Cooperative Multi-Agent Reinforcement Learning (Coop MARL)

Part 2. Policy-based Methods

Acknowledgement: Most slides were contributed by 廖唯辰、何國豪, and organized by 廖唯辰.

Outline

- Introduction to Cooperative MARL
- Value-based Cooperative MARL
- Policy-based Cooperative MARL

Reference

- [marlgt]
  - Yang, Yaodong, and Jun Wang. "An overview of multi-agent reinforcement learning from game theoretical perspective." arXiv preprint arXiv:2011.00583 (2020). (https://arxiv.org/abs/2011.00583)
- [RL]
  - Sutton, Richard S., and Andrew G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.
- [gcoop]
  - Yang, Y. (n.d.). A GENERAL SOLUTION FRAMEWORK FOR COOPERATIVE MULTI-AGENT REINFORCEMENT LEARNING. Retrieved April 16, 2023, from https://www.yangyaodong.com/_files/ugd/ddd18b_969600e4558b43f29a250e573d618cbf.pdf
- [mc]
  - Cooperative MARL slides by Kuo-Hao Ho

Reference

- [vdn]
  - Sunehag, Peter, et al. "Value-decomposition networks for cooperative multi-agent learning." arXiv preprint arXiv:1706.05296 (2017). (https://arxiv.org/abs/1706.05296)
- [qmix]
  - Rashid, Tabish, et al. "Monotonic value function factorisation for deep multi-agent reinforcement learning." The Journal of Machine Learning Research 21.1 (2020): 7234-7284. (https://arxiv.org/abs/1803.11485)
- [qtran]
  - Son, Kyunghwan, et al. "Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning." International conference on machine learning. PMLR, 2019. (https://proceedings.mlr.press/v97/son19a.html)

Reference

- [maddpg]
  - Lowe, Ryan, et al. "Multi-agent actor-critic for mixed cooperative-competitive environments." Advances in neural information processing systems 30 (2017). (https://proceedings.neurips.cc/paper/2017/hash/68a9750337a418a86fe06c1991a1d64c-Abstract.html)
- [coma]
  - Foerster, Jakob, et al. "Counterfactual multi-agent policy gradients." Proceedings of the AAAI conference on artificial intelligence. Vol. 32. No. 1. 2018. (https://ojs.aaai.org/index.php/AAAI/article/view/11794)
- [ippo]
  - de Witt, Christian Schroeder, et al. "Is independent learning all you need in the starcraft multi-agent challenge?." *arXiv preprint arXiv:2011.09533* (2020). (https://arxiv.org/abs/2011.09533)
- [mappo]
  - Yu, Chao, et al. "The surprising effectiveness of ppo in cooperative multi-agent games." Advances in Neural Information Processing Systems 35 (2022): 24611-24624. (https://proceedings.neurips.cc/paper_files/paper/2022/hash/9c1535a02f0ce079433344e14d910597-Abstract-Datasets_and_Benchmarks.html)

Reference

- [hatrpo]
  - Kuba, J. G., Chen, R., Wen, M., Wen, Y., Sun, F., Wang, J., & Yang, Y. (2022). Trust Region Policy Optimisation in Multi-Agent Reinforcement Learning. International Conference on Learning Representations. (https://openreview.net/forum?id=EcGGFkNTxdJ)
- [mat]
  - Wen, Muning, et al. "Multi-agent reinforcement learning is a sequence modeling problem." Advances in Neural Information Processing Systems 35 (2022): 16509-16521. (https://proceedings.neurips.cc/paper_files/paper/2022/hash/69413f87e5a34897cd010ca698097d0a-Abstract-Conference.html)
- [bmarl]
  - Papoudakis, Georgios, et al. "Benchmarking Multi-Agent Deep Reinforcement Learning Algorithms in Cooperative Tasks." Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1). 2021. (https://openreview.net/forum?id=cIrPX-Sn5n)

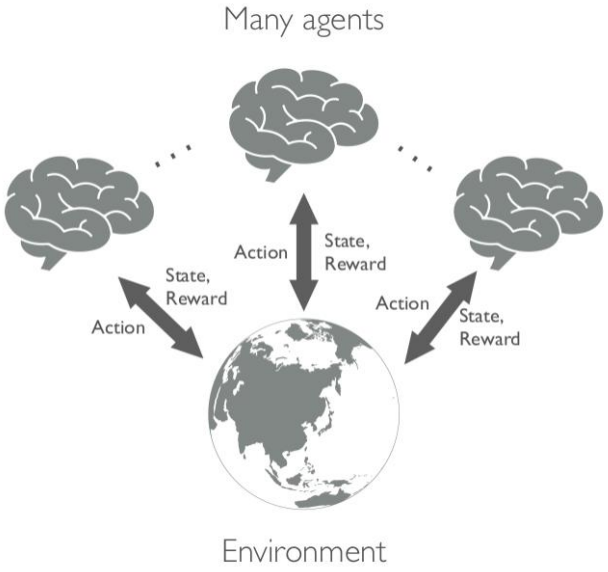
Reference

- [seps]
  - Filippos Christianos, Georgios Papoudakis, Arrasy Rahman, and Stefano V. Albrecht. Scaling multiagent reinforcement learning with selective parameter sharing. In International Conference on Machine Learning, 2021. (http://proceedings.mlr.press/v139/christianos21a.html)

Multi-Agent RL (MARL)

- N agent(s) in the environment
  - $N = 1$: degrade to Single-Agent RL

Single-Agent

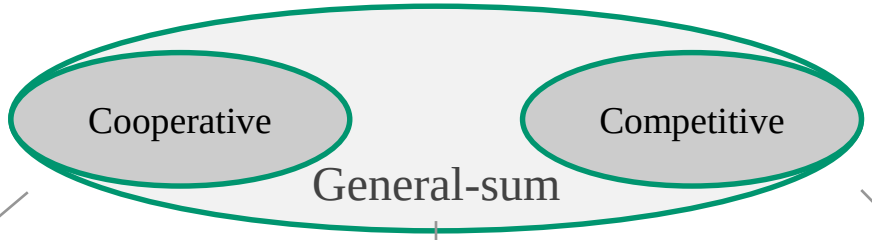


Multi-Agent

Source of image: [marlgt]

Categories of MARL

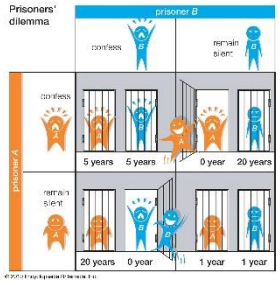
Categorized by Game Types



All agents have identical interests

E.g. Robotic Cooperation

Mixture of the other two game types



E.g. Prisoners' Dilemma

Agents share opposite interests and act competitively

E.g. Go

Source of images:
https://www.youtube.com/watch?v=5kIQ0F8iN8U
https://spectrum.ieee.org/mobile-robots-cooperate-to-3d-print-structures
https://www.britannica.com/science/game-theory/N-person-games

Cooperative MARL: Problem Formulation

- Consider a Partially-Observable Markov game: $\langle \mathcal{N}, \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{Z}, P, \mathcal{R}, \gamma \rangle$
  - $t \in \mathbb{Z}_0^+$: times step
    - All agents act simultaneously at each time step
  - $\mathcal{N} = \{1, \dots, N\}$
    - $\mathcal{N}$: the set of agents
    - N: number of agents
    - Agent $i \in \mathcal{N}$
  - $s \in \mathcal{S}$
    - $\mathcal{S}$: state space
    - s: global environment state
    - s_t: global environment state at time step t
  - $\boldsymbol{a} \coloneqq (a^i)_{i \in \mathcal{N}} \in \mathcal{A}^N$
    - $\boldsymbol{a}$: joint action
    - $\mathcal{A}$: action space
    - a^i: agent i's action
  - $o^i = \mathcal{Z}(s; i)$
    - $\mathcal{O}$: Local observation space
    - $o^i \in \mathcal{O}$: agent i's local observation
    - $\mathcal{Z}$: observation function
  - Agents optimize towards one shard reward: $\mathcal{R}(s, \boldsymbol{a}): \mathcal{S} \times \mathcal{A}^N \to \mathbb{R}$
  - $\tau^i \in \mathcal{T} \coloneqq (\mathcal{O} \times \mathcal{A})^{t-1} \times \mathcal{O}$
    - τ^i: agent i's action-observation
    - $\boldsymbol{\tau} \coloneqq (\tau^i)_{i \in \mathcal{N}} \in \mathcal{T}^N$: all of the agent histories
  - For $a, \boldsymbol{a}, o, \boldsymbol{o}, \tau, \boldsymbol{\tau}$, etc.:
    - With subscript t: at time step t
    - With superscript i: of agent i
    - E.g. a_t^i means the action of agent i at time step t
  - $P(s'|s, \boldsymbol{a}) \coloneqq \mathcal{S} \times \mathcal{A}^N \times \mathcal{S} \to [0,1]$: state transition dynamics
  - $\gamma \in [0,1)$: discount factor

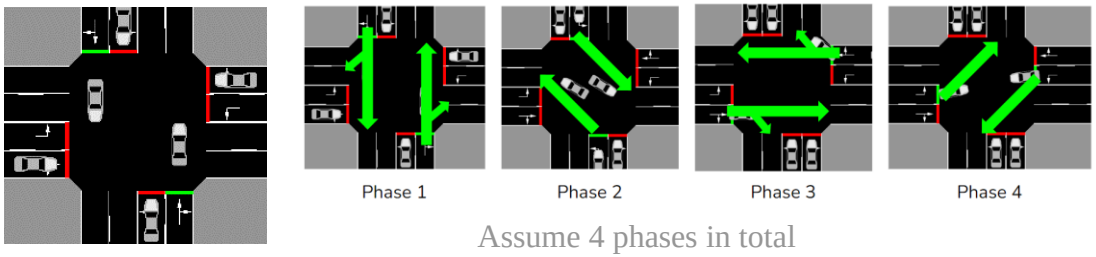
Problem Formulation and Notation

- $\boldsymbol{\pi} \coloneqq (\pi^i)_{i \in \mathcal{N}}$
  - $\boldsymbol{\pi}$: joint policy
  - π^i: agent i's policy
  - Different policy settings
    - $\pi^i(a^i|\tau^i): \mathcal{T} \times \mathcal{A} \to [0,1]$
    - $\pi^i(a^i|o^i): \mathcal{O} \times \mathcal{A} \to [0,1]$
    - $\pi^i(a^i|s): \mathcal{S} \times \mathcal{A} \to [0,1]$
- $G_t \coloneqq \sum_j^{\infty} \gamma^j R_{t+i}$: total accumulative rewards
- $\rho_{\boldsymbol{\pi}}(s) \coloneqq \sum_{t=0}^{\infty} \gamma^t \rho_{\boldsymbol{\pi}}^t(s)$
  - $\rho_{\boldsymbol{\pi}}(s)$: improper marginal state distribution
  - $\rho_{\boldsymbol{\pi}}^t(s)$: marginal state distribution at time t under joint policy $\boldsymbol{\pi}$
  - $\rho^0(\mathrm{s})$: initial state distribution
- Value function
  - Use $u/\boldsymbol{u}$ to denote the input instance of $a/\boldsymbol{a}$
  - $V_{\boldsymbol{\pi}}(s) \coloneqq \mathbb{E}_{\boldsymbol{a}_{0:\infty} \sim \boldsymbol{\pi}, s_{1:\infty} \sim \mathcal{P}}[\sum_{t=0}^{\infty} \gamma^t R_t | s_0 = s]$
  - $Q_{\boldsymbol{\pi}}(s, \boldsymbol{u}) \coloneqq \mathbb{E}_{s_{1:\infty} \sim \mathcal{P}, \boldsymbol{a}_{1:\infty} \sim \boldsymbol{\pi}}[\sum_{t=0}^{\infty} \gamma^t R_t | s_0 = s, \boldsymbol{a}_0 = \boldsymbol{u}]$
  - $A_{\boldsymbol{\pi}}(s, \boldsymbol{u}) \coloneqq Q_{\boldsymbol{\pi}}(s, \boldsymbol{u}) - V_{\boldsymbol{\pi}}(s)$
- Consider a fully-cooperative setting where all agents share the same reward function, aiming to maximize the expected total reward:

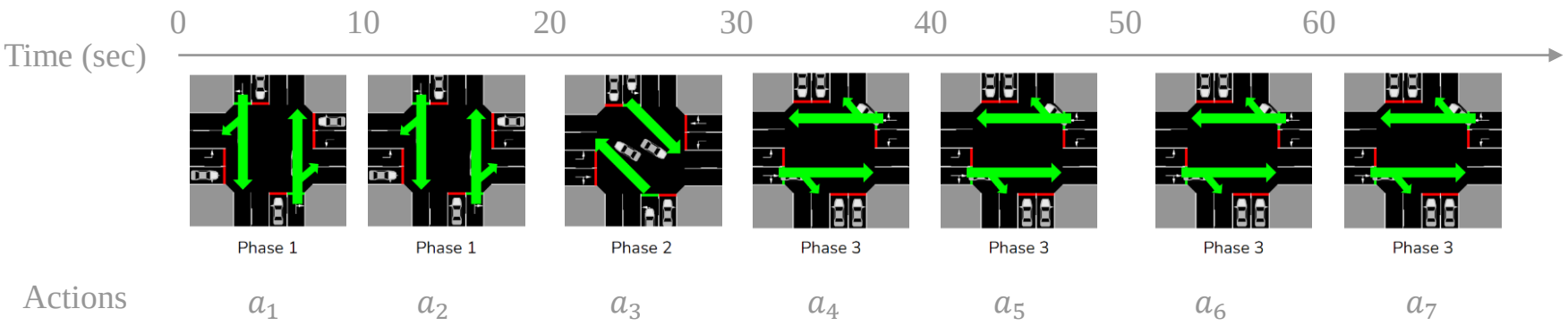
$$J(\boldsymbol{\pi}) \coloneqq \mathbb{E}_{s_{0:\infty} \sim \rho_{\boldsymbol{\pi}}^{0:\infty}, \boldsymbol{a}_{0:\infty} \sim \boldsymbol{\pi}}[\sum_{t=0}^{\infty} \gamma^t R_t]$$

Case Study: Traffic Signal Control (TSC)

- Example:
  - Action definition: select next phase every 10 seconds

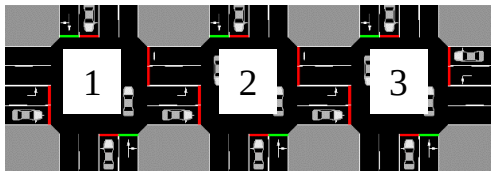


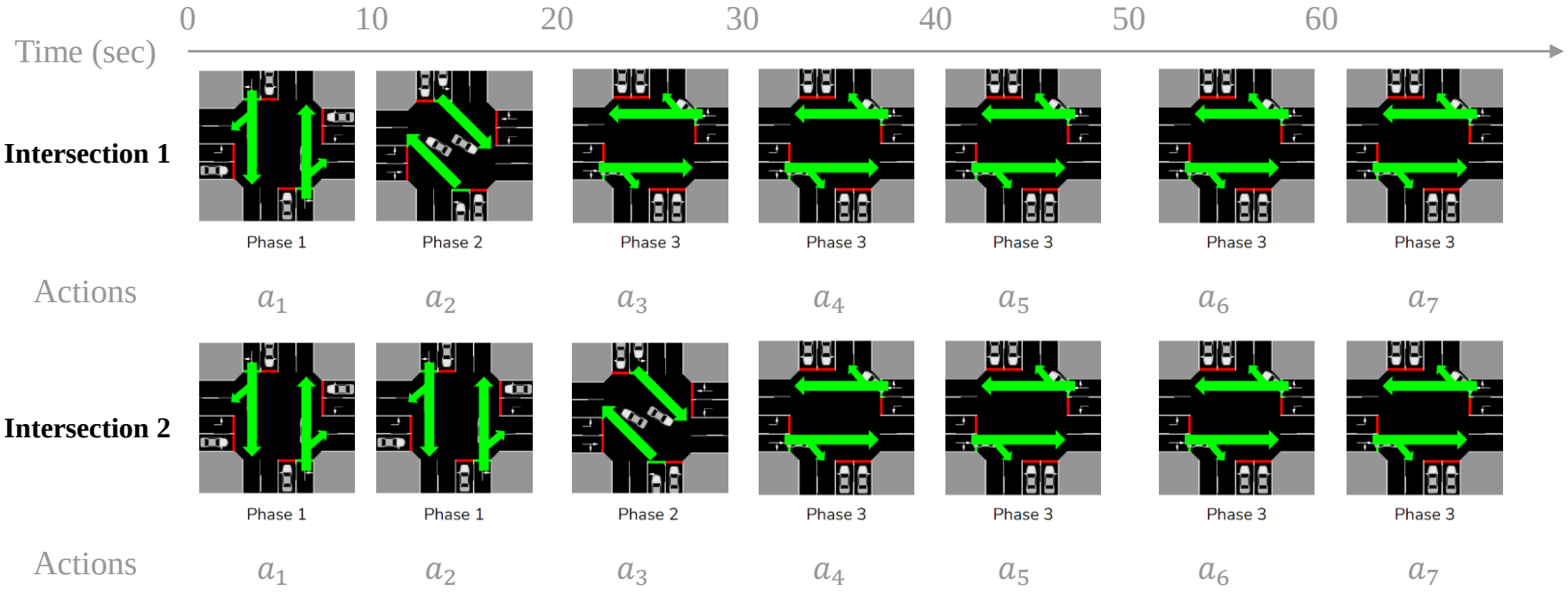
Assume 4 phases in total



Case Study: Traffic Signal Control (TSC)

- Multiple intersections





Policy-based Cooperative MARL

- IA2C, IPPO, MADDPG, COMA, MAA2C, MAPPO
- Parameter Sharing
- HATRPO/HAPPO
- Multi-Agent Transformer (MAT)

Policy-based Cooperative MARL

- IA2C, IPPO, MADDPG, COMA, MAA2C, MAPPO
- Parameter Sharing
- HATRPO/HAPPO
- Multi-Agent Transformer (MAT)

Reference

- Benchmarking Multi-Agent Deep Reinforcement Learning Algorithms in Cooperative Tasks (NeurIPS Datasets and Benchmarks Track 2021)
  - [bmarl] Papoudakis, Georgios, et al. "Benchmarking Multi-Agent Deep Reinforcement Learning Algorithms in Cooperative Tasks." Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1). 2021. (https://openreview.net/forum?id=cIrPX-Sn5n)
  - Source code: https://github.com/uoe-agents/epymarl
    - Lots of algorithms
      - IQL
      - IA2C
      - IPPO
      - MADDPG
      - COMA
      - MAA2C
      - MAPPO
      - VDN
      - QMIX

Reference

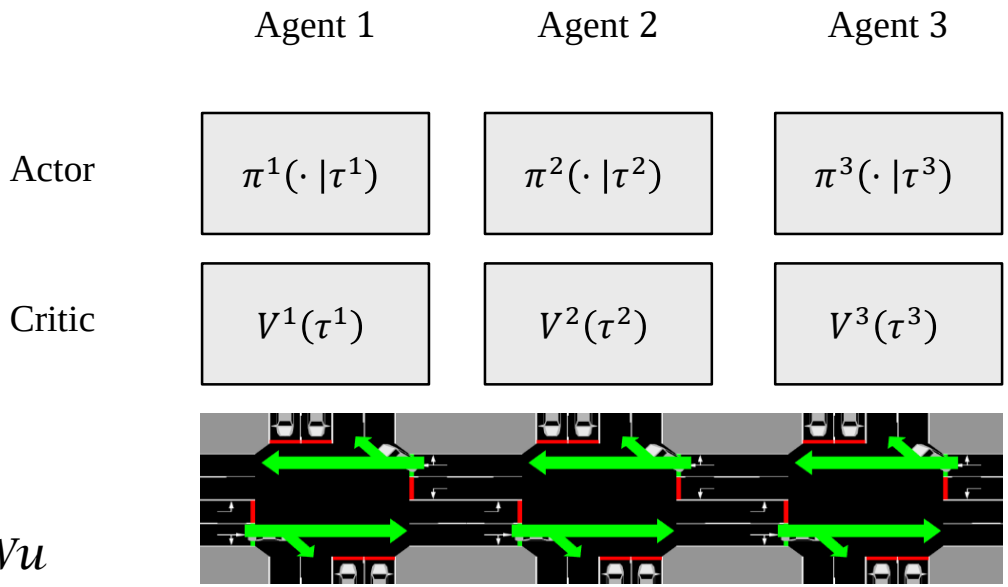
- More details
  - [maddpg] (NeurIPS 2017)
    - Lowe, Ryan, et al. "Multi-agent actor-critic for mixed cooperative-competitive environments." Advances in neural information processing systems 30 (2017). (https://proceedings.neurips.cc/paper/2017/hash/68a9750337a418a86fe06c1991a1d64c-Abstract.html)
  - [coma] (AAAI 2018)
    - Foerster, Jakob, et al. "Counterfactual multi-agent policy gradients." Proceedings of the AAAI conference on artificial intelligence. Vol. 32. No. 1. 2018. (https://ojs.aaai.org/index.php/AAAI/article/view/11794)
  - [ippo]
    - de Witt, Christian Schroeder, et al. "Is independent learning all you need in the starcraft multi-agent challenge?." *arXiv preprint arXiv:2011.09533* (2020). (https://arxiv.org/abs/2011.09533)
  - [mappo] (NeurIPS 2022)
    - Yu, Chao, et al. "The surprising effectiveness of ppo in cooperative multi-agent games." Advances in Neural Information Processing Systems 35 (2022): 24611-24624. (https://proceedings.neurips.cc/paper_files/paper/2022/hash/9c1535a02f0ce079433344e14d910597-Abstract-Datasets_and_Benchmarks.html)

Algorithms

- IL (Independent Learning)
  - IA2C
  - IPPO
- CTDE (Centralized Training Decentralized Execution)
  - MADDPG
  - MAA2C
  - COMA
  - MAPPO

Independent Learning (IL)

- For IL, each agent is learning independently and perceives the other agents as part of the environment
- Methods
  - IA2C (Independent Advantage Actor-Critic)
  - IPPO (Independent Proximal Policy Optimization)
  - …



CTDE Methods

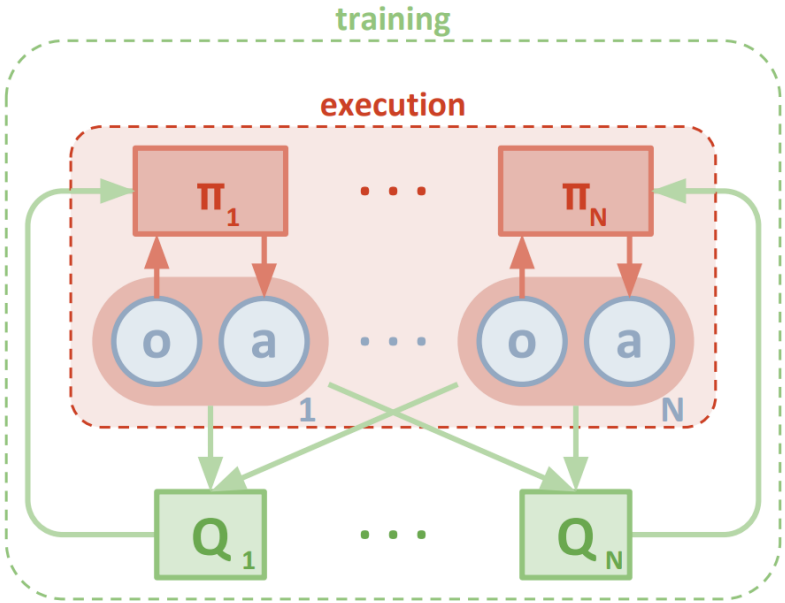
- Centralized Training Decentralized Execution (CTDE)
  - Training: allows sharing of information during training
  - Execution: policies are only conditioned on local observations
  - Methods
    - MADDPG
    - MAA2C
    - COMA
    - MAPPO
    - …

CTDE Methods

Algorithm	Actor Loss (Use local information)	Critic Type (Use global information)
MADDPG	Deterministic Policy Gradient loss	Q-function
MAA2C	A2C loss	V-function
COMA	Policy Gradient computed by the Advantage with Counterfactual Baseline	Q-function
MAPPO	PPO-Clip loss	V-function

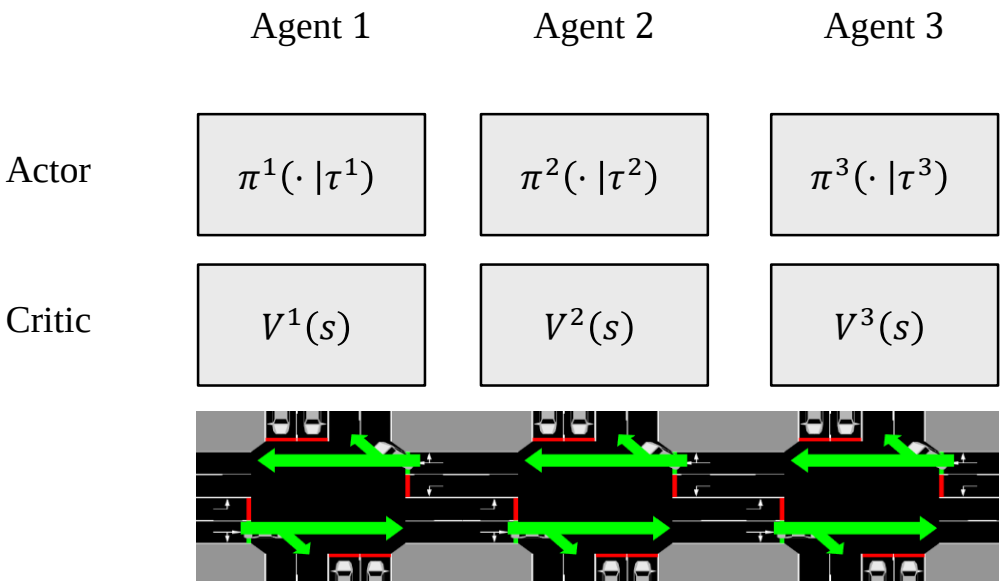
MADDPG

- MADDPG (Multi-Agent DDPG)
  - A variation of the DDPG algorithm for MARL
  - Critic (Q-function): trained on the joint observation and action
  - Actor: conditioned on the local information



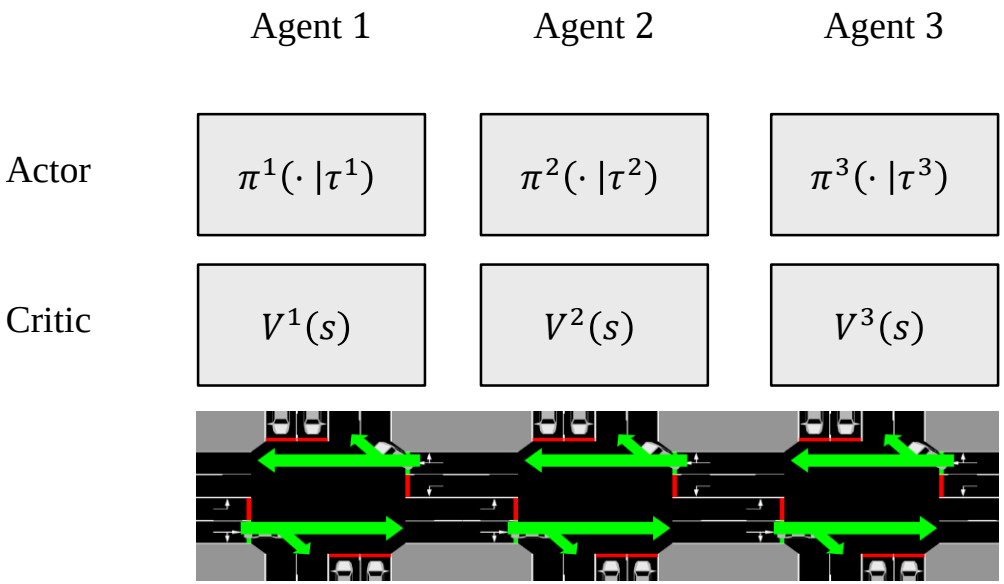
MAA2C

- MAA2C (Multi-Agent A2C)
  - A variation of the A2C algorithm for MARL
  - Critic (V-function): learn state value function
  - Actor: conditioned on the local information



MAPPO

- MAPPO (Multi-Agent PPO)
  - A variation of the PPO algorithm for MARL
  - Critic (V-function): learn state value function (similar to MAA2C)
  - Actor: conditioned on the local information

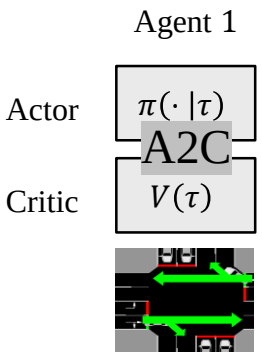


Comparisons

- Use A2C as an example

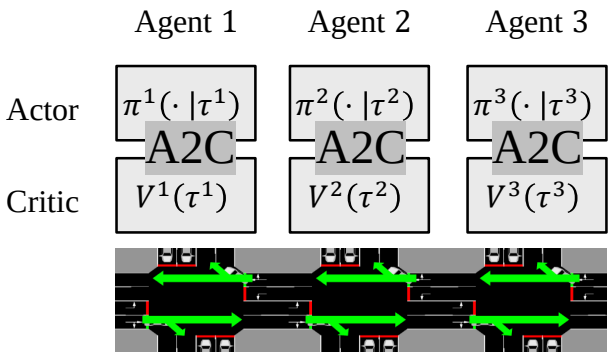
Single-Agent

A2C



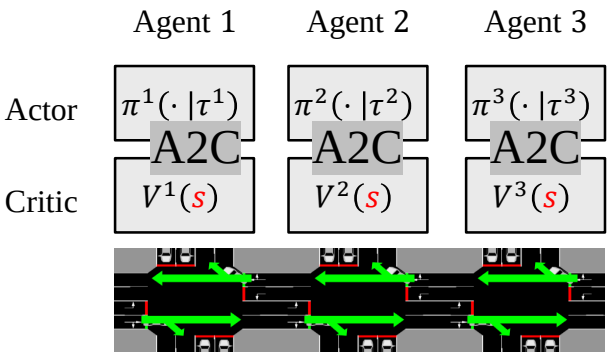
Multi-Agent
(Independent Learning)

IA2C



Multi-Agent
(CTDE)

MAA2C



COMA

- COMA (Counterfactual Multi-Agent Policy Gradient)
  - Critic (Q-function): trained on state and joint actions
  - Actor: conditioned on the local history
    - Compute policy gradient using counterfactual baseline to estimate advantage
      - Expectation of following current π^i while fixing the actions of other agents

$$A^i(s, \boldsymbol{u}) = Q(s, \boldsymbol{u}) - \sum_{u'^i \in \mathcal{A}} \pi^i\left(u'^i \middle| \tau^i\right) Q\left(s, \left(\boldsymbol{u}^{-i}, u'^i\right)\right)$$

      - Recall Advantage:

$$A(s,u) = Q(s,u) - V(s) = Q(s,u) - \sum_{u' \in \mathcal{A}} \pi(u'|s) Q(s,u')$$

COMA

s

State

Action Space

$A_{\boldsymbol{\pi}}(s,$  $) =?$ $Q_{\boldsymbol{\pi}}(s,$  $) - Baseline$

$i=1$ $i=2$ $i=3$ $i=1$ $i=2$ $i=3$

$$Baseline = V_{\boldsymbol{\pi}}(s)$$
$$= \mathbb{E}_{\boldsymbol{a}_{0:\infty}\sim\boldsymbol{\pi}, s_{1:\infty}\sim\mathcal{P}}[G_0|s_0=s]$$

$$Baseline = B_{\boldsymbol{\pi}}^{COMA}(s,\boldsymbol{u}^i)$$
$$= \mathbb{E}_{a_{0:\infty}^i\sim\pi^i, \boldsymbol{a}_{1:\infty}^{-i}\sim\boldsymbol{\pi}^{-i}, s_{1:\infty}\sim\mathcal{P}}[G_0|s_0=s, \boldsymbol{a}_0^{-i}=\boldsymbol{u}^{-i}]$$
$$= \sum_{u'^i\in\mathcal{A}} \pi^i\left(u'^i\middle|\tau^i\right) Q\left(s,\left(\boldsymbol{u}^{-i},u'^i\right)\right)$$

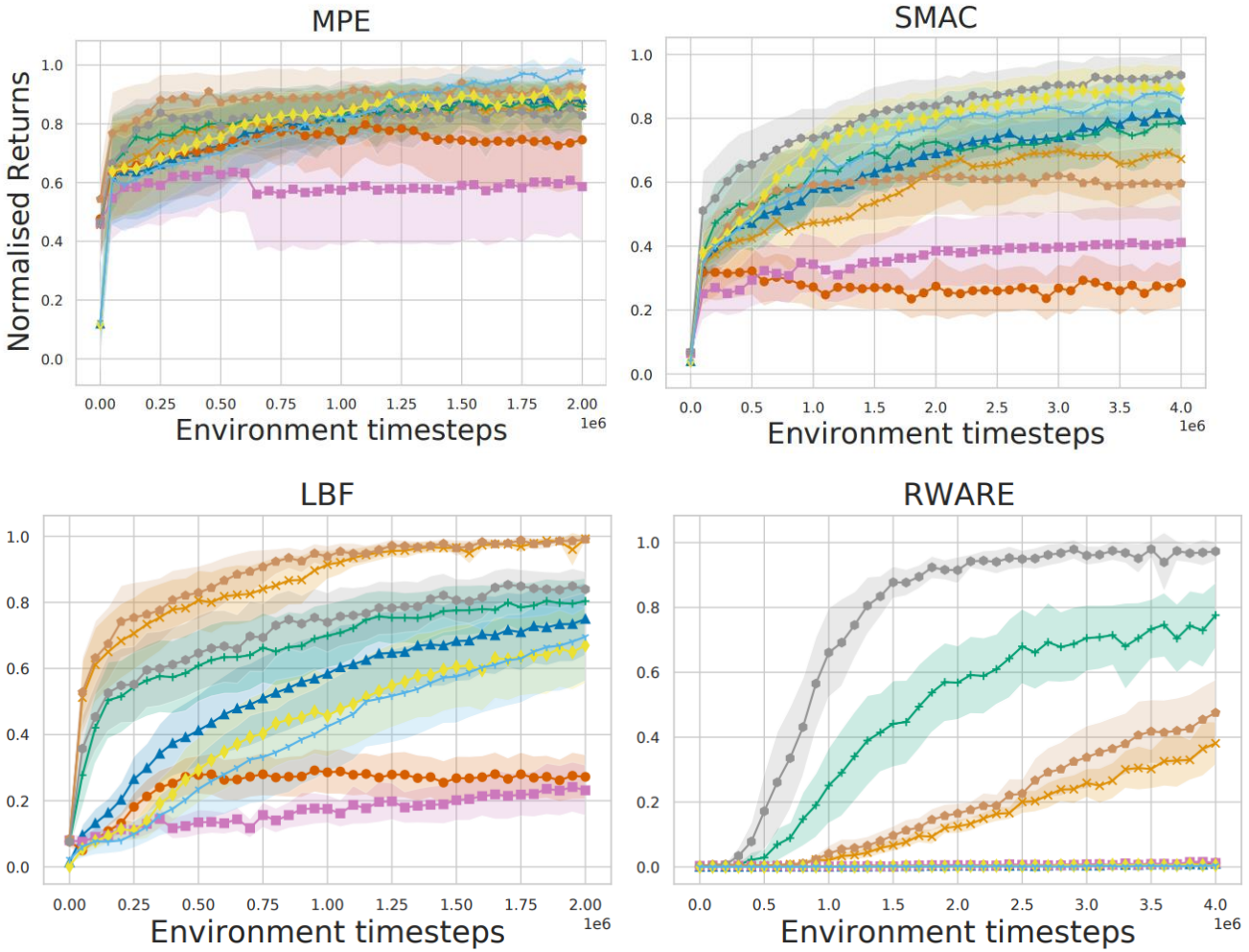
	$i=1$	$i=2$	$i=3$
t	$a_t^i\sim\pi^i$	$a_t^i\sim\pi^i$	$a_t^i\sim\pi^i$
$t+1$	$a_{t+1}^i\sim\pi^i$	$a_{t+1}^i\sim\pi^i$	$a_{t+1}^i\sim\pi^i$

	$i=1$	$i=2$	$i=3$
t	$a_t^i\sim\pi^i$		
$t+1$	$a_{t+1}^i\sim\pi^i$	$a_{t+1}^i\sim\pi^i$	$a_{t+1}^i\sim\pi^i$

Comparisons

Table 1: Overview of algorithms and their properties.

	Centr. Training	Off-/On-policy	Value-based	Policy-based
IQL	✗	Off	✓	✗
IA2C	✗	On	✓	✓
IPPO	✗	On	✓	✓
MADDPG	✓	Off	✓	✓
COMA	✓	On	✓	✓
MAA2C	✓	On	✓	✓
MAPPO	✓	On	✓	✓
VDN	✓	Off	✓	✗
QMIX	✓	Off	✓	✗



※ Implementation assuming that agents **share parameters**

Policy-based Cooperative MARL

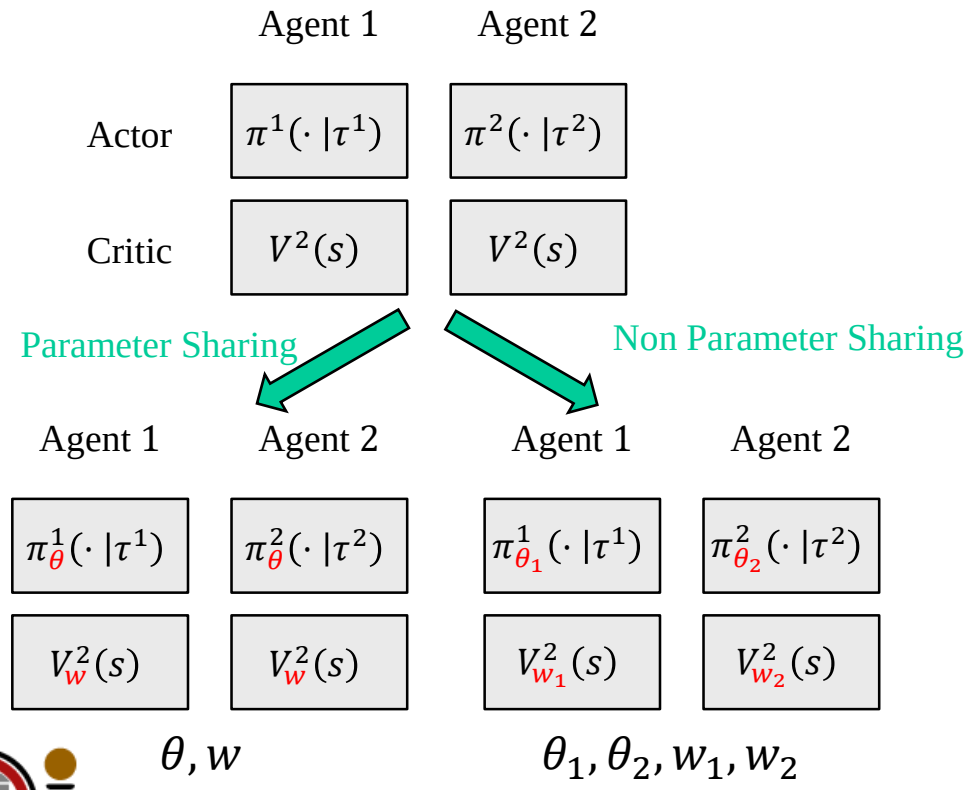
- IA2C, IPPO, MADDPG, COMA, MAA2C, MAPPO
- Parameter Sharing
- HATRPO/HAPPO
- Multi-Agent Transformer (MAT)

Parameter Sharing (PS)

- Two typical configuration:
  - Without parameter sharing:
    - Each agent uses its own set of parameters for its networks
  - With parameter sharing:
    - All agents share the same set of parameters for their networks
    - Typically, the policy and critic (if there is one) additionally receive the identity of each agent as a one-hot vector
    - Observation sizes and action sizes may need padding to ensure the same input dimensionality

Parameter Sharing (PS)

- Example: MAPPO
  - The authors of [mappo] implements parameter sharing in their paper

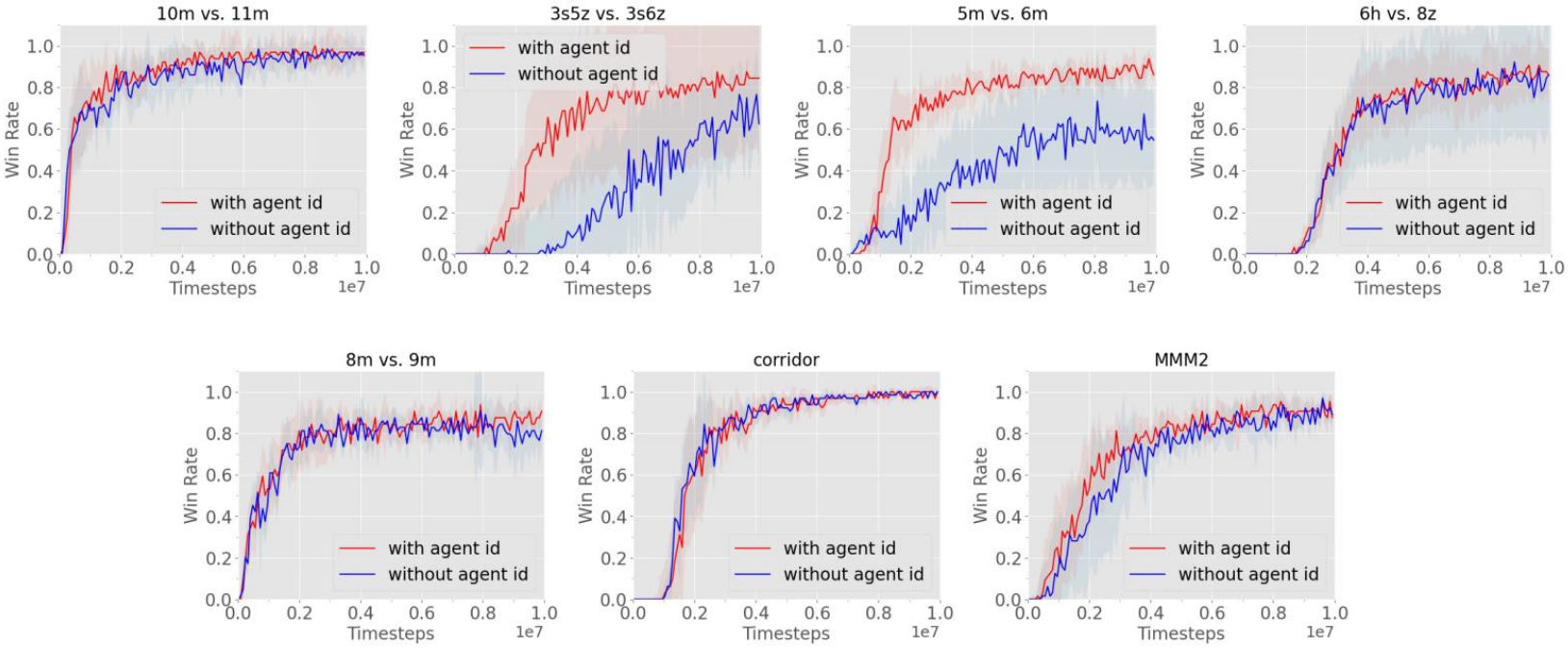


	(w/ PS)	(w/o PS)
Map	MAPPO	MAPPO-Ind
1c3s5z	**100.0(0.0)**	99.1(0.7)
2s3z	**100.0(0.7)**	99.1(0.9)
3s_vs_5z	**100.0(0.6)**	93.8(1.8)
3s5z	**96.9(0.7)**	80.4(3.3)
3s5z_vs_3s6z	84.4(34.0)	37.8(5.6)
5m_vs_6m	**89.1(2.5)**	44.4(2.9)
6h_vs_8z	**88.3(3.7)**	11.4(2.5)
10m_vs_11m	**96.9(4.8)**	78.4(2.7)
corridor	**100.0(1.2)**	82.2(1.8)
MMM2	90.6(2.8)	13.0(3.7)

SMAC evaluation win rate

Parameter Sharing (PS)

- Example: MAPPO
  - W/ or w/o agent id

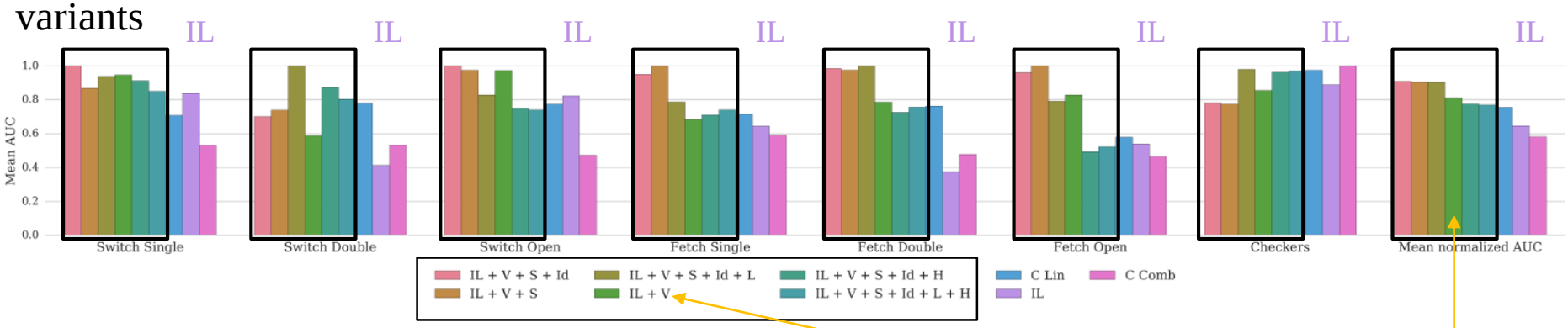


SMAC domain

Parameter Sharing (PS)

- VDN
  - Various combinations are tried in the experiments

VDN variants



"S" means parameter sharing here

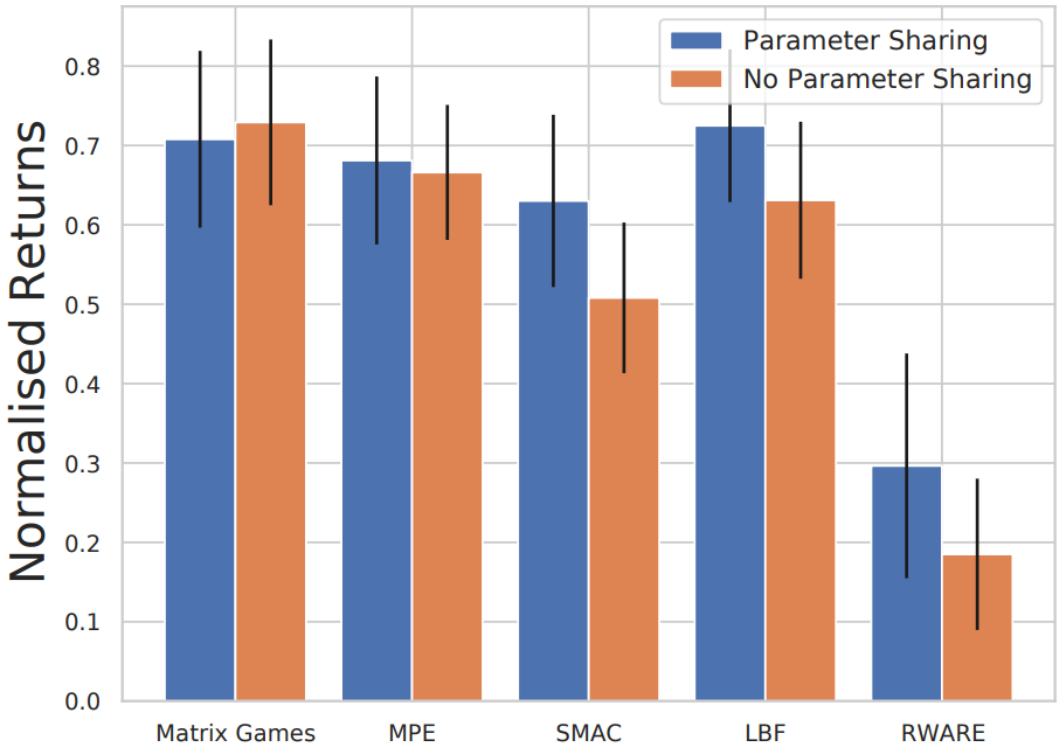
Without sharing

Parameter Sharing (PS)

- QMIX
  - Share parameter sharing in the paper's implementation
  - Agent id is concatenated onto each agent's observations

Parameter Sharing (PS)

- PS vs no PS in [bmarl]



Normalized maximum returns averaged over all algorithms with/without parameter sharing (with standard error)

Parameter Sharing (PS)

- Pros of PS
  - Decreases the number of trainable parameters
  - Share latent representation
- Cons of PS
  - Parameter sharing can act as an information bottleneck, especially in environments with heterogeneous agents
- The impact of parameter-sharing methods is dependent on the environments

Policy-based Cooperative MARL

- IA2C, IPPO, MADDPG, COMA, MAA2C, MAPPO
- Parameter Sharing
- HATRPO/HAPPO
- Multi-Agent Transformer (MAT)

Reference

- [hatrpo] (ICLR 2022)
  - Kuba, J. G., Chen, R., Wen, M., Wen, Y., Sun, F., Wang, J., & Yang, Y. (2022). Trust Region Policy Optimisation in Multi-Agent Reinforcement Learning. International Conference on Learning Representations. (https://openreview.net/forum?id=EcGGFkNTxdJ)

HATRPO/HAPPO

- Full names
  - Heterogeneous-Agent Trust Region Policy Optimization (HATPRO)
  - Heterogeneous-Agent Proximal Policy Optimization (HAPPO)
- Summary
  - Extend the theory of trust region learning to Coop MARL
  - Justify in theory the monotonic improvement property
  - Not need parameter sharing or assuming homogeneous agents

Limitation of Existing Trust Region Methods in Coop MARL

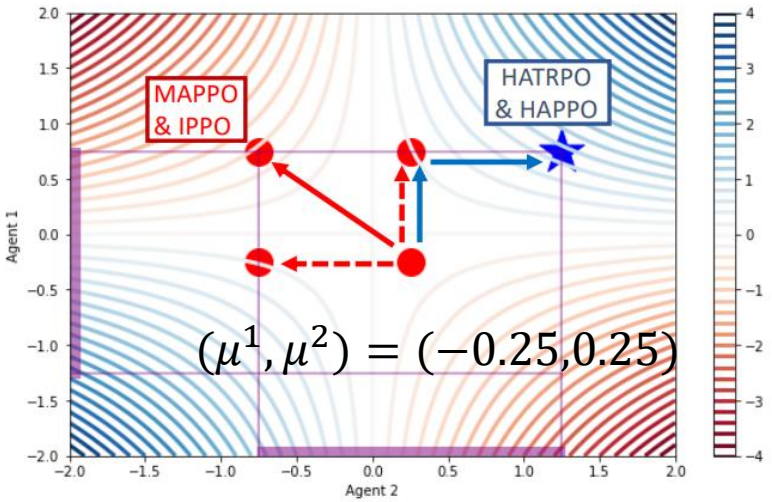
- Existing methods: IPPO, MAPPO
- Limitations:
  - IPPO/MAPPO use Parameter Sharing
    - May lead to suboptimal policies due to parameter sharing
      - Enforcing parameter sharing equals to putting a constraint: $\theta^i = \theta^j, \forall i, j \in \mathcal{N}$

    → HATRPO/HAPPO do not use parameter sharing
  - IPPO/MAPPO cannot offer a rigorous guarantee of monotonic improvement during training
    - Agents' local improvements in performance can jointly lead to a worse outcome
      - See the example in the next page

    → HATRPO/HAPPO offer practical solutions to apply trust region learning in MARL having the monotonic improvement property

Limitation of Existing Trust Region Methods in Coop MARL

- Two-player game settings
  - Action space: [-2, 2]
  - $r(a^1, a^2) = a^1 a^2$
  - Initial Gaussian policies
    - Agent 1: $\mu^1 = -0.25$
    - Agent 2: $\mu^2 = 0.25$



- Agents' local improvements in performance can jointly lead to a worse outcome using MAPPO or IPPO

Multi-Agent Trust Region Learning

- Notation
  - $i_{1:m}$: a set of m agents
  - $j_{1:k}$: a set of k agents
  - $i_{1:m}$ and $j_{1:k}$ are disjoint sets
  - $-i_{1:m}$ refers to $i_{1:m}$'s complement

Multi-Agent Value Function

- V/Q functions in Multi-Agent case
  - Multi-agent state-action value function

$$Q_{\boldsymbol{\pi}}^{i_{1:m}}\left(s, \boldsymbol{u}^{i_{1:m}}\right) \stackrel{\text{def}}{=} \mathbb{E}_{\boldsymbol{a}^{-i_{1:m}} \sim \boldsymbol{\pi}^{-i_{1:m}}}\left[Q_{\boldsymbol{\pi}}\left(s, \boldsymbol{u}^{i_{1:m}}, \boldsymbol{a}^{-i_{1:m}}\right)\right]$$

  - Multi-agent advantage function

$$A_{\boldsymbol{\pi}}^{i_{1:m}}\left(s, \boldsymbol{u}^{j_{1:k}}, \boldsymbol{u}^{i_{1:m}}\right) \stackrel{\text{def}}{=} Q_{\boldsymbol{\pi}}^{j_{1:k}, i_{1:m}}\left(s, \boldsymbol{u}^{j_{1:k}}, \boldsymbol{u}^{i_{1:m}}\right) - Q_{\boldsymbol{\pi}}^{j_{1:k}}\left(s, \boldsymbol{u}^{j_{1:k}}\right)$$

Multi-Agent Value Function

- V/Q functions in Single-Agent Case

$$V_\pi(s) = \mathbb{E}_{a_{0:\infty}\sim\pi, s_{1:\infty}\sim\mathcal{P}}[\sum_{j=0}^{\infty}\gamma^j R_j \,|s_0 = s] \quad Q_\pi(s,u) = \mathbb{E}_{s_{1:\infty}\sim\mathcal{P}, a_{1:\infty}\sim\pi}[\sum_{j=0}^{\infty}\gamma^j R_j \,|s_0 = s, a_0 = u]$$

Assume there are 3 agents

$$V_{\boldsymbol{\pi}}(s) \coloneqq \mathbb{E}_{\boldsymbol{a}_{0:\infty}\sim\boldsymbol{\pi}, s_{1:\infty}\sim\mathcal{P}}\left[\sum_{t=0}^{\infty}\gamma^t R_t | s_0 = s\right]$$

	$i=1$	$i=2$	$i=3$
t	$a_t^i \sim \pi^i$	$a_t^i \sim \pi^i$	$a_t^i \sim \pi^i$
$t+1$	$a_{t+1}^i \sim \pi^i$	$a_{t+1}^i \sim \pi^i$	$a_{t+1}^i \sim \pi^i$

$$Q_{\boldsymbol{\pi}}(s, \boldsymbol{u}) \coloneqq \mathbb{E}_{s_{1:\infty}\sim\mathcal{P}, \boldsymbol{a}_{1:\infty}\sim\boldsymbol{\pi}}\left[\sum_{t=0}^{\infty}\gamma^t R_t | s_0 = s, \boldsymbol{a}_0 = \boldsymbol{u}\right]$$

	$i=1$	$i=2$	$i=3$
t	$a_t^i = u^i$	$a_t^i = u^i$	$a_t^i = u^i$
$t+1$	$a_{t+1}^i \sim \pi^i$	$a_{t+1}^i \sim \pi^i$	$a_{t+1}^i \sim \pi^i$

$$Q_{\boldsymbol{\pi}}^{i_{1:m}}(s, \boldsymbol{u}^{i_{1:m}}) \overset{\text{def}}{=} \mathbb{E}_{\boldsymbol{a}^{-i_{1:m}}\sim\boldsymbol{\pi}^{-i_{1:m}}}\left[Q_{\boldsymbol{\pi}}(s, \boldsymbol{u}^{i_{1:m}}, \boldsymbol{a}^{-i_{1:m}})\right]$$

Let $i_{1:m}$ be the set containing agent {1}

	$i=1$	$i=2$	$i=3$
t	$a_t^i = u^i$	$a_t^i \sim \pi^i$	$a_t^i \sim \pi^i$
$t+1$	$a_{t+1}^i \sim \pi^i$	$a_{t+1}^i \sim \pi^i$	$a_{t+1}^i \sim \pi^i$

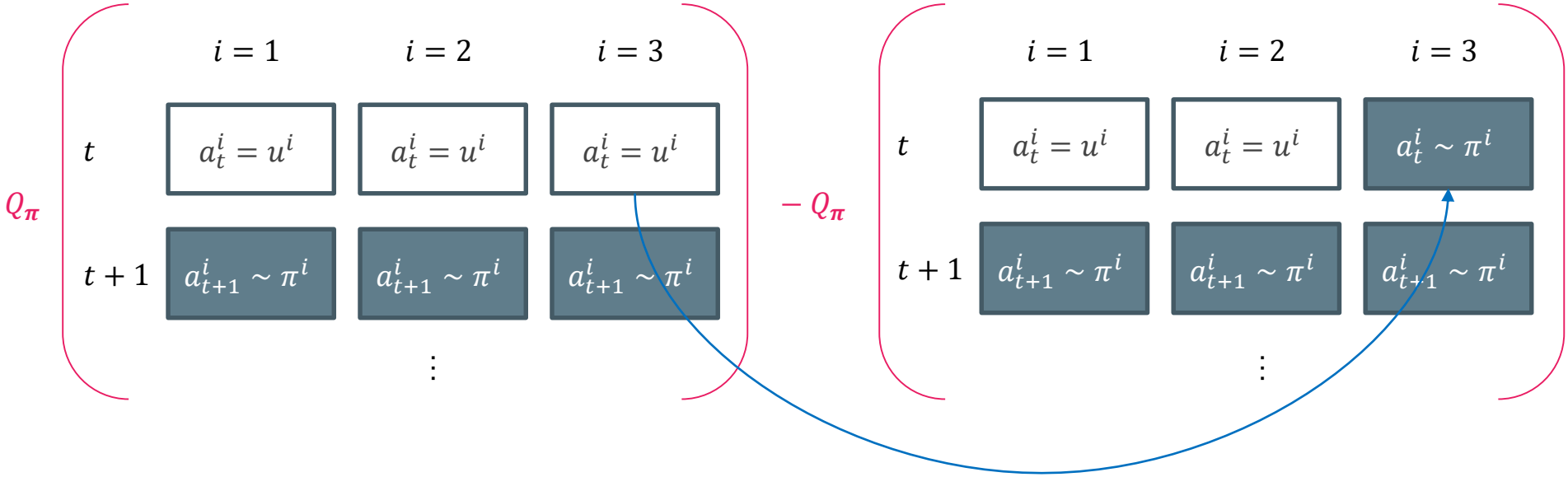
Let $i_{1:m}$ be the set containing agent {1,2}

	$i=1$	$i=2$	$i=3$
t	$a_t^i = u^i$	$a_t^i = u^i$	$a_t^i \sim \pi^i$
$t+1$	$a_{t+1}^i \sim \pi^i$	$a_{t+1}^i \sim \pi^i$	$a_{t+1}^i \sim \pi^i$

Multi-Agent Advantage Function

$$A_{\boldsymbol{\pi}}^{i_{1:m}}(s, \boldsymbol{u}^{j_{1:k}}, \boldsymbol{u}^{i_{1:m}}) \stackrel{\text{def}}{=} Q_{\boldsymbol{\pi}}^{j_{1:k}, i_{1:m}}(s, \boldsymbol{u}^{j_{1:k}}, \boldsymbol{u}^{i_{1:m}}) - Q_{\boldsymbol{\pi}}^{j_{1:k}}(s, \boldsymbol{u}^{j_{1:k}})$$

Assume $i_{1:m}$ contains a set of agent {3}
and $j_{1:k}$ contains a set of agent {1,2}



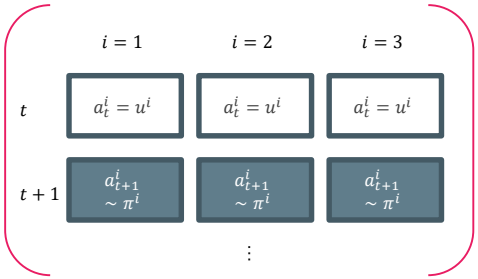
Effect of specifying one more action instead of random sampling

Multi-Agent Advantage Function

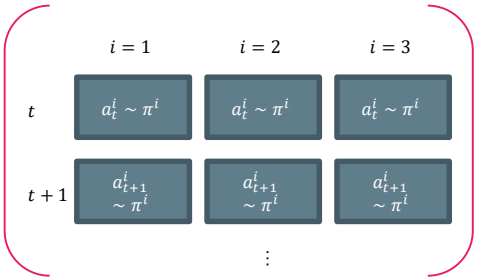
$$A_{\boldsymbol{\pi}}^{i_{1:m}}(s, \boldsymbol{u}^{j_{1:k}}, \boldsymbol{u}^{i_{1:m}}) \stackrel{\text{def}}{=} Q_{\boldsymbol{\pi}}^{j_{1:k}, i_{1:m}}(s, \boldsymbol{u}^{j_{1:k}}, \boldsymbol{u}^{i_{1:m}}) - Q_{\boldsymbol{\pi}}^{j_{1:k}}(s, \boldsymbol{u}^{j_{1:k}})$$

$$A_{\boldsymbol{\pi}}(s, \boldsymbol{u}) \coloneqq Q_{\boldsymbol{\pi}}(s, \boldsymbol{u}) - V_{\boldsymbol{\pi}}(s)$$
$$= Q_{\boldsymbol{\pi}}^{\phi, \{1,\dots,N\}}(s, \boldsymbol{u}^{\phi}, \boldsymbol{u}^{\{1,\dots,N\}}) - Q_{\boldsymbol{\pi}}^{\phi}(s, \boldsymbol{u}^{\phi})$$

$$A_{\boldsymbol{\pi}}^{\{1,2,3\}}(s, u^{\phi}, u^{\{1,2,3\}}) = Q_{\boldsymbol{\pi}}$$



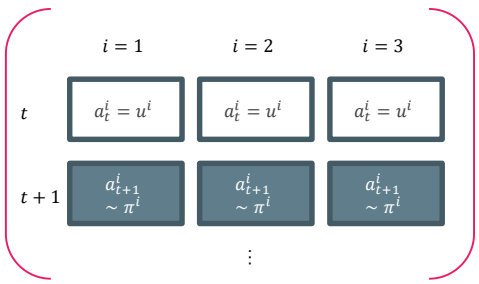
$$- Q_{\boldsymbol{\pi}}$$



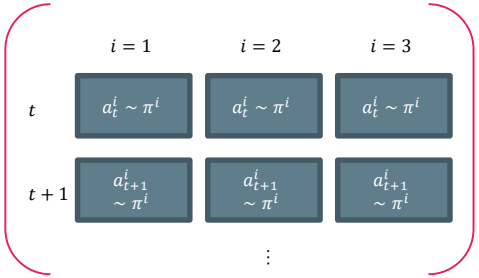
Multi-Agent Advantage Decomposition Lemma

$$A_{\boldsymbol{\pi}}^{i_{1:m}}\left(s, \boldsymbol{u}^{j_{1:k}}, \boldsymbol{u}^{i_{1:m}}\right) \stackrel{\text{def}}{=} Q_{\boldsymbol{\pi}}^{j_{1:k}, i_{1:m}}\left(s, \boldsymbol{u}^{j_{1:k}}, \boldsymbol{u}^{i_{1:m}}\right) - Q_{\boldsymbol{\pi}}^{j_{1:k}}\left(s, \boldsymbol{u}^{j_{1:k}}\right)$$

$$A_{\boldsymbol{\pi}}(s, \boldsymbol{u}) = Q_{\boldsymbol{\pi}}(s, \boldsymbol{u}) - V_{\boldsymbol{\pi}}(s) = A_{\boldsymbol{\pi}}^{\{1,2,3\}}\left(s, u^{\phi}, u^{\{1,2,3\}}\right) = Q_{\boldsymbol{\pi}}$$



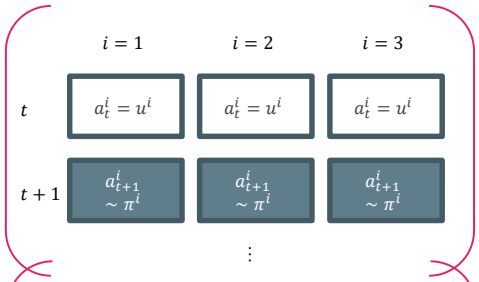
$- Q_{\boldsymbol{\pi}}$



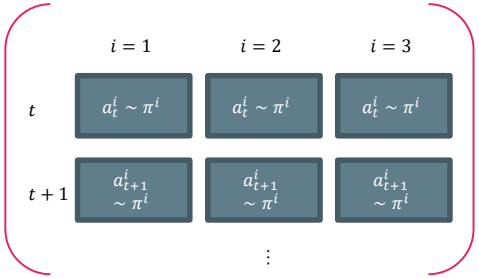
Multi-Agent Advantage Decomposition Lemma

$$A_{\pi}^{i_{1:m}}(s, \boldsymbol{u}^{j_{1:k}}, \boldsymbol{u}^{i_{1:m}}) \stackrel{\text{def}}{=} Q_{\pi}^{j_{1:k}, i_{1:m}}(s, \boldsymbol{u}^{j_{1:k}}, \boldsymbol{u}^{i_{1:m}}) - Q_{\pi}^{j_{1:k}}(s, \boldsymbol{u}^{j_{1:k}})$$

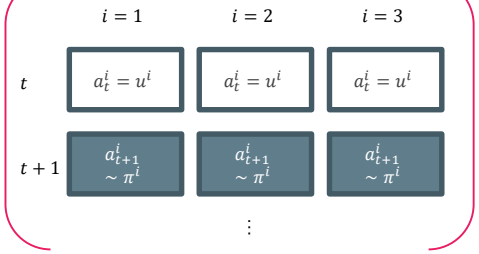
$$A_{\pi}(s, \boldsymbol{u}) =$$
$$Q_{\pi}(s, \boldsymbol{u}) - V_{\pi}(s) =$$
$$A_{\pi}^{\{1,2,3\}}(s, u^{\phi}, u^{\{1,2,3\}}) = Q_{\pi}$$



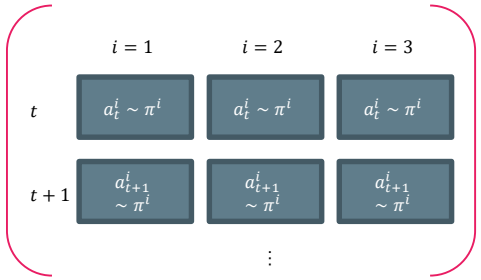
$- Q_{\pi}$



$= Q_{\pi}$



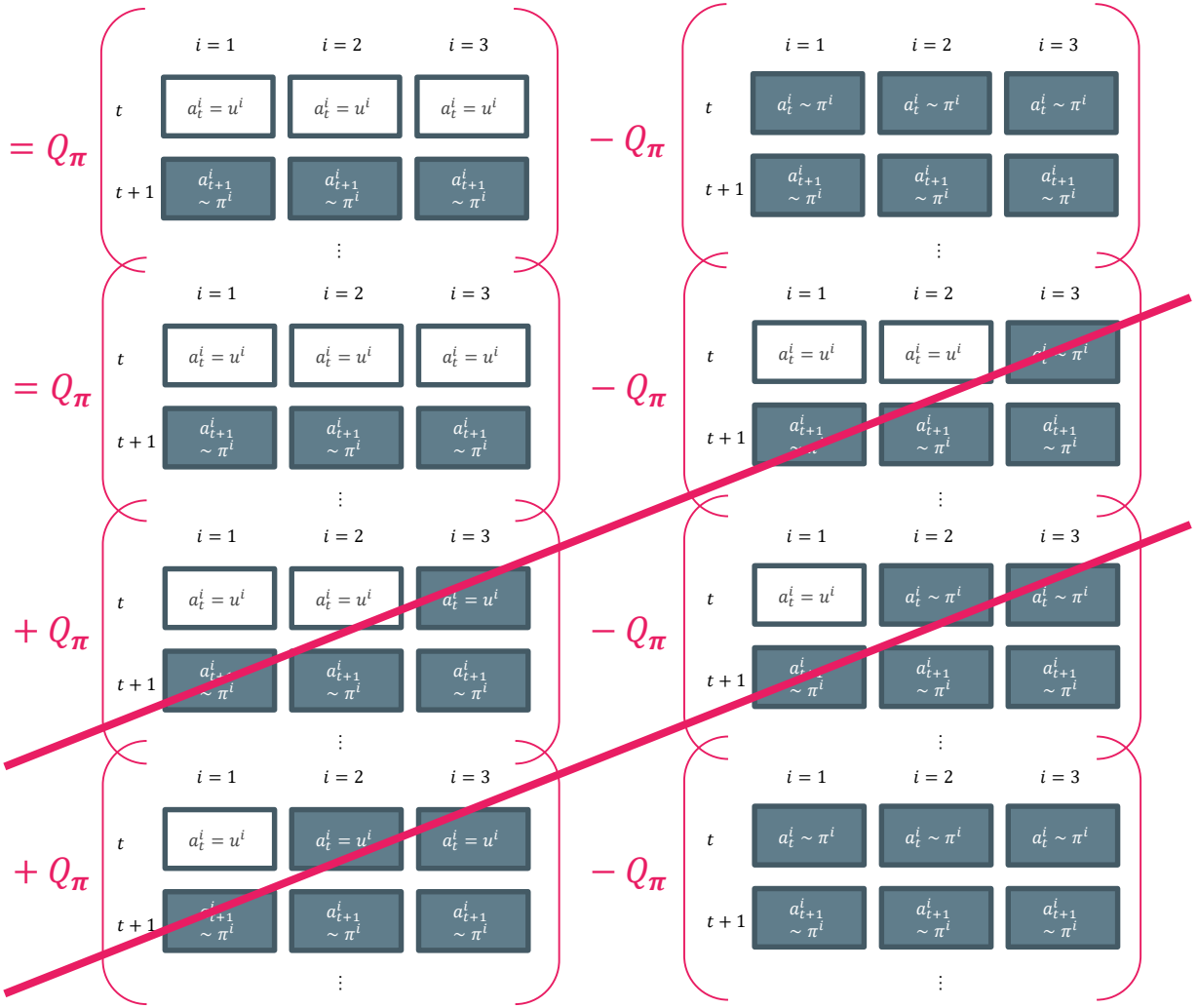
$- Q_{\pi}$



Multi-Agent Advantage Decomposition Lemma

$$A_{\pi}^{i_{1:m}}(s, \boldsymbol{u}^{j_{1:k}}, \boldsymbol{u}^{i_{1:m}}) \stackrel{\text{def}}{=} Q_{\pi}^{j_{1:k}, i_{1:m}}(s, \boldsymbol{u}^{j_{1:k}}, \boldsymbol{u}^{i_{1:m}}) - Q_{\pi}^{j_{1:k}}(s, \boldsymbol{u}^{j_{1:k}})$$

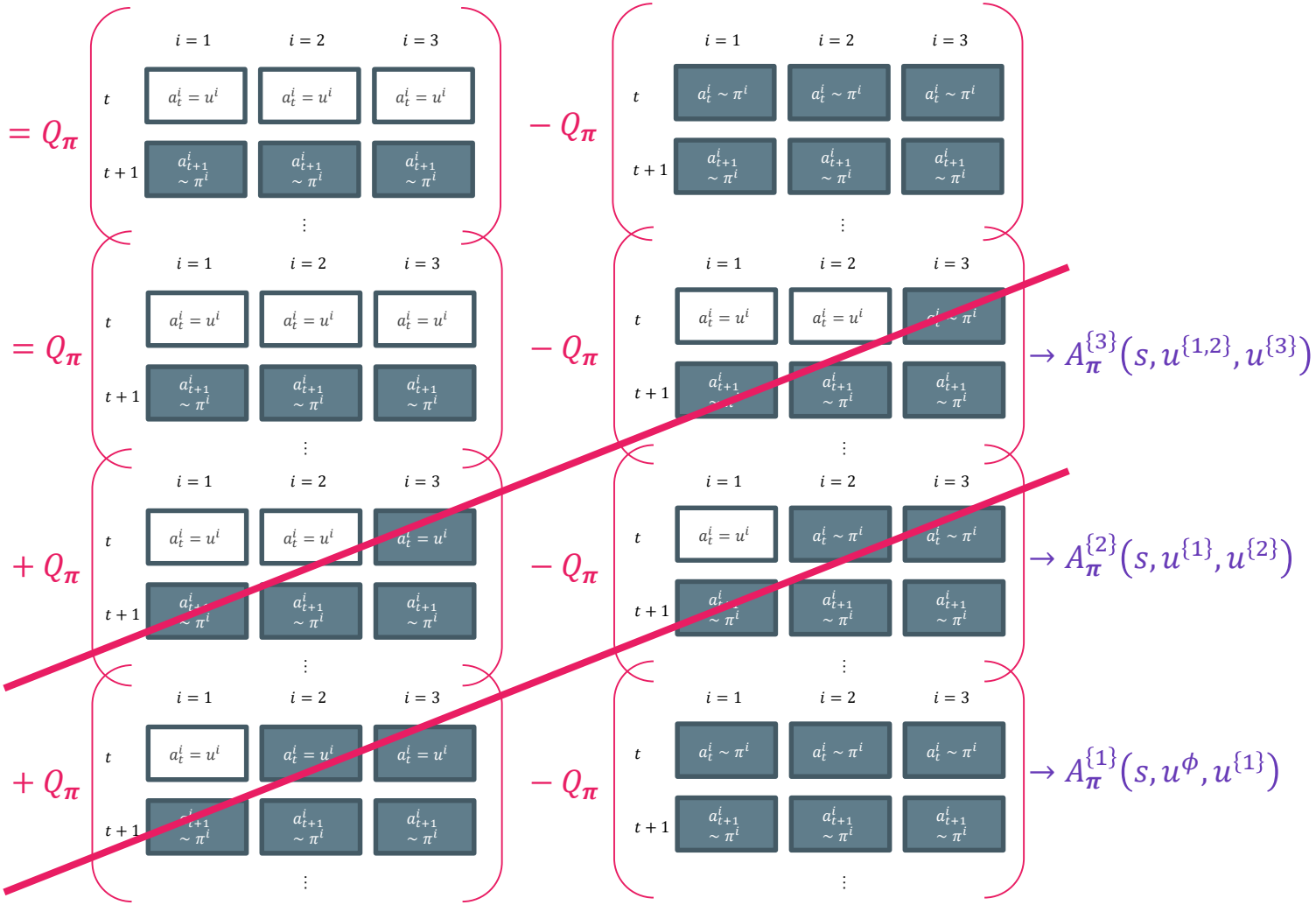
$$A_{\pi}(s, \boldsymbol{u}) =$$
$$Q_{\pi}(s, \boldsymbol{u}) - V_{\pi}(s) =$$
$$A_{\pi}^{\{1,2,3\}}(s, u^{\phi}, u^{\{1,2,3\}}) = Q_{\pi}$$



Multi-Agent Advantage Decomposition Lemma

$$A_{\pi}^{i_{1:m}}(s, \boldsymbol{u}^{j_{1:k}}, \boldsymbol{u}^{i_{1:m}}) \stackrel{\text{def}}{=} Q_{\pi}^{j_{1:k}, i_{1:m}}(s, \boldsymbol{u}^{j_{1:k}}, \boldsymbol{u}^{i_{1:m}}) - Q_{\pi}^{j_{1:k}}(s, \boldsymbol{u}^{j_{1:k}})$$

$$A_{\pi}(s, \boldsymbol{u}) =$$
$$Q_{\pi}(s, \boldsymbol{u}) - V_{\pi}(s) =$$
$$A_{\pi}^{\{1,2,3\}}(s, u^{\phi}, u^{\{1,2,3\}}) = Q_{\pi}$$



Multi-Agent Advantage Decomposition Lemma

$$A_{\boldsymbol{\pi}}^{i_{1:m}}(s, \boldsymbol{u}^{j_{1:k}}, \boldsymbol{u}^{i_{1:m}}) \stackrel{\text{def}}{=} Q_{\boldsymbol{\pi}}^{j_{1:k}, i_{1:m}}(s, \boldsymbol{u}^{j_{1:k}}, \boldsymbol{u}^{i_{1:m}}) - Q_{\boldsymbol{\pi}}^{j_{1:k}}(s, \boldsymbol{u}^{j_{1:k}})$$

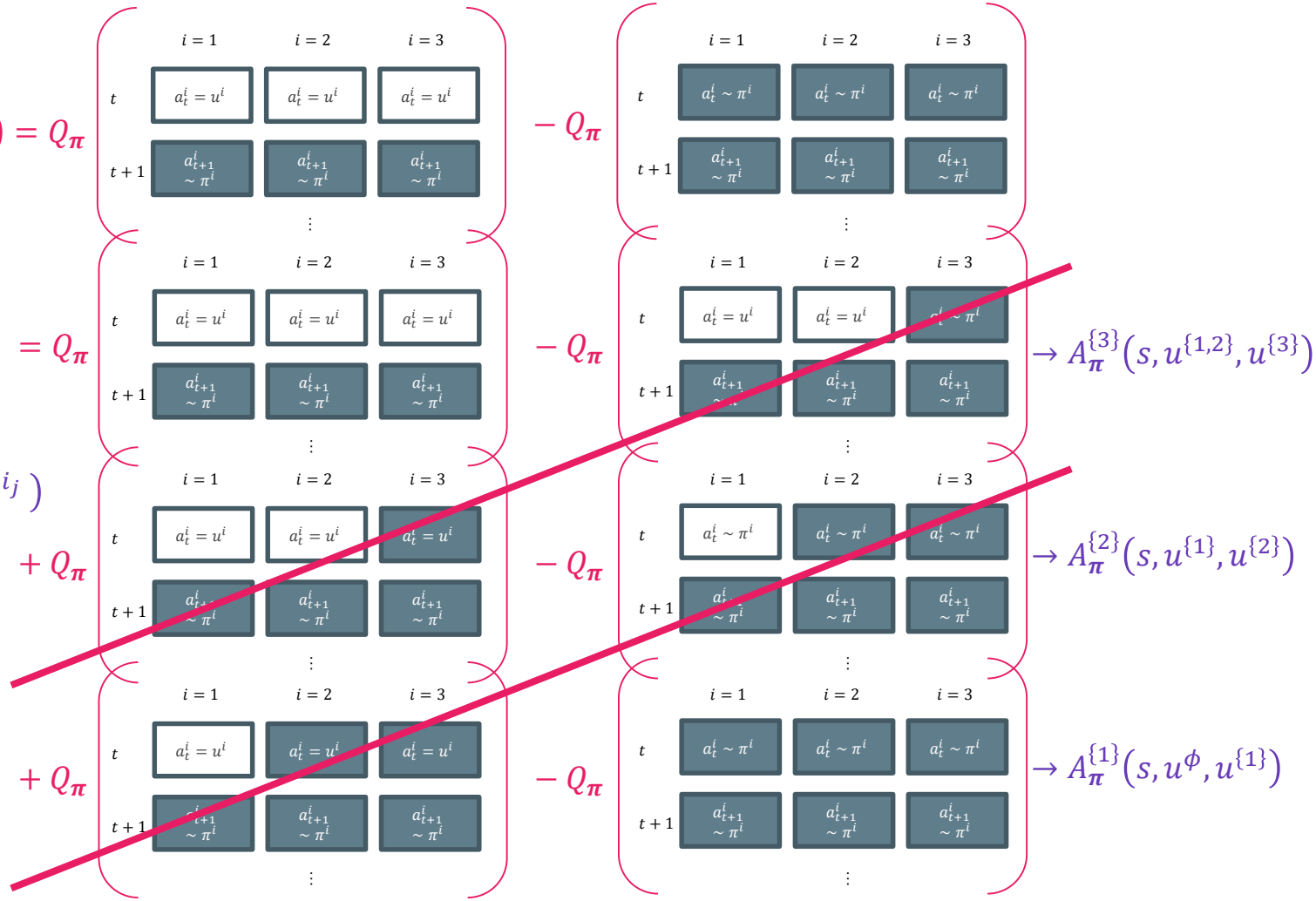
$$A_{\boldsymbol{\pi}}(s, \boldsymbol{u}) =$$
$$Q_{\boldsymbol{\pi}}(s, \boldsymbol{u}) - V_{\boldsymbol{\pi}}(s) =$$
$$A_{\boldsymbol{\pi}}^{\{1,2,3\}}(s, u^{\phi}, u^{\{1,2,3\}}) = Q_{\boldsymbol{\pi}}$$

$$= Q_{\boldsymbol{\pi}} - Q_{\boldsymbol{\pi}} \rightarrow A_{\boldsymbol{\pi}}^{\{3\}}(s, u^{\{1,2\}}, u^{\{3\}})$$

$$+ Q_{\boldsymbol{\pi}} - Q_{\boldsymbol{\pi}} \rightarrow A_{\boldsymbol{\pi}}^{\{2\}}(s, u^{\{1\}}, u^{\{2\}})$$

$$+ Q_{\boldsymbol{\pi}} - Q_{\boldsymbol{\pi}} \rightarrow A_{\boldsymbol{\pi}}^{\{1\}}(s, u^{\phi}, u^{\{1\}})$$

$$A_{\boldsymbol{\pi}}^{i_{1:m}}(s, \boldsymbol{u}^{i_{1:m}}) = \sum_{j=1}^{m} A_{\boldsymbol{\pi}}^{i_j}(s, \boldsymbol{u}^{i_{1:j-1}}, u^{i_j})$$



Multi-Agent Advantage Decomposition Lemma

- In any cooperative Markov games, given a joint policy, for any state s, and any agent subset $i_{1:m}$, the below equations holds.

$$A_{\boldsymbol{\pi}}^{i_{1:m}}\left(s, \boldsymbol{u}^{i_{1:m}}\right) = \sum_{j=1}^{m} A_{\boldsymbol{\pi}}^{i_j}\left(s, \boldsymbol{u}^{i_{1:j-1}}, u^{i_j}\right)$$

Improvement of the Joint Policy

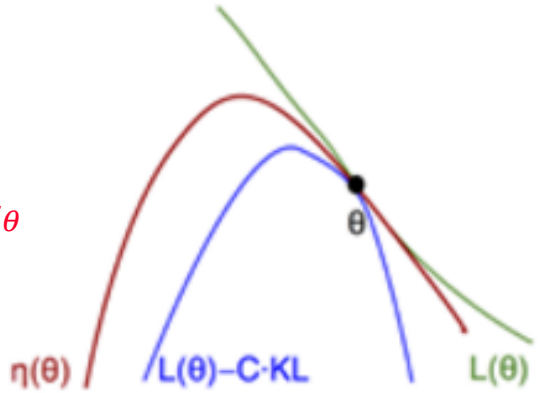
Lemma 8. *Let* $\boldsymbol{\pi} = \prod_{i=1}^{n} \pi^i$ *and* $\bar{\boldsymbol{\pi}} = \prod_{i=1}^{n} \bar{\pi}^i$ *be joint policies. Then*

$$\mathrm{D}_{KL}^{max}\left(\boldsymbol{\pi}, \bar{\boldsymbol{\pi}}\right) \leq \sum_{i=1}^{n} \mathrm{D}_{KL}^{max}\left(\pi^i, \bar{\pi}^i\right)$$

Definition 2. *Let* $\boldsymbol{\pi}$ *be a joint policy,* $\bar{\boldsymbol{\pi}}^{i_{1:m-1}} = \prod_{j=1}^{m-1} \bar{\pi}^{i_j}$ *be some* ***other*** *joint policy of agents* $i_{1:m-1}$*, and* $\hat{\pi}^{i_m}$ *be some* ***other*** *policy of agent* i_m*. Then*

$$L_{\boldsymbol{\pi}}^{i_{1:m}}\left(\bar{\boldsymbol{\pi}}^{i_{1:m-1}}, \hat{\pi}^{i_m}\right) \triangleq \mathbb{E}_{\mathrm{s}\sim\rho_{\boldsymbol{\pi}}, \mathbf{a}^{i_{1:m-1}}\sim\bar{\boldsymbol{\pi}}^{i_{1:m-1}}, \mathrm{a}^{i_m}\sim\hat{\pi}^{i_m}}\left[A_{\boldsymbol{\pi}}^{i_m}\left(\mathrm{s}, \mathbf{a}^{i_{1:m-1}}, \mathrm{a}^{i_m}\right)\right].$$

Recall L_{π_θ} and η_{π_θ} for TRPO/PPO (see RL4-3)



Improvement of the Joint Policy

Lemma 2. *Let $\boldsymbol{\pi}$ be a joint policy. Then, for any joint policy $\bar{\boldsymbol{\pi}}$, we have*

$$J(\bar{\boldsymbol{\pi}}) \geq J(\boldsymbol{\pi}) + \sum_{m=1}^{n} \left[L_{\boldsymbol{\pi}}^{i_{1:m}} \left(\bar{\boldsymbol{\pi}}^{i_{1:m-1}}, \bar{\pi}^{i_m} \right) - C \mathrm{D}_{KL}^{max}(\pi^{i_m}, \bar{\pi}^{i_m}) \right]$$

Proof. By Theorem 1

$$J(\bar{\boldsymbol{\pi}}) \geq L_{\boldsymbol{\pi}}(\bar{\boldsymbol{\pi}}) - C \mathrm{D}_{\mathrm{KL}}^{\max}(\boldsymbol{\pi}, \bar{\boldsymbol{\pi}}) \quad \text{(Schulman et al,2015a)}$$

$$= J(\boldsymbol{\pi}) + \mathbb{E}_{\mathrm{s} \sim \rho_{\boldsymbol{\pi}}, \mathbf{a} \sim \bar{\boldsymbol{\pi}}} \left[A_{\boldsymbol{\pi}}(\mathrm{s}, \mathbf{a}) \right] - C \mathrm{D}_{\mathrm{KL}}^{\max}(\boldsymbol{\pi}, \bar{\boldsymbol{\pi}})$$

which by Lemma 1 equals (Multi-Agent Advantage Decomposition Lemma)

$$= J(\boldsymbol{\pi}) + \mathbb{E}_{\mathrm{s} \sim \rho_{\boldsymbol{\pi}}, \mathbf{a} \sim \bar{\boldsymbol{\pi}}} \left[\sum_{m=1}^{n} A_{\boldsymbol{\pi}}^{i_m} \left(\mathrm{s}, \mathbf{a}^{i_{1:m-1}}, \mathrm{a}^{i_m} \right) \right] - C \mathrm{D}_{\mathrm{KL}}^{\max}(\boldsymbol{\pi}, \bar{\boldsymbol{\pi}})$$

and by Lemma 8 this is at least

$$\geq J(\boldsymbol{\pi}) + \mathbb{E}_{\mathrm{s} \sim \rho_{\boldsymbol{\pi}}, \mathbf{a} \sim \bar{\boldsymbol{\pi}}} \left[\sum_{m=1}^{n} A_{\boldsymbol{\pi}}^{i_m} \left(\mathrm{s}, \mathbf{a}^{i_{1:m-1}}, \mathrm{a}^{i_m} \right) \right] - \sum_{m=1}^{n} C \mathrm{D}_{\mathrm{KL}}^{\max}(\pi^{i_m}, \bar{\pi}^{i_m})$$

$$= J(\boldsymbol{\pi}) + \sum_{m=1}^{n} \mathbb{E}_{\mathrm{s} \sim \rho_{\boldsymbol{\pi}}, \mathbf{a}^{i_{1:m-1}} \sim \bar{\boldsymbol{\pi}}^{i_{1:m-1}}, \mathrm{a}^{i_m} \sim \bar{\pi}^{i_m}} \left[A_{\boldsymbol{\pi}}^{i_m} \left(\mathrm{s}, \mathbf{a}^{i_{1:m-1}}, \mathrm{a}^{i_m} \right) \right] - \sum_{m=1}^{n} C \mathrm{D}_{\mathrm{KL}}^{\max}(\pi^{i_m}, \bar{\pi}^{i_m})$$

$$= J(\boldsymbol{\pi}) + \sum_{m=1}^{n} \left(L_{\boldsymbol{\pi}}^{i_{1:m}} \left(\bar{\boldsymbol{\pi}}^{i_{1:m-1}}, \bar{\pi}^{i_m} \right) - C \mathrm{D}_{\mathrm{KL}}^{\max}(\pi^{i_m}, \bar{\pi}^{i_m}) \right).$$

Multi-Agent Policy Iteration with Monotonic Improvement Guarantee

Algorithm 1 Multi-Agent Policy Iteration with Monotonic Improvement Guarantee

1: Initialise the joint policy $\boldsymbol{\pi}_0 = (\pi_0^1, \ldots, \pi_0^n)$.
2: **for** $k = 0, 1, \ldots$ **do**
3: Compute the advantage function $A_{\boldsymbol{\pi}_k}(s, \boldsymbol{a})$ for all state-(joint)action pairs $(s, \boldsymbol{a})$.
4: Compute $\epsilon = \max_{s,\boldsymbol{a}} |A_{\boldsymbol{\pi}_k}(s, \boldsymbol{a})|$ and $C = \frac{4\gamma\epsilon}{(1-\gamma)^2}$.
5: Draw a permutaion $i_{1:n}$ of agents at random.
6: **for** $m = 1 : n$ **do**
7: Make an update $\pi_{k+1}^{i_m} = \arg\max_{\pi^{i_m}} \left[L_{\boldsymbol{\pi}_k}^{i_{1:m}} \left(\boldsymbol{\pi}_{k+1}^{i_{1:m-1}}, \pi^{i_m} \right) - C \mathrm{D}_{\mathrm{KL}}^{\max}(\pi_k^{i_m}, \pi^{i_m}) \right]$.
8: **end for**
9: **end for**

Theorem 2. *A sequence $(\boldsymbol{\pi}_k)_{k=0}^{\infty}$ of joint policies updated by Algorithm 1 has the monotonic improvement property, i.e., $J(\boldsymbol{\pi}_{k+1}) \geq J(\boldsymbol{\pi}_k)$ for all $k \in \mathbb{N}$.*

Converge to Nash Equilibrium?

- Converge due to the monotonic improvement ✔
  - But may not converge at a Nash Equilibrium

$$\pi_{k+1}^{i_m} = \arg\max_{\pi^{i_m}} \left[L_{\boldsymbol{\pi}_k}^{i_{1:m}} \left(\boldsymbol{\pi}_{k+1}^{i_{1:m-1}}, \pi^{i_m} \right) - C\mathrm{D}_{\mathrm{KL}}^{\max}(\pi_k^{i_m}, \pi^{i_m}) \right]$$

$$L_{\boldsymbol{\pi}}^{i_{1:m}} \left(\bar{\boldsymbol{\pi}}^{i_{1:m-1}}, \hat{\pi}^{i_m} \right) \triangleq \mathbb{E}_{\mathrm{s}\sim\rho_{\boldsymbol{\pi}}, \mathbf{a}^{i_{1:m-1}}\sim\bar{\boldsymbol{\pi}}^{i_{1:m-1}}, \mathrm{a}^{i_m}\sim\hat{\pi}^{i_m}} \left[A_{\boldsymbol{\pi}}^{i_m} \left(\mathrm{s}, \mathbf{a}^{i_{1:m-1}}, \mathrm{a}^{i_m} \right) \right]$$

  - Random permutation: lead to Nash Equilibrium

Definition 3. *In a fully-cooperative game, a joint policy* $\boldsymbol{\pi}_* = (\pi_*^1, \ldots, \pi_*^n)$ *is a Nash equilibrium (NE) if for every* $i \in \mathcal{N}$, $\pi^i \in \Pi^i$ *implies* $J(\boldsymbol{\pi}_*) \geq J\left(\pi^i, \boldsymbol{\pi}_*^{-i}\right)$.

Theorem 3. *Supposing in Algorithm 1 any permutation of agents has a fixed non-zero probability to begin the update, a sequence* $(\boldsymbol{\pi}_k)_{k=0}^{\infty}$ *of joint policies generated by the algorithm, in a cooperative Markov game, has a non-empty set of limit points, each of which is a Nash equilibrium.*

Practical Algorithms

Monotonic Improvement Policy Iteration (Single-Agent)

$$\pi_{k+1} = argmax_{\pi}\left(L_{\pi_k}(\pi) - CD_{KL}^{max}(\pi_k, \pi)\right)$$

$\pi_{k+1} = \arg\max_{\pi} L_{\pi_k}(\pi)$,
subject to $\mathbb{E}_{s\sim\rho_{\pi_k}}[D_{KL}(\pi_k, \pi)] \leq \delta$

TRPO

$\pi_{k+1} = \arg\max_{\pi} L_{\pi_k}^{PPO}(\pi)$, where $L_{\pi_k}^{PPO}(\pi) :=$
$\mathbb{E}_{s\sim\rho_{\pi_k}, a\sim\pi_k}[\min\left(\frac{\pi(a|s)}{\pi_k(a|s)}A_{\pi_k}(s,a), \text{clip}(\frac{\pi(a|s)}{\pi_k(a|s)} \pm \epsilon)A_{\pi_k}(s,a)\right)]$

PPO

Monotonic Improvement Policy Iteration (Multi-Agent)

$$\pi_{k+1}^{i_m} = \arg\max_{\pi^{i_m}}\left[L_{\boldsymbol{\pi}_k}^{i_{1:m}}\left(\boldsymbol{\pi}_{k+1}^{i_{1:m-1}}, \pi^{i_m}\right) - C\mathrm{D}_{\mathrm{KL}}^{\max}(\pi_k^{i_m}, \pi^{i_m})\right].$$

$\theta_{k+1}^{i_m} = \arg\max_{\theta^{i_m}} \mathbb{E}_{\mathrm{s}\sim\rho_{\boldsymbol{\pi}_{\boldsymbol{\theta}_k}}, \mathbf{a}^{i_{1:m-1}}\sim\boldsymbol{\pi}_{\boldsymbol{\theta}_{k+1}}^{i_{1:m-1}}, \mathrm{a}^{i_m}\sim\pi_{\theta^{i_m}}^{i_m}}\left[A_{\boldsymbol{\pi}_{\boldsymbol{\theta}_k}}^{i_m}(\mathrm{s}, \mathbf{a}^{i_{1:m-1}}, \mathrm{a}^{i_m})\right]$,
subject to $\mathbb{E}_{\mathrm{s}\sim\rho_{\boldsymbol{\pi}_{\boldsymbol{\theta}_k}}}\left[\mathrm{D}_{\mathrm{KL}}\left(\pi_{\theta_k^{i_m}}^{i_m}(\cdot|\mathrm{s}), \pi_{\theta^{i_m}}^{i_m}(\cdot|\mathrm{s})\right)\right] \leq \delta$.

HATRPO

$\mathbb{E}_{\mathrm{s}\sim\rho_{\boldsymbol{\pi}_{\boldsymbol{\theta}_k}}, \mathbf{a}\sim\boldsymbol{\pi}_{\boldsymbol{\theta}_k}}\left[\min\left(\frac{\pi_{\theta^{i_m}}^{i_m}(\mathrm{a}^i|\mathrm{s})}{\pi_{\theta_k^{i_m}}^{i_m}(\mathrm{a}^i|\mathrm{s})}M^{i_{1:m}}(\mathrm{s}, \mathbf{a}), \text{clip}\left(\frac{\pi_{\theta^{i_m}}^{i_m}(\mathrm{a}^i|\mathrm{s})}{\pi_{\theta_k^{i_m}}^{i_m}(\mathrm{a}^i|\mathrm{s})}, 1\pm\epsilon\right)M^{i_{1:m}}(\mathrm{s}, \mathbf{a})\right)\right]$

HAPPO

HATRPO

Algorithm 2 HATRPO

Input: Stepsize α, batch size B, number of: agents n, episodes K, steps per episode T, possible steps in line search L, line search acceptance threshold κ.
Initialize: Actor networks $\{\theta_0^i,\ \forall i \in \mathcal{N}\}$, Global V-value network $\{\phi_0\}$, Replay buffer $\mathcal{B}$
for $k = 0, 1, \ldots, K-1$ **do**
  Collect a set of trajectories by running the joint policy $\boldsymbol{\pi}_{\boldsymbol{\theta}_k} = (\pi^1_{\theta^1_k}, \ldots, \pi^n_{\theta^n_k})$.
  Push transitions $\{(o^i_t, a^i_t, o^i_{t+1}, r_t), \forall i \in \mathcal{N}, t \in T\}$ into $\mathcal{B}$.
  Sample a random minibatch of B transitions from $\mathcal{B}$.
  Compute advantage function $\hat{A}(\mathrm{s}, \mathbf{a})$ based on global V-value network with GAE.
  Draw a random permutation of agents $i_{1:n}$.
  Set $M^{i_1}(\mathrm{s}, \mathbf{a}) = \hat{A}(\mathrm{s}, \mathbf{a})$.
  for agent $i_m = i_1, \ldots, i_n$ **do**
    Estimate the gradient of the agent's maximisation objective

$$\hat{\boldsymbol{g}}_k^{i_m} = \frac{1}{B} \sum_{b=1}^{B} \sum_{t=1}^{T} \nabla_{\theta_k^{i_m}} \log \pi_{\theta_k^{i_m}}^{i_m}\left(a_t^{i_m} \mid o_t^{i_m}\right) M^{i_{1:m}}(s_t, \boldsymbol{a}_t).$$

    Use the conjugate gradient algorithm to compute the update direction

$$\boldsymbol{x}_k^{i_m} \approx (\hat{\boldsymbol{H}}_k^{i_m})^{-1} \boldsymbol{g}_k^{i_m},$$

    where $\hat{\boldsymbol{H}}_k^{i_m}$ is the Hessian of the average KL-divergence

$$\frac{1}{BT} \sum_{b=1}^{B} \sum_{t=1}^{T} \mathrm{D}_{\mathrm{KL}}\left(\pi_{\theta_k^{i_m}}^{i_m}(\cdot|o_t^{i_m}), \pi_{\theta^{i_m}}^{i_m}(\cdot|o_t^{i_m})\right).$$

    Estimate the maximal step size allowing for meeting the KL-constraint

$$\hat{\beta}_k^{i_m} \approx \sqrt{\frac{2\delta}{(\hat{\boldsymbol{x}}_k^{i_m})^T \hat{\boldsymbol{H}}_k^{i_m} \hat{\boldsymbol{x}}_k^{i_m}}}.$$

    Update agent i_m's policy by

$$\theta_{k+1}^{i_m} = \theta_k^{i_m} + \alpha^j \hat{\beta}_k^{i_m} \hat{\boldsymbol{x}}_k^{i_m},$$

    where $j \in \{0, 1, \ldots, L\}$ is the smallest such j which improves the sample loss by at least $\kappa \alpha^j \hat{\beta}_k^{i_m} \hat{\boldsymbol{x}}_k^{i_m} \cdot \hat{\boldsymbol{g}}_k^{i_m}$, found by the backtracking line search.
    Compute $M^{i_{1:m+1}}(\mathrm{s}, \mathbf{a}) = \frac{\pi_{\theta_{k+1}^{i_m}}^{i_m}(\mathrm{a}^{i_m}|\mathrm{o}^{i_m})}{\pi_{\theta_k^{i_m}}^{i_m}(\mathrm{a}^{i_m}|\mathrm{o}^{i_m})} M^{i_{1:m}}(\mathrm{s}_t, \mathbf{a}_t)$. //Unless $m = n$.
  end for
  Update V-value network by following formula:

$$\phi_{k+1} = \arg\min_{\phi} \frac{1}{BT} \sum_{b=1}^{B} \sum_{t=0}^{T} \left(V_\phi(s_t) - \hat{R}_t\right)^2$$

end for

HAPPO

Algorithm 3 HAPPO

Input: Stepsize α, batch size B, number of: agents n, episodes K, steps per episode T.
Initialize: Actor networks $\{\theta_0^i,\ \forall i \in \mathcal{N}\}$, Global V-value network $\{\phi_0\}$, Replay buffer $\mathcal{B}$
for $k = 0, 1, \ldots, K-1$ **do**
  Collect a set of trajectories by running the joint policy $\boldsymbol{\pi}_{\boldsymbol{\theta}_k} = (\pi^1_{\theta^1_k}, \ldots, \pi^n_{\theta^n_k})$.
  Push transitions $\{(o^i_t, a^i_t, o^i_{t+1}, r_t), \forall i \in \mathcal{N}, t \in T\}$ into $\mathcal{B}$.
  Sample a random minibatch of B transitions from $\mathcal{B}$.
  Compute advantage function $\hat{A}(\mathrm{s}, \mathbf{a})$ based on global V-value network with GAE.
  Draw a random permutation of agents $i_{1:n}$.
  Set $M^{i_1}(\mathrm{s}, \mathbf{a}) = \hat{A}(\mathrm{s}, \mathbf{a})$.
  for agent $i_m = i_1, \ldots, i_n$ **do**
    Update actor i^m with $\theta^{i_m}_{k+1}$, the argmax of the PPO-Clip objective

$$\frac{1}{BT}\sum_{b=1}^{B}\sum_{t=0}^{T}\min\left(\frac{\pi^{i_m}_{\theta^{i_m}}\left(a^{i_m}_t|o^{i_m}_t\right)}{\pi^{i_m}_{\theta^{i_m}_k}\left(a^{i_m}_t|o^{i_m}_t\right)}M^{i_{1:m}}(s_t, \boldsymbol{a}_t),\ \mathrm{clip}\left(\frac{\pi^{i_m}_{\theta^{i_m}}\left(a^{i_m}_t|o^{i_m}_t\right)}{\pi^{i_m}_{\theta^{i_m}_k}\left(a^{i_m}_t|o^{i_m}_t\right)}, 1\pm\epsilon\right)M^{i_{1:m}}(s_t, \boldsymbol{a}_t)\right).$$

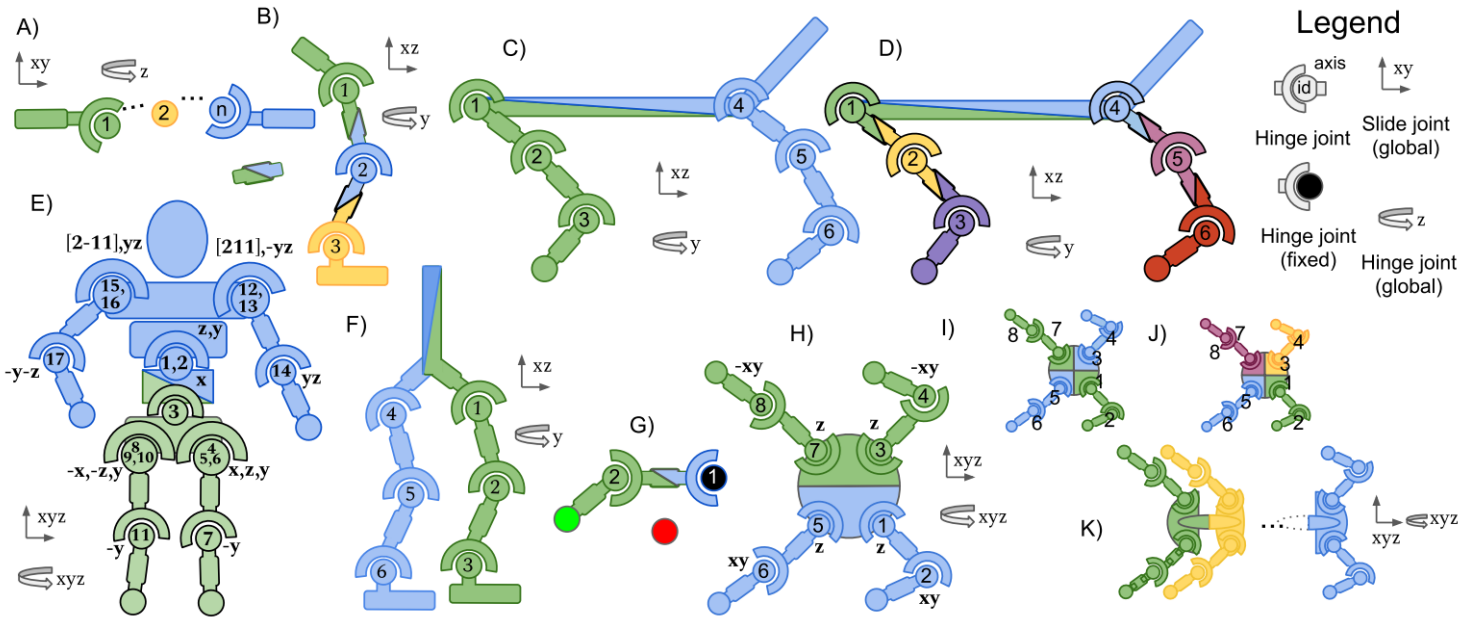
    Compute $M^{i_{1:m+1}}(\mathrm{s}, \mathbf{a}) = \frac{\pi^{i_m}_{\theta^{i_m}_{k+1}}\left(\mathrm{a}^{i_m}|\mathrm{o}^{i_m}\right)}{\pi^{i_m}_{\theta^{i_m}_k}\left(\mathrm{a}^{i_m}|\mathrm{o}^{i_m}\right)}M^{i_{1:m}}(\mathrm{s}, \mathbf{a})$. //Unless $m = n$.
  end for
  Update V-value network by following formula:

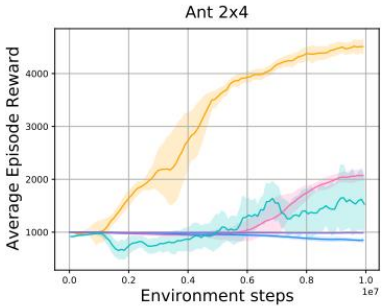
$$\phi_{k+1} = \arg\min_\phi \frac{1}{BT}\sum_{b=1}^{B}\sum_{t=0}^{T}\left(V_\phi(s_t) - \hat{R}_t\right)^2$$

end for

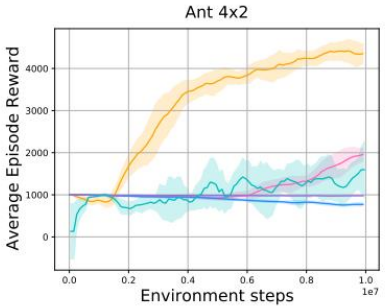
Experiment Result

- Multi-Agent MuJoCo
  - Models each part of a robot as an independent agent

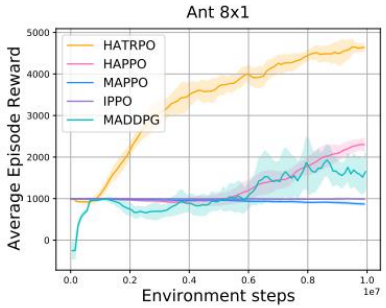




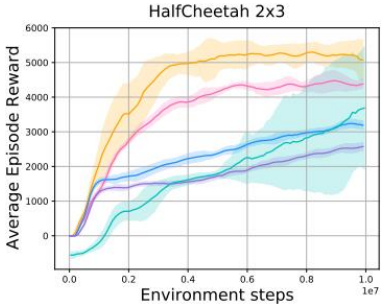
(a) 2x4-Agent Ant



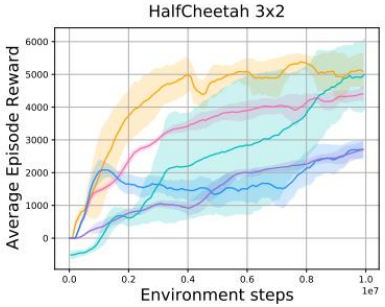
(b) 4x2-Agent Ant



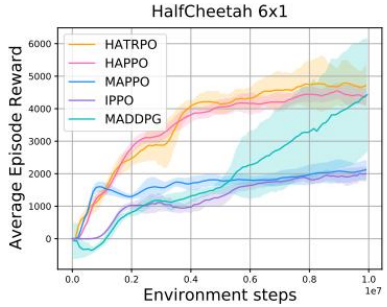
(c) 8x1-Agent Ant



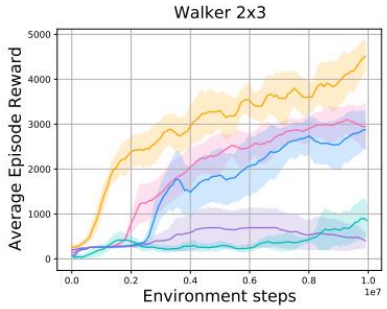
(d) 2x3-Agent HalfCheetah



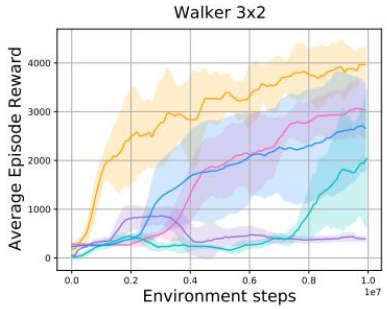
(e) 3x2-Agent HalfCheetah



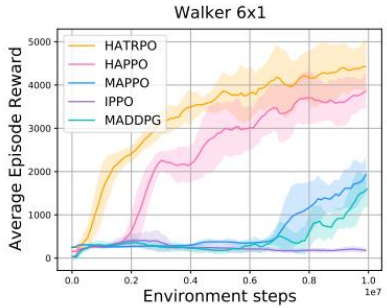
(f) 6x1-Agent HalfCheetah



(g) 2x3-Agent Walker



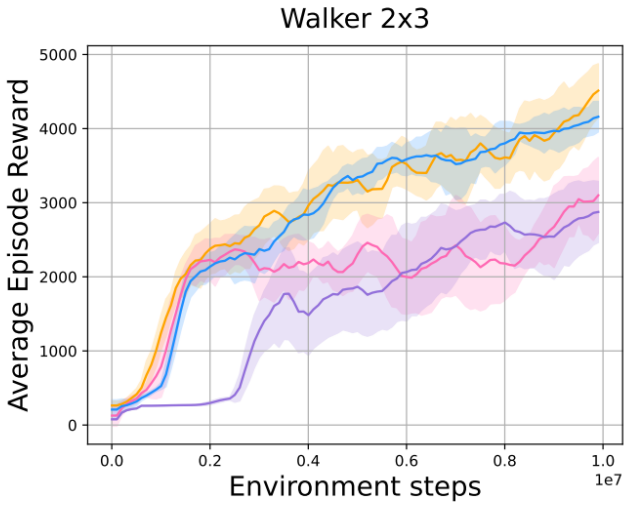
(h) 3x2-Agent Walker



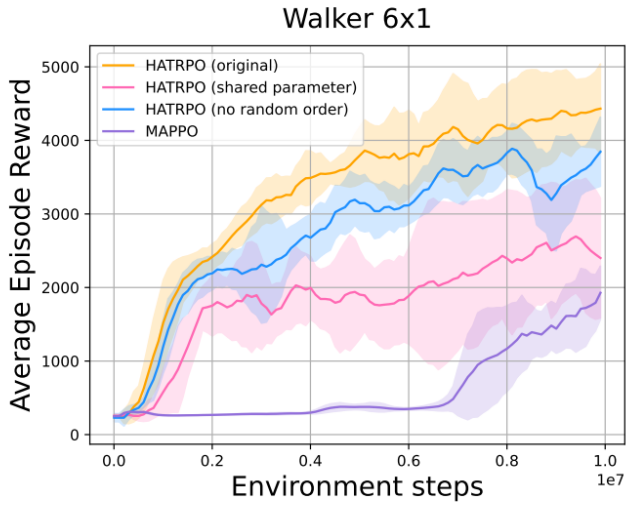
(i) 6x1-Agent Walker

Experiment Result

- Ablation
  - HATRPO without parameter sharing:
    - Outperform its counterpart
  - HATRPO with random permutation may be better



(a) 2-Agent Walker



(b) 6-Agent Walker

Policy-based Cooperative MARL

- IA2C, IPPO, MADDPG, COMA, MAA2C, MAPPO
- Parameter Sharing
- HATRPO/HAPPO
- Multi-Agent Transformer (MAT)

Reference

- [mat] (NeurIPS 2022)
  - Wen, Muning, et al. "Multi-agent reinforcement learning is a sequence modeling problem." Advances in Neural Information Processing Systems 35 (2022): 16509-16521. (https://proceedings.neurips.cc/paper_files/paper/2022/hash/69413f87e5a34897cd010ca698097d0a-Abstract-Conference.html)

Recall: Multi-Agent Advantage Decomposition Lemma

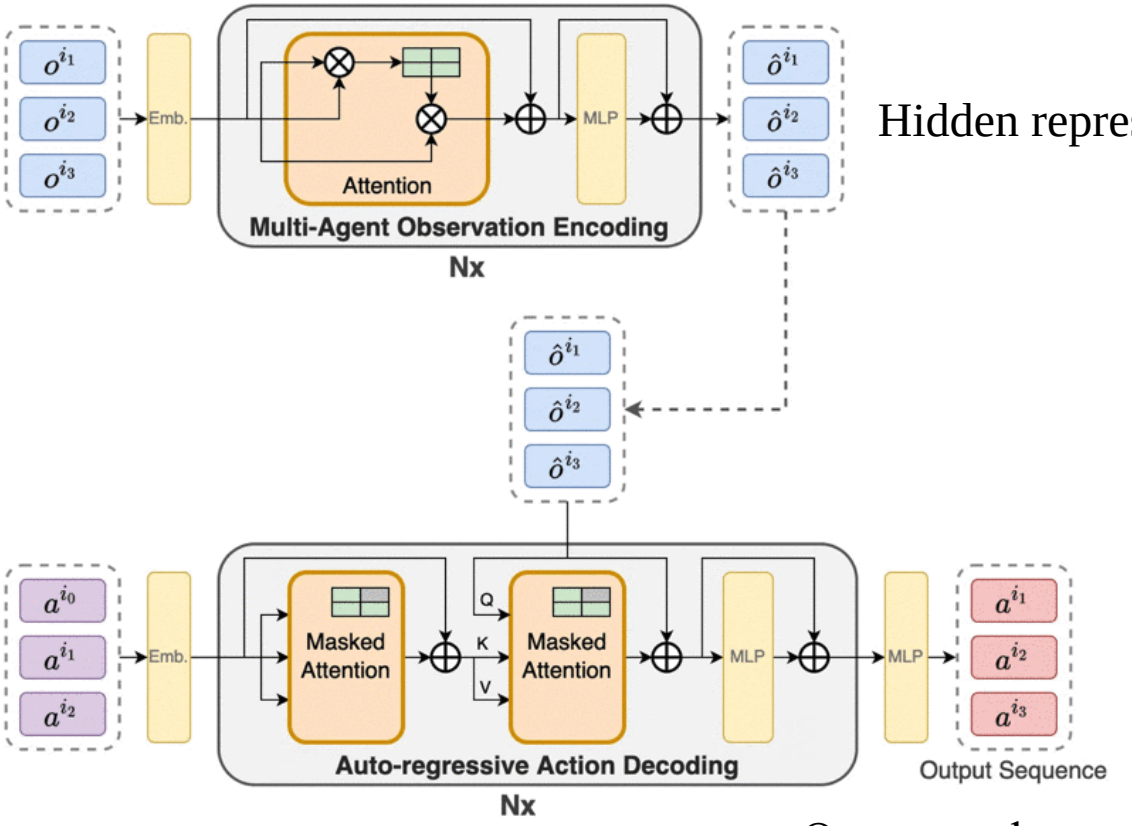
- In any cooperative Markov games, given a joint policy, for any state s, and any agent subset $i_{1:m}$, the below equations holds.

$$A_{\boldsymbol{\pi}}^{i_{1:m}}\left(s, \boldsymbol{a}^{i_{1:m}}\right) = \sum_{j=1}^{m} A_{\boldsymbol{\pi}}^{i_j}\left(s, \boldsymbol{a}^{i_{1:j-1}}, a^{i_j}\right)$$

Multi-Agent Transformer (MAT)

※n: number of agents

Input: each agent's observation

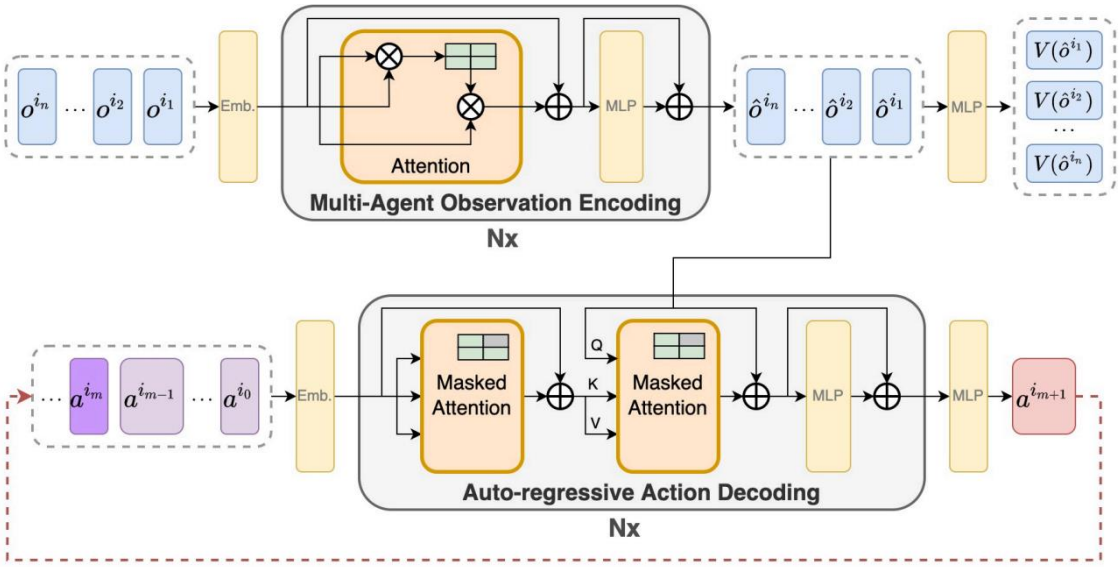


Hidden representation

Output each agent's action distribution auto-regressively

Multi-Agent Transformer (MAT)

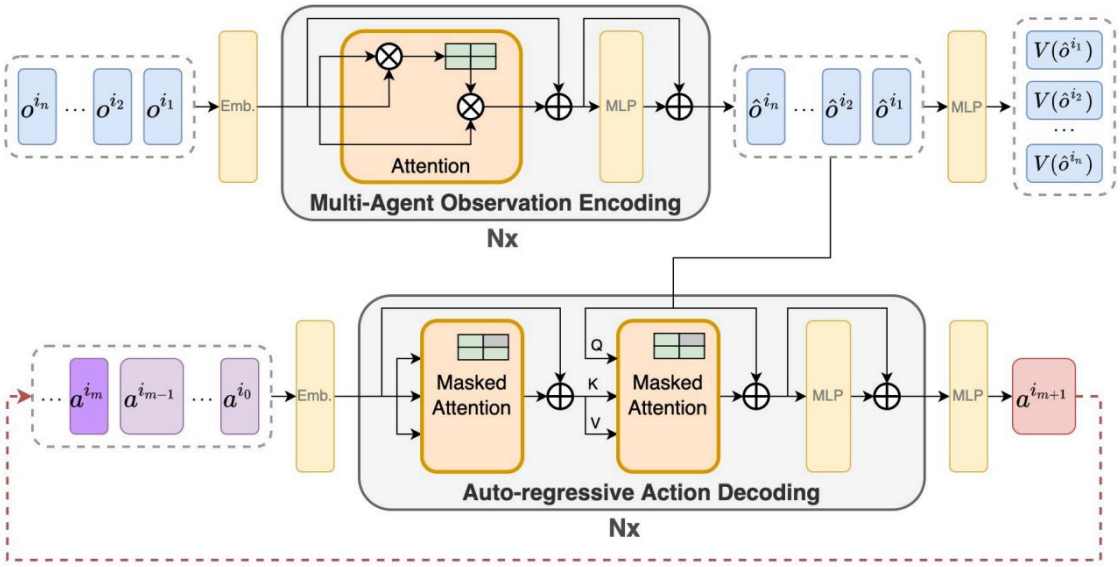
- Value function
  - $\hat{V}_t = \frac{1}{n}\sum_{m=1}^{n} V(\widehat{o_t^{i_m}})$



Multi-Agent Transformer (MAT)

- Encoder loss: critic loss
  - Loss of encoder: $$L_{\text{Encoder}}(\phi) = \frac{1}{Tn}\sum_{m=1}^{n}\sum_{t=0}^{T-1}\left[R(\mathbf{o}_t,\mathbf{a}_t) + \gamma V_{\bar{\phi}}(\hat{\mathbf{o}}_{t+1}^{i_m}) - V_{\phi}(\hat{\mathbf{o}}_t^{i_m})\right]^2$$

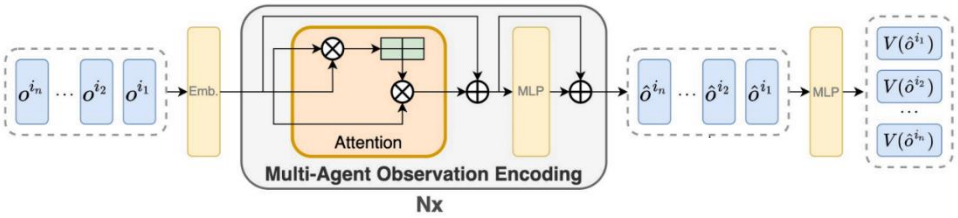
(TD Learning)



Multi-Agent Transformer (MAT)

- Encoder loss: critic loss
  - Loss of encoder: $L_{\text{Encoder}}(\phi) = \frac{1}{Tn}\sum_{m=1}^{n}\sum_{t=0}^{T-1}\left[R(\mathbf{o}_t, \mathbf{a}_t) + \gamma V_{\bar{\phi}}(\hat{\mathbf{o}}_{t+1}^{i_m}) - V_{\phi}(\hat{\mathbf{o}}_t^{i_m})\right]^2$

(TD Learning)



$R(\boldsymbol{o}_t, \boldsymbol{a}_t)$

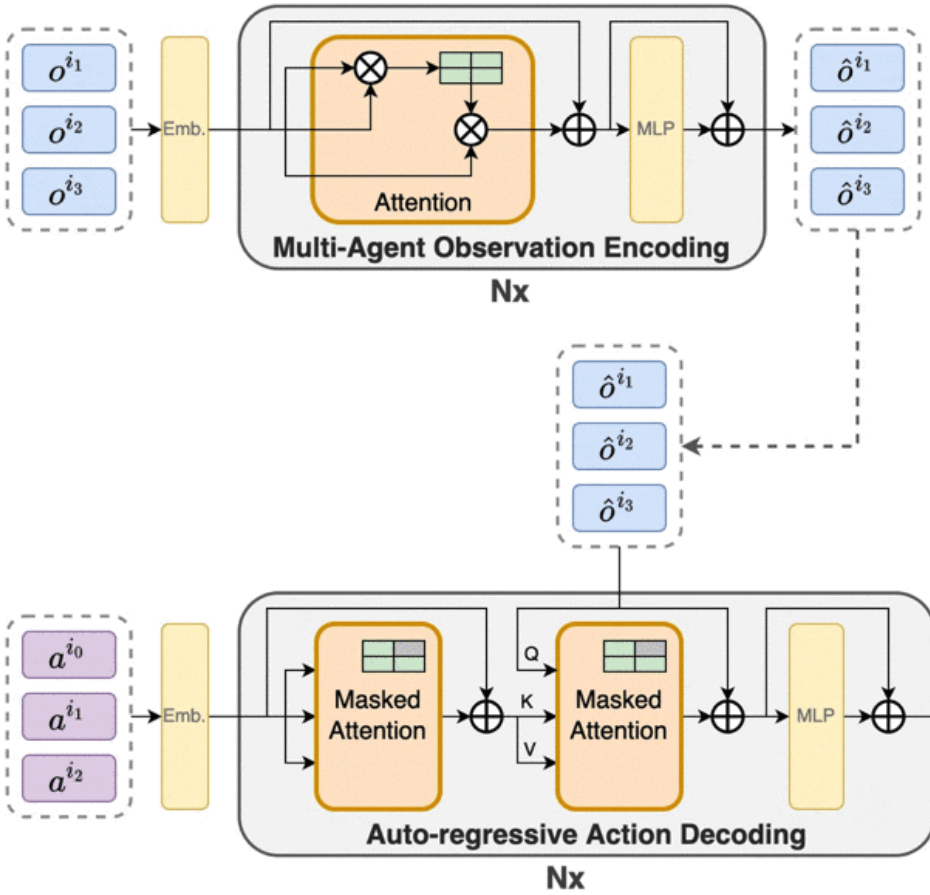
$+$

$\boldsymbol{o}_{t+1} = \begin{bmatrix} o_{t+1}^1 \\ o_{t+1}^2 \\ \vdots \\ o_{t+1}^N \end{bmatrix}$ Encoder $\widehat{\boldsymbol{o}_{t+1}} = \begin{bmatrix} \widehat{o_{t+1}^1} \\ \widehat{o_{t+1}^2} \\ \vdots \\ \widehat{o_{t+1}^N} \end{bmatrix}$ MLP $V(\widehat{\boldsymbol{o}_{t+1}}) = \begin{bmatrix} V\left(\widehat{o_{t+1}^1}\right) \\ V\left(\widehat{o_{t+1}^2}\right) \\ \vdots \\ V\left(\widehat{o_{t+1}^N}\right) \end{bmatrix}$

$-$

$\boldsymbol{o}_t = \begin{bmatrix} o_t^1 \\ o_t^2 \\ \vdots \\ o_t^N \end{bmatrix}$ $\widehat{\boldsymbol{o}_t} = \begin{bmatrix} \widehat{o_t^1} \\ \widehat{o_t^2} \\ \vdots \\ \widehat{o_t^N} \end{bmatrix}$ $V(\widehat{\boldsymbol{o}_t}) = \begin{bmatrix} V\left(\widehat{o_t^1}\right) \\ V\left(\widehat{o_t^2}\right) \\ \vdots \\ V\left(\widehat{o_t^N}\right) \end{bmatrix}$

Multi-Agent Transformer (MAT)



Multi-Agent Advantage Decomposition Lemma

$$A_{\boldsymbol{\pi}}^{i_{1:N}}(s, \boldsymbol{a}^{i_{1:N}}) = \sum_{j=1}^{N} A_{\boldsymbol{\pi}}^{i_j}(s, \boldsymbol{a}^{i_{1:j-1}}, a^{i_j})$$

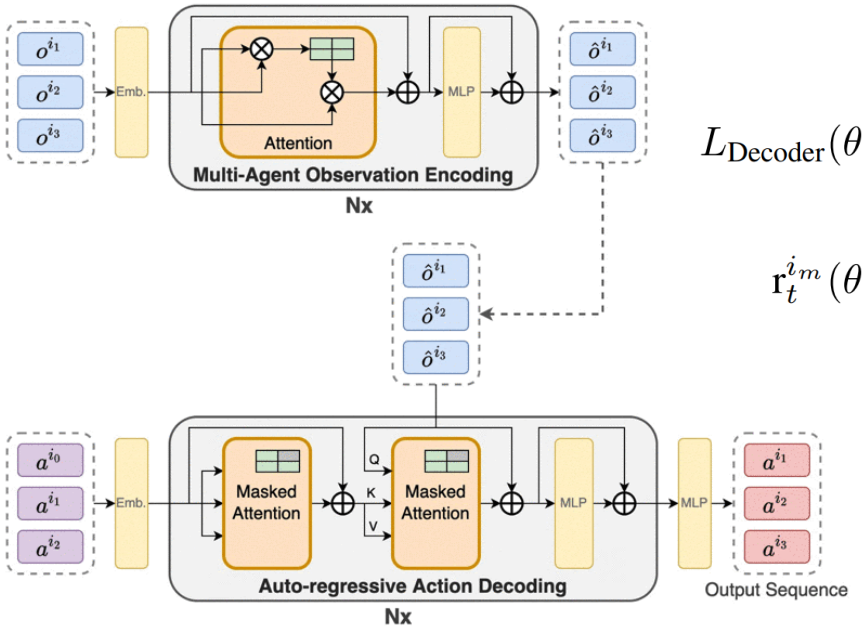
$$A_{\boldsymbol{\pi}}^{i_1}(s, a^{i_0}, a^{i_1})$$

$$A_{\boldsymbol{\pi}}^{i_2}(s, a^{i_0}, a^{i_1}, a^{i_2})$$

$$A_{\boldsymbol{\pi}}^{i_3}(s, a^{i_0}, a^{i_1}, a^{i_2}, a^{i_3})$$

Multi-Agent Transformer (MAT)

- Decoder loss: actor loss
  - Can use GAE to estimate $\widehat{A_t}$ with $\hat{V}_t = \frac{1}{n}\sum_{m=1}^{n} V(\widehat{o_t^{i_m}})$



$$L_{\text{Decoder}}(\theta) = -\frac{1}{Tn}\sum_{m=1}^{n}\sum_{t=0}^{T-1}\min\left(\mathrm{r}_t^{i_m}(\theta)\hat{A}_t, \mathrm{clip}(\mathrm{r}_t^{i_m}(\theta), 1\pm\epsilon)\hat{A}_t\right)$$

$$\mathrm{r}_t^{i_m}(\theta) = \frac{\pi_\theta^{i_m}(\mathrm{a}_t^{i_m}|\hat{\mathbf{o}}_t^{i_{1:n}}, \hat{\mathbf{a}}_t^{i_{1:m-1}})}{\pi_{\theta_{\text{old}}}^{i_m}(\mathrm{a}_t^{i_m}|\hat{\mathbf{o}}_t^{i_{1:n}}, \hat{\mathbf{a}}_t^{i_{1:m-1}})},$$

(Similar to PPO-Clip)

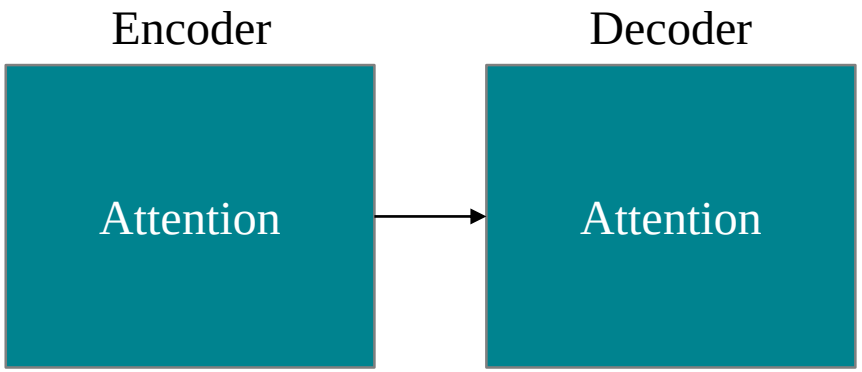
Multi-Agent Transformer (MAT)

Algorithm 1 Multi-Agent Transformer

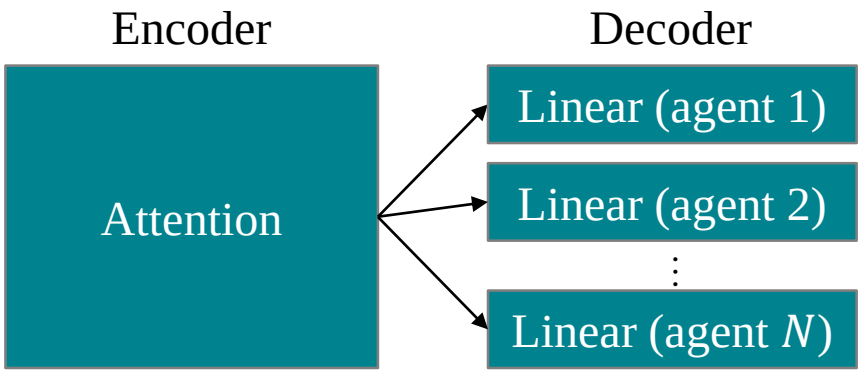
Input: Stepsize α, batch size B, number of agents n, episodes K, steps per episode T.
Initialize: Encoder $\{\phi_0\}$, Decoder $\{\theta_0\}$, Replay buffer $\mathcal{B}$.
for $k = 0, 1, \ldots, K-1$ **do**
  for $t = 0, 1, \ldots, T-1$ **do**
    Collect a sequence of observations $o_t^{i_1}, \ldots, o_t^{i_n}$ from environments.
    {// The Inference Phase}
    Generate representation sequence $\hat{o}_t^{i_1}, \ldots, \hat{o}_t^{i_n}$ by feeding observations to the encoder.
    Input $\hat{o}_t^{i_1}, \ldots, \hat{o}_t^{i_n}$ to the decoder.
    for $m = 0, 1, \ldots, n-1$ **do**
      Input $a_t^{i_0}, \ldots, a_t^{i_m}$ and infer $a_t^{i_{m+1}}$ with the auto-regressive decoder.
    end for
    Execute joint actions $a_t^{i_0}, \ldots, a_t^{i_n}$ in environments and collect the reward $R(\boldsymbol{o}_t, \boldsymbol{a}_t)$.
    Insert $(\boldsymbol{o}_t, \boldsymbol{a}_t, R(\boldsymbol{o}_t, \boldsymbol{a}_t))$ in to $\mathcal{B}$.
  end for
  {// The Training Phase}
  Sample a random minibatch of B steps from $\mathcal{B}$.
  Generate $V_\phi(\hat{o}^{i_1}), \ldots, V_\phi(\hat{o}^{i_n})$ with the output layer of the encoder.
  Calculate $L_{\text{Encoder}}(\phi)$ with Equation (4).
  Compute the joint advantage function $\hat{A}$ based on $V_\phi(\hat{o}^{i_1}), \ldots, V_\phi(\hat{o}^{i_n})$ with GAE.
  Input $\hat{o}^{i_1}, \ldots, \hat{o}^{i_n}$ and $a^{i_0}, \ldots, a^{i_{n-1}}$, generate $\pi_\theta^{i_1}, \ldots, \pi_\theta^{i_n}$ at once with the decoder.
  Calculate $L_{\text{Decoder}}(\theta)$ with Equation (5).
  Update the encoder and decoder by minimising $L_{\text{Encoder}}(\phi) + L_{\text{Decoder}}(\theta)$ with gradient descent.
end for

Multi-Agent Transformer (MAT)

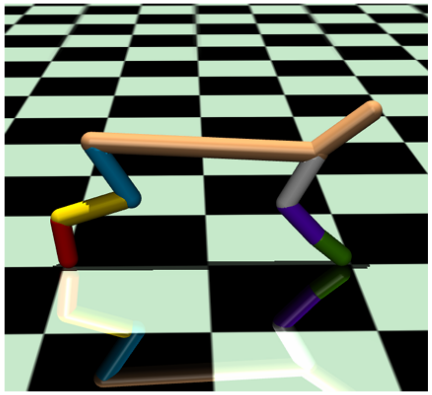
- MAT



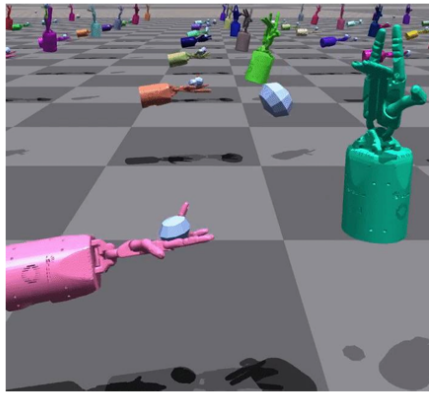
- MAT-Dec



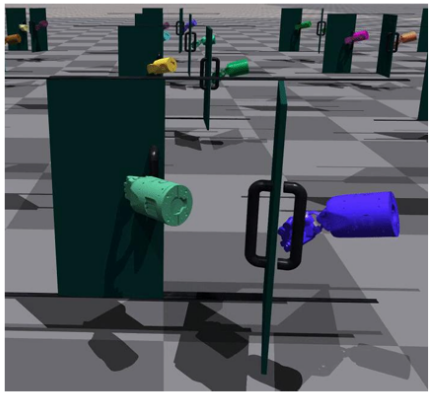
Experiment Results



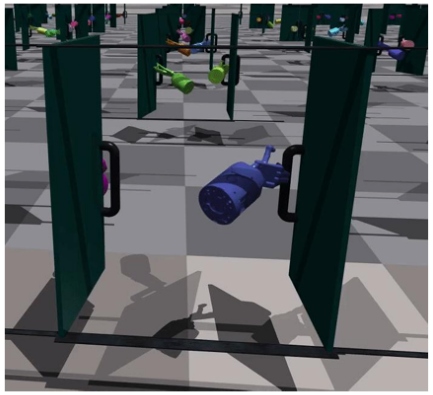
(a) HalfCheetah



(b) CatchOver2Underarm

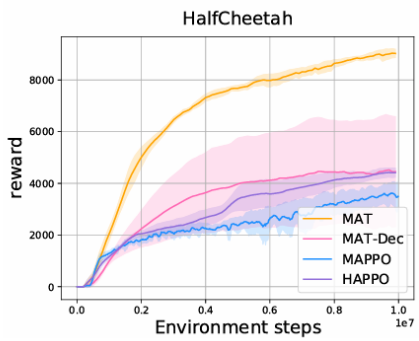


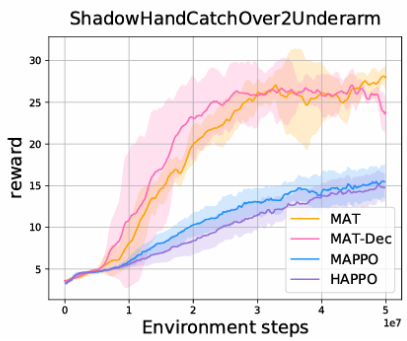
(c) DoorOpenInward

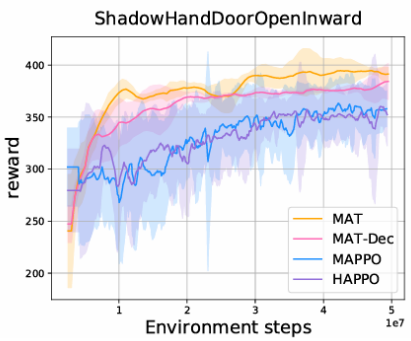


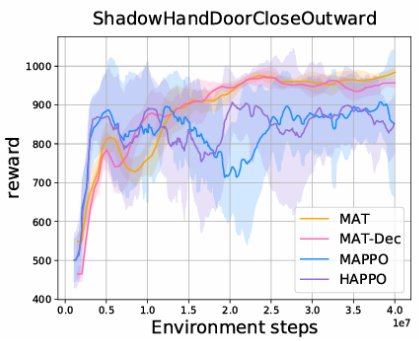
(d) DoorCloseOutward

Figure 3: Demonstrations of the Bi-DexHands and the HalfCheetah environments.









Google Research Football

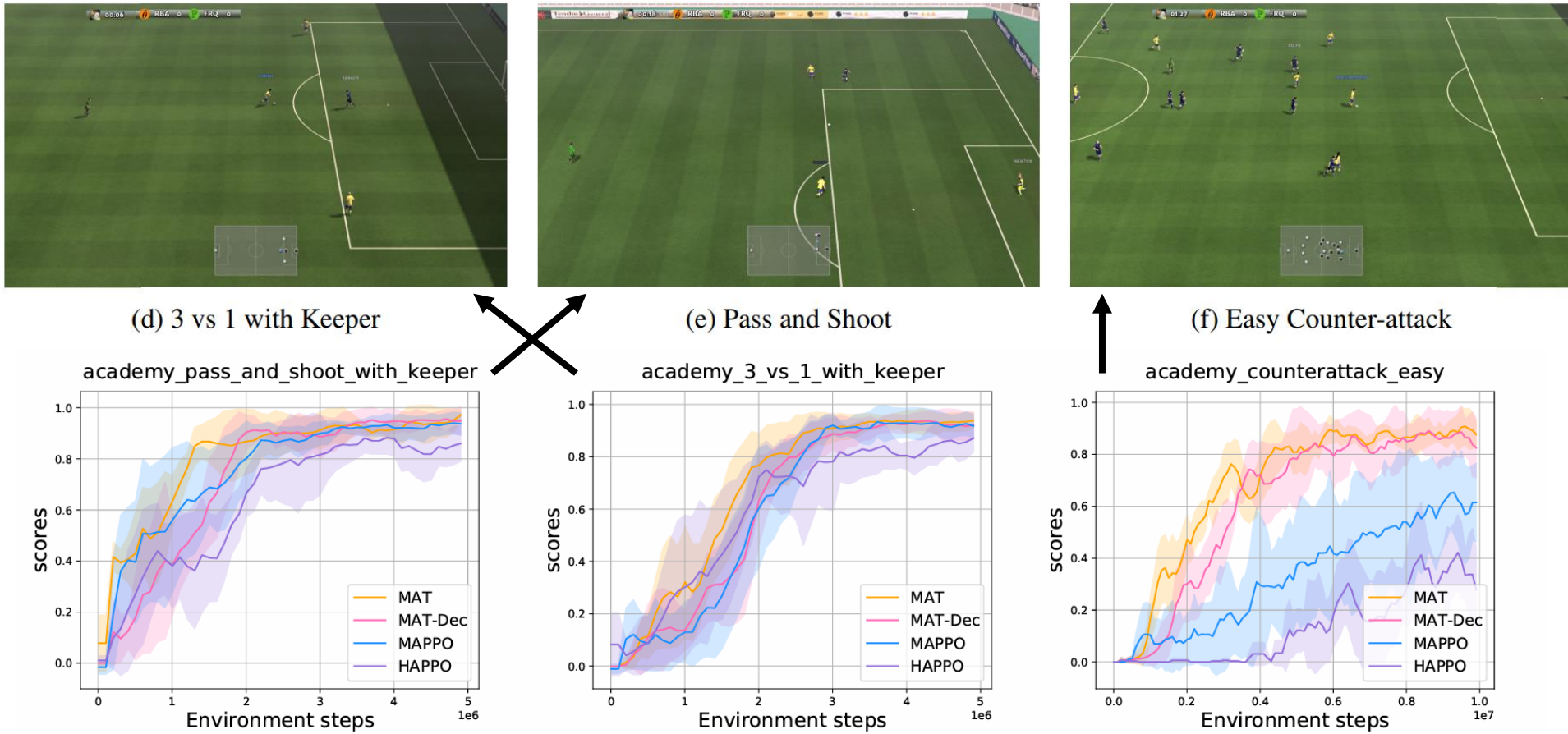


Figure 5: Performance comparison on the Google Research Football tasks with 2-4 agents from left to right respectively.

SMAC (StarCraft Multi-Agent Challenge)

27m vs 30m

MMM2

SMAC (StarCraft Multi-Agent Challenge)

Task	Difficulty	MAT	MAT-Dec	MAPPO	HAPPO	QMIX	UPDeT	Steps
3m	Easy	**100.0**(1.8)	**100.0**(1.1)	**100.0**(0.4)	**100.0**(1.2)	96.9 1.3	**100.0**(5.2)	5e5
8m	Easy	**100.0**(1.1)	97.5(2.5)	96.8(2.9)	97.5(1.1)	97.7 1.9	96.3(9.7)	1e6
1c3s5z	Easy	**100.0**(2.4)	**100.0**(0.4)	**100.0**(2.2)	97.5(1.8)	96.9(1.5)	/	2e6
MMM	Easy	**100.0**(2.2)	98.1(2.1)	95.6(4.5)	81.2(22.9)	91.2(3.2)	/	2e6
2c vs 64zg	Hard	**100.0**(1.3)	95.9(2.3)	**100.0**(2.7)	90.0(4.8)	90.3(4.0)	/	5e6
3s vs 5z	Hard	**100.0**(1.7)	**100.0**(1.3)	**100.0**(2.5)	91.9(5.3)	92.3(4.4)	/	5e6
3s5z	Hard	**100.0**(1.9)	**100.0**(3.3)	72.5(26.5)	90.0(3.5)	84.3(5.4)	/	3e6
5m vs 6m	Hard	**90.6**(4.4)	83.1(4.6)	88.2(6.2)	73.8(4.4)	75.8(3.7)	**90.6**(6.1)	1e7
8m vs 9m	Hard	**100.0**(3.1)	95.0(4.6)	93.8(3.5)	86.2(4.4)	92.6(4.0)	/	5e6
10m vs 11m	Hard	**100.0**(1.4)	**100.0**(2.0)	96.3(5.8)	77.5(9.7)	95.8(6.1)	/	5e6
25m	Hard	**100.0**(1.3)	86.9(5.6)	**100.0**(2.7)	0.6(0.8)	90.2(9.8)	2.8(3.1)	2e6
27m vs 30m	Hard+	**100.0**(0.7)	95.3(2.2)	93.1(3.2)	0.0(0.0)	39.2(8.8)	/	1e7
MMM2	Hard+	**93.8**(2.6)	91.2(5.3)	81.8(10.1)	0.3(0.4)	88.3(2.4)	/	1e7
6h vs 8z	Hard+	**98.8**(1.3)	93.8(4.7)	88.4(5.7)	0.0(0.0)	9.7(3.1)	/	1e7
3s5z vs 3s6z	Hard+	**96.5**(1.3)	85.3(7.5)	84.3(19.4)	82.8(21.2)	68.8(21.2)	/	2e7

Conclusion

- Today we have discussed the following points:
  - Several policy-based methods
  - Talked about parameter sharing
  - Mentioned extending trust region learning to cooperative MARL
  - Introduced multi-agent transformer