ContriQ: Ally-Focused Cooperation and Enemy-Concentrated Confrontation in Multi-Agent Reinforcement Learning

Chenran Zhao 260109852@qq.com
National University of Defense Technology, Changsha 410073, China
Dianxi Shi dxshi@nudt.edu.cn
National University of Defense Technology, Changsha 410073, China
Artificial Intelligence Research Center (AIRC), National Innovation Institute of Defense Technology (NIIDT), Beijing 100166, China
Tianjin Artificial Intelligence Innovation Center (TAIIC), Tianjin 300457, China

Yaowen Zhang 250zyw@163.com
Environment Information Support 32282 Research Institute, Jinan 250000, China

Huanhuan Yang yanghh94@126.com
National University of Defense Technology, Changsha 410073, China

Shaowu Yang shaowu.yang@nudt.edu.cn
National University of Defense Technology, Changsha 410073, China

Yongjun Zhang yjzhang@nudt.edu.cn
Artificial Intelligence Research Center (AIRC), National Innovation Institute of Defense Technology (NIIDT), Beijing 100166, China

Editors: Vineeth N Balasubramanian and Ivor Tsang

Abstract

Centralized training with decentralized execution (CTDE) is an important setting for cooperative multi-agent reinforcement learning (MARL) due to communication constraints during execution and scalability constraints during training, which has shown superior performance but still suffers from challenges. One branch is to understand the mutual interplay between agents. Due to the communication constraints in practice, agents cannot exchange perceptual information, and thus, many approaches use a centralized attention network with scalability constraints. Contrary to these common approaches, we propose to learn to cooperate in a decentralized way by applying attention mechanism on the local observation so that each agent could focus on allied agents with a decentralized model, and therefore promote understanding. Another branch is to model how agents cooperate and simplify the learning process. Previous approaches that focus on value decomposition have achieved innovative results but still suffer from problems. These approaches either limit the representation expressiveness of their value function classes or relax the IGM consistency to achieve scalability, which may lead to poor performance. We combine value composition with game abstraction by modeling the relationships between agents as a bi-level graph. We propose a novel value decomposition network based on it through a bi-level attention network, which indicates the contribution of allied agents attacking enemies and the priority of attacking each enemy under the situation of each time step, respectively. We show that our method substantially outperforms existing state-of-the-art methods on battle games in StarCraftII, and attention analysis is also comprehensively discussed with sights.

Keywords: Graph, Value Decomposition, Centralized Training with Decentralized Execution, Attention Mechanism.

1. Introduction

Cooperative multi-agent reinforcement learning (MARL) has broad prospects for addressing many complex real-world problems, such as robot swarms(Hüttenrauch et al. (2017)) and autonomous cars(Cao et al. (2013)), where a system of agents learn towards coordinated policies to optimize the accumulated global rewards. This paper is interested in confrontation scenarios where allied agents have to win the battle against enemy agents. This can be modeled as cooperative-competitive multi-agent reinforcement learning (CC-MARL) settings, in which allied agents must develop an effective cooperative mechanism so that allied agents with different roles can learn to collaborate and defeat enemies.

Intelligent agents in real-world scenarios can significantly benefit from understanding the mutual interplay between agents. A natural idea to realize it is communication. Effective communication is a key ability for collaboration, and it has recently emerged as a wide-ranging approach in multi-agent environments. Examples include a wide range of communication-based MARL algorithms on promoting mutual understanding between agents: communication(Sukhbaatar et al. (2016); Singh et al. (2018)), mean-field theory(Yang et al. (2018)), causal influence(Jaques et al. (2019)) and so on. The approaches above assume that there are effective communication channels between agents, allowing them to exchange local sensory information and utilize their combined sensory experiences during the execution time so as to make better judgments and decisions. Unfortunately, this assumption is often unrealistic: in practice, agents are often independently deployed, and communications are disabled or prohibitive, and thus each agent has to predict its action conditioning on its partial observation trajectory. To overcome this challenge, some attention-based approaches(Iqbal and Sha (2018); Jiang et al. (2020)) take all these dynamics into account during the centralized training. However, it is costly and less helpful to consider all other allied agents because receiving a large amount of information incurs high computational complexity.

A salient property of human cooperation is the ability to hold targeted observations. Rather than obtaining the perception information of other agents through communication, why don't we use the local observations to promote the mutual interplay understanding? Besides, rather than the one-size-fits-all approach of considering all participating agents from the perspective of each agent as we discussed above, it can be useful to select which agents to focus on dynamically.

In this paper, we develop AFA, an Ally-Focused Actor for cooperative multi-agent deep reinforcement learning. Our key insight in AFA is to allow each agent to actively select which agents to cooperate with within local observations. It is beneficial for allied agents to know which one they should focus on or cooperate with at different stages of a complete trajectory of the battle. This targeted observation is operationalized via a simple soft attention mechanism: each agent exerts an attention mechanism on local observations, allowing the agent to pay attention to the surrounding allied agents selectively. By refining the local observation through AFA, the observation is potentially simplified for understanding and reasoning, which is indeed helpful for the understanding of the mutual interplay between agents. This enables a more flexible collaboration strategy in complex environments.

However, just using targeted observations is not enough. The interaction relationship between agents in multi-agent environments is complex. Therefore, modeling the cooperation between agents and simplifying the learning process is a crucial research area. Luckily, recent research has shown how to overcome this challenge. On the one hand, the establishment of agents cooperation model can typically be solved by value decomposition(Sunehag et al. (2017); Rashid et al. (2018); Son et al. (2019)), which focuses on designing an effective scheme that could represent the centralized multi-agent action-value function Q^{tot}, and inducing decentralized policies that allow each agent to act based on individual observations. It establishes a connection between the individual agent and the entire Multi-Agent System (MAS) through an end-to-end framework. It makes the individual agent learn in a direction that is conducive to the MAS so that the individual decisions of the agents result in jointly optimal decisions for the MAS. On the other hand, game abstraction(Yang et al. (2018); Liu et al. (2019a); Yu et al. (2015)) has recently emerged as a successful approach in various fields. The main idea behind it is to learn the relationship between agents so as to simplify the problem model of the Markov game to a smaller game and reduce the complexity of solving the game equilibrium policy.

Our approach builds a connection between value decomposition and game abstraction. We represent all agents as a complete graph to achieve game abstraction, and we propose a novel Graph-based Value Decomposition (GVD) algorithm based on the game abstraction through a bi-level attention network. More specifically, the GVD is modeled as a bi-level graph where the first level models the contribution of allied agents attacking enemies and the second level models the priority of our team attacking enemies at each time step so that the GVD network can extract interactions relations between individual value functions Q^i and produce the value of Q^{tot}. Thus, it could automatically model the mutual interplay between agents, help with speeding up multi-agent reinforcement learning and promote cooperation as it does in human cooperation.

We adopt the framework of centralized training with decentralized execution (CTDE), where AFA learns an actor for each agent during the decentralized execution. It helps agents hold targeted observations and produce individual value function Q^i based on them. Then, GVD combines all Q^i into Q^{tot} through a bi-level graph framework during the centralized training. Since the key to our approach is that collective interest is closely related to the individual contributions, this novel approach is named as ContriQ.

Experiments on the challenging MARL benchmark show that our method obtains significant improvement. The attention analysis shows that ContriQ captures proper allied agents that each agent should pay more attention to as well as proper importance weight of each agent for approximating Q^{tot} at each time step t, which reveals the internal workflow of the Q^{tot}'s approximation from Q^i.

2. Background

A fully cooperative multi-agent task can be described as a Dec-POMDP(Oliehoek and Amato (2016)) consisting of a tuple $G = \langle \mathcal{N}, S, A, P, Z, O, r, n, \gamma \rangle$. $s \in S$ describes the true state of the environment which contains the global information of all agents on the field.

At each time step, each agent $i \in \mathcal{N} \equiv \{1, ..., n\}$ chooses an action $a^i \in A$, forming a joint action $\mathbf{a} \in \mathbf{A} \equiv A^n$. This causes a transition on the environment according to the state transition function $P(s'|s, \mathbf{a}) : S \times \mathbf{A} \times S \to [0, 1]$. All agents share the same global reward function $r(s, \mathbf{a}) : S \times \mathbf{A} \to \mathbb{R}$ and $\gamma \in [0, 1)$ is a discount factor. We specially consider a partially observable scenario in which each agent i draws local observations $o^i \in O$ instead of global state s according to the observation probability function $Z(o, a) : O \times A \to O$. Each agent i has an observation-action history $\tau^i \in T \equiv (Z \times A)^*$, on which it conditions a stochastic policy $\pi^i(a^i|\tau^i) : T \times A \to [0, 1]$, aiming at maximizing global rewards. The joint policy π has a joint action-value function: $Q^\pi(s_t, \mathbf{a}_t) = \mathbb{E}_{s_{t+1:\infty}, \mathbf{a}_{t+1:\infty}}[R_t|s_t, \mathbf{a}_t]$, where $R_t = \sum_{j=0}^{\infty} \gamma^j r_{t+j}$ is the discounted return.

2.1. Attention-based Algorithms

MAAC(Iqbal and Sha (2018)) applied soft-attention mechanism to the critic so as to dynamically select which agents to attend to at each time step for each agent during the training. ATOC(Jiang and Lu (2018)) follows the structure of decentralized critics with centralized policies, where the information is shared between the policies.

Instead of simply taking the cooperative dynamics into account by applying soft attention to the centralized critic, our approach considers the cooperative mechanism from the perspective of each agent without any assumptions about communication, which achieves the cooperative goal with decentralized models.

2.2. Value Decomposition

Value Decomposition is aimed at integrating individual values Q^i into collective values Q^{tot} and giving update signals for decentralized agent policies when the value decomposition network performs backward-update. The idea of value decomposition has shown superior performance in MARL. VDN(Sunehag et al. (2017)) simply adds up the individual value Q^i of each agent without considering the difference and dynamics between agents. QMIX(Rashid et al. (2018)) considers the difference between agents on the basis of VDN by simply adding a positive weight learned from the global state s for each Q^i, but it still ignores the dynamics between agents. QTRAN(Son et al. (2019)) proposes a factorization method expressing the complete value function space induced by the IGM consistency, but its exact implementation is known to be computationally intractable.

Different from the approaches proposed before, we derive a generalized formalization of Q^{tot} and Q^i for any number of cooperative agents. It takes advantage of a bi-level graph to extract the relationship between allied agents and enemy agents through attention mechanism and produce the value of Q^{tot} from individual Q-values Q^i, which helps understand agents' mutual interplay and explicitly reveals the internal workflow of the Q^{tot}'s approximation from Q^i.

2.3. Game Abstraction and Graph Network

Game abstraction has recently emerged as an interesting approach in various fields. Recent work(Jiang et al. (2020); Iqbal and Sha (2018)) uses attention mechanism(Vaswani et al. (2017)) to learn the importance distribution of the other agents for each agent. Interestingly, the idea of learning the relationship between agents also appeared in the work of

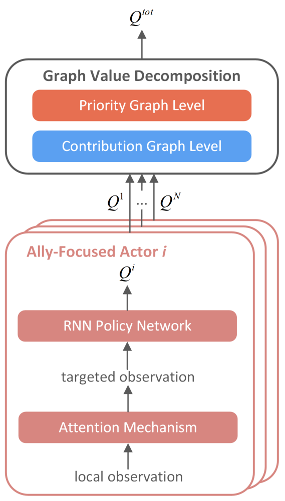


Figure 1: ContriQ

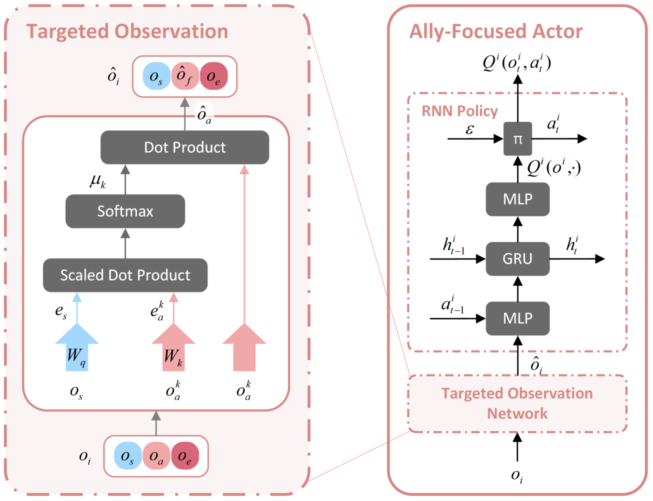


Figure 2: The overall structure of Ally-Focused Actor.

DGN(Jiang et al. (2020)) and G2ANET(Liu et al. (2019b)). The problem statements in the aforementioned papers differ slightly, but in essence, they are solving the same problem: modeling the interaction relationships between agents.

Both of the above use graph mechanism or game abstraction to obtain the contribution from other agents, which apply the graph to the actor of each agent helping make decisions, or to the centralized critic helping calculate individual Q-values Q^i. Compared with them, we propose to apply graphs to the value decomposition network, which explicitly models how agents collaborate and thus promotes reasonable credit assignment.

3. Our Approach

This section proposes a novel approach under the CTDE framework called ContriQ. As illustrated in Fig.1, ContriQ is composed of two parts: Ally-Focused Actor (AFA) and Graph Value Decomposition (GVD). We refine the observation through AFA in order to promote allied agents to focus on other allied agents selectively and dynamically. And during the centralized training, we construct a bi-level graph structure to extract the cooperation relationship and confrontation relationship between agents in order to integrate individual Q-values Q^i into global joint Q-value Q^{tot}.

3.1. Ally-Focused Actor

In a football game, each player needs to pay attention to other allied units' actions and positions all the time in order to perform cooperative actions such as passing and assisting. Similarly, we believe that paying more attention to allied agents is beneficial to the cooperation between allied agents. As illustrated in Fig.2, AFA uses decentralized attention actors to adaptively extract information about nearby allied agents in the field of view of each agent itself, which is proven to be effective for helping agents focus on other allied agents and learn cooperative strategies.

Football players can observe the movements of other players. Similarly, each allied agent can observe other agents located within the sight. For agent i, the local observation o (omit i for convenience) is separated into three parts according to the prior knowledge: o_e represents enemy information, o_a represents ally information and o_s represents self information, as Eq.(1) shows. The o_a is composed of n allies' information o_a^k $(k \in \{1, ..., n\})$. The embedding vector e_s and e_a^k could be obtained by a one or two-layer embedding transformation from o_s and o_a^k $(k \in \{1, ..., n\})$.

$$o = (o_e, o_a, o_s). \tag{1}$$

We use attention architecture between e_s and e_a^k $(k \in \{1, ..., n\})$ in the field of view of agent i, trying to focus on allied agents and promote effective interaction within local observation o. Specifically, we pass the similarity value between the allied agent k's embedding vector e_a^k and the individual embedding vector e_s into a softmax.

$$\hat{o_a} = \sum_{k=1}^{n} \mu_k \cdot e_a^k. \tag{2}$$

$$\mu_k \propto \exp(e_s \cdot W_k \cdot W_q \cdot e_a^k). \tag{3}$$

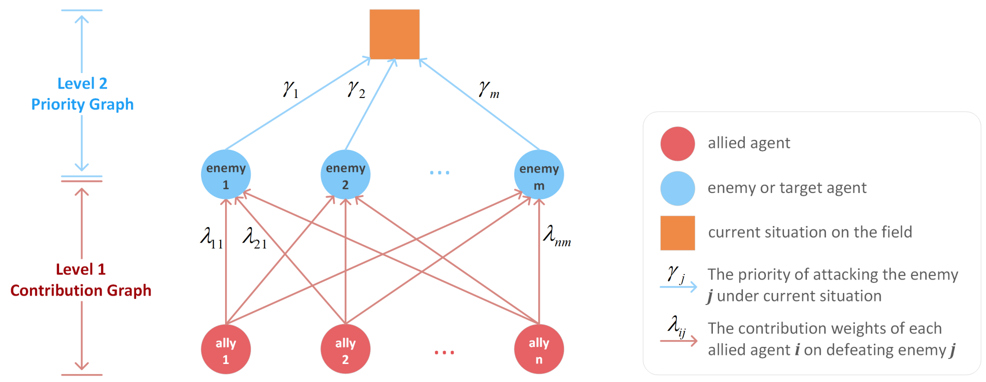
where W_q transforms e_a^k into the global query and W_k transforms e_s into the key of each allied agent.

$$\hat{o} = (o_e, \hat{o_a}, o_s). \tag{4}$$

The ally information o_a is rewritten by the attention mechanism while the rest remains the same, as Eq.(4) shows. Then the features of all parts of observation are recombined and concatenated into $\hat{o}$, which will be fed into the RNN policy network. By applying the attention mechanism to local observation, agents are motivated to hold targeted observations and being mutually helpful to strategize better as a team.

3.2. Graph Value Decomposition

Each agent plays a different role for a specific target enemy, and each enemy receives the attention that changes over time. In other words, each allied agent exerts a time-varying impact on each enemy, and the priority of attacking each enemy changes over time. The cooperation and confrontation relationship is changing violently during the battle, and it's vital for us to extract these relationships adaptively. So we construct a bi-level graph structure consisting of Contribution Graph (Level 1) and Priority Graph (Level 2), as shown in Fig.3. We construct the Graph Value Decomposition (GVD) network based on it to help model the mutual interplay between agents, including both cooperation and confrontation, and thus helps establish a connection between individual interests and collective interest, and promotes the learning of individual policy in a direction that is conducive to the collective interest. Here we describe these two levels of GVD separately.



Level 2
Priority Graph
Level 1
Contribution Graph
enemy 1
enemy 2
enemy m
ally 1
ally 2
ally n
γ_1
γ_2
γ_m
λ_{11}
λ_{21}
λ_{nm}
allied agent
enemy or target agent
current situation on the field
γ_j The priority of attacking the enemy j under current situation
λ_{ij} The contribution weights of each allied agent i on defeating enemy j

Figure 3: The whole structure of the bi-level graph consisting of the Contribution Graph and the Priority Graph. The Contribution Graph (Level 1) tries to figure out how much contribution λ_{ij} each allied agent i does to attack each enemy agent j, and it is composed of m subgraphs. The Priority Graph (Level 2) focuses on quantifying the priority γ_j of attacking each enemy j.

3.2.1. Level 1 of Graph Value Decomposition network

As discussed above, every agent is connected to the whole team, each agent learns to cooperate with other allied agents defeating their enemies at different time steps during the whole execution time. From the perspective of each enemy, it is under the attack of different combinations of allied agents at different stages, which means each allied agent exerts a time-varying impact on each enemy. Inspired by this principle, we define the Contribution Graph as below.

Definition 1 *(Contribution Graph) The contribution of all allied agents attacking enemies is defined as a directed graph as $G_1 = (N_e, N_a, E_c)$ consisting of the set N_e of target (or enemy) nodes , the set N_a of ally nodes and the set E_c of edges which are ordered pairs from N_a to N_e. Each target (or enemy) node e_i represents an enemy agent entry. Each ally node a_j represents an allied agent entry. If a_j contributes to attack e_i, then there is an edge from a_j to e_i.*

We train a multi-head attention model to learn the G_1's weights of contributions of all allied agents attacking their target enemies. Unlike the traditional multi-head attention network, each head of GVD's Level 1 network focuses on a different enemy. Thus, the number of heads is equal to the number of enemies. And each head is built based on G_1's subgraph G_1^j, which describes allied agents' contribution attacking the target enemy j. Here comes the definition of Contribution Subgraph, and it is illustrated in Fig.4.

Definition 2 *(Contribution Subgraph) The contribution of all allied agents attacking enemy j at time step is defined as a directed subgraph of G_1 as $G_1^j = (j, N_a, E_c^j)$ consisting of the enemy node j, the set N_a of ally nodes (same as in Definition 1) and the edge set E_c^j which is the subset of the edge set E_c in G_1, and it only contains the edges that pointing to enemy j. The weight of edge from allied agent i to enemy j is defined as λ_{ij}:*

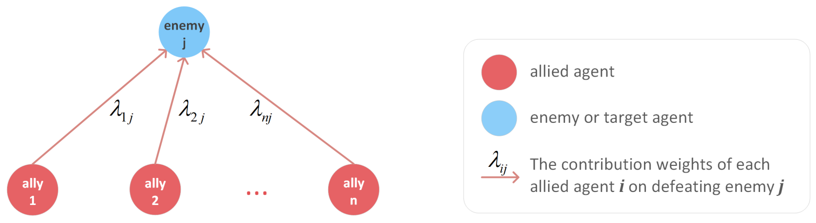


Figure 4: The structure of Contribution Subgraph.

$$\lambda_{ij} = \frac{\exp(f(e_t^j, a_t^i))}{\sum_{k=1}^{n} \exp(f(e_t^j, a_t^k))}. \tag{5}$$

where a_t^i and e_t^j are the embedding vector of ally agent i and enemy j at time step t.

Since the global state s contains information about all agents in the environment and it is available during centralized training, we can use it to help build the embedding vectors of both ally and enemy agents in each subgraph. First, we separate the global state s into two parts according to the prior knowledge: s_a represents ally information which is composed of n allies' information s_a^i $(i \in \{1, ..., n\})$ and s_e represents enemy information which is composed of m enemies' information s_e^j $(j \in \{1, ..., m\})$.

$$s = (s_a, s_e). \tag{6}$$

Then we encode the information about the target enemy j and allied agent i in state s and some additional information into embedding vectors e_t^j and a_t^i respectively by MLP as shown in Fig.5 at each time step t. More specifically, the contribution of each allied agent i is connected to its global information s_a^i and one-hot representation of action α^i, both of which are concatenated into the state-action binary (s_a^i, α^i) followed by a one or two-layer embedding transformation to obtain the ally embedding vector a_t^i. We construct a one-hot representation β^j for each enemy j for unique marking, corresponding to the one-hot representation of action α^i for each allied agent i. We combine each enemy j's global information s_e^j and its unique one-hot mark β^j into state-mark binary (s_e^j, β^j) followed by a one or two-layer embedding transformation to obtain the target embedding vector e_t^j. Combining with one-hot representation, the attention mechanism can perceive the interaction between allied agents and the target enemy more effectively.

Now we can use embedding vectors e_t^j and a_t^i $(i \in \{1, ..., n\})$ to learn the weights of contribution edges in subgraph G_1^j $(j \in \{1, ..., m\})$ according to Eq.(5). Then we mix the individual value function Q^i according to the contribution weights in G_1^j and produce the head Q-values Q_h^j for each target enemy j:

$$Q_h^j = \sum_{i=1}^{n} \lambda_{ij} \cdot Q^i. \tag{7}$$

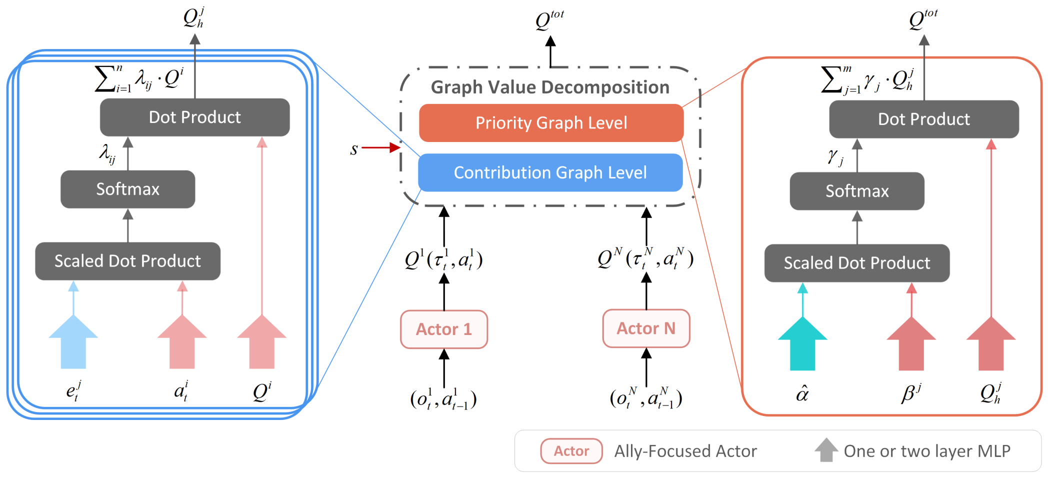


Figure 5: The overall architecture of Graph Value Decomposition (GVD). It mixes $Q^i(\tau_t^i, a_t^i)$ together with the enemy information and ally information inside s_t and produce the global joint Q-value Q^{tot}.

3.2.2. Level 2 of Graph Value Decomposition network

The priority of attacking each enemy is of great importance, and it changes over time. That is to say, allied agents should focus on different enemies based on the priority weights of attacking enemies at different time steps during a complete trajectory of battle. According to the principle above, we define the Priority Graph as below.

Definition 3 *(Priority Graph) The priority of attacking each enemy at time step t is defined as a directed graph as $G_2 = (N_e, s, E_p)$ consisting of the set N_e of enemy nodes (same as in Contribution Graph), the abstract node s and the set E_p of edges which are ordered pairs from N_e to s. The abstract node s represents the situation of current time step. If attacking enemy j is valuable for the current situation s, then there is an edge from j to s and the priority weight of edge from enemy j to s is defined as γ_j:*

$$\gamma_j = \frac{\exp(f(\hat{\alpha}, \beta^j))}{\sum_{k=1}^{m} \exp(f(\hat{\alpha}, \beta^k))}. \tag{8}$$

where $\hat{\alpha}$ and β^j are the embedding vector of current situation s and enemy j at time step t.

The one-hot representation of the agent's action contains information about the target of the attack. It reflects the ally team's judgment on the current situation to some extent, so we sum the one-hot representation α_i of all agents up as $\hat{\alpha}$ representing the embedding vector of the current situation. And we encode the enemy j's global information in state s into embedding vector β^j through a one or two-layer MLP.

Then we use embedding vectors $\hat{\alpha}$ and β^j to learn the weights of priority edges in G_2 according to Eq.(8) through a soft-attention model as shown in Fig.5. Then we mix the head Q-values Q_h^j according to the priority weights in G_2 and produce the value of Q^{tot}.

$$Q^{tot} = \sum_{j=1}^{m} \gamma_j \cdot Q_h^j. \tag{9}$$

In summary, our Graph Value Decomposition (GVD) network is a combination of the attention network of G_1 and G_2. More specifically, as shown in Fig.5, GVD is a feed-forward neural network that takes the individual value function Q^i as input and mixes them according to the Contribution Graph and the Priority Graph through a bi-level multi-head attention network using the global state s, producing the values of Q^{tot} admitting the following form.

$$Q^{tot}(s, \mathbf{a}) = \sum_{j=1}^{m} \gamma_j(s) \sum_{i=1}^{n} \lambda_{ij}(s) Q^i(s, a^i). \tag{10}$$

where $\lambda_{ij}(s)$ is the contribution weight of each allied agent i attacking enemy j and $\gamma_j(s)$ is the priority weights of attacking each enemy agent j under the current situation.

4. Experimental Evaluation

4.1. Settings

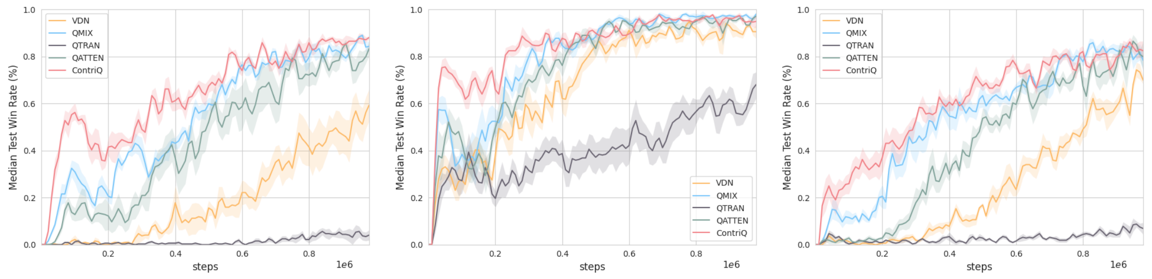
In this section, we evaluate ContriQ in the StarCraft Multi-Agent Challenge (SMAC) environment(Samvelyan et al. (2019)), which has become a common used benchmark for evaluating state-of-the-art MARL approaches that adopt the framework of CTDE such as VDN(Sunehag et al. (2017)), QMIX(Rashid et al. (2018)), QTRAN(Son et al. (2019)) and Qatten(Yang et al. (2020)). We consider six different scenarios with difficulty levels. Here we briefly introduce them in Table 1. Training and evaluation schedules are kept the same as QMIX in SMAC. The percentage of episodes where the agents defeat all enemy units, i.e., test win rate, is reported as the performance of algorithms. All the results are averaged over five independent runs with different seeds.

Table 1: Maps in easy and hard scenarios.

Level	Name	Ally Units	Enemy Units	Type
easy	2s3z	2 Zealots 3 Stalkers	2 Zealots 3 Stalkers	Symmetric & Heterogeneous
	3s5z	3 Zealots 5 Stalkers	3 Zealots 5 Stalkers	Symmetric & Heterogeneous
	2s_vs_1sc	2 Zealots	1 Spine Crawler	Asymmetric & Homogeneous
	1s3s5z	1 Colossi 3 Zealots 5 Stalkers	1 Colossi 3 Zealots 5 Stalkers	Symmetric & Heterogeneous
hard	3s_vs_5z	3 Zealots	5 Stalkers	Asymmetric & Homogeneous
	5m_vs_6m	5 Marines	6 Marines	Asymmetric & Homogeneous

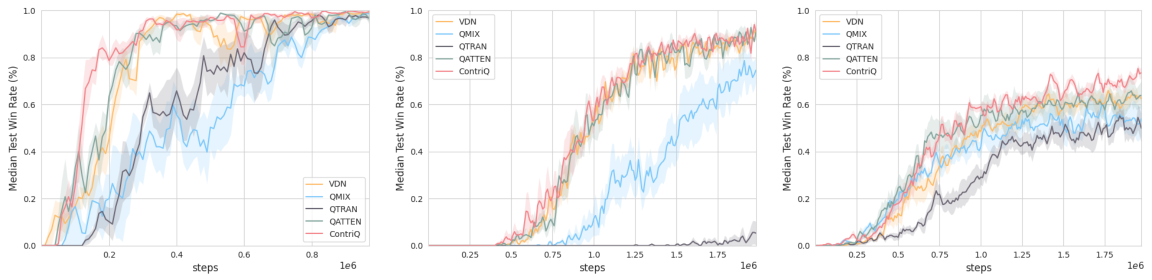
4.2. Validation

Fig.6 and Fig.7 show the median test win rate of different MARL algorithms in symmetric and asymmetric scenarios. Among all the baseline algorithms, QMIX and VDN are the state-of-the-art algorithms in the value decomposition area. QMIX could master heterogeneous scenarios while it has relatively poor performance in homogeneous scenarios. On the contrary, VDN could master homogeneous scenarios while it doesn't perform well in heterogeneous scenarios. And these results demonstrate that ContriQ achieves competitive performance with QMIX, VDN and other popular methods in both heterogeneous and homogeneous scenarios. On the one hand, there is a significant gap between ContriQ and the second-best baselines for each scenario at the beginning of training in all these homogeneous scenarios. On the other hand, in the heterogeneous scenarios, while QMIX fails to perform well and VDN exceeds all other baseline algorithms, our ContriQ is able to compete with VDN, especially on 5m_vs_6m map where ContriQ beats all other approaches. More importantly, we observe that the early test median win rate has increased very quickly, which means our ContriQ is indeed helpful for speeding up the multi-agent reinforcement learning.



(a) 3s5z (b) 2s3z (c) 1c3s5z

Figure 6: Median Test Win Rate on Symmetric & Heterogeneous Scenarios



(a) 2s_vs_1sc (b) 3s_vs_5z (c) 5m_vs_6m

Figure 7: Median Test Win Rate on Asymmetric & Homogeneous Scenarios

4.3. Attention Analysis

In addition to evaluate the performance of the trained model in Section 4.2, we are more curious about how much effect the attention mechanism actually contributes to the decision making in AFA and Q^{tot} calculating in GVD. In order to figure out what has been learned in the attention mechanism, we propose to visualize the attention weights explicitly at each time step in a complete trajectory. We choose a representative map 2s3z for illustration.

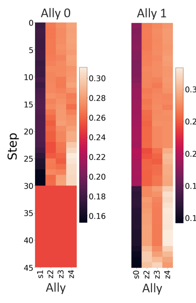
4.3.1. Attention Analysis of Ally-Focused Actor

Fig.8 has shown the attention weights (μ_k) of each allied agent focusing on other allied agents k ($k \in \{0, 1, 2, 3, 4\}$) at each time step in a complete trajectory of 2s3z battle. The Vertical ordination of the heatmap represents time steps that increase from top to bottom. Each heatmap represents the attention of a specific agent A from our team to the other four allied agents except itself at each time step. The head title of this heatmap indicates A's identity, including agent type (s means Stalker, z means Zealot) and agent id (from 0 to 4). More specifically, each column of this heatmap represents the attention of A to an allied agent B at each time step, and the bottom label of this column indicates B's identity.

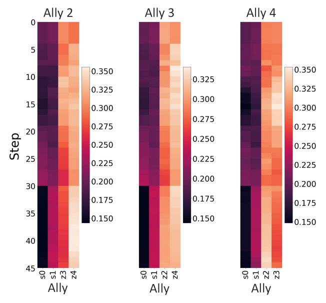
Fig.8(a) shows the attention of two Stalkers to other allied agents except itself. Obviously, these two strong Stalkers pay more attention to the fragile Zealots, which means Stalkers have learned to protect Zealots and cooperate with them to destroy the enemy. As revealed by Fig.9(a), ally Stalker 1 helps ally Zealot 2 attack enemy Zealot 3 instead of evenly matched enemy Stalkers.

Fig.8(b) shows the attention weights of three Zealots to other allied agents. Compared with strong Stalkers, Zealots always pay more attention to the other two Ally Zealots. As revealed by Relative x property and Relative y property in Fig.8(c), three Ally Zealots always stick together, and they would not be far away from each other. Fig.9(b) provides a clear explanation that these fragile Zealots (Ally 3, 4) learns to stick together so that they can focus fire on Enemy Zealot 2.

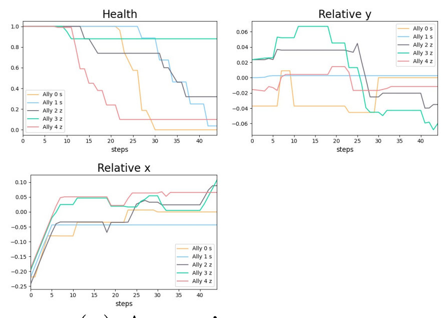
As revealed by Health property in Fig.8(c), Agent 0 is dead at time step 30. In Fig.8(a-b), we observe that the attention weights in Agent 0's heatmap no longer changes after time step 30, and the corresponding attention weights in the heatmap of other allied agents turn



(a)



(b)



(c) Agents' property

Figure 8: Attention Analysis in AFA

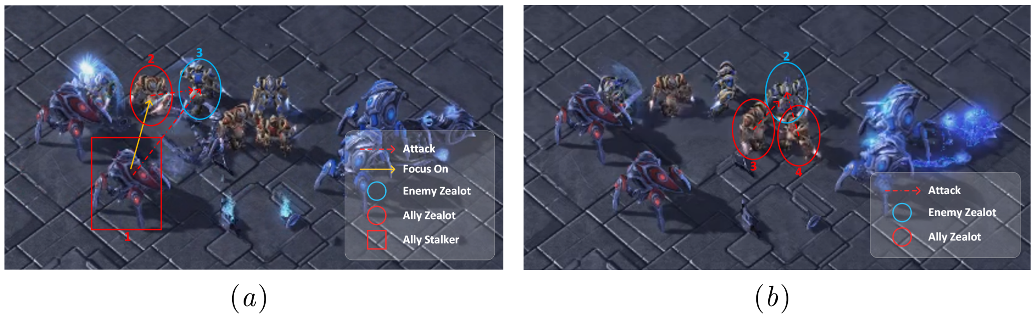


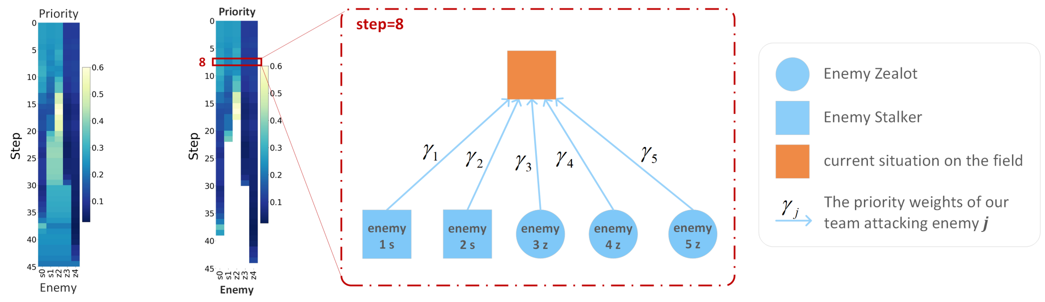
Figure 9: Snapshots of allied agents during a complete trajectory of 2s3z battle

to be darker after time step 30 compared to other living agents. In contrast, those living agents' corresponding attention weights in others' heatmap turn to be brighter compared to the part before time step 30, indicating that the living agents turn their attention to other allied agents after Agent 0 is dead.

The qualitative analysis indicates that AFA is indeed helpful for holding targeted observations and thus promote better decision-making.

4.3.2. Attention Analysis of Graph Value Decomposition

Fig.10(a) has shown the priority weights (γ_j) of our team attacking each enemy j under the situation of each time step in a complete trajectory of 2s3z battle. Each column of the heatmap in Fig.10(a) represents our team's judgment on the priority of attacking a specific enemy agent C at each time step. The bottom label of this column indicates C's identity. The five grids in each row of the heatmaps in Fig.10(a) respectively represent our team's judgment on the priority of attacking five enemy agents at a certain time step.



(a) Priority heatmap (b) The cropped version heatmap and the Priority Graph

Figure 10: Heatmap of Priority Graph Level

For better understanding, we cut out part of the heatmap where the corresponding enemy is dead. The cropped version is shown in Fig.10(b). Besides, we attach a Priority Graph of 2s3z battle in Fig.10(b) to explain the priority heatmap. As you can see, our team constructs a Priority Graph representing our team's judgment on the current situation of the battle at each time step during the battle (we take time step 8 for example).

One main tendency that we have observed in Fig.10(b) is that the priority of our team attacking enemy agent D is getting higher and higher several time steps before D is dying, which means our team has learned to focus fire on the most vulnerable enemy agent so as to achieve the goal of weakening the enemy combat power as soon as possible.

Fig.11(a-b) has shown the contribution weights (λ_{ij}) of our agents i ($i \in \{0, 1, 2, 3, 4\}$) attacking each enemy j under the situation of each time step in a complete trajectory of 2s3z battle. Each heatmap represents the contribution of our team attacking a specific enemy E at each time step. The head title of this heatmap indicates E's identity. Each column of this heatmap represents the contribution of an agent F from our team attacking E at each time step, and the bottom label of this column indicates F's identity.

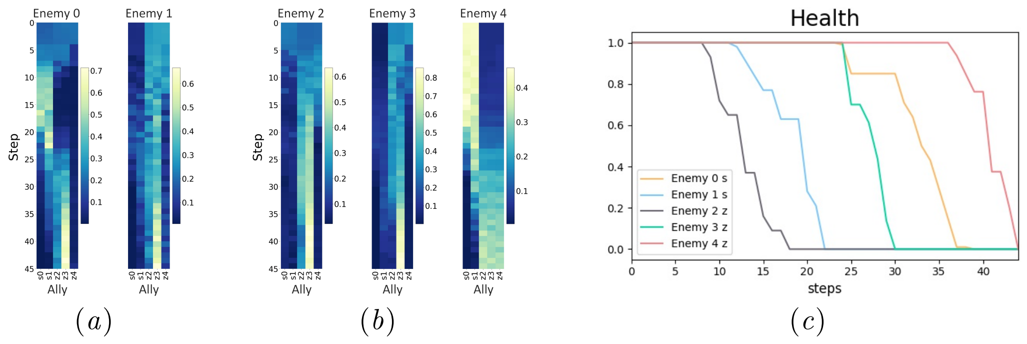


Figure 11: Heatmap of Contribution Graph Level

After analyzing the correlation of attention weights and agent features in Fig.11, we notice that two Ally Stalkers contribute more to attacking Enemy 0 and Enemy 4 at the beginning of the battle. Gradually, Ally Zealots become the main contributors to Enemy 0 and Enemy 4. Besides, three Ally Zealots contribute more to attacking Enemy 1-3.

Since Ally 0 is dead at time step 30 (as the health property in Fig.8(c) shows), Ally 0's corresponding part in heatmaps becomes darker and darker before time step 30 as revealed by Fig.11(a-b), which means that the dying alive unit (Agent 0) exerts a fewer and fewer contribution on attacking enemies. And once Ally 0 is dead (after time step 30), Ally 0's corresponding part in heatmaps becomes completely dark. I believe that our GVD could encourage agents to survive for contributing and winning.

As discussed in Section 3.2.1, the attention weights within a complete trajectory of battle are changing violently. Under these circumstances, GVD could model the interaction relationships between agents properly and thus approximate the sophisticated relations between Q^i and Q^{tot}, and help with speeding up the learning process.

5. Conclusion

This paper presented ContriQ, a deep multi-agent reinforcement learning method, which learns an ally-focused policy-maker for each agent in a decentralized way. Besides, for the first time, we derive a decomposition formula of Q^{tot} and Q^i through a bi-level graph structure using a multi-head attention mechanism. Our empirical results carried on the battle games in StarCraftII demonstrate that our method induces better-trained policy compared with several state-of-the-art competitors. We further perform the attention analysis to visualize the attention weights of each agent focusing on other allied units and the contribution weights of each allied agent on defeating enemies. The results provide intuitive explanations on the effects of the learned attention mechanism and, to some degree, reveals the internal workflow of the Q^{tot}'s approximation from Q^i.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 61803375).

References

Yongcan Cao, Wenwu Yu, Wei Ren, and Guanrong Chen. An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Trans. Ind. Informatics*, 9(1): 427–438, 2013. doi: 10.1109/TII.2012.2219061. URL https://doi.org/10.1109/TII.2012.2219061.

Maximilian Hüttenrauch, Adrian Sosic, and Gerhard Neumann. Guided deep reinforcement learning for swarm systems. *CoRR*, abs/1709.06011, 2017. URL http://arxiv.org/abs/1709.06011.

Shariq Iqbal and Fei Sha. Actor-attention-critic for multi-agent reinforcement learning. *CoRR*, abs/1810.02912, 2018. URL http://arxiv.org/abs/1810.02912.

Natasha Jaques, Angeliki Lazaridou, Edward Hughes, Caglar Gulcehre, Pedro Ortega, DJ Strouse, Joel Z Leibo, and Nando De Freitas. Social influence as intrinsic motivation for multi-agent deep reinforcement learning. In *International Conference on Machine Learning*, pages 3040–3049. PMLR, 2019.

Jiechuan Jiang and Zongqing Lu. Learning attentional communication for multi-agent cooperation. *CoRR*, abs/1805.07733, 2018. URL http://arxiv.org/abs/1805.07733.

Jiechuan Jiang, Chen Dun, Tiejun Huang, and Zongqing Lu. Graph convolutional reinforcement learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=HkxdQkSYDB.

Yong Liu, Yujing Hu, Yang Gao, Yingfeng Chen, and Changjie Fan. Value function transfer for deep multi-agent reinforcement learning based on n-step returns. In Sarit Kraus, editor, *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 457–463. ijcai.org, 2019a. doi: 10.24963/ijcai.2019/65. URL https://doi.org/10.24963/ijcai.2019/65.

Yong Liu, Weixun Wang, Yujing Hu, Jianye Hao, Xingguo Chen, and Yang Gao. Multi-agent game abstraction via graph attention neural network. *CoRR*, abs/1911.10715, 2019b. URL http://arxiv.org/abs/1911.10715.

Frans A. Oliehoek and Christopher Amato. *A Concise Introduction to Decentralized POMDPs*. Springer Briefs in Intelligent Systems. Springer, 2016. ISBN 978-3-319-28927-4. doi: 10.1007/978-3-319-28929-8. URL https://doi.org/10.1007/978-3-319-28929-8.

Tabish Rashid, Mikayel Samvelyan, Christian Schröder de Witt, Gregory Farquhar, Jakob N. Foerster, and Shimon Whiteson. QMIX: monotonic value function factorisation for deep multi-agent reinforcement learning. *CoRR*, abs/1803.11485, 2018. URL http://arxiv.org/abs/1803.11485.

Mikayel Samvelyan, Tabish Rashid, Christian Schröder de Witt, Gregory Farquhar, Nantas Nardelli, Tim G. J. Rudner, Chia-Man Hung, Philip H. S. Torr, Jakob N. Foerster, and Shimon Whiteson. The starcraft multi-agent challenge. *CoRR*, abs/1902.04043, 2019. URL http://arxiv.org/abs/1902.04043.

Amanpreet Singh, Tushar Jain, and Sainbayar Sukhbaatar. Learning when to communicate at scale in multiagent cooperative and competitive tasks. *CoRR*, abs/1812.09755, 2018. URL http://arxiv.org/abs/1812.09755.

Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Hostallero, and Yung Yi. QTRAN: learning to factorize with transformation for cooperative multi-agent reinforcement learning. *CoRR*, abs/1905.05408, 2019. URL http://arxiv.org/abs/1905.05408.

Sainbayar Sukhbaatar, Arthur Szlam, and Rob Fergus. Learning multiagent communication with backpropagation. *CoRR*, abs/1605.07736, 2016. URL http://arxiv.org/abs/1605.07736.

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinícius Flores Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. Value-decomposition networks for cooperative multi-agent learning. *CoRR*, abs/1706.05296, 2017. URL http://arxiv.org/abs/1706.05296.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL http://arxiv.org/abs/1706.03762.

Yaodong Yang, Rui Luo, Minne Li, Ming Zhou, Weinan Zhang, and Jun Wang. Mean field multi-agent reinforcement learning. *CoRR*, abs/1802.05438, 2018. URL http://arxiv.org/abs/1802.05438.

Yaodong Yang, Jianye Hao, Ben Liao, Kun Shao, Guangyong Chen, Wulong Liu, and Hongyao Tang. Qatten: A general framework for cooperative multiagent reinforcement learning. *CoRR*, abs/2002.03939, 2020. URL https://arxiv.org/abs/2002.03939.

Chao Yu, Minjie Zhang, Fenghui Ren, and Guozhen Tan. Multiagent learning of coordination in loosely coupled multiagent systems. *IEEE Trans. Cybern.*, 45(12):2853–2867, 2015. doi: 10.1109/TCYB.2014.2387277. URL https://doi.org/10.1109/TCYB.2014.2387277.