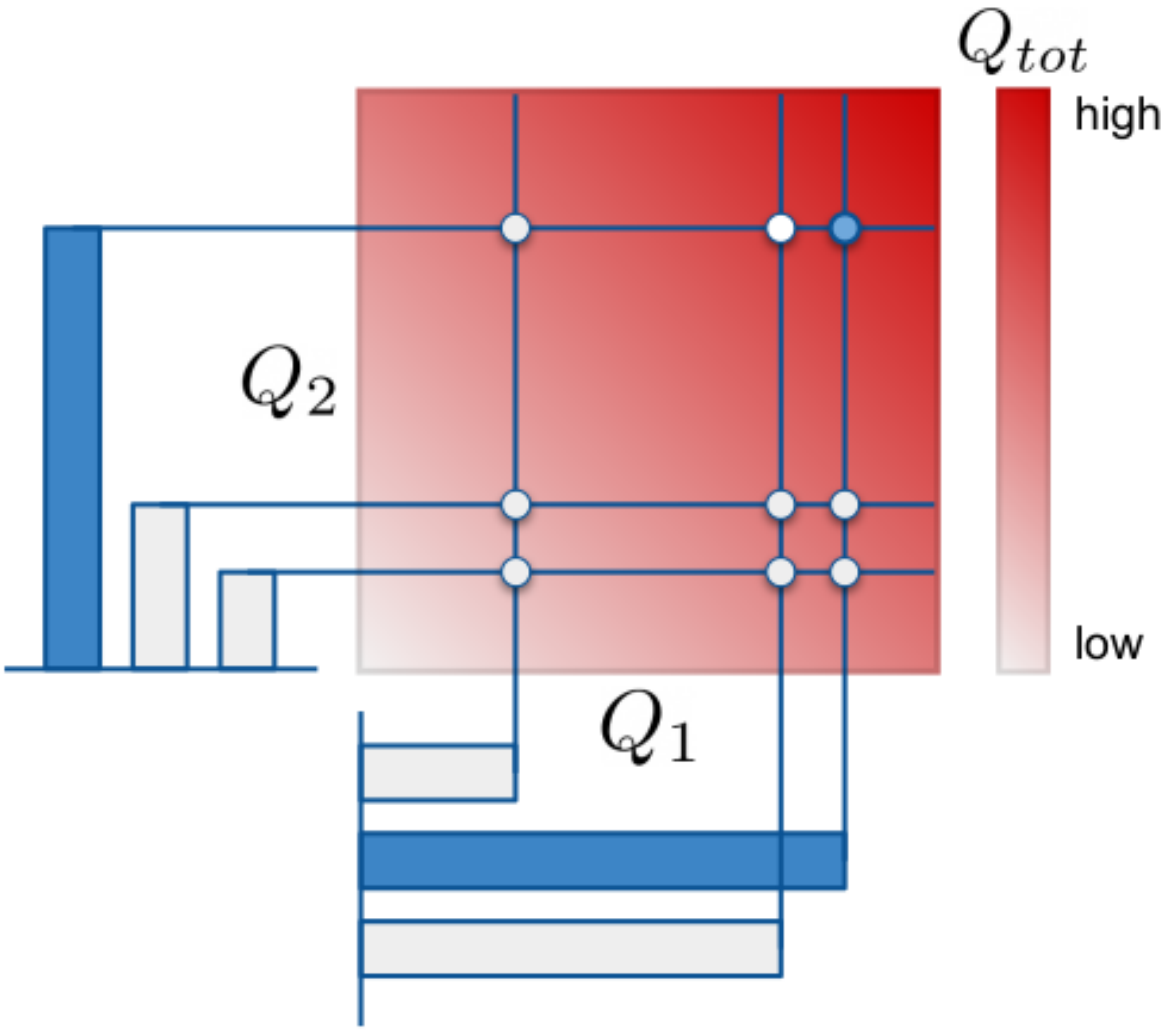
- No more hard greedification $\implies$ Q-learning, not actor-critic:

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^{b} \left[\left(y_i^{\text{tot}} - Q_{tot}(\boldsymbol{\tau}, \mathbf{u}; \boldsymbol{\theta}) \right)^2 \right],$$

$$y_i^{\text{tot}} = r_i + \gamma \max_{\mathbf{u}'} Q_{tot}(\boldsymbol{\tau}'_i, \mathbf{u}'; \boldsymbol{\theta}^-)$$

QMIX's Monotonicity Constraint

To decentralise max / arg max, it suffices to enforce: $\frac{\partial Q_{tot}}{\partial Q_a} \geq 0, \ \forall a \in A$



Representational Capacity

It'll never work: monotonic mixing still can't capture the benefit of coordination

Agent 1 \ Agent 2	A	B
A	0	1
B	1	2

linear & monotonic

VDN & QMIX

Agent 1 \ Agent 2	A	B
A	0	1
B	1	8

nonlinear & monotonic

Just QMIX

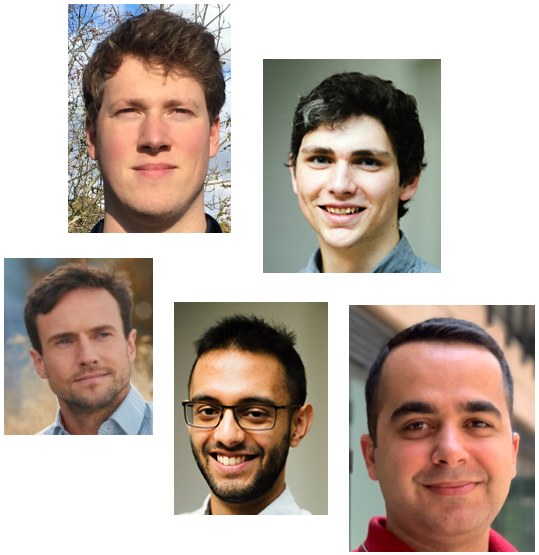
Agent 1 \ Agent 2	A	B
A	2	1
B	1	8

nonlinear & nonmonotonic

Neither

Bootstrapping

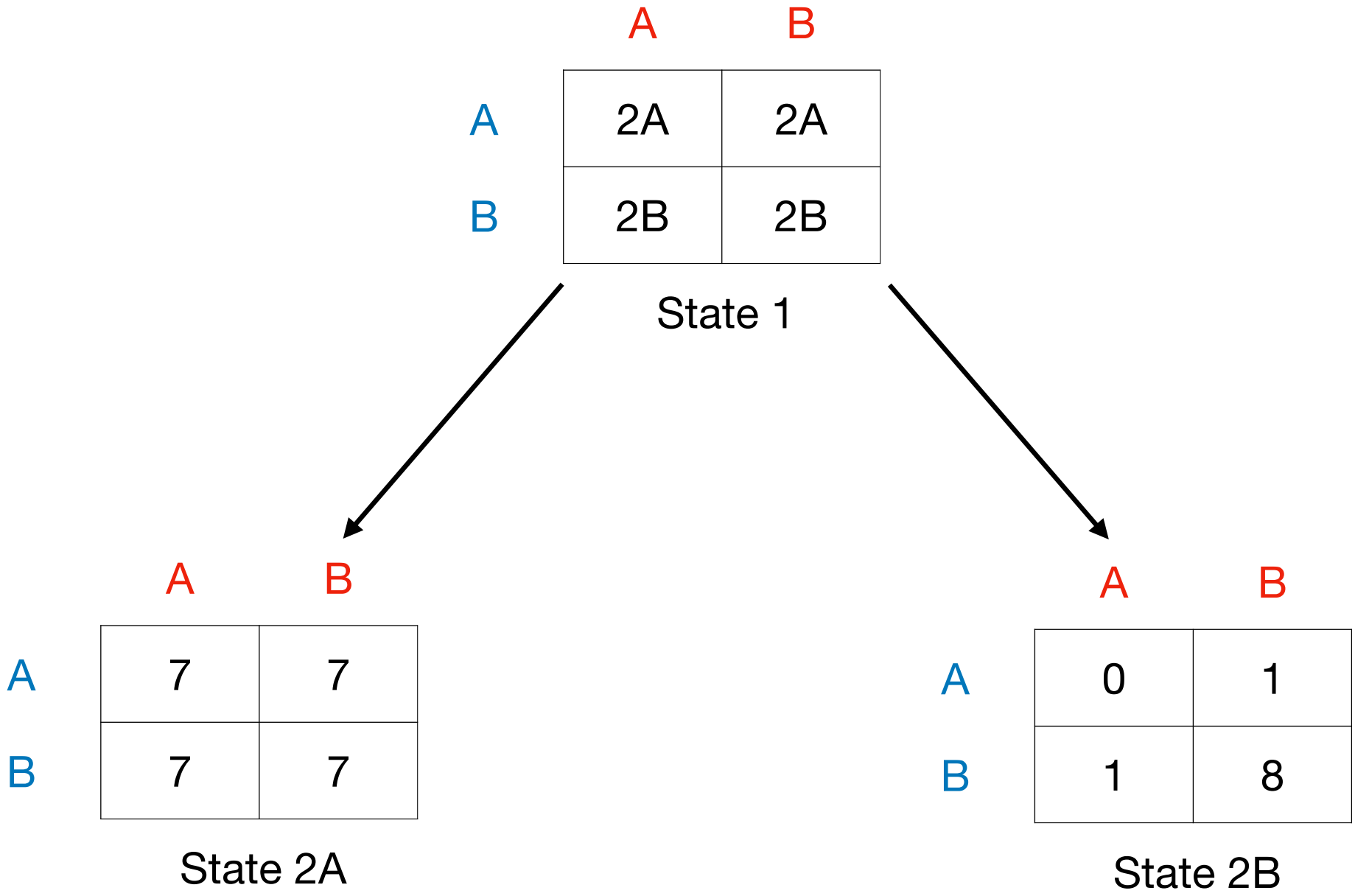
It matters because of *bootstrapping*



$$\mathcal{L}(\theta) = \sum_{i=1}^{b} \left[\left(y_i^{\text{tot}} - Q_{tot}(\boldsymbol{\tau}, \mathbf{u}, s; \theta) \right)^2 \right],$$

$$y_i^{\text{tot}} = r_i + \gamma \max_{\mathbf{u}'} Q_{tot}(\boldsymbol{\tau}'_i, \mathbf{u}', s'; \theta^-)$$

Two-Step Game



Two-Step Game Results

Ground Truth

State 1

	A	B
A	7	7
B	8	8

State 2A

	A	B
A	7	7
B	7	7

State 2B

	A	B
A	0	1
B	1	8

VDN

State 1

	A	B
A	6.94	6.94
B	6.35	6.36

State 2A

	A	B
A	6.99	7.02
B	6.99	7.02

State 2B

	A	B
A	-1.87	2.31
B	2.33	6.51

QMIX

State 1

	A	B
A	6.93	6.93
B	7.92	7.92

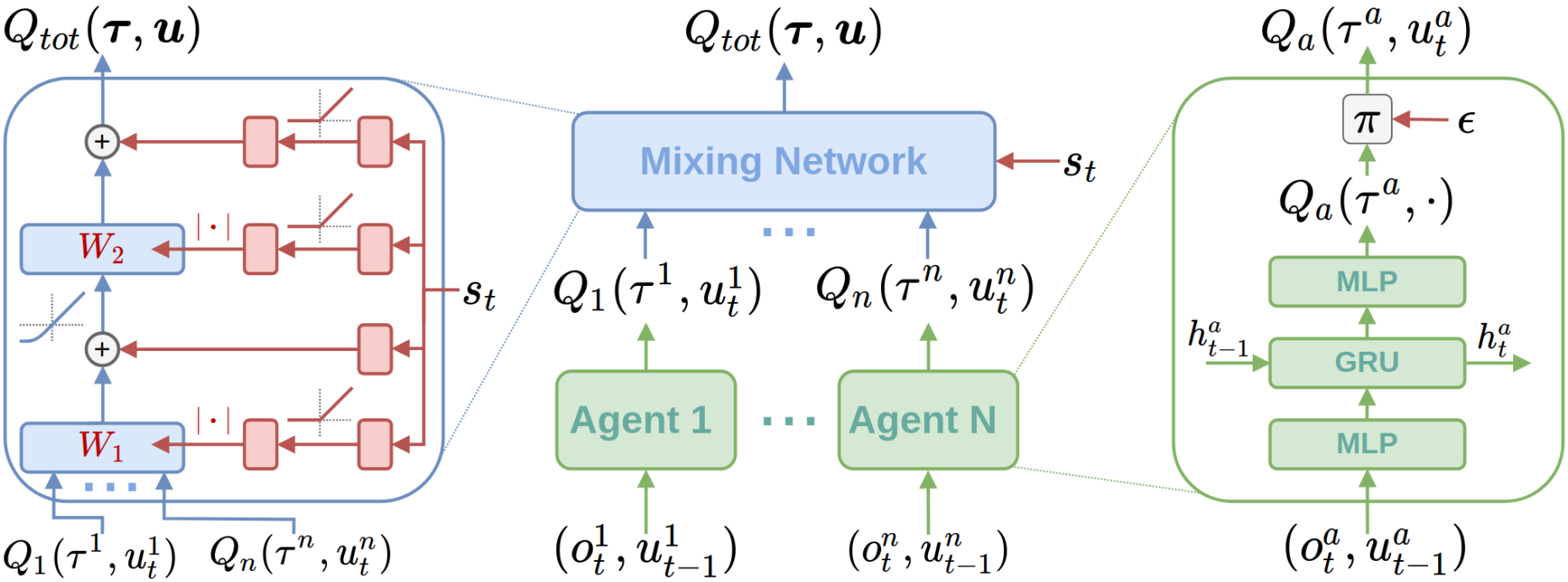
State 2A

	A	B
A	7.00	7.00
B	7.00	7.00

State 2B

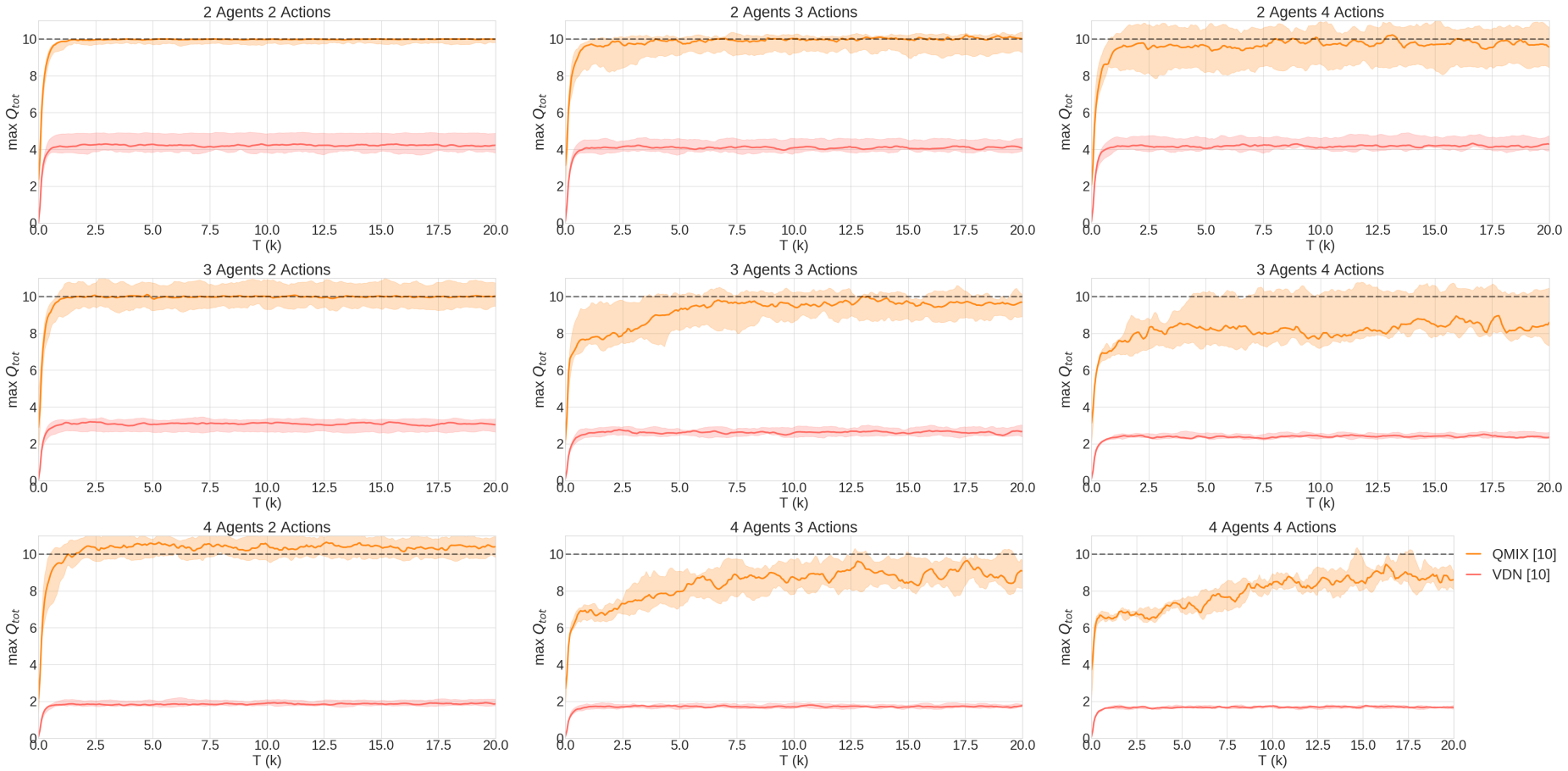
	A	B
A	0.00	1.00
B	1.00	8.00

QMIX [Rashid et al. 2018]



- Agent network: represents $Q_i(\tau^a, u^a; \theta^a)$
- Mixing network: represents $Q_{tot}(\boldsymbol{\tau})$ using nonnegative weights
- Hypernetwork: generates weights of hypernetwork based on global s

Random Matrix Games (The Students Were Right)



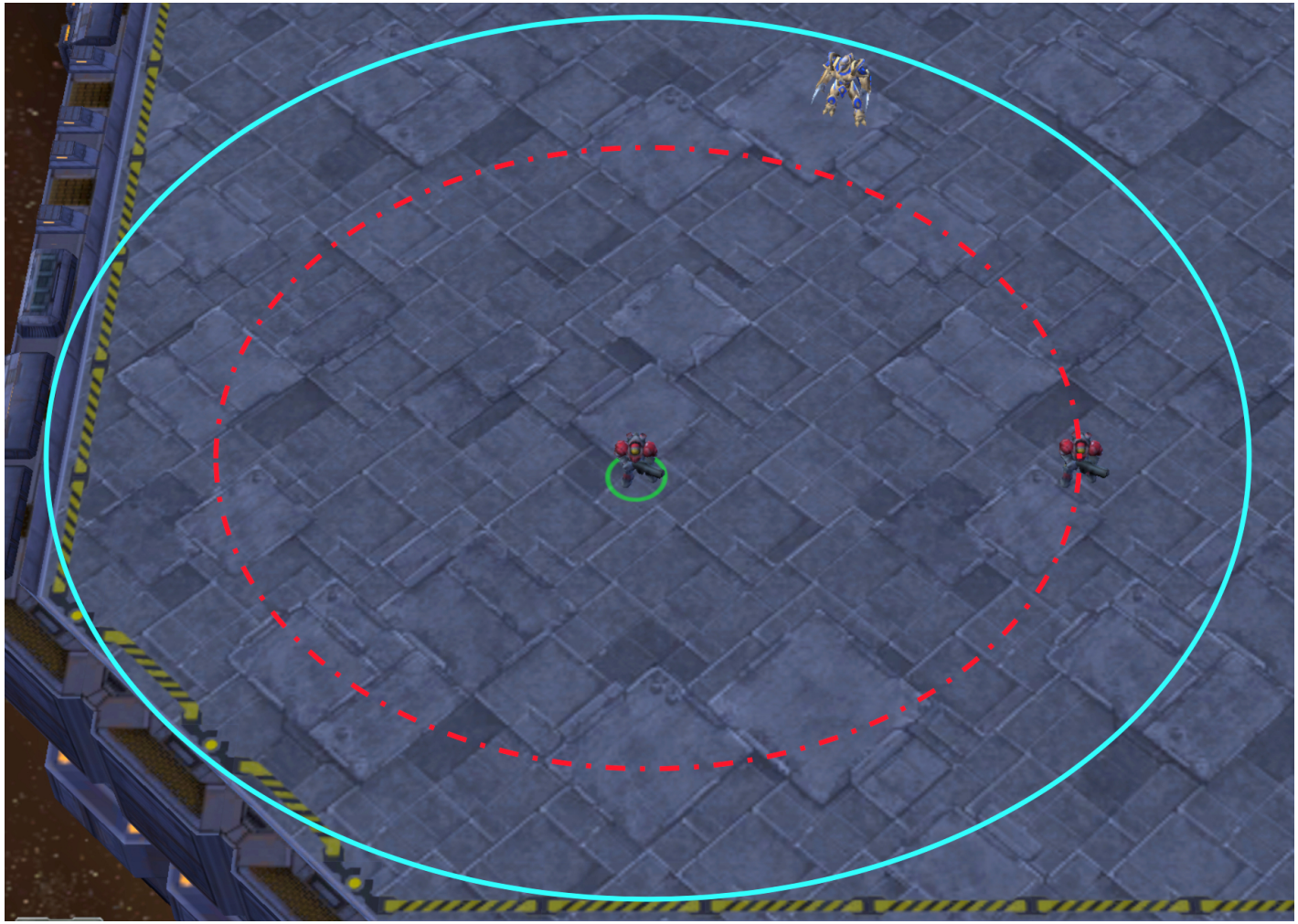
StarCraft Multi-Agent Challenge (SMAC)

[Samvelyan et al. 2019]

https://github.com/oxwhirl/smac

https://github.com/oxwhirl/pymarl

Partial Observability in SMAC

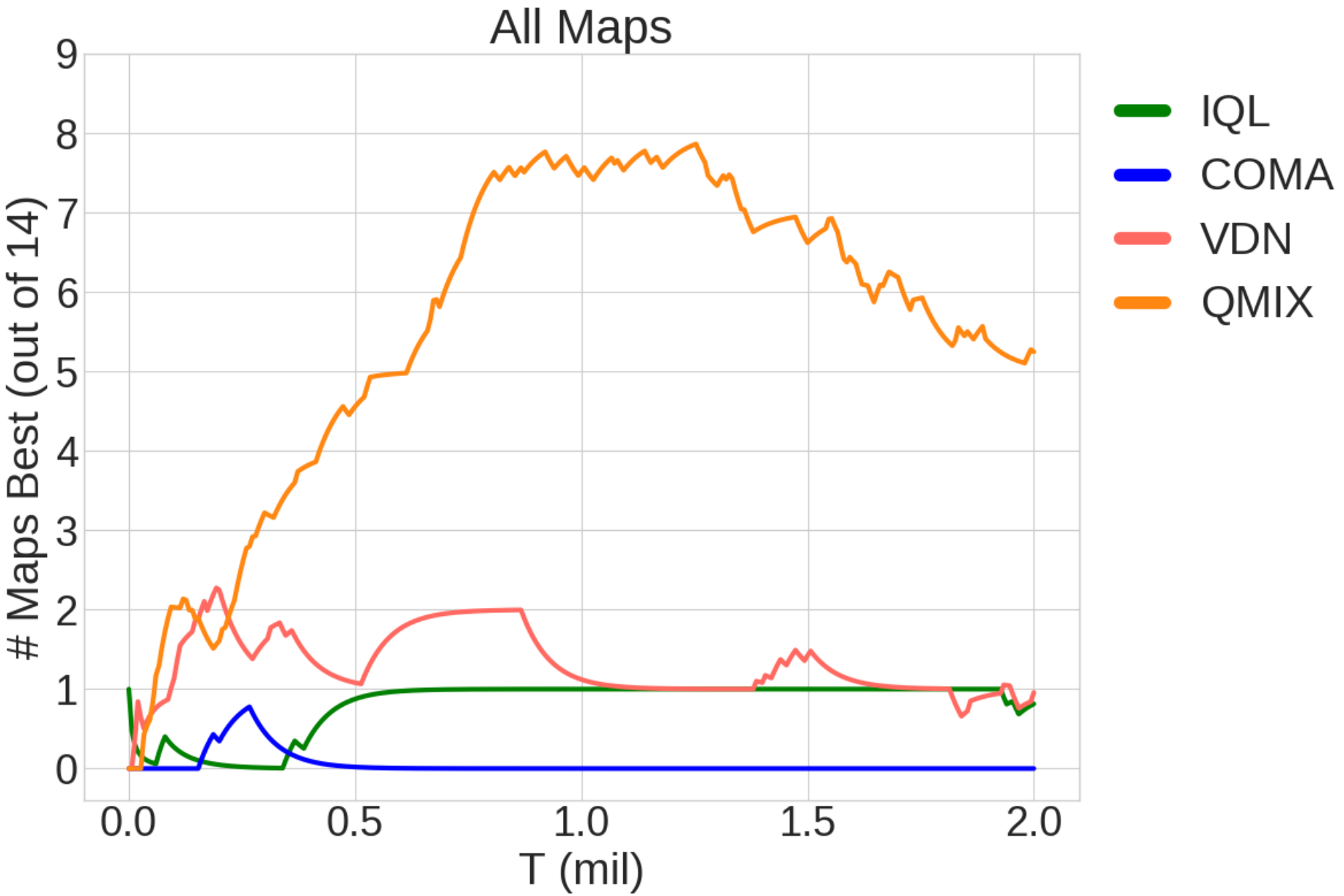


Cyan = sight range Red = shooting range

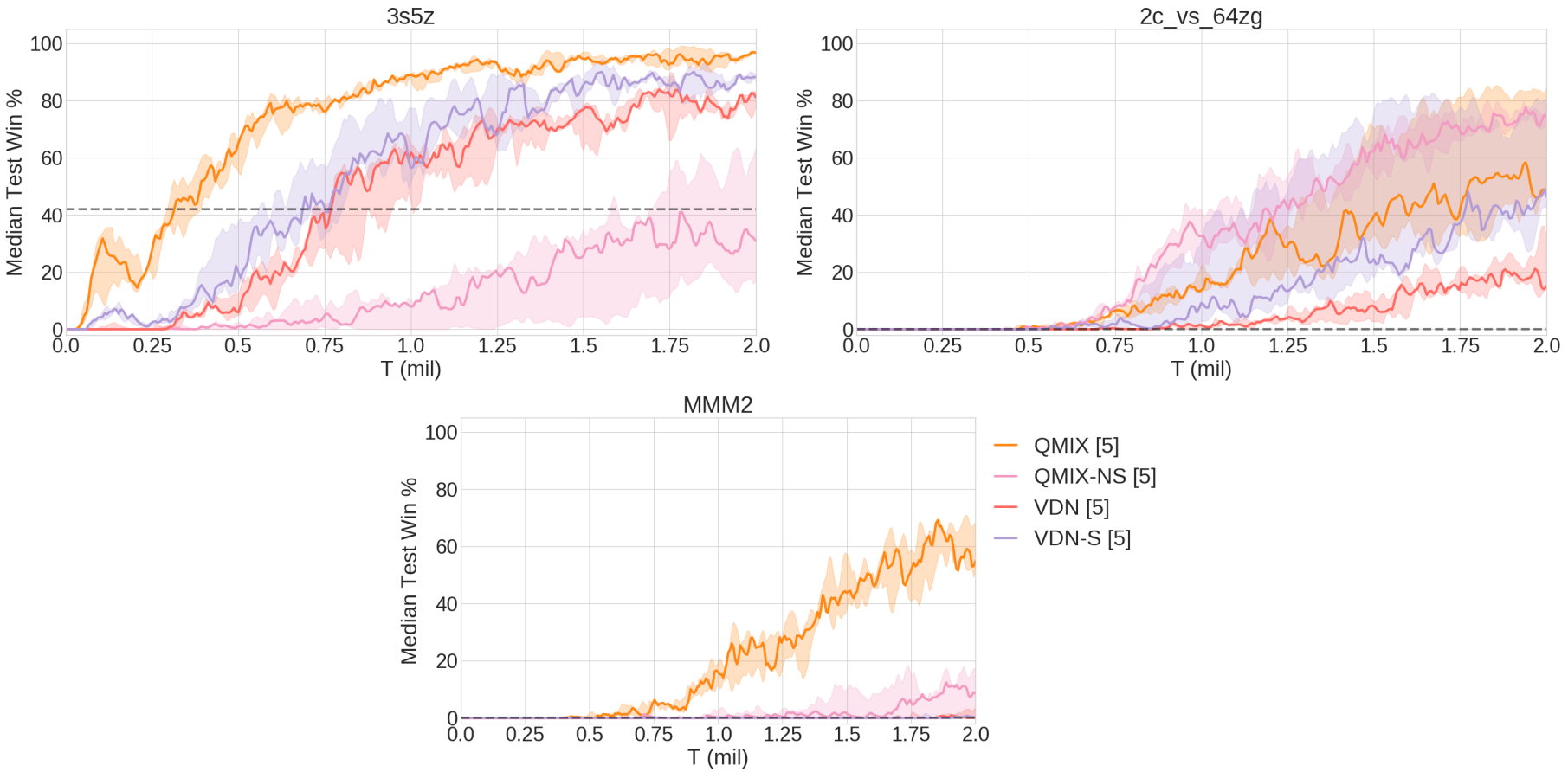
SMAC Maps

Name	Ally Units	Enemy Units
2s3z	2 Stalkers & 3 Zealots	2 Stalkers & 3 Zealots
3s5z	3 Stalkers & 5 Zealots	3 Stalkers & 5 Zealots
1c3s5z	1 Colossus, 3 Stalkers & 5 Zealots	1 Colossus, 3 Stalkers & 5 Zealots
5m_vs_6m	5 Marines	6 Marines
10m_vs_11m	10 Marines	11 Marines
27m_vs_30m	27 Marines	30 Marines
3s5z_vs_3s6z	3 Stalkers & 5 Zealots	3 Stalkers & 6 Zealots
MMM2	1 Medivac, 2 Marauders & 7 Marines	1 Medivac, 3 Marauders & 8 Marines
2s_vs_1sc	2 Stalkers	1 Spine Crawler
3s_vs_5z	3 Stalkers	5 Zealots
6h_vs_8z	6 Hydralisks	8 Zealots
bane_vs_bane	20 Zerglings & 4 Banelings	20 Zerglings & 4 Banelings
2c_vs_64zg	2 Colossi	64 Zerglings
corridor	6 Zealots	24 Zerglings

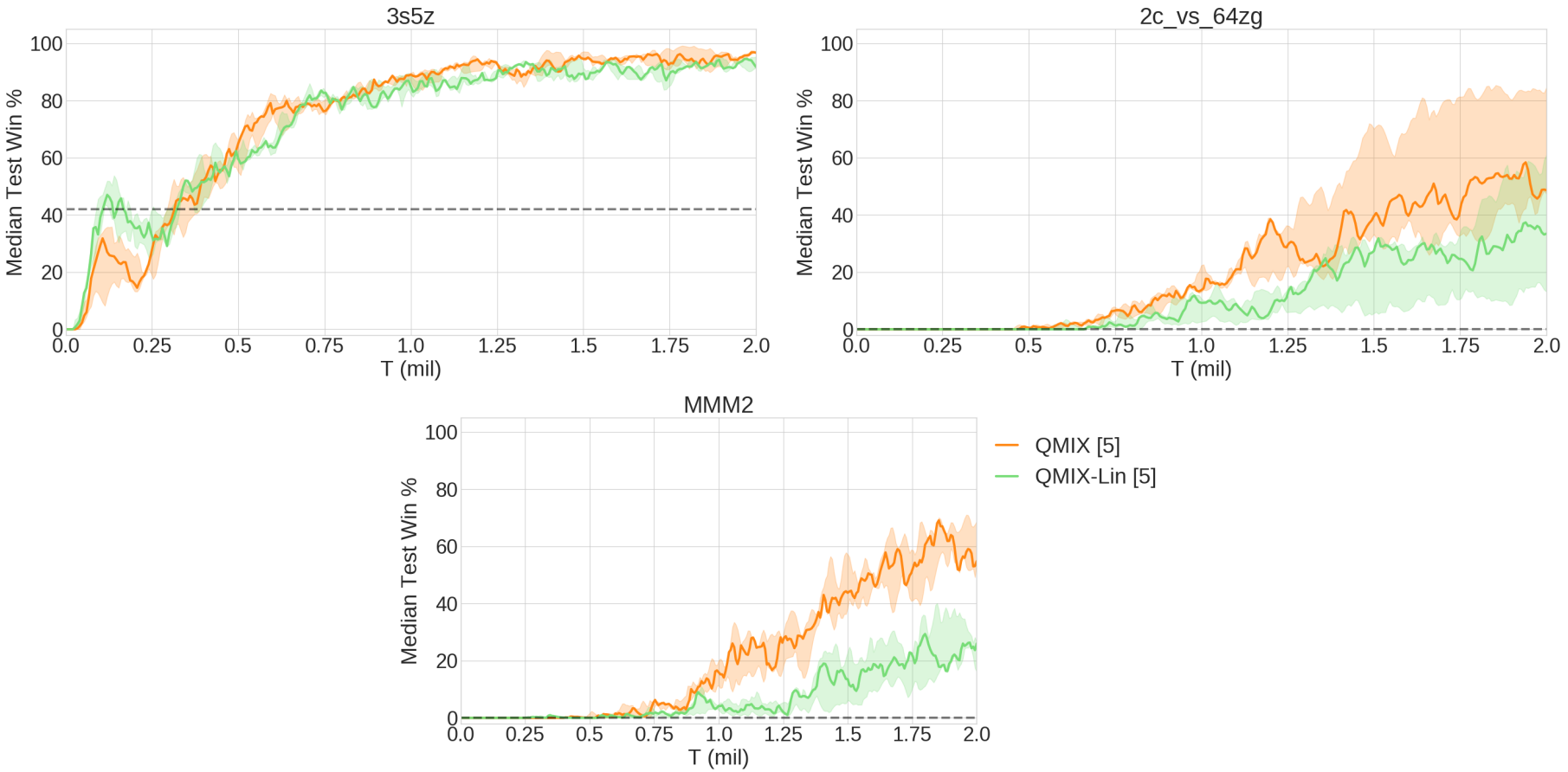
Overall Results (The Students Were Right)



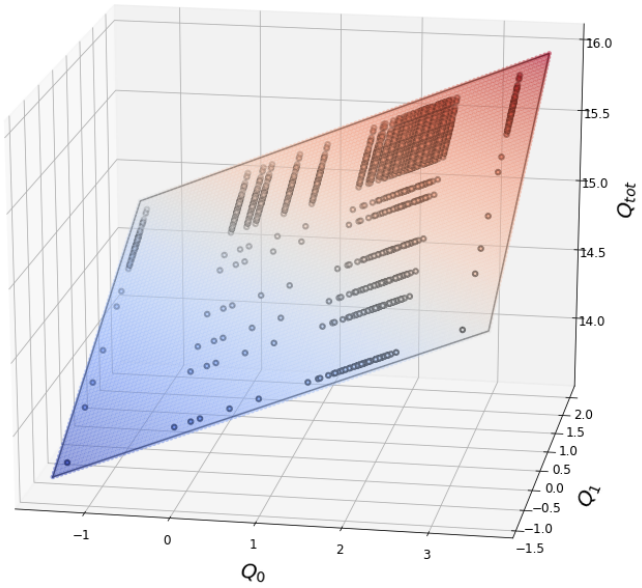
State Ablations



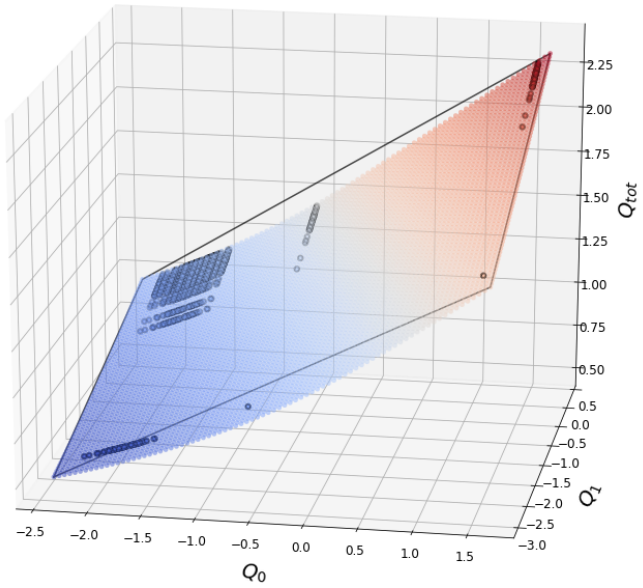
Linear Ablations



Learned Mixing Functions (2c_vs_64zg)

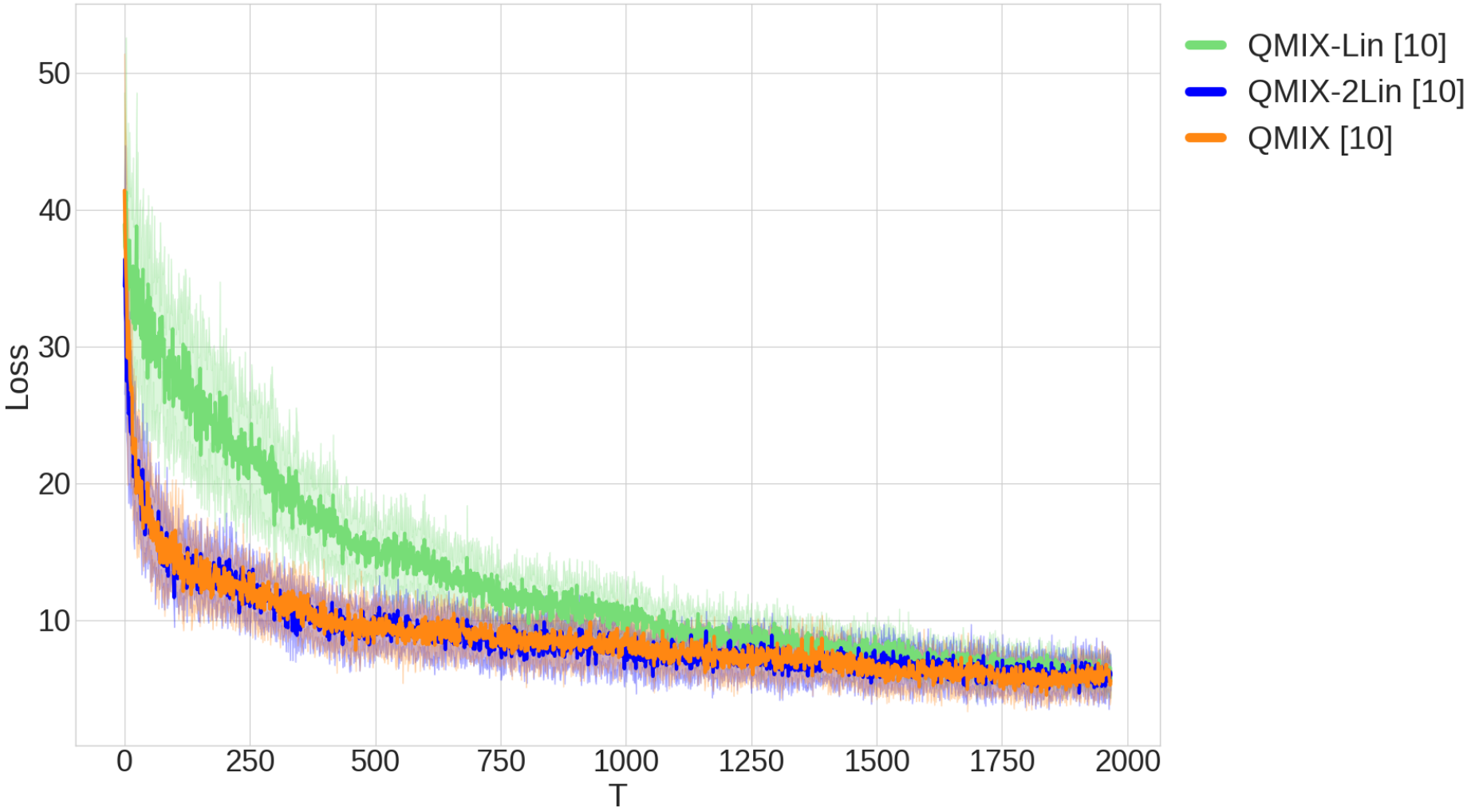


$t = 0$

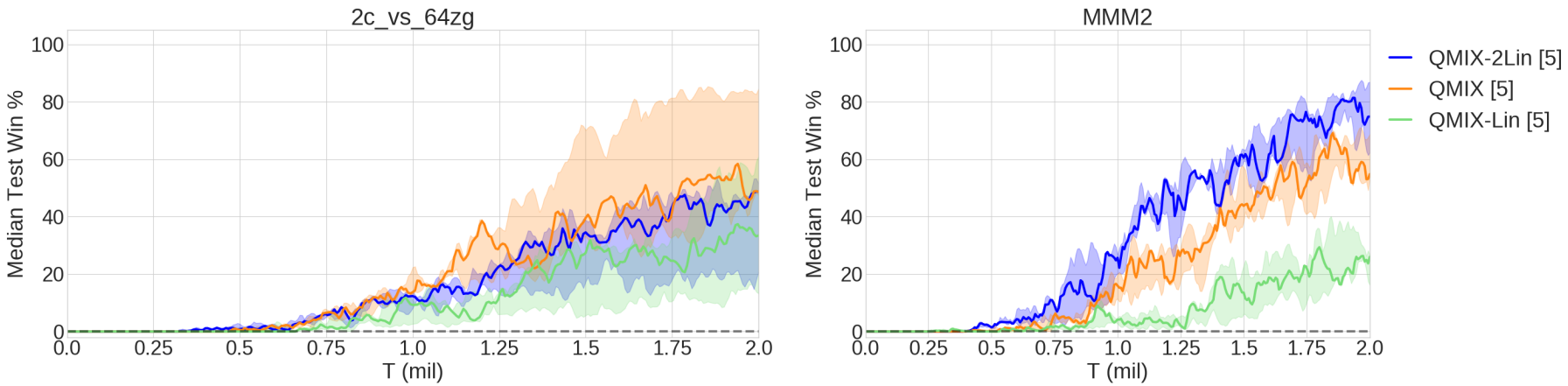


$t = 50$

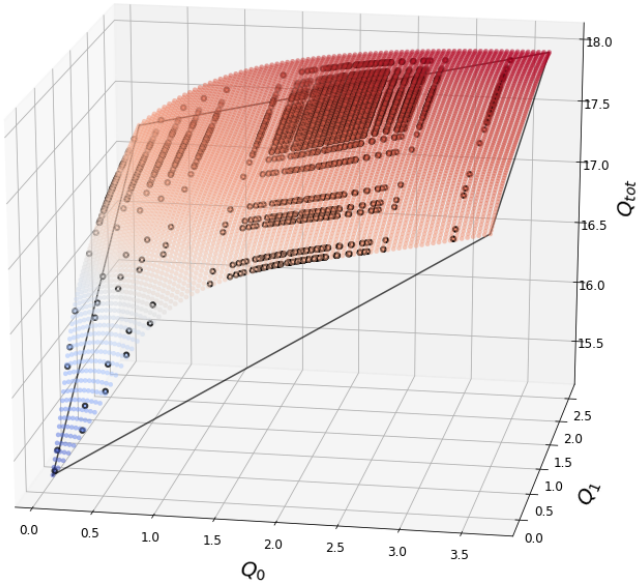
Multi-Layer Linear Mixing (Regression)



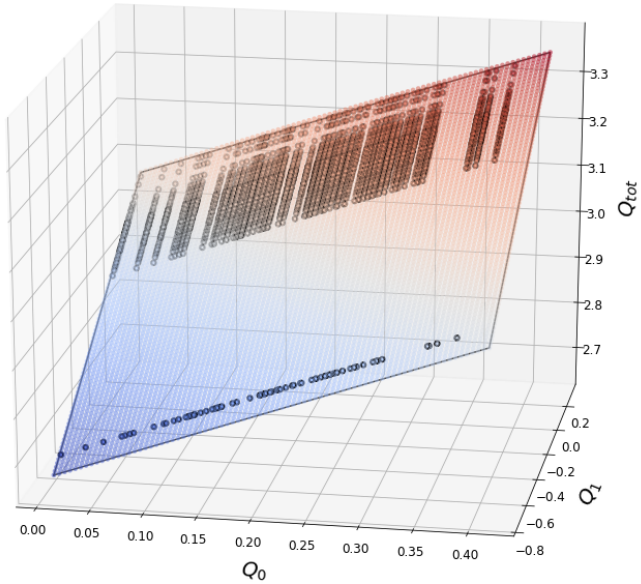
Multi-Layer Linear Mixing (SMAC)



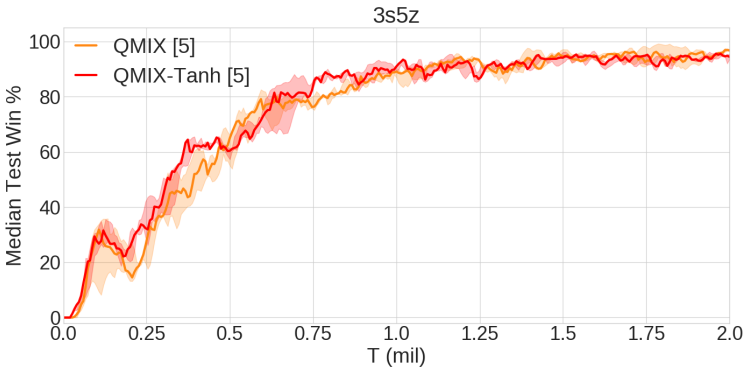
Tanh Activation

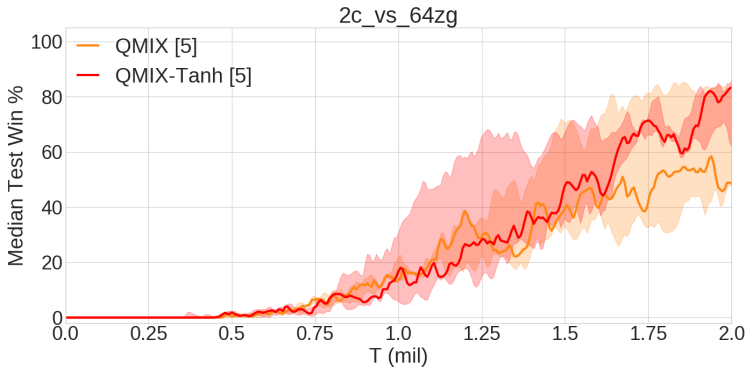


$t = 0$



$t = 50$





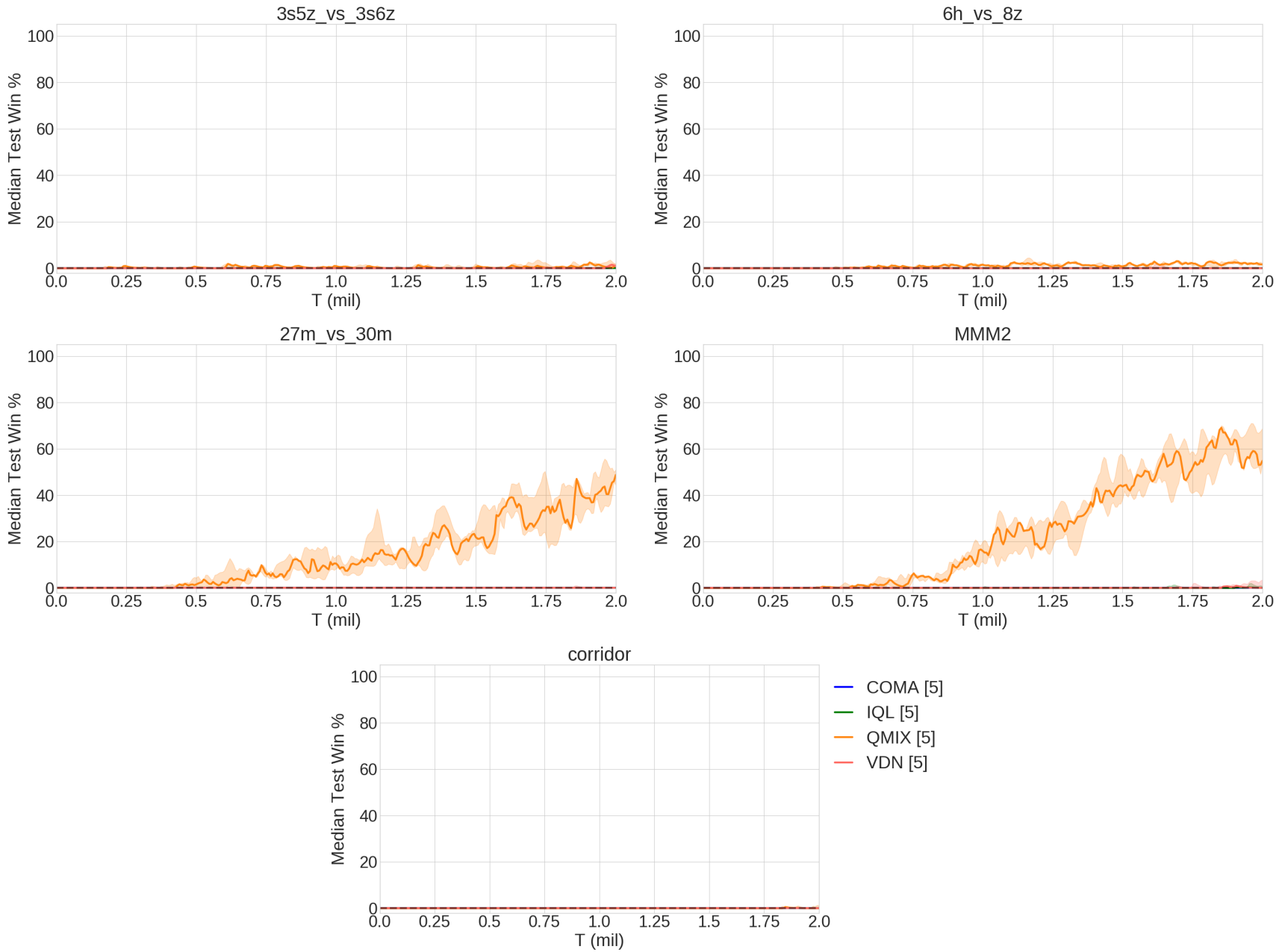
QMIX Takeaways

- Value function factorisation is crucial
- Flexible conditioning on central state is crucial
- Richly parameterised mixing is crucial
- Nonlinear mixing is not crucial (on SMAC)

Multi-Agent RL Methods from WhiRL

- DIAL [Foerster et al. 2015]
- Multi-Agent Fingerprints [Foerster et al. 2017]
- COMA [Foerster et al. 2018]
- QMIX [Rashid et al. 2018]
- LOLA [Foerster et al. 2019]
- SOS [Letcher et al. 2019]
- MACKRL [Schroeder de Witt et al. 2019]
- *MAVEN [Mahajan et al. 2019]*
- WQMIX [Rashid et al. 2020]
- COMIX [Schroeder et al. 2020]

Super Hard Maps



Hypotheses

Hypothesis 1

Super Hard Maps in SMAC require nonmonotonic mixing functions

But then why does COMA also fail?

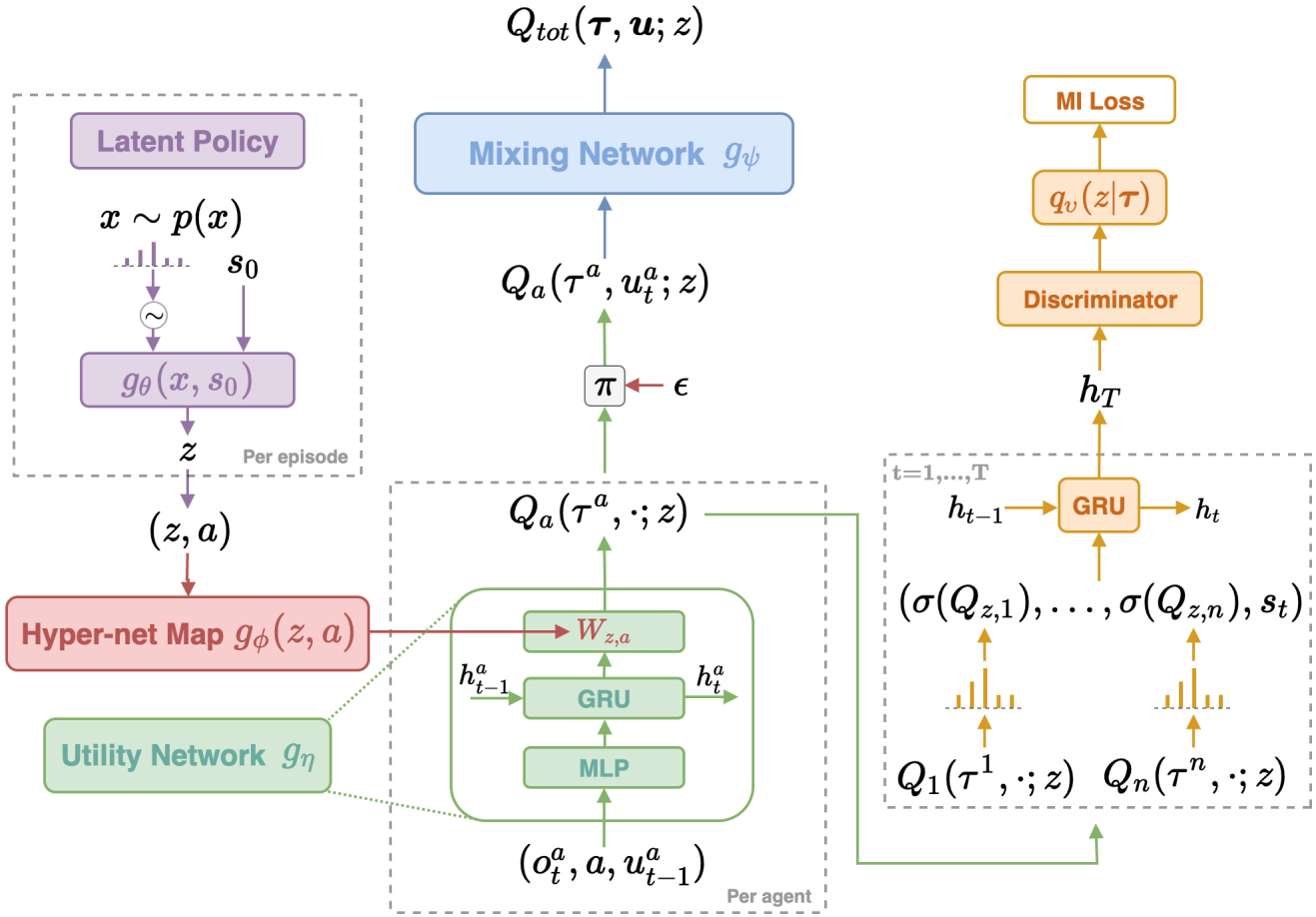
Hypothesis 2

Learning nonmonotonic mixing functions requires smart exploration

QMIX uses naive ϵ-greedy and is sensitive to annealing schedule

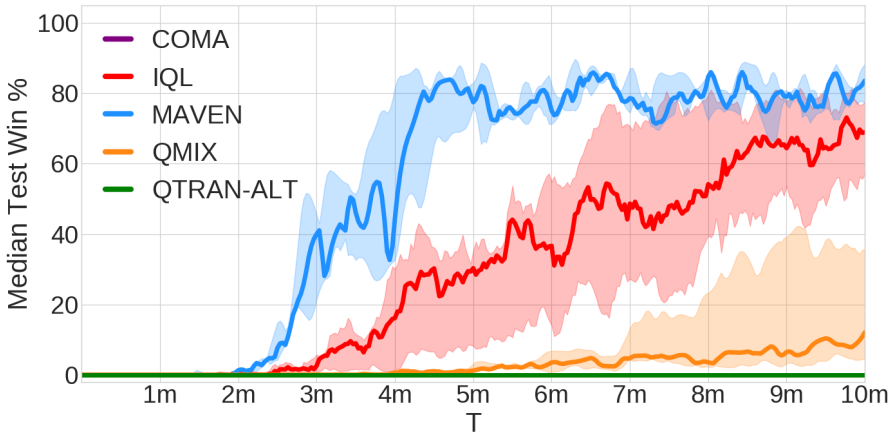
Multi-Agent Variational Exploration (MAVEN)

[Mahajan et al. 2019]

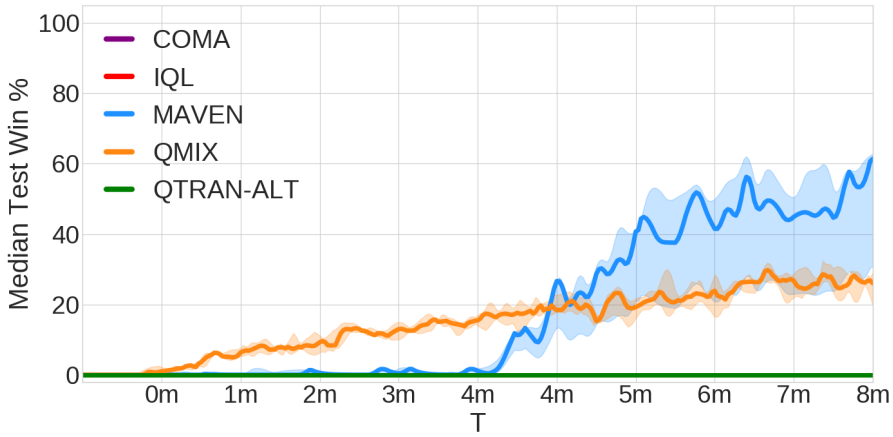


$$\max_{\upsilon,\phi,\eta,\psi,\theta} \mathcal{J}_{RL}(\theta) + \lambda_{MI}\mathcal{J}_V(\upsilon, \phi, \eta, \psi) - \lambda_{QL}\mathcal{L}_{QL}(\phi, \eta, \psi) \qquad (1)$$

MAVEN Results on Super Hard Maps

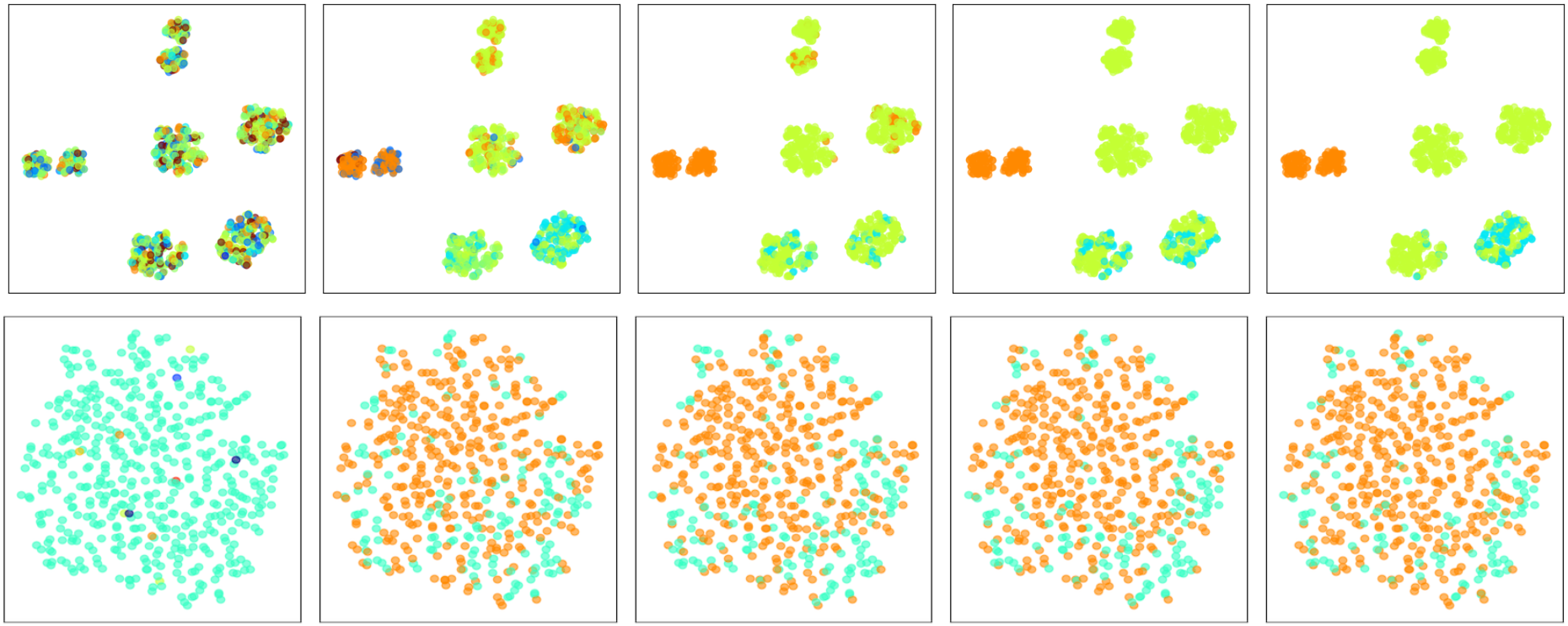


corridor



6h_vs_8z

MAVEN Latent Space



Top: 3s5z Bottom: corridor

Papers

QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning, **ICML-18**

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder de Witt, Gregory Farquhar, Jakob Foerster, & Shimon Whiteson

Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning, **JMLR-20 (To Appear)**

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder de Witt, Gregory Farquhar, Jakob Foerster, & Shimon Whiteson

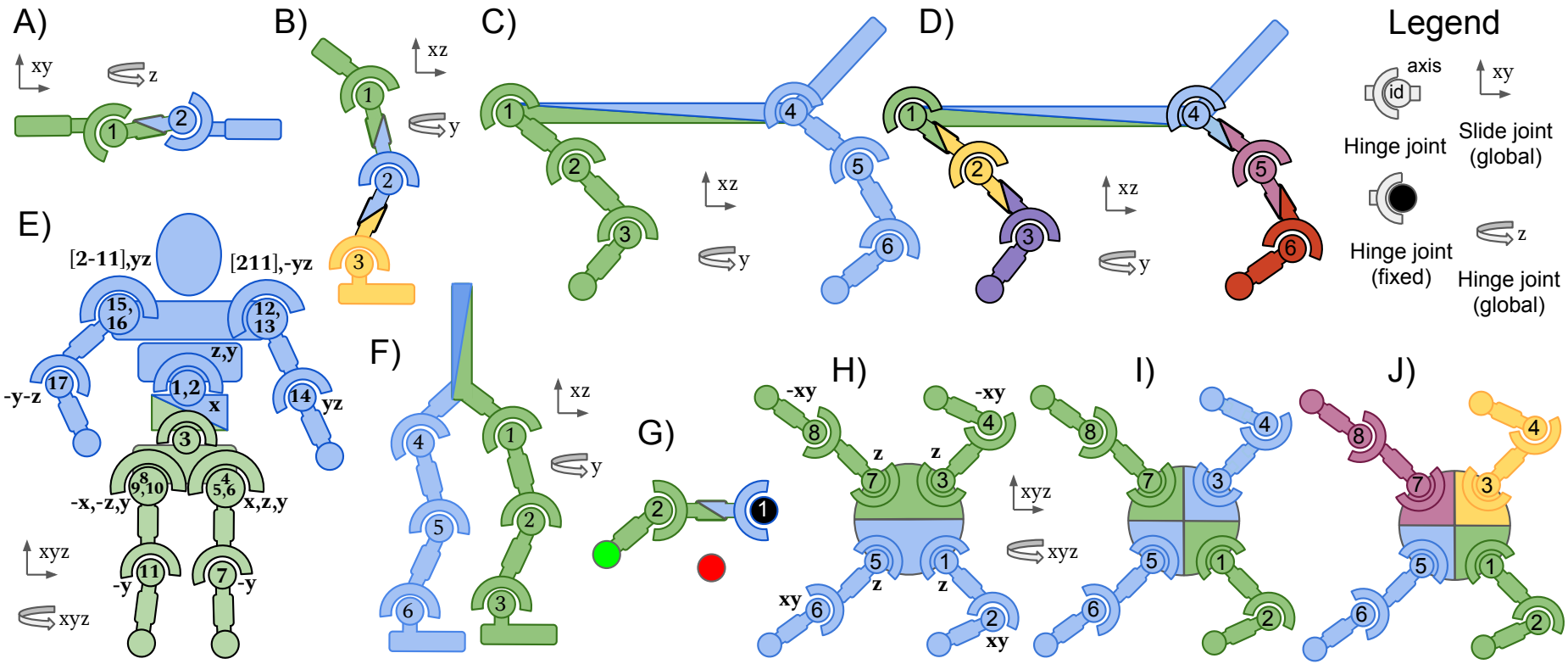
MAVEN: Multi-Agent Variational Exploration, **NeurIPS-19**

Anuj Mahajan, Tabish Rashid, Mikayel Samvelyan, & Shimon Whiteson

Conclusions

- QMIX: simple, effective value factorisation for MARL
- Extensive results suggest factorisation is crucial
- Confounder: COMA is on-policy actor-critic
- Controlled comparisons:
  - COMIX: continuous-action QMIX [Schroeder de Witt et al. 2020]
  - MADDPG: continuous-action off-policy actor-critic [Lowe et al. 2017]

Multi-Agent Mujoco [Schroeder de Witt et al. 2020]



$$COMIX > MADDPG$$

$$COMIX = FacMADDPG$$

New Frontiers

- *New factorisations*: tensor decomposition
- *New algorithms*: PPO-based methods upending SOTA
- *New architectures*: transformer-based approaches for transfer
- *New applications*: Google Research Football

Figure from [Kurach et al. 2019]