计算机科学
COMPUTER SCIENCE

稀疏异质多智能体环境下基于强化学习的课程学习框架

罗睿卿，曾坤，张欣景

引用本文

罗睿卿，曾坤，张欣景. 稀疏异质多智能体环境下基于强化学习的课程学习框架[J]. 计算机科学，2024，51(1)：301-309.

LUO Ruiqing, ZENG Kun, ZHANG Xinjing. Curriculum Learning Framework Based on Reinforcement Learning in Sparse HeterogeneousMulti-agent Environments [J]. Computer Science, 2024, 51(1): 301-309.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.230500146

稀疏异质多智能体环境下基于强化学习的课程学习框架

罗睿卿[1]　曾　坤[1]　张欣景[2]
1 中山大学计算机学院　广州 510006
2 中国人民解放军 91976 部队　广州 510430
(ruiqingluo1@163.com)

摘　要　现代战争的战场较大且兵种较多,利用多智能体强化学习(MARL)进行战场推演可以加强作战单位之间的协同决策能力,从而提升战斗力。当前 MARL 在兵棋推演研究和对抗演练中的应用普遍存在两个简化:各个智能体的同质化以及作战单位分布稠密。实际战争场景中并不总是满足这两个设定,可能包含多种异质的智能体以及作战单位分布稀疏。为了探索强化学习在更多场景中的应用,分别就这两方面进行改进研究。首先,设计并实现了多尺度多智能体抢滩登陆环境 M2ALE,M2ALE 针对上述两个简化设定做了针对性的复杂化,添加了多种异质智能体和作战单位分布稀疏的场景,这两种复杂化设定加剧了多智能体环境的探索困难问题和非平稳性,使用常用的多智能体算法通常难以训练。其次,提出了一种异质多智能体课程学习框架 HMACL,用于应对 M2ALE 环境的难点。HMACL 包括 3 个模块:1)任务生成模块(STG),用于生成源任务以引导智能体训练;2)种类策略提升模块(CPI),针对多智能体系统本身的非平稳性,提出了一种基于智能体种类的参数共享(Class Based Parameter Sharing)策略,实现了异质智能体系统中的参数共享;3)训练模块(Trainer),通过从 STG 获取源任务,从 CPI 获取最新的策略,使用任意 MARL 算法训练当前的最新策略。HMACL 可以缓解常用 MARL 算法在 M2ALE 环境中的探索难问题和非平稳性问题,引导多智能体系统在 M2ALE 环境中的学习过程。实验结果表明,使用 HMACL 使得 MARL 算法在 M2ALE 环境下的采样效率和最终性能得到大幅度的提升。

关键词:多智能体强化学习;作战仿真;课程学习;参数共享;多智能体环境设计
中图分类号　TP183

Curriculum Learning Framework Based on Reinforcement Learning in Sparse Heterogeneous Multi-agent Environments

LUO Ruiqing[1],ZENG Kun[1] and ZHANG Xinjing[2]
1 School of Computer Science and Engineering,Sun Yat-sen University,Guangzhou 510006,China
2 91976 Unit,People's Liberation Army of China,Guangzhou 510430,China

Abstract　The battlefield of modern warfare is large and has a variety of units,and the use of multi-agent reinforcement learning (MARL) in battlefield simulation can enhance the collaborative decision-making ability among combat units and thus improve combat effectiveness. Current applications of Multi-agent reinforcement learning(MARL) in military simulation often rely on two simplifications:the homogeneity of agents and dense distribution of combat units,real-world warfare scenarios may not always adhere to these assumptions and may include various heterogeneous agents and sparsely distributed combat units. In order to explore the potential applications of reinforcement learning in a wider range of scenarios,this paper proposes improvements in these two aspects. Firstly,a multi-scale multi-agent amphibious landing environment(M2ALE) is designed to address the simplifications,incorporating various heterogeneous agents and scenarios with sparsely distributed combat units. These complex settings exacerbate the exploration difficulty and non-stationarity of multi-agent environments,making it difficult to train with commonly used multi-agent algorithms. Secondly,a heterogeneous multi-agent curriculum learning framework(HMACL) is proposed to address the challenges in the M2ALE environment. HMACL consists of three modules:source task generating(STG) module,class policy improving(CPI) module,and Trainer module. The STG module generates source tasks to guide agent training,while the CPI module proposes a class-based parameter sharing strategy to mitigate the non-stationarity of the multi-agent system and implement parameter sharing in a heterogeneous agent system. The Trainer module trains the latest policy using any MARL algo-

到稿日期:2023-05-22　返修日期:2023-09-20
基金项目:国家自然科学基金(U1711266);广东省基础与应用基础研究基金联合基金(2019A1515011078)
The work was supported by the National Natural Science Foundation of China(U1711266) and Guangdong Basic and Applied Basic Research Foundation(2019A1515011078).
通信作者:曾坤(zengkun2@mail.sysu.edu.cn)

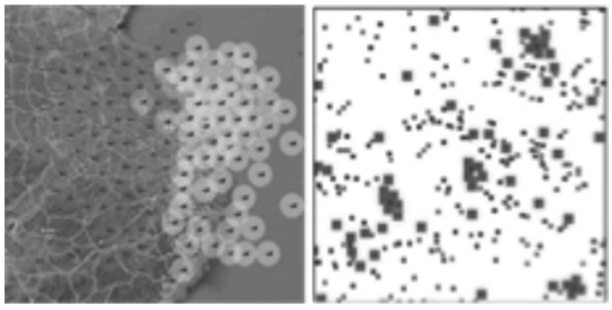
rithm with the source tasks generated by the STG and the latest policy from the CPI. HMACL can alleviate the exploration difficulty and non-stationarity issues of commonly used MARL algorithms in the M2ALE environment and guide the learning process of the multi-agent system. Experiments show that using HMACL significantly improves the sampling efficiency and final performance of MARL algorithms in the M2ALE environment.

Keywords Multi-agent reinforcement learning, Combat simulation, Curriculum learning, Parameter sharing, Multi-agent environment design

1 引言

目前多智能体深度强化学习(Multi-Agent Deep Reinforcement Learning, MARL)在很多任务上取得了突破性的进展,表现达到甚至超越了人类水平,如 MPE[1], SMAC[2], Pettingzoo[3]等。随着现代战争的复杂程度日益加深,可以利用多智能体深度强化学习方法应对战场模拟和决策问题。

MARL 在战场仿真中主要应用于大规模无人机群[4]或类似多智能体场景[5],其研究通常存在两个简化的设定。1)同质化的智能体,即所有智能体为同一类。目前的大多数研究工作采用的环境都是同质化的智能体,这样的设定降低了动作和状态空间的多样性和复杂性。2)作战单位分布稠密。如果作战单位分布稠密,智能体之间或智能体与环境中的其他因素就更容易产生交互,进而更容易探索出有意义的策略;但是,如果作战单位分布稀疏,智能体面对空旷的环境则需要更多的探索才能找到有意义的动作,如图 1 所示。在上述简化设置下,MARL 算法容易达到较好的效果。



(a)大规模无人机群[4] (b)MAgent[5]

图 1 普遍采用的简化设定

Fig. 1 Simplifying assumptions commonly adopted

但由于战场模拟的复杂性,单纯使用 MARL 算法难以适应环境的变化。具体而言,环境中有多个智能体,且分为若干个种类,例如在本研究中,有 10 个以上智能体,分为 4 个种类;其次地图会较大,这使得智能体分布会比较稀疏。在现实场景中,上述两个简化设定一旦被打破,常用的 MARL 算法就难以学习到有意义的策略。因此如果想要探索多智能体强化学习在现实场景中的应用,需要消除上述两个简化设定,以训练鲁棒的智能体集群,实现在具有多种功能各异的非同质化多智能体的环境中提高算法的效率和性能,以及智能体在作战单位分布稀疏的环境中依然能够高效地探索和学习。

为了探索多智能体强化学习在此类环境中的应用,本文首先提出了一个全新的多尺度多智能体抢滩登陆环境 M2ALE(Multi-scale Multi-agent Amphibious Landing Environment),环境中有多种不同类型的智能体和不同大小尺度的环境设置。M2ALE 环境一定程度上弥合了当前研究与现实场景的差距,同时也带来了更大的挑战,核心问题就是如何在稀疏的、充满多样性的环境中平衡探索(Explore)与利用(Exploit)[6]的问题。

作为强化学习中的重要问题,智能体对环境的探索与利用是智能体选择继续探索未知的状态以寻找更优的策略,还是基于当前的经验选择最优的策略。随着环境复杂度的增加,探索的难度也增加。使用强化学习对战争环境建模的一个难点是高复杂度带来的极高的探索成本。目前强化学习在各种特定的模拟环境中取得了长足的进步,但绝大多数环境的状态空间较小,使用常用的探索方法(如 Epsilon-greedy)[7]可以使算法收敛。在 M2ALE 中,随着环境规模的增大,探索的复杂度呈指数增长。这种情况下,单纯的 MARL 算法难以进行高效的训练,需要借助额外的方法来引导智能体学习。

多智能体强化学习需要解决的另外一个重要问题就是多智能体环境本身的非平稳性。这种非平稳性主要来自于同时有多个智能体在根据当前环境和奖励改变自己的策略,在算法层面缓解这个问题的常用方式是采用中心化的评论家(Critic)网络[8-9]。另外,多个智能策略采用参数共享(Parameter Sharing)的方式也可以极大地缓解此问题[10-11]。在异质智能体之间,往往具有不同的动作空间和状态空间,并且如果在异质智能体之间使用参数共享,不同性质的智能体之间会相互影响,往往达不到理想的效果,甚至会阻碍智能体集群的训练过程[12]。

针对 M2ALE 环境中存在的两个重要问题中的难点,本文提出了一个异质多智能体课程学习框架 HMACL(Heterogeneous Multi Agents Curriculum Learning),该框架能够将 M2ALE 及其类似环境中的异质多智能体集群迁移到大规模、分布稀疏的场景中。异质多智能体系统(Heterogeneous Multi Agents Systems, HMAS)[13-14]指多智能体系统中包含多种性质不同的智能体,智能体性质的不同可能导致智能体观测空间和动作空间不一致。强化学习中的自动课程学习(Automatic Curriculum Learning for RL, ACL)[15]是训练智能体的一种方式,通过为智能体系统生成一系列适应智能体的课程来调整训练数据的分布。异质多智能体课程学习框架(HMACL)由 3 个模块组成。

1)源任务生成(Source Task Generating, STG)模块。考虑到军事实战中作战单位的信息和视野具有局限性,某几个作战单位之间只有局部的有限信息,因此局部的协同作战其实是整个问题的子问题。故可以通过源任务生成的方式来生成多种尺度的环境,从而逐步引导(bootstrap)智能体学习。先训练智能体在小规模问题上的能力,然后慢慢扩大问题规模,最终使得智能体能够适应超大规模的战场环境,缓解大规模

战争环境中高复杂度带来的探索难问题。

2) 基于种类的策略提升(Class Based Policy Improving, CPI) 模块。由于环境中存在多种异质智能体,对全局智能体使用参数共享往往达不到理想的效果,甚至比非参数共享下的训练效果更差。HMACL 对每种智能体创建种类策略(Class Policy),同一种智能体可以共享种类策略参数。采用这样的方式训练的智能体更加鲁棒,能够缓解多智能体策略带来的非平稳性。

3)训练(Trainer)模块。训练模块从 STG 获取任务,从 CPI 获取最新的策略,然后根据策略和任务继续训练当前策略,训练完成后交给 CPI 评估并更新最新的策略。HMACL 允许训练模块采用任意的 MARL 算法训练。

需要强调的是,HMACL 的目的不是为了探索环境的多样性[16],或者训练智能体的零样本迁移能力[17],而是缓解在分布稀疏的异质多智能体任务中的探索难问题和非平稳性问题。

本文的主要贡献和创新点如下:

1)设计并实现了 M2ALE 环境,并为环境提供了 OpenAI. gym 风格接口。M2ALE 为多智能体算法提供新的环境,通过环境配置可以创建具备多种智能体、分布稀疏的强化学习环境。

2)提出了异质多智能体课程学习框架 HMACL,缓解了 MARL 算法在分布稀疏的异质多智能体任务中的探索难问题和非平稳性问题。高效地引导 M2ALE 中的智能体从分布稠密的环境迁移到分布稀疏的环境,降低了探索的复杂度,提高了智能体的鲁棒性。

3)实验结果表明,HMACL 可以大大提高原始 MARL 算法在大规模、稀疏环境中的采样率和最终性能。

2 相关工作

本文主要研究在抢滩登陆场景下,高效训练多智能体协同作战的方法,并提出了多尺度的抢滩登陆合作对抗模拟环境 M2ALE 和使用 MARL 算法高效训练 M2ALE 的课程学习框架 HMACL。下文从军事战场模拟、MARL 算法以及强化学习(Reinforcement Learning,RL)中的课程学习 3 个方面描述相关工作。

2.1 军事仿真与模拟

近年来,随着人工智能及强化学习的快速发展,军事领域也开始逐渐应用这些技术进行战场仿真和算法模拟。首先,文献[18]提出了一种基于强化学习的多智能体协同对抗算法。该算法通过智能体之间的信息交流和协作来完成对抗任务,实现了协同作战和智能化决策。同样基于通信的方法还有文献[19]。这些方法与 HMACL 的区别在于,HMACL 是智能体之间通过各自的观测来决策,而不需要与友军通信。文献[20]提出了一种基于仿真推演的海战评估方法。该方法通过仿真模拟,对海战过程进行评估,为制定战术决策提供了重要参考。同时,该方法还可以与强化学习算法相结合,帮助实现智能化决策和协同作战评估。文献[21]提出了一种基于后悔值的多 Agent 冲突博弈强化学习模型。该模型可以用于军事战场中的决策制定和策略优化,实现协同作战和智能化决策。该模型通过仿真实验验证了其有效性和可行性。文献[22]提出了结合多目标优化与强化学习结合的空战决策方法,目的是优化单个智能体在多目标环境中的策略。文献[4]提出了一种基于多智能体强化学习的大规模无人机集群对抗。该算法可以用于实现无人机的智能化控制和决策,实现协同作战和智能化决策。这也是与本文的工作比较接近的一项研究,但是文献[4]的环境中使用了同质化的智能体,以及高密度的作战单位。类似的环境设定还有文献[5]提出的 MAgent 模拟环境,其环境中的设定同样是大规模的同质化智能体。本文的目标是在具备多种非同质化智能体的环境中,将智能体集群迁移到大规模环境中。

2.2 MARL 算法

多智能体强化学习由最初的单智能体强化学习[23-24]扩展而来,最初 IQL[25]把环境中的其他智能体看作环境的一部分,智能体之间彼此独立学习(IL)。IL 架构最大的问题就是忽略了所有智能体会共同影响环境,独立的学习策略导致环境呈非稳态,虽然 IQL,IA2C[26],IPPO[27]等算法在较小的环境中能训练出较好的策略,但是这些算法不能从理论上证明算法的收敛性,当环境复杂度上升时效果并不理想。

为了缓解独立学习的环境非平稳性,中心化训练去中心化执行架构(CTDE)使智能体在训练期间需要共享全局信息,而在执行阶段只需要基于自身的观测值来决策。CTDE 架构主要有两个方向,第一个方向是中心化策略梯度方法,在训练阶段共享全局的状态信息来训练只在训练阶段需要的 Critic 网络,而执行阶段需要的 Actor 网络则只依赖于智能体自身的观测值,代表性的算法有 MADDPG[28],MAPPO[29],COMA[30];另外一个方向是基于值分解的方法,智能体通过基于值的算法训练局部 Q 网络,然后将所有的局部 Q 值输入一个 mixer 网络,通过对 mixer 网络的反向传播施加一定的单调性限制,将联合 Q 函数分解到各个局部的 Q 网络中。此外,mixer 网络还可以输入额外的全局信息来辅助训练,代表性的算法有 VDN[31],QMIX[32]等。

除了基于 CTDE 架构的 MARL 算法,对合作设定下的 MARL 算法采用参数共享也可以很大程度缓解环境的非平稳性问题[10,33]。

2.3 课程学习

目前强化学习中的课程学习主要有两种方式。第一种方式是通过生成能够帮助目标任务训练的源任务来进行课程学习。文献[34]通过预先指定环境的参数来制定课程,如在快速国际象棋的环境中预先定义好棋盘的大小和棋子的数量等。这是一种手动的任务生成方法,需要在训练之前设计完成所有的课程,由于源任务和目标任务之间没有明确的界限,设计出来的源任务可能无法解决,最终导致智能体在目标任务上甚至无法达到不使用课程学习的效果。文献[35]提出了一种基于最小最大后悔值的自动环境设计,将神经网络的输出作为转换函数的输入,损失函数的目的是最大化主角(Protagonist)的后悔值来生成难度更高的环境,而主角则通过最小化最大后悔值来优化策略(Minimax Regret)。这种方法能够保证源任务的可解性,但是只能应用于单智能体环境中。上述任务都是针对环境内部元素的设计,目标任务与源任务

没有明确的界限。如果考虑环境的尺度变化，而智能体的局部观测值不变，那么上述方法依然无法训练有效的智能体。

第二种方式是通过适当的排列智能体的经验(轨迹)来帮助智能体训练，即为了把智能体迁移到难的任务中，从简单到容易排列训练轨迹，逐步引导智能体。文献[36]提出了一种基于优先级的经验回放，给予当前期望对智能体帮助大的轨迹更高的采样概率来加速训练。文献[37]利用所有的经验轨迹来加速训练，包括达到预期的经验轨迹和未达到预期的经验轨迹，为这样的经验设置与轨迹相同的优化目标。经验重排通过改变轨迹的采样顺序来加速训练，这样的方式可以被无缝地结合到各种方法中。但是在一些复杂的多智能体环境中，单纯依靠上述方法不足以训练高效鲁棒的多智能体集群。

3 环境与方法

本章包含两个小节，3.1 小节介绍了多尺度多智能体抢滩登陆环境 M2ALE，3.2 小节介绍了异质多智能体课程学习框架 HMACL。

3.1 M2ALE 环境

M2ALE(Multi-scale Multi-agent Amphibious Landing Environment)环境是以抢滩登陆为背景的多智能体强化学习环境，主要目的是削弱当前研究中普遍存在的两个简化设定，模拟战场环境下智能体分布稀疏和多种异质智能体的特点，如图 2 所示，图中包含战斗机、无人机、舰艇等作战单位及其视野范围，Attacker 代表防守方，Defender 代表进攻方，可以通过配置更改环境的尺度来控制作战单位的稀疏程度。本文针对海战战场这样一种不完全、不完美的信息博弈，以及抢滩登陆的特点做出了模拟，可以通过更改 M2ALE 环境参数，来达到生成任务的目的。

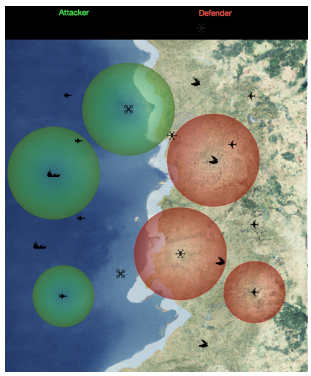


图 2 M2ALE 环境

Fig. 2 M2ALE environment

3.1.1 M2ALE 概览

M2ALE 是一个具备多种非同质化多智能体的多尺度环境，其中存在两方对战，分别为进攻方(Attacker)和防守方(Defender)。环境大小为 $n*n$，一半为海面，一半为陆地。进攻方从海面发动进攻，目的是消灭全体敌方单位，以便登陆攻占陆地；防守方则占领陆地，目标是通过消灭全体敌方单位或坚持一定的时间步数来击退敌方。双方分别拥有一定数量的作战单位，如舰艇、战斗机等，这些作战单位各自的功能，将在 3.1.2 节中具体介绍。

M2ALE 提供了部分参数可供选择，以便为算法提供验证，设置了几种难度不同的初始场景。环境的主要参数如表 1 所列。

表 1 M2ALE 环境的主要参数

Table 1 Major parameters of M2ALE

参数	作用	默认值
tag_map_size	目标任务的大小	50 * 50
map_size	初始源任务的大小	8 * 8
pos_init_strategy	智能体位置初始化策略	random
reward_type	奖励类型	dense
max_step_length	episode 最大步长	200
scenario	指定预设场景	5u_vs_5u
expand_degree	源任务的扩展度	2
attacker	进攻方智能体设置	
defender	防守方智能体设置	

对于参数 pos_init_strategy，可以有 random 和 specified 两种选择。其中 random 指作战双方从己方阵营的边缘随机选择，例如对于地图大小为 20 * 20 的场景，进攻方初始位置为$[x,1]$，防守方初始位置为$[x,20]$，x 为随机选择；specified 则为每个作战单位的指定位置。另外环境还提供了一个 expand 函数，用于扩展地图的大小。如果环境 e 的初始大小为 $s_e=[x,y]$，$expand\_degree=2$，那么调用 expand 扩展环境 $e'=e.\text{expand}(2)$，此时 $s_{e'}=[x+2,y+2]$。

3.1.2 智能体设定

为了模拟抢滩登陆环境的多兵种特点，M2ALE 中引入了多种功能特点各异的智能体，包括舰艇、战斗机、炮台、无人侦查机。依据各自的性质和功能，每个智能体会配置一个可执行动作的集合，动作包含移动、操作两部分。移动指智能体在环境中改变位置，不同的智能体可移动的范围也不一样，如舰艇只能在海面移动；操作指智能体执行一个作用于环境的除移动以外的动作。下面分别介绍各种智能体的功能和特点。

舰艇：可以在地图中左右移动来靠近或远离岸边(从舰艇视角来看是前后移动)，舰艇只能在水面移动；舰艇可以攻击被观察到的、射程范围内的敌方单位。

战斗机：可以在战场范围的上空前后左右移动，可以攻击被观察到的、射程范围内的敌方单位。

移动炮台：可以在陆地范围内前后左右移动，可以攻击被观察到的、射程范围内的敌方单位。

侦查无人机：可以在战场范围的上空前后左右移动。侦查无人机的主要战略作用是为友军发现敌军目标的位置和状态，并将信息同步给附近的友军。

3.1.3 奖励函数

M2ALE 提供稀疏(sparse)和稠密(dense)两种奖励函数以供选择。对于稀疏奖励，只在 episode 结束时，给予胜利方 1 的奖励和失败方 −1 的奖励；对于稠密奖励，M2ALE 根据每回合对敌方单位造成的伤害和被敌方单位造成的伤害的加和来计算，会在每个 step 结束时返回。

3.2 HMACL 框架

HMACL 框架包含 3 个部份，如图 3 所示。

1)种类策略提升模块(CPI)。CPI 是维护和提升智能体集群的最优策略。由于 M2ALE 环境中包含多种异质智能体，因此 CPI 提出了一种应用于异质多智能系统的

参数共享策略。

2)源任务生成模块(STG)。通过生成难度逐渐增加的任务来引导智能体训练,同时评估过往任务学习潜力来回放对智能体集群提升较大的过往任务。

3)训练模块(Trainer)。从 STG 获取源任务训练,从 CPI 获取最新的种类策略并训练。

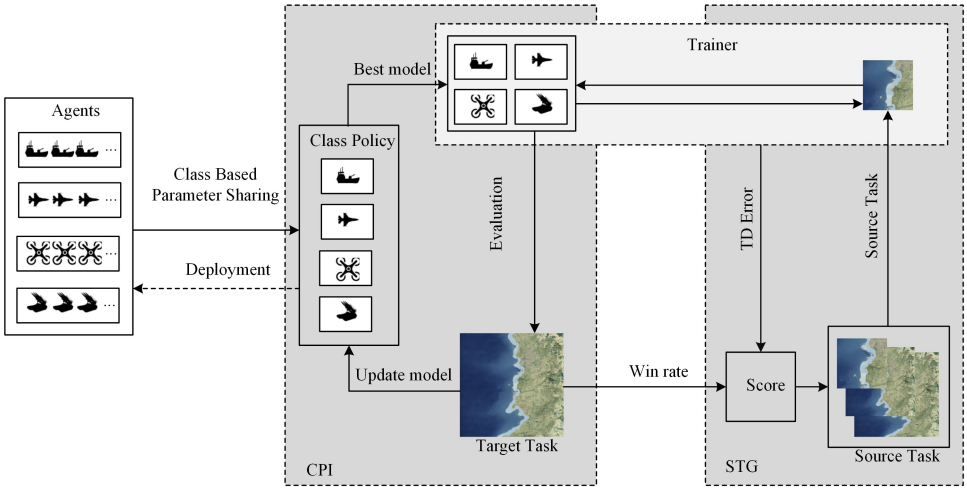


图 3 HMACL 框架

Fig. 3 HMACL framework

HMACL 框架的流程如算法 1 所示。

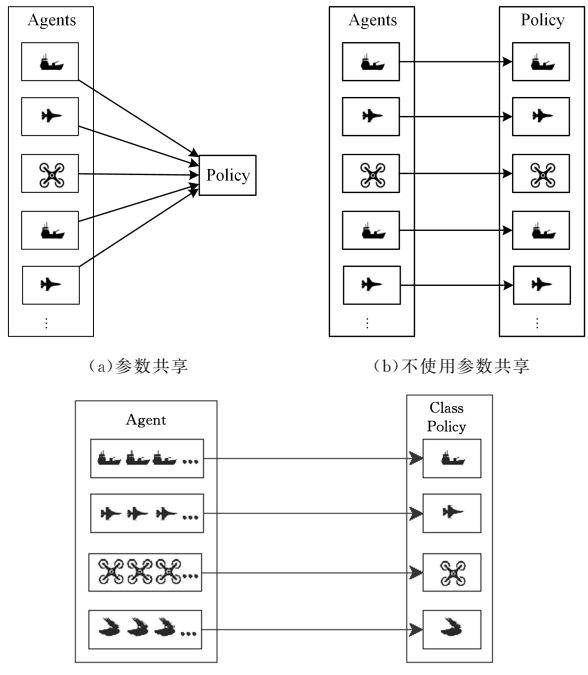
算法 1 HMACL 框架

1. 源任务生成模块为 STG
2. 种类策略提升模块为 CPI
3. 训练模块为 Trainer
4. 智能体数量为 N,智能体集群为 $\mathbf{a}\leftarrow a^i \mid i\in(1,\cdots,N)$
5. 智能体种类数量为 C,种类策略为 $\boldsymbol{\pi}\leftarrow\pi^j \mid j\in(1,\cdots,C)$
6. 智能体到种类策略的映射为 $\pi^j\leftarrow ap2cp(a^i)$
7. 当前源任务 e_{src},$\boldsymbol{\pi}$ 在 e_{src} 上的 TD 误差 L_{src}^{π}
8. 目标任务 e_{tag},$\boldsymbol{\pi}$ 在 e_{tag} 上的性能度量 M_{tag}^{π}
9. 初始化 episodei←0,最大 episode 为 E,每 n 个 episode 切换源任务 e_{src}
10. while i<E do
11. $\boldsymbol{\pi}', L_{src}^{\pi'}\leftarrow$Trainer($\boldsymbol{\pi}, e_{src}, \mathbf{a}$, ap2cp, n) / * Trainer 训练当前策略 * /
12. $\pi, M_{tag}^{\pi'}\leftarrow$CPI($\boldsymbol{\pi}, \boldsymbol{\pi}', e_{tag}, \mathbf{a}$, ap2cp) / * CPI 评估 $\boldsymbol{\pi}'$ 并更新 $\boldsymbol{\pi}$ * /
13. $e_{src}\leftarrow$STG($e_{src}, L_{src}^{\pi'}, M_{tag}^{\pi'}$) / * STG 更新源任务 e_{src} * /
14. i←i+n
15. end while
16. $\boldsymbol{\pi}_{tag}\leftarrow$Trainer($\boldsymbol{\pi}, e_{tag}, \mathbf{a}$, ap2cp, n) / * 在目标任务 e_{tag} 上训练得到最终的策略 $\boldsymbol{\pi}_{tag}$ * /

3.2.1 种类策略提升(CPI)

CPI 提出了一种异质多智能体环境中的参数共享策略,解决了常用参数共享策略在异质智能体环境中由于智能体之间观测和动作空间不一致导致的实践上的困难,能够缓解 HMAS 本身的非平稳性。CPI 维护一组种类策略 π_θ,当 Trainer 完成一个源任务 e_{src} 的训练,会将训练完成的策略 $\boldsymbol{\pi}_\theta'$ 传给 CPI,CPI 评估 $\boldsymbol{\pi}_\theta$ 和 $\boldsymbol{\pi}_\theta'$ 在目标任务 e_{tag} 上的性能,将 $\boldsymbol{\pi}_\theta'$ 的参数更新到 $\boldsymbol{\pi}_\theta$。

由于多智能体系统本身具有非平稳性,当前基于合作的 MARL 算法普遍采用参数共享的方式,并取得了很大的性能提升[16-18]。并且,参数共享中多个智能体使用同一个策略或模型,使得训练大规模的多智能体集群的时间和空间效率大大提高,如图 4(a)和图 4(b)所示。上述工作的参数共享主要是在同质化的智能体之间实现的,当把同样的方法应用到非同质化的智能体中时,不仅要解决智能体之间可能存在的动作和状态空间不一致带来的实践上的问题,而且还会受到异质智能体之间由于性质不同带来的训练中的扰动和智能体意图偏差的影响。这些问题通常会使得非同质化智能体之间参数共享的效果不好,甚至比非参数共享时更差[27]。



(a)参数共享 (b)不使用参数共享

(c)基于种类策略的参数共享

图 4 参数共享策略对比

Fig. 4 Comparison between parameter sharing strategies

CPI 使用了基于智能体种类的参数共享方法,为每一类智能体训练单独的策略,即同类的智能体共享一个策略,而不是所有的智能体共享一个策略,或者每一个智能体独享一个独立的策略;同时建立了从智能体到种类策略的映射

ap2cp，如图 4(c)所示。当智能体要决策时，即可通过 ap2cp 定位到自身的种类策略。

种类策略的目的是提升在目标任务中的性能而不是源任务。因此虽然算法模块基于源任务训练最新策略，但 CPI 使用目标任务来评估并更新策略参数。模块提供两个参数更新方式，在使用硬更新(hard update)时，会对比两种策略在目标任务上的性能，如果新的策略比当前策略更好，则将当前策略替换为新策略；在使用软更新(soft update)时，会使用一部分新的策略参数去更新当前策略。

$$\theta \leftarrow (1-\tau)\theta + \tau\theta' \tag{1}$$

其中，τ 为平衡新策略和旧策略的超参数。CPI 的伪代码如算法 2 所示。

算法 2 ClassPolicyImproving(CPI)

输入：$(\boldsymbol{\pi}_\theta, \boldsymbol{\pi}_\theta{}', e_{tag}, \mathbf{a}, ap2cp)$

输出：(π_θ, M_{tag})

1. $M_{tag} \leftarrow Eval(\boldsymbol{\pi}_\theta, e_{tag}, \mathbf{a}, ap2cp)$
2. $M' \leftarrow Eval(\mathbf{a}_\theta{}', e_{tag}, \mathbf{a}, ap2cp)$
3. if 采用软更新 then
4. $\quad \theta \leftarrow (1-\tau)\theta + \tau\theta'$
5. $\quad M_{tag} \leftarrow Eval(\boldsymbol{\pi}_\theta, e_{tag}, \mathbf{a}, ap2cp)$
6. else if 采用硬更新，且 $M' > M_{tag}$ then
7. $\quad \theta \leftarrow \theta'$
8. $\quad M_{tag} \leftarrow M'$
9. end if

对于一个具有 N 个智能体的 HMAS，N 个智能体分为 C 类，在 CPI 中需要 C 个策略而不是 N 个。考虑到实际情况中 $C \leqslant N$，因此种类策略不仅有参数共享的优点，即缓解了环境的非平稳性，提高了时空效率，同时还避免了 HMAS 中参数共享在实践中存在的状态空间和动作空间不一致的问题。

3.2.2 源任务生成(STG)

STG 可以为 Trainer 生成适当的源任务 e_{src} 序列，通过不断地训练策略在源任务上的能力，使得策略可以平滑地迁移到目标任务 e_{tag} 上。

在部份可观测问题中，智能体观测空间有限，当环境规模较小时能够进行有效的探索，但是随着环境增大，探索成本也迅速增大。因此引导智能体进行高效的探索，而不是盲目地检索所有的状态空间十分重要。为了引导智能体逐步地学习，最终成功扩展到大规模环境中，STG 给智能体生成了难度逐渐增加的源任务。源任务生成的关键问题是如何控制生成任务的难度。如果给予智能体太简单的任务，智能体会在简单任务上迅速地过拟合；如果任务太难，智能体长期得不到有意义的奖励信号，会导致学习过程崩溃。

STG 让智能体先适应在较小空间中的探索，然后给予智能体规模逐渐增大的环境。与此同时，为了保证策略在难度不断增加的源任务上的质量，STG 会评估已学习任务的难度来回放已经学习过的智能体策略表现较差的任务。通过这样的回放机制，来完成源任务生成从难到易的平滑过渡，防止策略在其中的某一个任务上训练不足，导致训练过程崩溃，如图 5 和算法 3 所示。

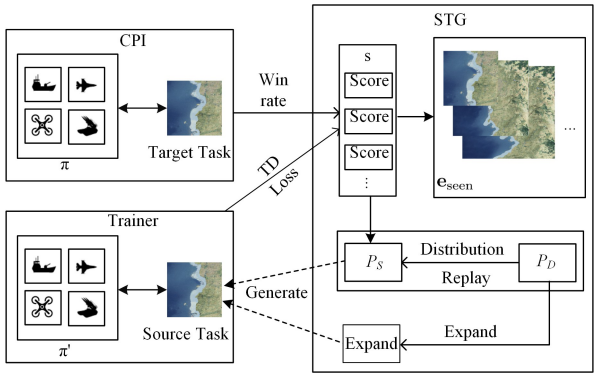


图 5 源任务生成

Fig. 5 STG

算法 3 SourceTaskGenerating(STG)

输入：$(e_{src}, L_{src}^{\pi'}, M_{tag}^{\pi'})$

输出：e'_{src}

1. e_{train} 为所有任务，e_{seen} 为已经生成的任务
2. $S \leftarrow \emptyset$，对于 $\{s^e \in S | e \in e_{train}\}$ 为任务 e 的分数，S 包括 M_{tag} 和 L_{src} 两部分
3. 根据 S 初始化概率分布 P_S，环境扩展度 d
4. 根据 $|e_{train}|$ 和 $|e_{seen}|$ 初始化概率分布 P_D
5. if $s^e{}_{src} \notin S$ then
6. $\quad$ 将 $s^e{}_{src}$ 加入 S，e_{src} 加入 e_{seen}
7. $\quad |S| \leftarrow |S|+1, |e_{seen}| \leftarrow |e_{seen}|+1$
8. $\quad P_D \leftarrow P(\cdot \parallel e_{train}|, |e_{seen}|) /*$ 更新分布 P_D $*/$
9. end if
10. $s^e{}_{src} \leftarrow Score(L_{src}^{\pi'}, M_{tag}^{\pi'}) /*$ 更新任务 e_{src} 的分数 $*/$
11. 更新分布 $P_S \leftarrow P(\cdot | S)$
12. $D \sim P_D /*$ 从 P_D 中采样 $*/$
13. if $D==0$ then $/*$ 生成新的任务 $*/$
14. $\quad e'_{src} \leftarrow Expand(e_{src}, d)$
15. else $/*$ 从旧的任务 e_{seen} 中采样 $*/$
16. $\quad e'_{src} \sim P_S$
17. end if

因此 STG 需要解决的问题是：1)怎样决定生成新任务的难度；2)怎样选择生成新任务(扩大环境)或回放旧任务；3)回放旧任务时，选择哪个旧任务。

首先是生成新任务。对于一个大规模的目标任务 e_{tag}，需要同时初始化一个小规模的初始源任务 e^0_{src} 和一个任务的扩展度 d。在需要生成源任务时，M2ALE 通过扩展度 d 在最大规模的源任务的基础上继续扩展：$e^0_{src} = Expand(e^1_{src}, d)$。对于其他的环境，需要预先指定所有的源任务和目标任务，或提供一个类似的生成源任务的函数。

STG 通过所有的任务和已生成的任务数量建立伯努利分布(Bernoulli Distribution)来选择生成新任务还是回放旧任务。所有的任务集合为 $e_{train} = e_{src} \cup e_{tag}$，已生成任务集合为 e_{seen}。STG 建立分布 $P_D = P(\cdot \parallel e_{seen} \parallel e_{train}|)$，在 $|e_{seen}|$ 较小时，应该更多地生成新任务；随着 $|e_{seen}|$ 逐渐增大，STG 需要考虑对已生成任务的充分利用，因此以概率 $P_D(d=0)$ 生成新的任务，以概率 $P_D(d=1)$ 回放旧的任务。

当选择旧的任务进行回放时，应该选择具有较大学习潜力的任务，STG 会对所有已生成任务的学习潜力进行评分，表示为集合 $\boldsymbol{S}$。评分越高代表具有越大的学习潜力，应该有

更大的概率被回放。因此 STG 利用 $\boldsymbol{S}$ 建立分布 $P_S=P(\cdot|\boldsymbol{S})$，从分布中采样 $e^i=P_{\boldsymbol{S}}$ 即可。

STG 的评分由两部分构成。第一部分是最新策略在目标任务 e_{tag} 上的性能 M_{tag}（如胜率或回报）。我们的目标是提升智能体集群在 e_{tag} 上的性能，当策略在 e_{seen} 上训练完之后，STG 会从 CPI 接收当前策略在 e_{tag} 上的 M_{tag}，使用集合 M 记录。M_{tag} 越高，说明当前任务对策略在 e_{tag} 上的性能帮助越大，因此给予越高的概率回放当前任务。当前策略对 e_{tag} 进行 K 个 episode 的评估，平均胜率可以表示为：

$$w=\frac{1}{K}\sum_{i=0}^{K}w^i \tag{2}$$

评分的第二部分是最新策略在当前源任务上的 TD 误差，给予 TD 误差大的任务更高的回放概率，有助于策略稳定地提升到更高难度的任务中。对于一个轨迹 τ，第 t 步的 TD 误差为：

$$\delta_i=r_t+\gamma V(s_{t+1})-V(s_t) \tag{3}$$

因此对于 K 个长度为 T 的轨迹，总的 TD 误差为：

$$\delta=\frac{1}{K}\frac{1}{T}|\delta_i| \tag{4}$$

对于 w 和 δ，分别使用优先级函数 $h(S_i)=\text{rank}(S_i)$ 和温度参数 β 的归一化输出：

$$P_X(e^i|X)=\frac{h(X^i)^{\frac{1}{\beta}}}{\sum_i h(X^i)^{\frac{1}{\beta}}} \tag{5}$$

其中，温度参数 β 是超参数，用于控制分布 P_X 对排序靠后的 M_{tag} 或 TD 误差的关注程度。β 越大，则分布越平滑，对 e_{seen} 整体的关注度更高；β 越小，则更多地关注排序靠前的任务。通过式(5)，可以分别基于 M_{tag} 构建分布 $P_M(e^i|M_{\text{tag}})$ 和基于 TD 误差构建 $P_\delta(e^i|\delta)$。综合两个指标得到最终的分布：

$$P_{\boldsymbol{S}}=\sigma P_\delta(e^i|\delta)+(1-\sigma)P_M(e^i|M_{\text{tag}}) \tag{6}$$

通过调整超参数 σ 可以平衡对 M_{tag} 和 TD 误差 δ 的关注度。

3.2.3 训练模块(Trainer)

Trainer 从 STG 获取任务，从 CPI 获取最新的策略。Trainer 需要做的就是根据当前 STG 提供的源任务 e_{src} 在 CPI 维护的最新策略 $\boldsymbol{\pi}_\theta$ 的基础上继续训练，训练完成后交给 CPI 评估并更新 $\boldsymbol{\pi}_\theta$。HMACL 允许 Trainer 采用任意的 MARL 来训练。Trainer 需要依赖 CPI 中维护的从智能体到种类策略的映射函数 ap2cp 来找到每一个智能体对应的种类策略。

单纯的 MARL 算法会在训练开始之前设置总的时间步 num_timestep 或者总的 episode 数量 num_episode，并且在整个训练过程中环境保持不变。而在 HAMCL 中，整个训练过程被切分成了若干段，每一段训练固定的时间步或 episode，完成之后将得到的策略交给 CPI 评估和更新，将 TDloss 发送给 STG。然后从 CPI 得到更新后的策略，从 STG 得到下一个源任务，重复上述过程。

4 实验

4.1 实验设置

实验以基于值的多智能体算法 VDN 为基本算法，基本环境为 M2ALE。进攻方智能体采用 HMACL 结合 VDN 算法训练，防守方为硬编码策略。目的是训练进攻方智能体集群，优化目标(M)为进攻方的胜率(win_rate)。每一次对局(episode)，我们将步长(num_step)限制为 200 步，即如果 200 步以内双方不能分出胜负(消灭对方全体单位)，则算进攻方失败。

实验场景为 5u_vs_5u，代表进攻方 5 个进攻单位(包含舰艇、战斗机、侦查无人机)对抗防守方 5 个进攻单位(包含移动炮台、战斗机、侦查无人机)，50×50 为目标任务的战场大小。

除了上述重要的设置以外，本文还设置了一些对实验结果影响较大的参数，主要有两部分：1) HMACL 超参数，如表 2 所列；2) MARL 算法超参数，如表 3 所列。

表 2 HMACL 重要超参数

Table 2 Important hyperparameters of HMACL

超参数	描述	值
t_max	总时间步	4 000 000
t_update	cpi 更新步数	20 000
σ	loss_coef	0.1
β	temperature	0.1
improve_type	cpi 更新方式	hard
M	评估性能度量	win_rate

表 3 MARL 算法重要超参数

Table 3 Important hyperparameters of MARL algorithm

超参数	描述	值
algorithm	算法	vdn
hidden_dim	隐藏层大小	128
lr	学习率	0.0005
epsilon_anneal_time	epsilon 退火步数	100 000
optimizer	优化器	Adam
buffer_size	经验回放池大小	5000

初始的源任务大小为 8×8，$expand\_degree=2$。作战单位的初始化策略为 random，奖励类型为 dense。实验数据为在 5 个随机种子下的平均值，策略评估指标为 200 个 episode 的平均值。

4.2 HMACL

为了展示 HMACL 的有效性，证明 HMACL 可以高效地将异质多智能体集群迁移到大规模稀疏的目标任务上，实验将使用 HMACL 训练的结果与不使用 HMACL 的结果进行对比，如图 6 所示。

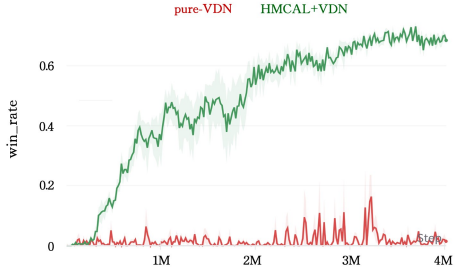


图 6 HMACL+VDN 与 VDN 的训练过程胜率变化

Fig. 6 Win rate changes during training process of HMACL+VDN and pure VDN

使用 HMACL 结合 VDN 算法可以使策略在目标任务 e_{tag} 上的胜率在结束训练时达到 68.5%，而单纯使用 VDN

算法则始终无法学习到可靠的策略，最终胜率只有 1.5%。因此，相比直接使用 VDN 算法，使用 HMACL 结合 VDN 极大地提升了最终性能和采样率。

4.3 消融实验

为了验证 CPI 和 STG 在 HMACL 框架中的有效性，分别对两个模块进行消融实验。

4.3.1 CPI

CPI 实现了一种基于智能体种类的参数共享策略，并维护和更新种类策略。本文对比了 3.2.1 节中 3 种参数共享策略对胜率变化的影响，如图 7 所示。

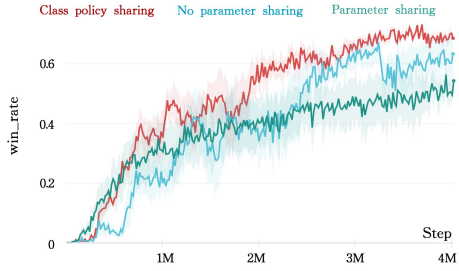


图 7 3 种参数共享策略对胜率的影响

Fig. 7 Impact of three parameter sharing strategies on win rate

从实验结果中可以发现，使用种类参数共享不仅在最终性能上超过了其他两种方法，且稳定性更高。

4.3.2 STG

STG 生成一系列难度逐渐提升的课程，帮助策略平滑地迁移到目标任务上，而单纯的 VDN 算法则直接在目标任务上训练。对比使用 STG 和不使用 STG 这两种方式的结果如图 8 所示。

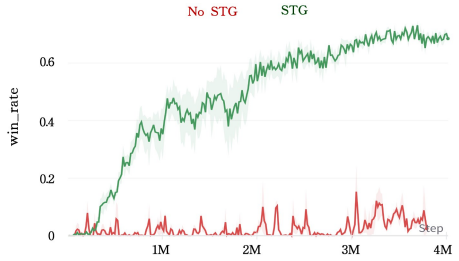


图 8 使用 STG 生成的课程训练和直接在目标任务上训练

Fig. 8 Training with and without curriculum that STG generated

可以发现，HMACL 框架去掉了 STG 模块后，训练效果对比单纯使用 VDN 算法几乎没有提升。因此，STG 模块的源任务生成是 HMACL 的核心。

结束语 本文通过分析当前多智能体强化学习在战场模拟方面的研究，发现普遍存在两点简化设定：1)智能体同质化；2)作战单位密度大。为了增加环境的实用性，本文提出了 M2ALE 环境，实现了多种功能各异的异质作战单位。M2ALE 可以通过地图的尺度变换来改变作战单位的密度，增加智能体探索动作的多样性和环境的复杂性。同时，为了应对智能体在 M2ALE 环境及类似任务中面临的非平稳性和探索难问题，本文提出了 HMACL 课程学习框架，包含 3 个模块：种类策略提升、源任务生成、算法模块。通过种类策略参数共享实现了异质多智能体系统中的参数共享，缓解了多智能体环境中的非平稳问题；源任务生成模块通过生成难度逐步增加的任务，为智能体提供了由易到难的课程，引导智能体机群扩展到大规模稀疏环境中；算法模块负责持续提升种类策略在当前源任务上的能力。实验数据表明，HMACL 在 M2ALE 这样一个具有多种异质智能体且作战单位分布稀疏的场景中，能够大幅度提高多智能体算法的采样率和性能，将多智能体强化学习算法扩展到超大规模的环境中。

参 考 文 献

[1] MORDATCH I, ABBEEL P. Emergence of grounded compositional language in multi-agent populations[C]// Proceedings of the AAAI Conference on Artificial Intelligence. 2018.

[2] SAMVELYAN M, RASHID T, DE WITT C S, et al. The starcraft multi-agent challenge[J]. arXiv: 1902.04043, 2019.

[3] TERRY J, BLACK B, GRAMMEL N, et al. Pettingzoo: Gym for multi-agent reinforcement learning[J]. Advances in Neural Information Processing Systems, 2021, 34: 15032-15043.

[4] WANG B H, WU T Y, LI W H, et al. Large-scale UAVs Confrontation Based on Multi-agent Reinforcement Learning[J]. Journal of System Simulation, 2021, 33(8): 1739-1753.

[5] ZHENG L, YANG J, CAI H, et al. Magent: A many-agent reinforcement learning platform for artificial collective intelligence [C]// Proceedings of the AAAI Conference on Artificial Intelligence. 2018.

[6] LI Y. Deep reinforcement learning: An overview [J]. arXiv: 1701.07274, 2017.

[7] WATKINS C J C H. Learning from delayed rewards[J/OL]. https://d1wqtxts1xzle7.cloudfront.net/50360235/Learning_from_delayed_rewards_20161116-28282-v2pwvq-libre.pdf?1479337768=&response-content-disposition=inline%3B+fi-lename%3DLearning_from_delayed_rewards.pdf&Expires=1697437463&Signature=DTEgpQ1CwNQSh73wVPYhujim-RY-brTt06a6MNFrAhzcQnOQ8jPb5K8AbuSd4o5HwMabnNv0N7-weYKFszXWSgDgnHC73-jwDsGIT3KhsE9wbR8H1PUyqXlR-lkr~kapd2K5NF~yj92hGkbtHxVT5YCm4t8bC3LFSMZvrd-D0i5z1AIgd97DF94bUdJ-YoR9-Ag6eaADJWZmow6WKki8oKhAvyGoOY9~pJi94w4dKLww-IqnrGNhiSCCITANWMVeH7rc5x-1MsDfd1iP31vWrdlDpF71nn1uh28tm35rr03HmESv4Tbnt-RxG410d4E7QeUe31ItR8Htrq5CWiIITEnulLBcg__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA.

[8] SUNEHAG P, LEVER G, GRUSLYS A, et al. Value-decomposition networks for cooperative multi-agent learning[J]. arXiv: 1706.05296, 2017.

[9] RASHID T, SAMVELYAN M, DE WITT C S, et al. Monotonic value function factorisation for deep multi-agent reinforcement learning[J]. The Journal of Machine Learning Research, 2020, 21(1): 7234-7284.

[10] TERRY J K, GRAMMEL N, HARI A, et al. Revisiting parameter sharing in multi-agent deep reinforcement learning[J]. arXiv: 2005.13625, 2020.

[11] GUPTA J K, EGOROV M, KOCHENDERFER M. Cooperative multi-agent control using deep reinforcement learning[C]// Autonomous Agents and Multiagent Systems: AAMAS 2017 Workshops, Best Papers, São Paulo, Brazil, May 8-12, 2017, Re-

vised Selected Papers 16. Springer International Publishing, 2017:66-83.

[12] CHRISTIANOS F, PAPOUDAKIS G, RAHMAN M A, et al. Scaling multi-agent reinforcement learning with selective parameter sharing[C]// International Conference on Machine Learning. PMLR, 2021:1989-1998.

[13] DORRI A, KANHERE S S, JURDAK R. Multi-agent systems: A survey[J]. IEEEAccess, 2018, 6:28573-28593.

[14] ZHENG Y, ZHU Y, WANG L. Consensus of heterogeneous multi-agent systems[J]. IET Control Theory & Applications, 2011, 5(16):1881-1888.

[15] PORTELAS R, COLAS C, WENG L, et al. Automatic curriculum learning for deep rl: A short survey[J]. arXiv:2003.04664, 2020.

[16] LIU I J, JAIN U, YEH R A, et al. Cooperative exploration for multi-agent deep reinforcement learning[C]// International Conference on Machine Learning. PMLR, 2021:6826-6836.

[17] DENNIS M, JAQUES N, VINITSKY E, et al. Emergent complexity and zero-shot transfer via unsupervised environment design[J]. Advances in Neural Information Processing Systems, 2020, 33:13049-13061.

[18] YU W W, YANG X Y, LI H C, et al. Attentional Intention and Communication for Multi-Agent Learning[J]. Acta Automatica Sinica, 2021, 47:1-16.

[19] ZANG R, WANG L, SHI T F. Multiagent reinforcement learning based on attentional message sharing[J]. Journal of Computer Applications, 2022, 42(11):3346-3353.

[20] ZHAO Y P, FAN Z J. Research into The Evaluation Method of Naval Warfare Based on Simulation Deduction[J]. Shipboard Electronic Countermeasure, 2019, 42(3):1-4.

[21] XIAO Z, ZHANG S Y. Reinforcement Learning Model Based on Regret for Multi-Agent Conflict Games[J]. Journal of Software, 2008, 19(11):2957-2967.

[22] DU H W, CUI M L, HAN T, et al. Maneuvering decision in air combat based on multi-objective optimization and reinforcement learning[J]. Journal of Beijing University of Aeronautics and Astronautics, 2018, 44(11):2247-2256.

[24] MNIH V, KAVUKCUOGLU K, SILVER D, et al. Human-level control through deep reinforcement learning[J]. Nature, 2015, 518(7540):529-533.

[25] TAN M. Multi-agent reinforcement learning: Independent vs. cooperative agents[C]// Proceedings of the Tenth International Conference on Machine Learning. 1993:330-337.

[26] MNIH V, BADIA A P, MIRZA M, et al. Asynchronous methods for deep reinforcement learning[C]// International Conference on Machine Learning. PMLR, 2016:1928-1937.

[27] SCHULMAN J, WOLSKI F, DHARIWAL P, et al. Proximal policy optimization algorithms[J]. arXiv:1707.06347, 2017.

[28] LOWE R, WU Y I, TAMAR A, et al. Multi-agent actor-critic for mixed cooperative-competitive environments[C]// Advances in Neural Information Processing Systems. 2017.

[29] YU C, VELU A, VINITSKY E, et al. The surprising effectiveness of ppo in cooperative multi-agent games[J]. Advances in Neural Information Processing Systems, 2022, 35:24611-24624.

[30] FOERSTER J, FARQUHAR G, AFOURAS T, et al. Counterfactual multi-agent policy gradients[C]// Proceedings of the AAAI Conference on Artificial Intelligence. 2018.

[31] SUNEHAG P, LEVER G, GRUSLYS A, et al. Value-decomposition networks for cooperative multi-agent learning[J]. arXiv:1706.05296, 2017.

[32] RASHID T, SAMVELYAN M, DE WITT C S, et al. Monotonic value function factorisation for deep multi-agent reinforcement learning[J]. The Journal of Machine Learning Research, 2020, 21(1):7234-7284.

[33] CHRISTIANOS F, PAPOUDAKIS G, RAHMAN M A, et al. Scaling multi-agent reinforcement learning with selective parameter sharing[C]// International Conference on Machine Learning. PMLR, 2021:1989-1998.

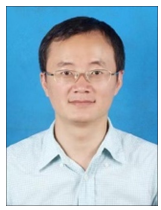
[34] NARVEKAR S, SINAPOV J, LEONETTI M, et al. Source task creation for curriculum learning[C]// Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems. 2016:566-574.

[35] DENNIS M, JAQUES N, VINITSKY E, et al. Emergent complexity and zero-shot transfer via unsupervised environment design[J]. Advances in Neural Information Processing Systems, 2020, 33:13049-13061.

[36] SCHAUL T, QUAN J, ANTONOGLOU I, et al. Prioritized experience replay[J]. arXiv:1511.05952, 2015.

[37] ANDRYCHOWICZ M, WOLSKI F, RAY A, et al. Hindsight experience replay[C]// Advances in Neural Information Processing Systems. 2017.

LUO Ruiqing, born in 1995, postgraduate. His main research interests include machine learning and reinforcement learning.



ZENG Kun, born in 1982, Ph.D, associate professor. His main research interests include computer vision, machine learning, and graphics.

(责任编辑:何杨)