Value Aggregation with Uncertainty in Online Dec-MARL

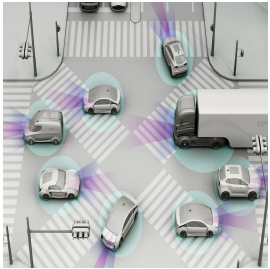
Ziyue Chu, Leonardo Stella

School of Computer Science
University of Birmingham

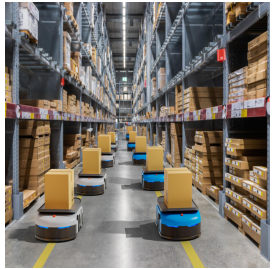
ICML26 Seoul, Republic of Korea July 2, 2026

Outline

Multi-agent System (MAS) Applications:



Intelligent Transport in Smart City[1]



Warehouse Robotics in Logistics[2]

Distributed Energy Grids[3]

Stock Market in Finance[4]

[1] Haydari, Ammar, et al. "Deep RL for intelligent transportation systems: A survey." IEEE Trans. on Intelligent Transportation Systems (2020)

[2] Sebastián, Eduardo, et al. "Physics-informed MARL for distributed multi-robot problems." IEEE Trans. on Robotics (2025)

[3] Z., Yang, et al. "Multistep MARL for optimal energy schedule strategy of charging stations in smart grid." IEEE Trans. on Cybernetics (2022)

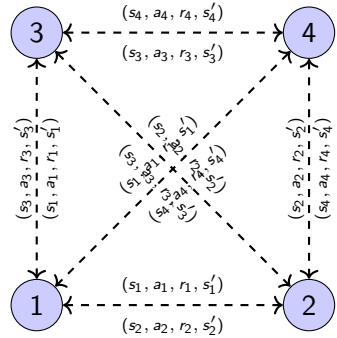
[4] Liu, Xiao-Yang, et al. "FinRL: A deep reinforcement learning library for automated stock trading in quantitative finance." arXiv (2020)

Background: Learning in Multi-Agent Systems

Learning Paradigms:

- **Centralized learning**[1]**:**
  - × Sample space ∼ Exp(agent#)
  - ✓ Global optimization (theory)
  - × Impractical → Bad perform.
- **Independent learning**[2]**:**
  - × Local optimization
  - ✓ Parallel computation
  - ? Sometimes work when decouplable
- ⋆ **Networked decentralized learning**[3]**:**
  - ✓ Balance efficiency & collaboration
  - ✓ Scalable with communication

Communication Network



Agents exchange experience through local communication

[1] Lu, Chengxuan, et al. "Centralized RL for MA-cooperative environments." Evolutionary Intelligence (2024).

[2] Tan, Ming. "MARL: Independent vs. cooperative agents." ICML. 1993.

[3] Zhang, Kaiqing, et al. "Fully decentralized MARL with networked agents." ICML. PMLR, 2018.

How to do Networked Decentralized Learning?

Framework in a nutshell:

- Local Independent Learning;
- Info exchange via network;
- Update Consensus feedback.

2 ways to exchange info[4]:

- Experience sharing;
- Parameter sharing, e.g. Federated Learning.

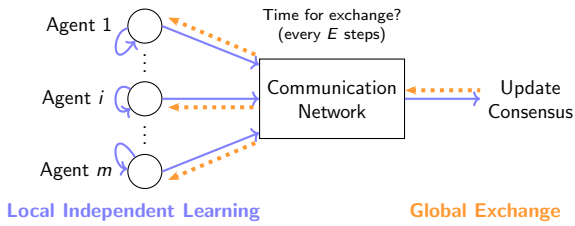


Figure: Networked Dec-Learning Framework[5]

[4] A., Stefano V., F. Christianos, and L. Schäfer. Multi-agent reinforcement learning: Foundations and modern approaches. MIT Press, 2024.

[5] Z., Chu, and L., Stella. "Value Aggregation with Uncertainty in Online Decentralized MARL." ICML. PMLR, 2026.

Current Gaps for Networked Dec-learning:

× *Data privacy* issue for direct experience sharing methods;

? Learning with *Online* samples from local interaction trajectories; Violating *Uniform Ergodic* assumption in current statistical convergence analysis;

- One naïve idea in MAS: More minds better than Smarter one.
- → Federated Learning shares params. and do *convex aggregation* for consensus;

! → Suffers from *learning rollback* issue in the online learning setting;

Problem: Learning Rollback Issue

1. LR in convex weights aggregation:

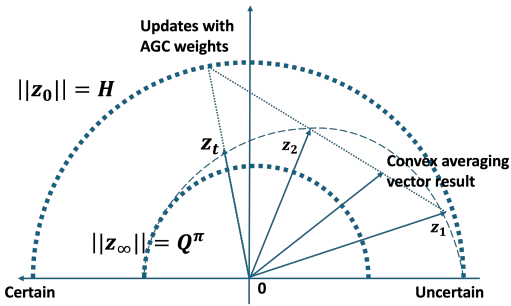


Figure: Convex weights update in vector space.

2. Formulation in online learning:

$$\hat{Q}_{t',m} \leftarrow \sum_{i\in[M]} w_{i,m}\hat{Q}_{t,i}, \forall m \in [M],$$

$$\text{s.t.} \quad \sum_{i\in[M]} w_{i,m} = 1; w_{i,m} \geq 0, \forall i \in [M],$$

$$||\hat{Q}_{t',m} - Q^*||_\infty < ||\hat{Q}_{t,i} - Q^*||_\infty.$$

Not solvable in online cases w/o uniform ergodic assump. for uniform convergence

★ Local t' not updating with global t

LR happens when convex aggregation over parameters with different uncertainty.

Why Does Federated Aggregation Fail? (Cont.)

Problem: Heterogeneous Online Updates

$$\sum_{s,a} t_i(s,a) = t, \quad \text{where } t_i(s,a) \neq \text{uniform}$$

This causes:

- Point-wise convergence instead of uniform convergence:

$$\left|(\hat{Q}_i^t - Q^*)(s,a)\right| < \epsilon_i \text{ with } \epsilon_i = T^{-1}(t_i(s,a))$$

- Errors $\max_i\{\epsilon_i\}$ depend on visit counts, not global time t
- Convex averaging does NOT guarantee uniform convergence

Key Insight:

$$\left|(\hat{Q}_m^{t'} - Q^*)(s,a)\right| < \max_i\{\epsilon_i\} \quad (\text{not } \epsilon)$$

Mismatch between $\max_i\{\epsilon_i\}$ and ϵ causes learning rollback.

Empirical Evidence: Learning Rollback

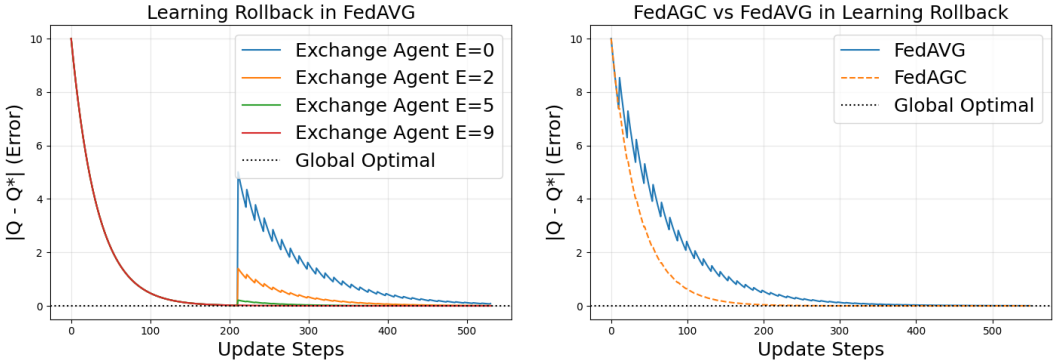


Figure: Learning Rollback in FedAVG (left) and Performance comparison with AGC (right).

How can we design an efficient online algorithm to bypass learning rollback for uncertain value aggregation in networked dec-MARL?

Our Contributions:

(1) Identify and formulate the *Rollback* issue in Online Dec-learning

(2) Propose a novel *Adaptive Global Consensus* (AGC) mechanism for parameter aggregation with different value uncertainty, favoring more informative updates.

(3) Develop a novel *DTDE learning dynamics* that *recursively* learns the AGC weights.

(4) Theoretical & Empirical guarantees of *faster convergence* and *bounded variance*.

Markov Game Formulation

$$\mathcal{M} = ([M], S, \{A_i\}_{i=1}^{M}, P, \{R_h\}_{h=1}^{H}, \gamma)$$

Assumption 2.1 (Homogeneity)

All agents share the same learning goal and symmetric policy effects:

$$V_i^{\pi_i} = V_{\sigma(i)}^{\langle \pi_{\sigma(1)}, \pi_{\sigma(2)}, \ldots, \pi_{\sigma(M)} \rangle}, \quad \forall i \in [M]$$

Under 2.1, the optimal joint policy consists of identical individual optimal policies:

$$\pi_i^* = \pi_j^*, \quad \forall i, j \in [M]$$

Policy Evaluation Objective: For agent i, maximize discounted cumulative reward:

$$\pi_i^* = \arg\max_{\pi_i} V_{1,i}^{\pi_i}, \qquad \boxed{V_{1,i}^{\pi_i} = \mathbb{E}\left[\sum_{h=1}^{H} \gamma^h r_{h,i}\right]}$$

Our Method (1): Local Policy Evaluation via TD

Local Q-Value Update (One-Step Policy Evaluation):

$$Q_{t+1}^{\pi_{i,m}}(s,a) \leftarrow r(s,a) + \gamma Q_t^{\pi_{i,m}}(s', \pi_{i,m}(s'))$$

Lemma 4.1 (Contraction Property): The iterations satisfy γ-contraction:

$$\left\| (Q_t^{\pi_{i,m}} - Q^{\pi_{i,m}})(s,a) \right\|_\infty \leq \gamma^t \frac{R_{\max}}{1-\gamma},$$

$$Q_t^{\pi_{i,m}}(s,a) = (1-\gamma^t) Q^{\pi_{i,m}}(s,a) + \gamma^t Q_0(s,a) + M_t$$

Policy Improvement:

$$\pi_{i+1,m}(s) \leftarrow \arg\max_{a'} Q_t^{\pi_{i,m}}(s,a')$$

Terminates when policy no longer changes: i.e., $Q^{\pi_m^*} = Q_m^*$

Our Method (2): Adaptive Global Consensus Weights

Adaptive Global Consensus Weights

Agent m with hetero. neighbor i updates:

$$W_m^t = \sum_{i \in \mathcal{N}_m} \frac{\text{Pow}(D_i^t) - \text{Pow}(D_m^t + \sum_j D_j^t)}{A_m(\text{Pow}(D_i^t) - \text{Pow}(D_m^t))}$$

$$W_{i,m}^t = \frac{\text{Pow}(D_m^t + \sum_j D_j^t) - \text{Pow}(D_m^t)}{A_m(\text{Pow}(D_i^t) - \text{Pow}(D_m^t))}$$

$\text{Pow}(D) = \gamma^D$: uncertainty $\sim$ update depth

Theorem 4.3

AGC weights resolve uniform convergence: $||\hat{Q}_{t',m} - Q^*||_\infty < ||\hat{Q}_{t,i} - Q^*||_\infty$

- Same bias level, i.e., $D_m^t = D_k^t$: $W_m^t = W_{k,m}^t = 1/|(\mathcal{N}_m| + 1 - A_m)$
- $\hat{Q}_m \leftarrow W_m^t \odot \hat{Q}_m + \sum_{i \in \mathcal{N}_m} W_{i,m}^t \odot \hat{Q}_i, \ \forall m$

Scenarios (neighbor agent i)	**Weight Behavior**
untrained ($D_i^t = 0$)	$W_m^t = 1, W_{i,m}^t = 0 \Rightarrow$ Keep own estimate
well-trained ($D_i^t \to \infty$)	$W_m^t = 0, W_{i,m}^t = 1 \Rightarrow$ Adopt neighbor's value
less confident ($D_m^t > D_i^t$)	$\mathbf{W_m^t > 1, W_{i,m}^t < 0} \Rightarrow$ **Extrapolate towards confident**

Our Method (3): AGC Weights Efficient Learning

Algorithm 1 Decentralized AGC Weights Matrix Learning

Require: Any agent m, neighbor set $\mathcal{N}_m$, masks $= [\cdot]$
Ensure: $W_{i,m}^t, W_m^t$, Updated $\hat{Q}_m$
$D_m^{sum} = \mathbf{0}$, $A_m = \mathbf{0}$, Parameters memory $\mathcal{P} = [[\cdot]]$
for each agent $i \in \mathcal{N}_m$ **do**
  $D_i^{dif} = D_i^t - D_m^t$, $\text{mask}_i = \mathbb{I}[D_i^{dif} = 0]$
  masks $\leftarrow$ masks $+ \text{mask}_i$, $W_{i,m}^t[\text{mask}_i] = 1$
  $W_{i,m}^t[1 - \text{mask}_i] \leftarrow 1/(\text{Pow}(D_i^{dif}) - 1)$
  $\mathcal{P} \leftarrow \mathcal{P} + [W_{i,m}^t, \hat{Q}_i]$, $A_m \leftarrow A_m + (1 - \text{mask}_i)$
  $D_m^{sum} \leftarrow D_m^{sum} + D_i^t \odot (1 - \text{mask}_i)$
end for
Denote $\mathcal{P}_0 = [W_{i,m}^t], \mathcal{P}_1 = [\hat{Q}_i]$ the 2 columns of $\mathcal{P}$
if masks$(i, s, a) \neq 1, \forall i, s, a \in \mathcal{N}_m, \mathcal{S}, \mathcal{A}$ **then**
  $[W_{i,m}^t] \leftarrow \mathcal{P}_0 \odot [(\text{Pow}(D_m^{sum}) - 1) \odot A_m]$
else $[W_{i,m}^t] \leftarrow 1/(|\mathcal{N}_m| + 1 - A_m)$
end if
$W_m^t \leftarrow 1 - \sum_{i \in \mathcal{N}_m} W_{i,m}^t$
$\hat{Q}_m \leftarrow W_m^t \odot \hat{Q}_m + \langle [W_{i,m}^t], \mathcal{P}_1 \rangle$

Lemma 5.3: Decentralized Equivalence

Algorithm 1 computes the AGC weights matrix in a fully decentralized manner.

Key Takeaways:

- Focus on $W_{i,m}^t$; $W_m^t = 1 - \sum_i W_{i,m}^t$
- Denoms: neighbor uncertainty difference
- Accumulated num.: Agent-invariant
- Fully decentralized: Enables DTDE implementation w/o central coordination.

Properties of the designed AGC weights matrix:

Lemma 5.1 (Normalization)

The sum of AGC weights matrix in equation (2) are normalized:

$$\sum_{i \in \mathcal{N}_m} W_{i,m}^t + W_m^t = \mathbf{1}, \quad \forall m \in [M].$$

Lemma 5.2 (Ranges of AGC Weights)

Given any fixed $m \in [M]$, $\forall i \in \mathcal{N}_m, (s, a) \in \mathcal{S} \times A$, the weights in AGC matrix (2) are bounded: $\frac{\gamma^{ME}-1}{1-\gamma} \leq W_{i,m}^t(s, a) \leq \frac{1-\gamma^{ME}}{1-\gamma}$, and $\frac{\gamma^{ME}-\gamma}{1-\gamma} \leq W_m^t(s, a) \leq \frac{2-\gamma^{ME}-\gamma}{1-\gamma}$.

Theorem 5.6 (Learning convergence & acceleration)

After one step of global consensus update, the learned parameters have the following property: $\forall(s,a) \in \mathcal{S} \times \mathcal{A},\ m \in [M]$,

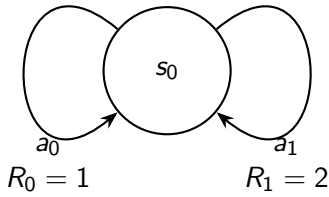
$$\mathbb{E}[\hat{Q}_m(s,a)] = \mathbb{E}[\mu_m] + \mathbb{E}[\alpha_m]\,\mathrm{Pow}\left(D_m^t + \sum_{i \in \mathcal{N}_m} D_i^t\right)(s,a),$$

with bounded variance $\mathrm{Var}[M_t]/M \leq \mathrm{Var}[\hat{Q}_m(s,a)] \leq ((Z+1)^2 + \frac{Z^2}{M-1})\mathrm{Var}[M_t]$.

Communication Bandwidth and Storage Analysis

- Exchange pair-wise params: $\hat{Q}_i$ and D_i^t; Bandwidth $\mathcal{O}(SA)$
- Memory stores $2M$ parallel $|S||A|$ matrices; Storage complexity $\mathcal{O}(MSA)$.
- Scalable parallel computation supported by Tensors.

1. Policy evaluation in symbolic MDP:

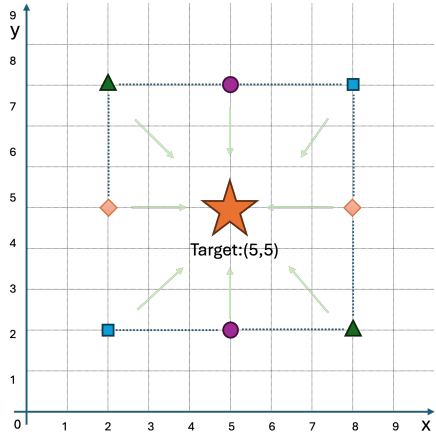


$$\pi^*(s_0) = a_1,$$

$$V^*(s_0) = \frac{R_1}{1-\gamma},$$

$$\hat{V}_t^{\pi_m} = V^*(s) + (V_0(s) - V^*(s))\gamma^m.$$

2. MPE for Cooperative Navigation:



Scalability and Convergence Acceleration[1]

Figure: Training performances for scalability

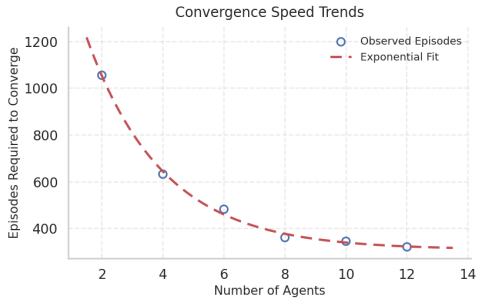
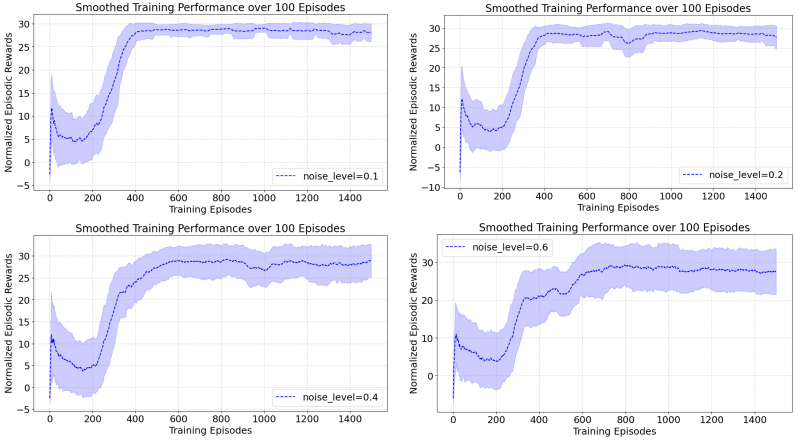


Figure: Convergence speeds with agent #

[1] Experiments on Scalability for MPE cooperative navigation task: $M \in \{2, 4, 6, 8, 10, 12\}$, 10×10 grid-world, $s_{tar} = (5, 5)$, $\epsilon = 0$, $K = 1500$, $H = 10$, $E = 5$, $\gamma = 0.9$

Play with $\mathrm{Var}[M_t]$ (Robustness to Stochastic Noise)



noise lv.	Mean	std
0.1	28.19	0.75
0.2	28.88	1.76
0.4	28.60	3.74
0.6	*24.41*	*6.81*

Figure: Training results with varying noise levels ($M = 8$).

Takeaway Question:

- Are more Heads better than Smarter ones?
- If total number of interactions ($E * M$) is fixed, shall we allocate them in parallel to evolve with the dummy or accumulate them to get a smarter agent first?

Future Work:

- Adaptive weights for non-linear operators, i.e., NN, general Bellman Operators.
- Complex network structures.
- Extension to hierarchical / heterogeneous MARL.

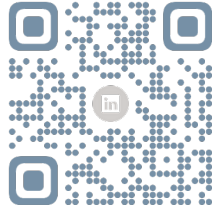
Thank you! Questions?

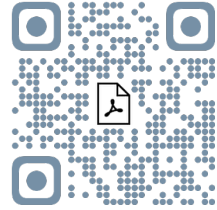
Presenter

Ziyue Chu*
MARL Researcher from DLC Research Group of UoBirmingham
zero-cooper.github.io

Research Directions:

- Networked-MARL Thoery
- Learning Cooperative Games
- UAV Swarms Navigation





zxc332@student.bham.ac.uk

Zero-Cooper_work