Reinforcement Learning China Summer School

RLChina 2021

Learning to Collaborate in Complex Environments

Chongjie Zhang

Institute of Interdisciplinary Information Sciences

Tsinghua University

August 19, 2021

AI Breakthrough in Pattern Recognition

What's the Next?

Intelligent decision-making in multi-agent environments

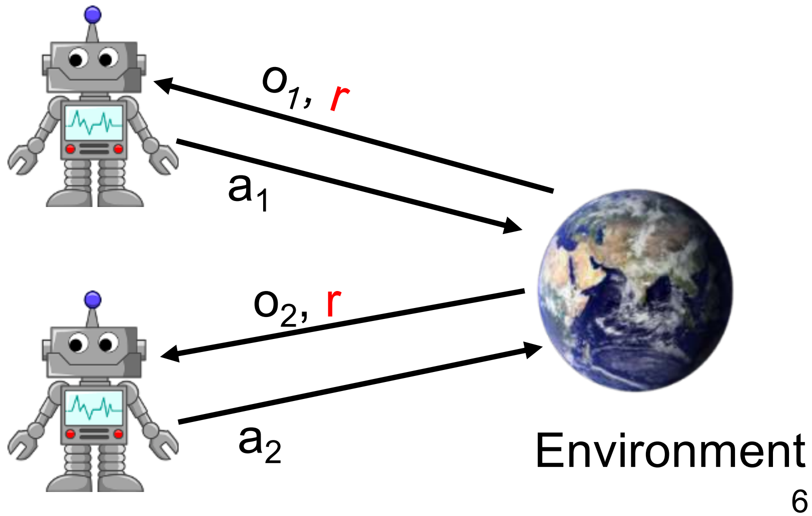
Types of Multi-Agent Systems

- Cooperative
  - Working together and coordinating their actions
  - Maximizing a shared team reward
- Competitive
  - Self-interested: maximizing an individual reward
  - Opposite rewards
  - Zero-sum games
- Mixed
  - Self-interested with different individual rewards (not opposite)
  - General-sum games

An Example in Starcraft II:
3 Stalkers vs 5 Zealots

Collaborative Multi-Agent Decision-Making

- Finding policies for agents to optimize team performance
- Model: decentralized partially observable Markov decision process (Dec-POMPD)
  - Multi-agent sequential decision-making under uncertainty
  - Extension of MDPs and POMDPs
- At each step, each agent *i* takes an action and receives:
  - A local observation o_i
  - A joint immediate reward *r*



Dec-POMDP

- Model
  - Agent: $i \in I = \{1, 2, \dots, N\}$
  - State: $s \in S$
  - Action: $a_i \in A$, $\boldsymbol{a} \in A^N$
  - Transition function: $P(s' \mid s, \boldsymbol{a})$
  - Reward: $R(s, \boldsymbol{a})$
  - Observation: $o_i \in \Omega$
  - Observation function: $o_i \in \Omega \sim O(s, i)$

Dec-POMDP

- Objective: to find policies for agents to jointly maximize the expected cumulative reward
- A local policy π_i for each agent i: mapping its observation-action history τ_i to its action
  - Action-observation history: $\tau_i \in T = (\Omega \times A)^*$
  - State is unknown, so beneficial to remember the history

Dec-POMDP

- Objective: to find policies for agents to jointly maximize the expected cumulative reward
- Joint policy $\boldsymbol{\pi} =< \pi_1, \dots, \pi_n >$
- Value function: $Q_{tot}^{\boldsymbol{\pi}}(\boldsymbol{s}, \boldsymbol{a}) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s, \boldsymbol{a}_0 = \boldsymbol{a}, \boldsymbol{\pi}\right]$
- Optimal Policy $\boldsymbol{\pi}^*= \mathrm{argmax}_{\pi} \max_{a} Q_{tot}^{\boldsymbol{\pi}}(\boldsymbol{s_0}, \boldsymbol{a})$

Multi-Agent Reinforcement Learning (MARL)

- MARL is promising for solving Dec-POMDPs
  - Dec-POMDP is NEXP (2002)
  - The environment model is often unknown
  - Learning policies by interacting with the environment
- Reinforcement learning in a nutshell
  - Exploration: add some randomness into action selection
  - If an action performs better than expected, do it more in the future; otherwise, do it less.
- MARL: learning policies for multiple agents
  - Where agents are interacting

MARL Challenges

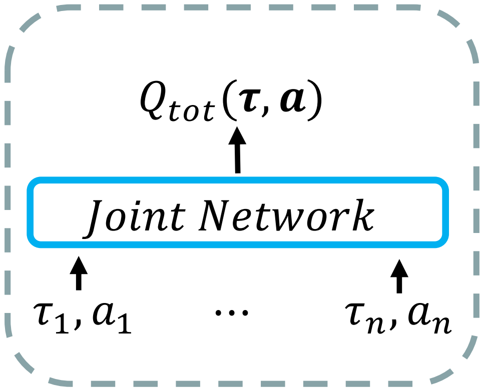
- Scalability
  - A large number of agents
- Credit Assignment
  - each agent's contribution to the team
- Uncertainty
  - Partial and noisy observations
- Heterogeneity
  - Requiring diverse behaviors of agents
- Exploration
  - Coordinated exploration among agents

Outlines

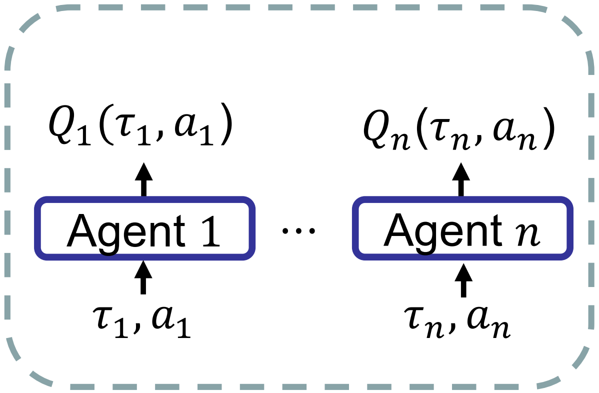
- Linearly factorized multi-agent learning [Arxiv 2020, ICLR 2021a]
  - Simple but scalable and effective
  - Properties: local convergence and implicit credit assignment
- QPLEX: non-linearly factorized learning [ICLR 2021b]
  - Strong representation with global convergence
  - State-of-the-art benchmark performance
- Extensions
  - Learning to communicate [ICLR 2020a]
  - Role-emergence learning [ICML 2020, ICLR 2021c]
  - Effective exploration [ICLR 2020b, Arxiv 2021]

Multi-Agent Reinforcement Learning (MARL)

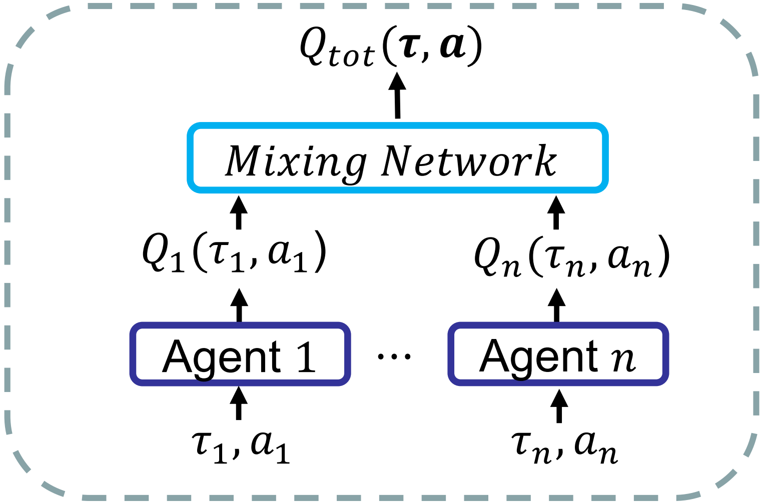
- Paradigms: learning cooperative policies or value functions



Centralized Value Functions



Decentralized Value Functions



Factorized Value Functions

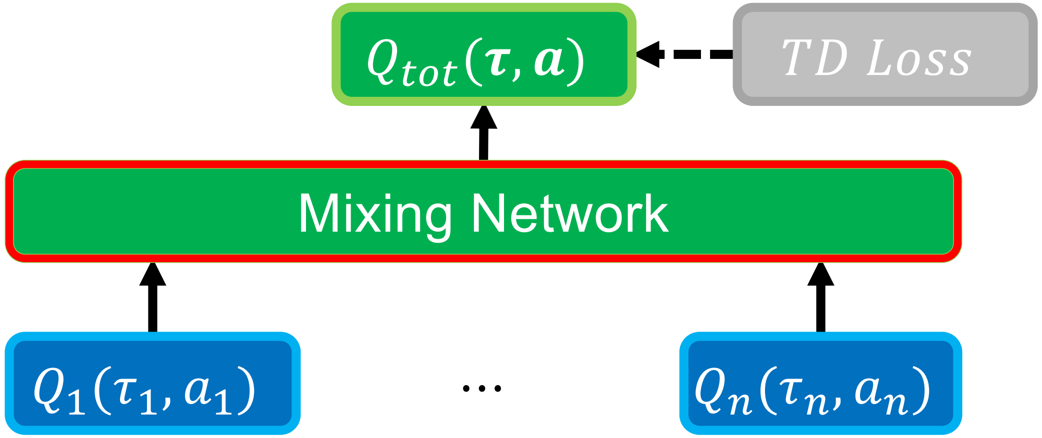
Scalability 

Non-stationarity
Credit assignment 

Centralized training
Decentralized execution 

Factorized Multi-Agent Reinforcement Learning

- Paradigm: centralized training with decentralized execution



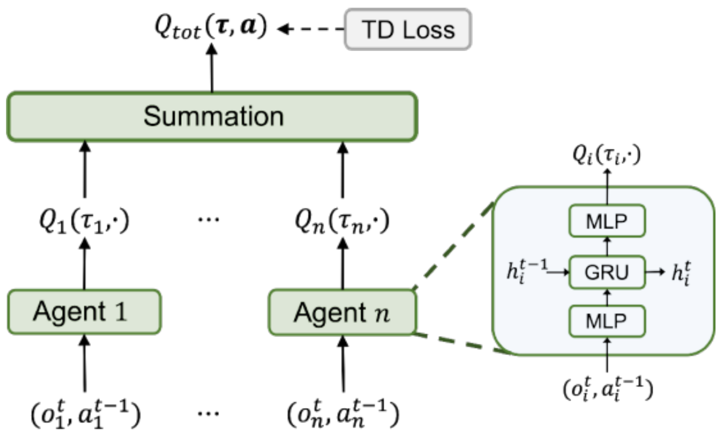
$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}\left[\left(r + \gamma V(\boldsymbol{\tau}'; \boldsymbol{\theta}^-) - Q(\boldsymbol{\tau}, \boldsymbol{a}; \boldsymbol{\theta})\right)^2\right]$$

$$V(\boldsymbol{\tau}'; \boldsymbol{\theta}^-) = \max_{\boldsymbol{a}'} Q(\boldsymbol{\tau}', \boldsymbol{a}'; \boldsymbol{\theta}^-)$$

- Individual-Global Maximization (IGM) Constraint
  - $\operatorname{argmax}_{\boldsymbol{a}} Q_{tot}(\boldsymbol{\tau}, \boldsymbol{a}) = \left(\operatorname{argmax}_{a_1} Q_1(\tau_1, a_1), \ldots, \operatorname{argmax}_{a_n} Q_n(\tau_n, a_n)\right)$
  - Consistent greedy action selection between joint and individuals

Linear Factorized Multi-Agent Learning

- Linear Mixing: $Q_{tot}(\boldsymbol{\tau}, \boldsymbol{a}) = \sum_i Q_i(\tau_i, a_i)$ [Sunehag et. al., 2017]
- Satisfying IGM Constraint
- Parameter sharing
- No specific reward for each agent
- Implicit credit assignment through gradient backpropagation

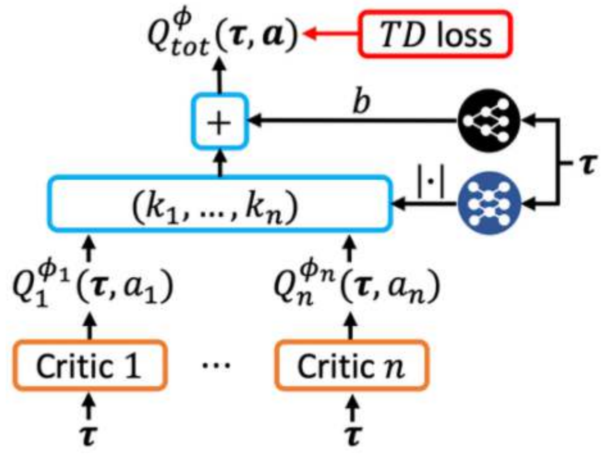


Implicit Credit Assignment Mechanism

- **counterfactual** credit assignment mechanism
  - $Q_i^{(t+1)}(s, a_i) = \underbrace{\mathbb{E}_{a'_{-i}}\left[y^{(t)}(s, a_i \oplus a'_{-i})\right]}_{\text{Evaluation of } a_i} - \frac{n-1}{n} \underbrace{\mathbb{E}_{\mathbf{a}'}\left[y^{(t)}(s, \mathbf{a}')\right]}_{\text{Baseline}}$
  - Target Q-value: $y^{(t)}(s, \mathbf{a}) = r + \gamma \max_{\mathbf{a}'} Q_{tot}^{(t)}(s', \mathbf{a}')$

[Arxiv 2020]

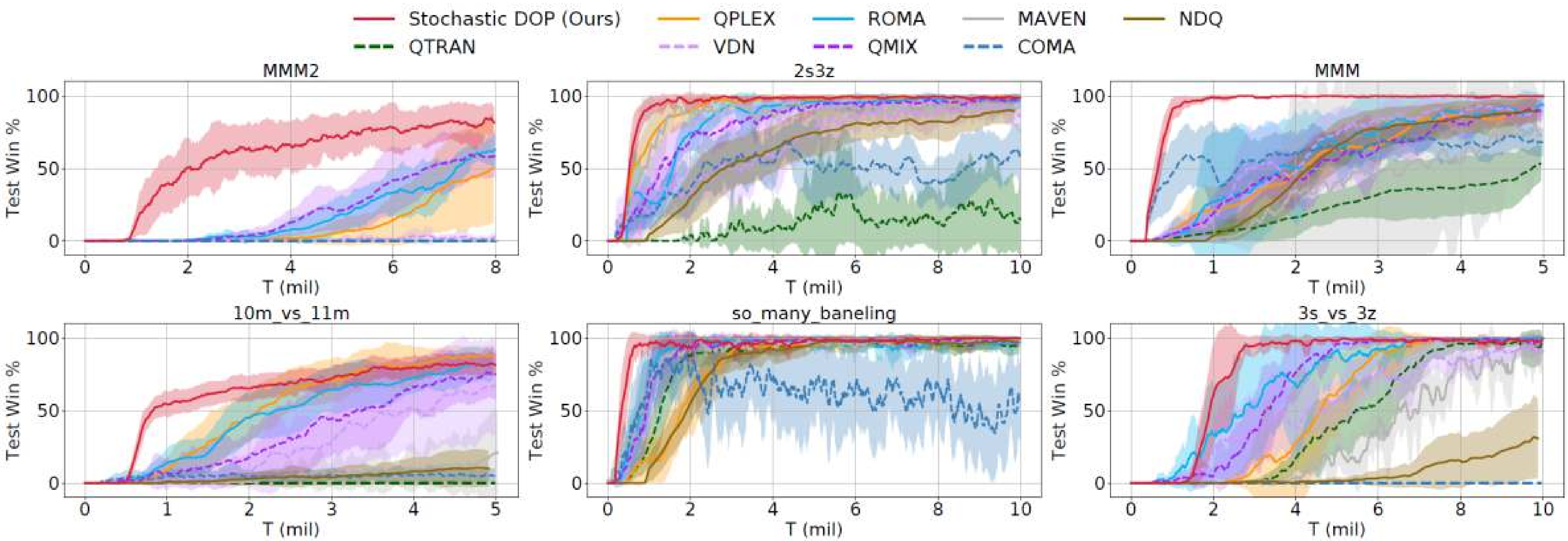
DOP: Off-Policy Decomposed Policy Gradient



- Introducing linearly decomposed critic
  - $Q_{tot}^{\boldsymbol{\pi}}(\boldsymbol{\tau}, \cdot) = \sum_i k_i(\boldsymbol{\tau}) Q_i(\boldsymbol{\tau}, \cdot) + b(\boldsymbol{\tau})$
- Policy gradient theorem
  - $\nabla J(\theta) = \mathbb{E}_{\boldsymbol{\pi}}[\sum_i \nabla_\theta \log \pi_i(a_i|\tau_i) k_i(\boldsymbol{\tau}) Q_i(\boldsymbol{\tau}, a_i)]$
- Benefits:
  - Simple but effective
  - Convergence guarantee with monotonic improvement
  - Work for both discrete and continuous action space

[ICLR 2021a]

Starcraft Micromanagement Benchmark



Learned Kiting Strategy in Starcraft II

Limitations on Linear Value Factorization

- $Q_{tot}(\boldsymbol{\tau}, \boldsymbol{a}) = \sum_i Q_i(\tau_i, a_i)$
- Limited Representation

Agent 1

Agent 2

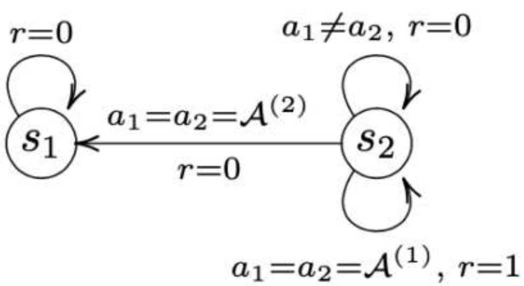
a_2 \ a_1	$\mathcal{A}^{(1)}$	$\mathcal{A}^{(2)}$	$\mathcal{A}^{(3)}$
$\mathcal{A}^{(1)}$	**8**	-12	-12
$\mathcal{A}^{(2)}$	-12	0	0
$\mathcal{A}^{(3)}$	-12	0	0

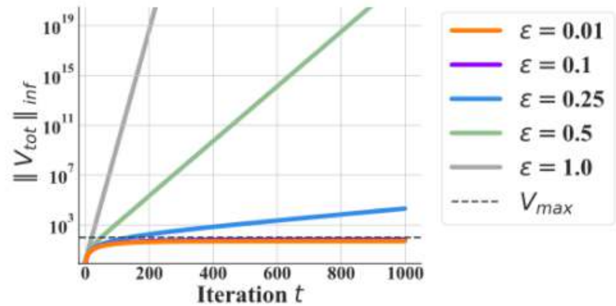
(a) Payoff of matrix game.

a_2 \ a_1	$\mathcal{A}^{(1)}$	$\mathcal{A}^{(2)}$	$\mathcal{A}^{(3)}$
$\mathcal{A}^{(1)}$	-6.5	-5.0	-5.0
$\mathcal{A}^{(2)}$	-5.0	-3.5	-3.5
$\mathcal{A}^{(3)}$	-5.0	**-3.5**	-3.5

(b) Q_{tot} of VDN.

- No global convergence guarantee with value-based learning





[Arxiv 2020]

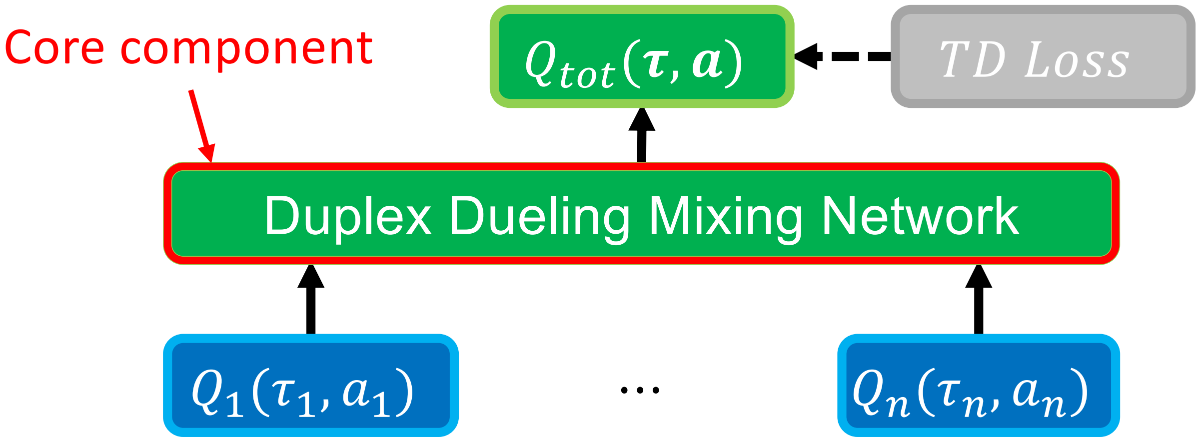
Outlines

- Linearly factorized multi-agent learning [Arxiv 2020, ICLR 2021a]
  - Simple but effective
  - Properties: local convergence and implicit credit assignment
- **QPLEX: non-linearly factorized learning** [ICLR 2021b]
  - Strong representation with global convergence
  - State-of-the-art benchmark performance
- Extensions
  - Learning to communicate [ICLR 2020a]
  - Role-emergence learning [ICML 2020, ICLR 2021c]
  - Effective exploration [ICLR 2020b, Arxiv 2021]

QPLEX: Duplex Dueling Mixing Network

[ICLR 2021b]

- Idea: fitting the maximum value and compensating the rest



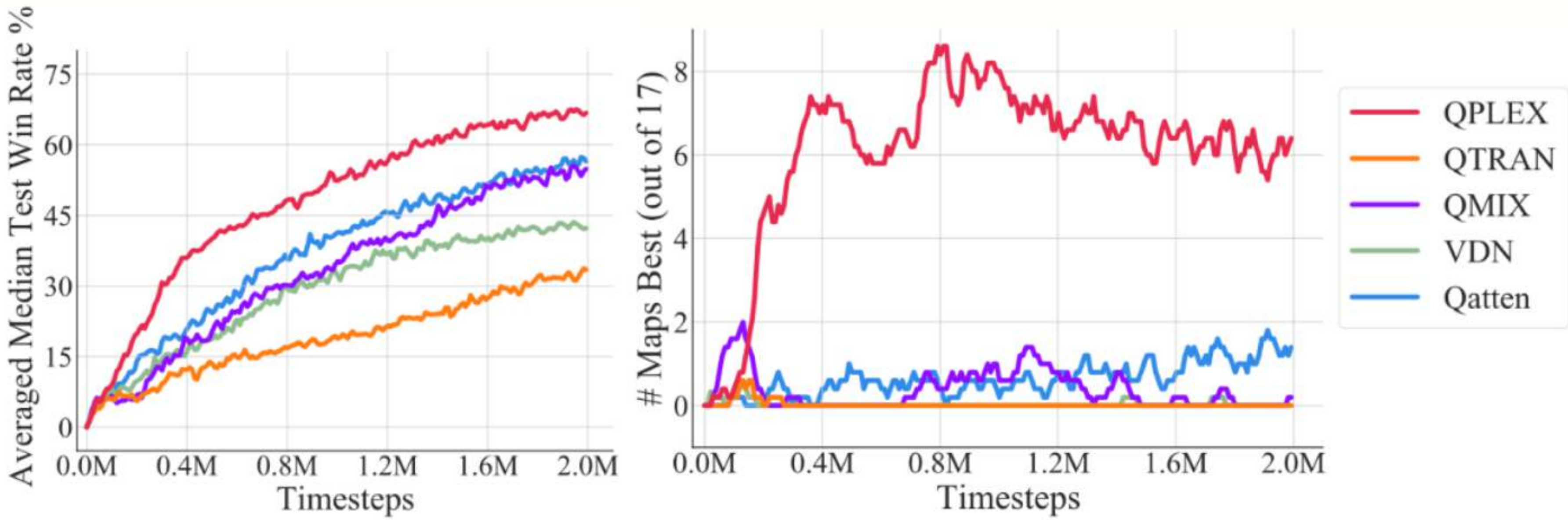
Linear Value Factorization

- QPLEX : $Q_{tot}(\boldsymbol{\tau}, \boldsymbol{a}) = \boxed{\sum_i Q_i(\boldsymbol{\tau}, a_i)} + \boxed{\sum_{i=1}^{n} (\lambda_i(\boldsymbol{\tau}, \boldsymbol{a}) - 1)\, A_i(\boldsymbol{\tau}, a_i)}$
  - Strong representation capacity
  - Easily realized and learned by neural networks

Theoretical Properties

Theorem 1 (Full Representation Capacity): *The joint action-value function class that QPLEX can realize is equivalent to what is induced by the IGM principle.*

StarCraft II Benchmark

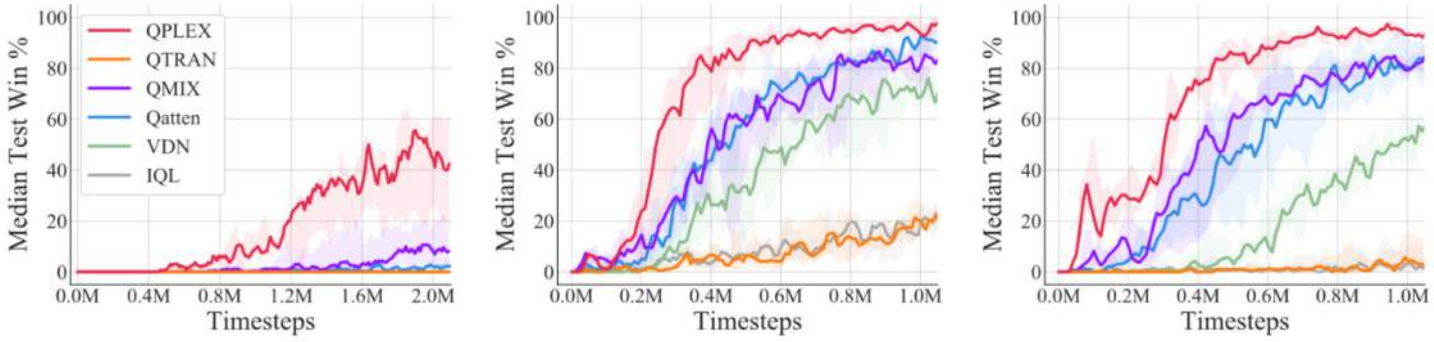


(a) Averaged test win rate

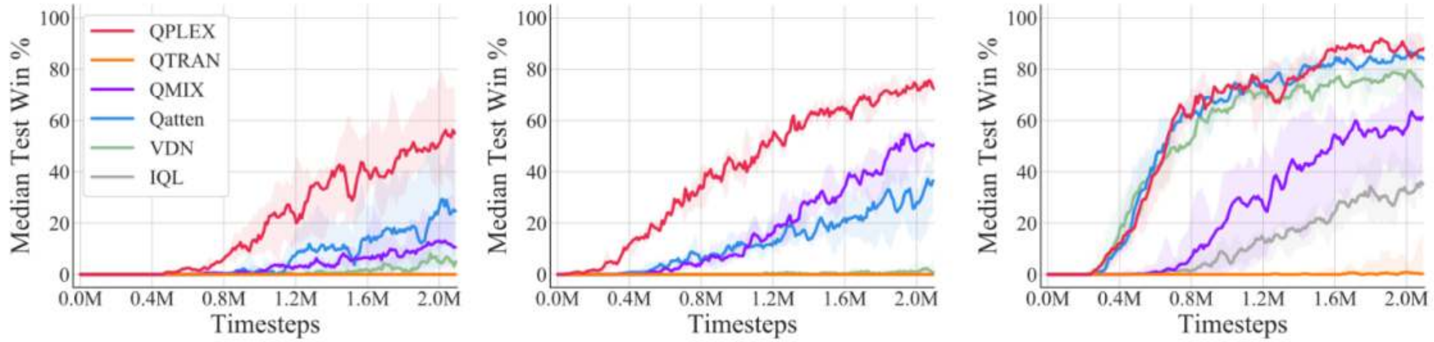
(b) # Maps best out of 17 scenarios

Figure: (a) The median test win %, averaged across all 17 scenarios. (b) The number of scenarios in which the algorithms' median test win % is the highest by at least 1/32 (smoothed).

StarCraft II Benchmark: Online Learning



(a) 5s10z (b) 1c3s5z (c) 3s5z

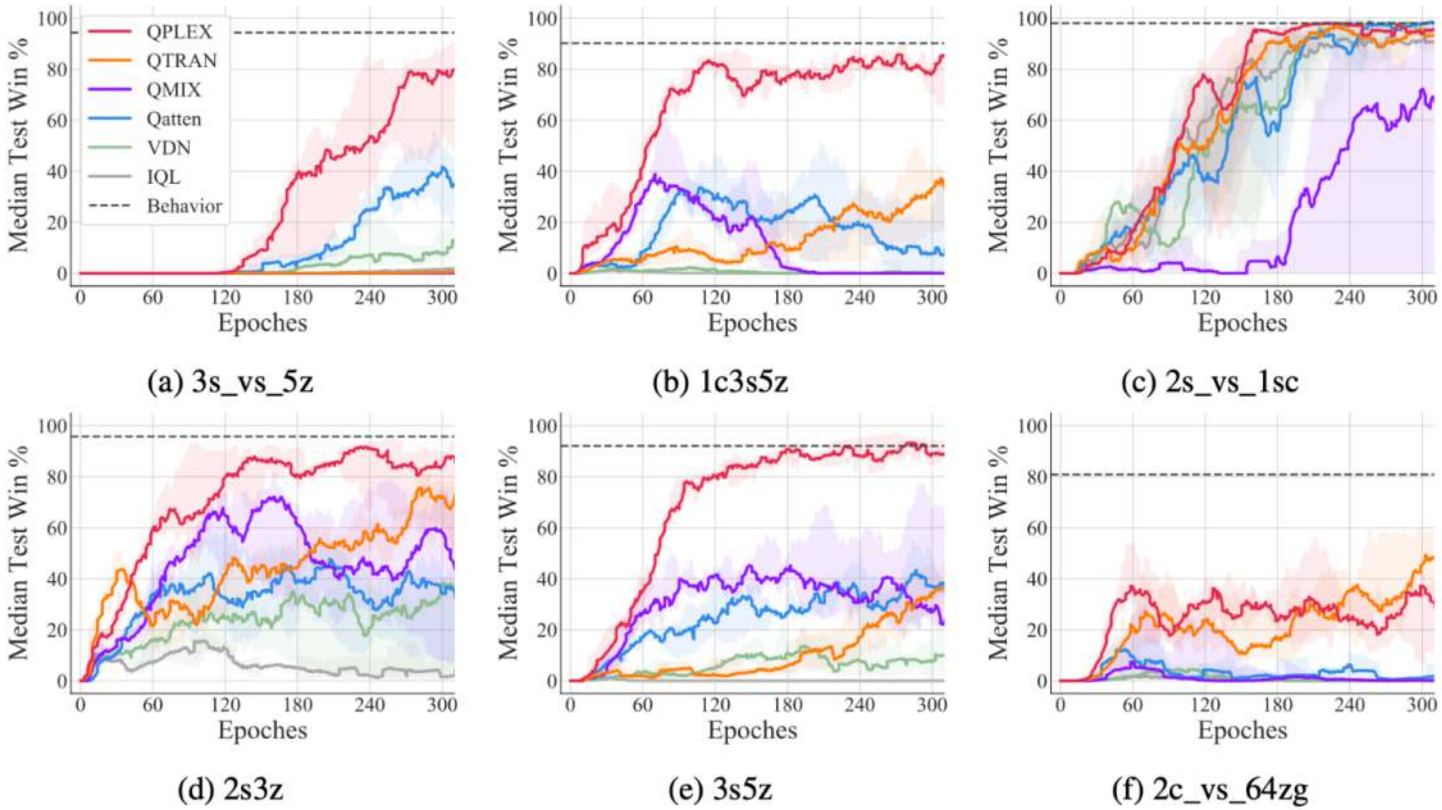


(a) 1c3s8z_vs_1c3s9z (b) 7sz (c) 3s_vs_5z

Learned Interesting Strategy in Starcraft II

StarCraft II Benchmark: **Offline Learning**

Data collected by a behavior policy learned by QMIX

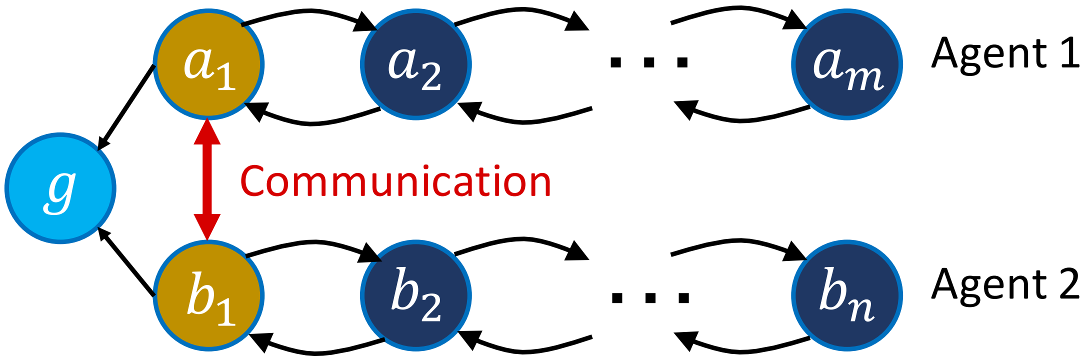


Outlines

- Linearly factorized multi-agent learning [Arxiv 2020, ICLR 2021a]
  - Simple but effective
  - Properties: local convergence and implicit credit assignment
- QPLEX: non-linearly factorized learning [ICLR 2021b]
  - Strong representation with global convergence
  - State-of-the-art benchmark performance
- Extensions
  - **Learning to communicate** [ICLR 2020a]
  - Role-emergence learning [ICML 2020, ICLR 2021c]
  - Effective exploration [ICLR 2020b, Arxiv 2021]

Limitations of Full Value Factorization

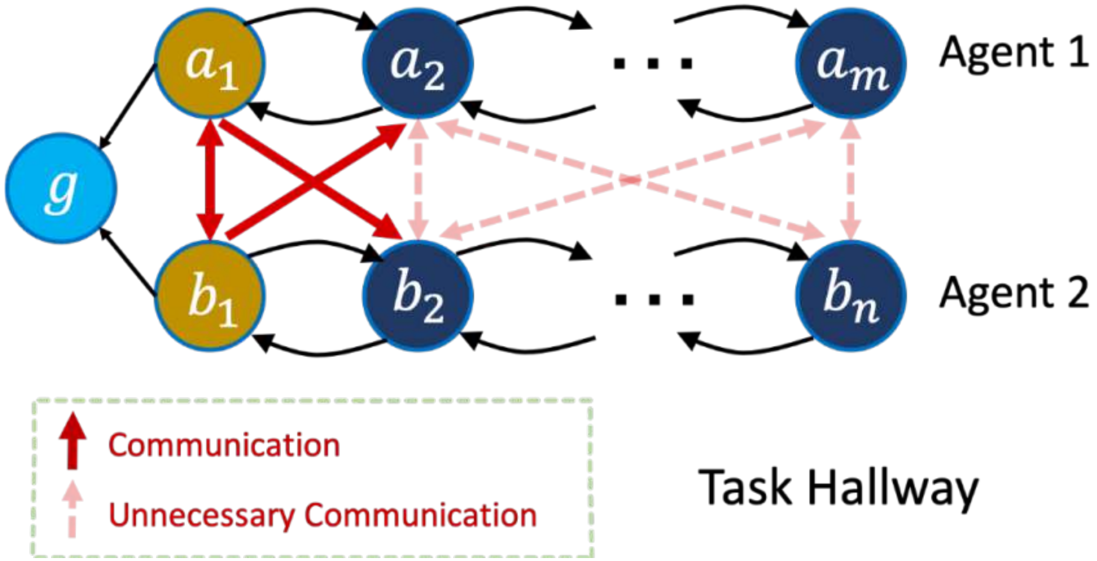
- Can cause miscoordinations during execution
  - Need Communication!



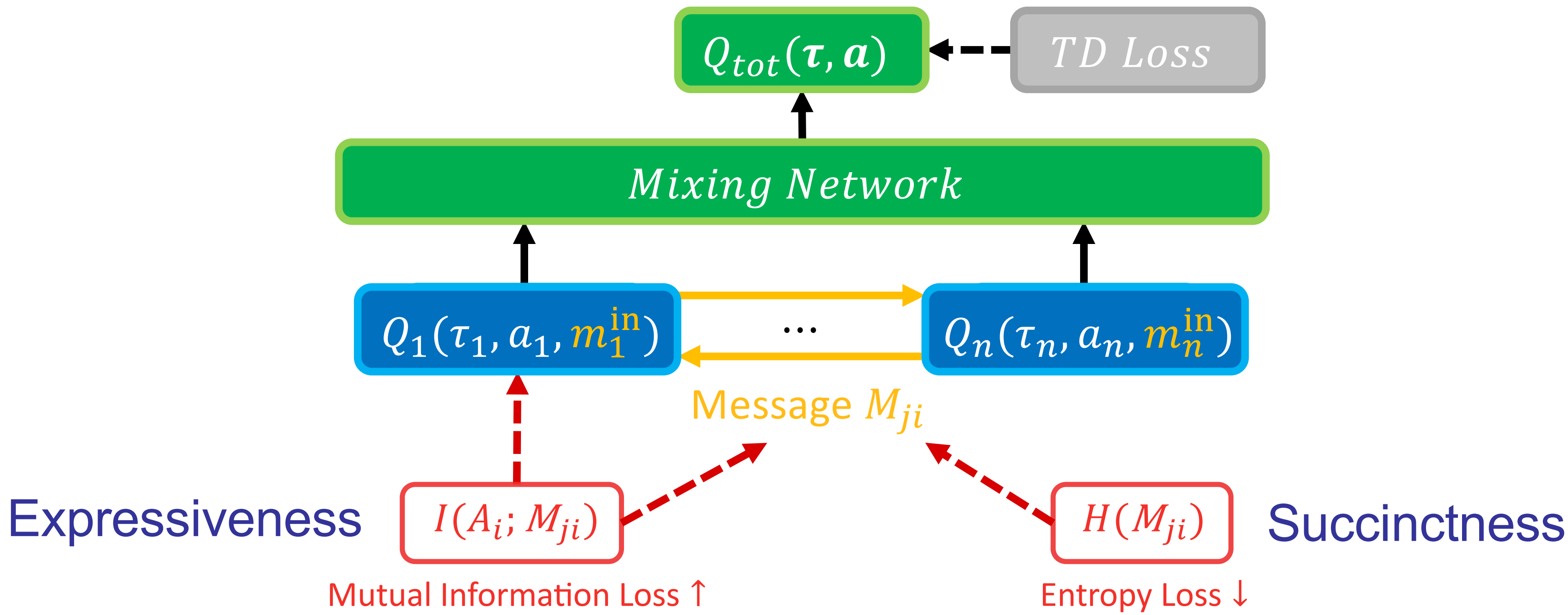
Task Hallway

Nearly Decomposable Q-Value Learning (NDQ)

- Allowing communication, but minimized
- Learn when, what, and with whom to communicate



NDQ Framework: Communication Optimization



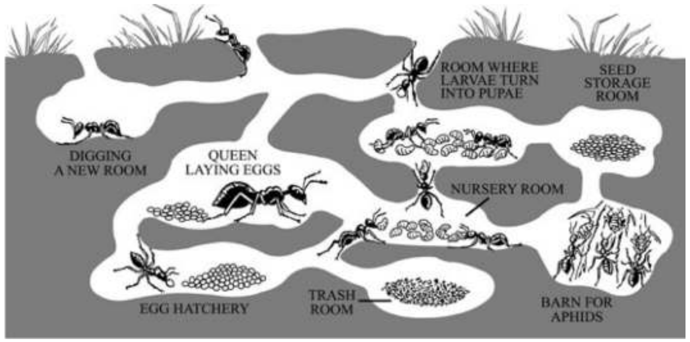
Wang, T., Wang, J., Zheng, C. and Zhang, C., 2019. Learning nearly decomposable value functions via communication minimization. *ICLR* 2020

Outlines

- Linearly factorized multi-agent learning [Arxiv 2020, ICLR 2021a]
  - Simple but effective
  - Properties: local convergence and implicit credit assignment
- QPLEX: non-linearly factorized learning [ICLR 2021b]
  - Strong representation with global convergence
  - State-of-the-art benchmark performance
- Extensions
  - Learning to communicate [ICLR 2020a]
  - **Role-emergence learning** [ICML 2020, ICLR 2021c]
  - Effective exploration [ICLR 2020b, Arxiv 2021]

Why role-based learning?

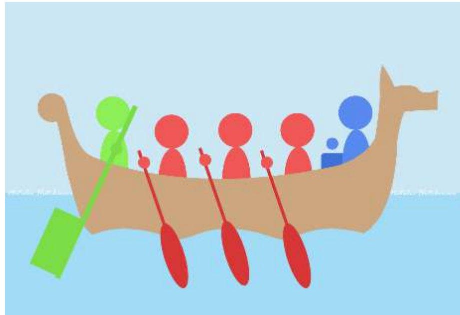
- Complex cooperative tasks require **diverse** behaviors among agents



- Learning a single shared policy network for agents[1-4]
  - Lack of diversity and requiring a high-capacity neural network
  - May result in slow, ineffective learning
- Learning independent policy networks is not efficient
  - Some agents perform similar sub-tasks, especially in large systems

[1] Rashid, et. al. QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. (ICML 2018)
[2] Vinyals, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. (Nature 2019)
[3] Baker, et al. Emergent tool use from multi-agent autocurricula. (ICLR 2020)
[4] Lowe, et al. Multi-agent actor-critic for mixed cooperative-competitive environments. (NeurIPS 2017)

ROMA: Multi-Agent Reinforcement Learning with Emerging Roles

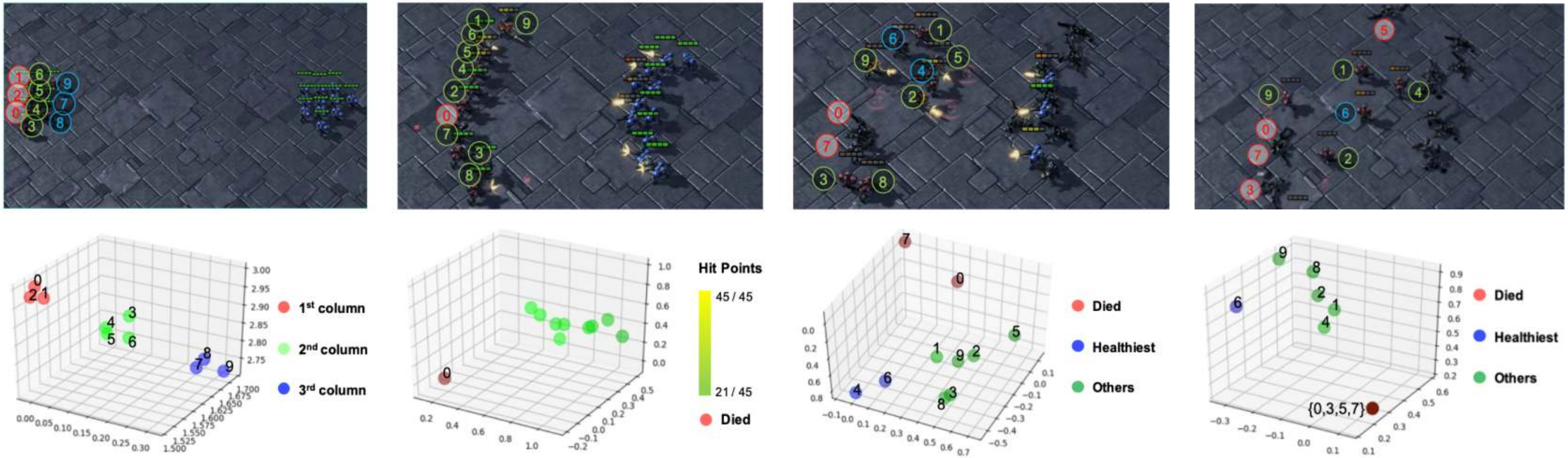


- Agents with similar roles have similar policies and share their learning
  - Similar roles ⟺ similar subtasks ⟺ similar behaviors
- Inferring an agent's roles based on the local observations and execution trajectories
- Agents learn policies conditioned on their roles
- An agent can change its roles in different situations

[ICML 2020, ICLR 2021c]

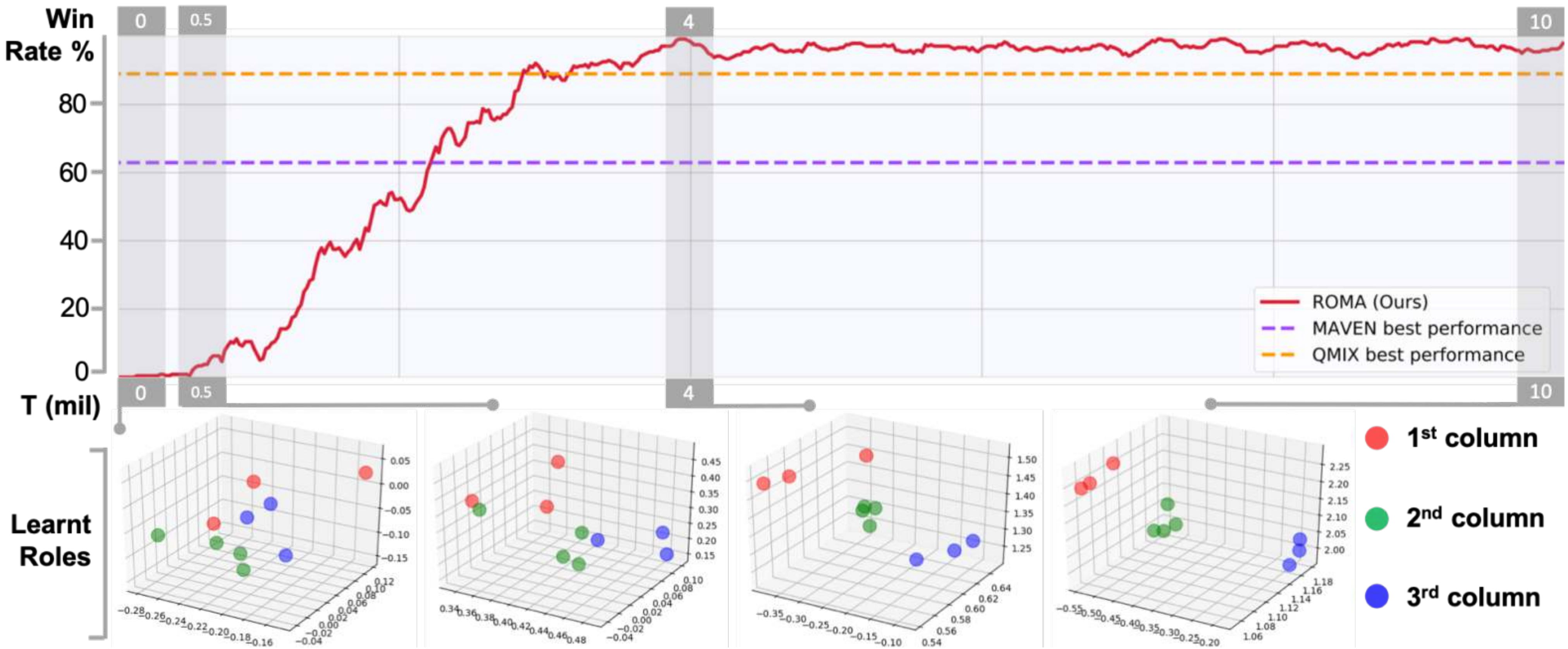
Starcraft II: 27 Marines vs 30 Marines

Dynamic Roles

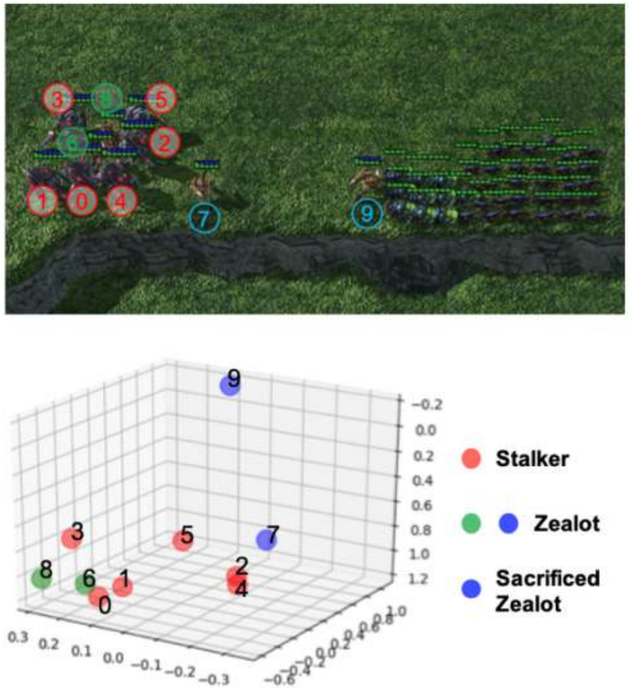


$t = 1$ $t = 8$ $t = 19$ $t = 27$

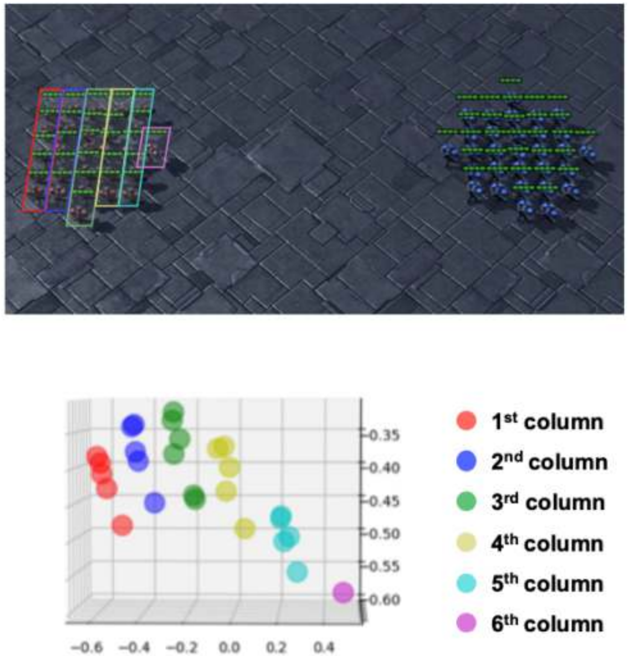
Role Emergence



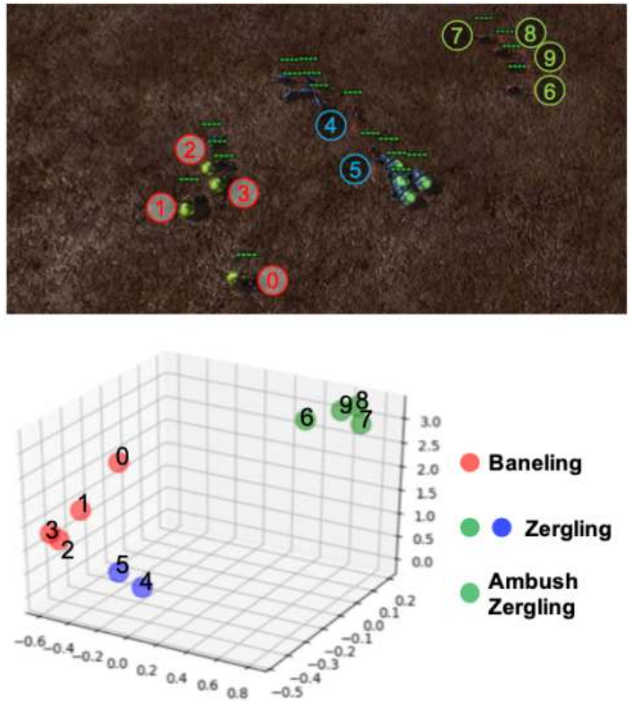
Specialized Roles



(a) Strategy: sacrificing Zealots 9 and 7 to minimize Banelings' splash damage.



(b) Strategy: forming an offensive concave arc quickly



(c) Strategy: green Zerglings hide away and Banelings kill most enemies by explosion.

Google Research Football

[Arxiv 2021]

Summary

- MARL plays a critical role for AI, but is at the early stage
- Value factorization enables scalable and effective MARL
- Communication is essential for dealing with uncertainty
- Role-based shared learning is promising for complex tasks.

- Future work:
  - Safe learning against opponents
  - Meta-learning for fast adaptation
  - Model-based MARL