文章编号:1671-6833(2025)03-0026-08

基于多智能体强化学习的 AMR 协作任务分配方法

张富强[1,2], 张焱锐[1,2], 丁 凯[1,2], 常丰田[1,2]

(1.长安大学 道路施工技术与装备教育部重点实验室,陕西 西安 710064;2.长安大学 智能制造系统研究所,陕西 西安 710064)

摘 要:为了解决 AMR 在柔性生产中运输任务的自主分配难题,采用一种基于改进多智能体强化学习算法的多智能体深度确定性策略梯度算法(MADDPG)。首先,引入注意力机制对算法进行改进,采用中心化训练分散式执行的框架,并对 AMR 的动作及状态进行设置;其次,根据奖励值的大小确定任务节点的覆盖程度以及任务的完成效果;最后,在 Pycharm 上进行仿真,结果表明:MADDPG 算法的平均奖励值较其他算法增幅为 3,训练次数减少了 300 次,在保证求解任务分配完成度的基础上,具有更快的学习速度和更稳定的收敛过程。

关键词:自主移动机器人; 多智能体; 强化学习; 协作; 任务分配

中图分类号:TP13 **文献标志码**:A **doi**:10.13705/j.issn.1671-6833.2025.03.001

基于制造执行系统(manufacturing execution system,MES)的集中计划运行模式可实现上层生产信息与下层车间之间的信息传递和资源调配,但也存在明显弊端[1]。运输资源之间缺乏必要的协作机制导致生产异常频发,难以实现实时响应和动态调度。为解决这一问题,自主移动机器人(autonomous mobile robot,AMR)作为一种具备环境感知、路径规划和避障功能的移动机器人,可以实现高效的自主协作与任务分配[2-4]。

针对传统配送模式单一固定的问题,可借鉴滴滴打车和美团外卖等软件的“抢单”配送模式,提高生产任务的灵活性以满足现代制造环境的需求[5]。“抢单”模式使配送方式从被动接受任务转变为主动参与,提高了运输效率。因此,本文主要研究 AMR 自主任务分配算法,多个 AMR 之间通过合作或竞争获取资源,快速响应任务变化并做出最佳分配决策。

关于运输任务分配,典型的研究有:刘广瑞等[6]建立了执行多目标无人机协同的多任务分配模型;Wu 等[7]提出了一种用于无人系统群体的同时进行任务分配和轨迹规划的方法,解决了任务分配和轨迹规划子问题;王俊英等[8]提出了一种任务分配概率自适应的蚁群算法,提升了算法的执行效率;吴蔚楠等[9]在多无人机执行任务的背景下,进行了任务分配的建模、算法设计和分析;鞠锴等[10]基于势博弈理论设计任务分配算法,在降低复杂度的同时,得到了近似最大化的期望全局效用的分配方案;施伟等[11]基于深度强化学习的多机协同空战决策流程的框架,针对 PPO(proximal policy optimization)算法设计了 4 种算法增强机制,提高了智能体之间的任务分配协同程度;Motes 等[12]提出了一种多机器人任务和运动集成方法,能够处理多个可分解任务中的顺序子任务依赖关系。

此外,强化学习模型的发展也为解决机器学习中各种顺序决策问题提供了新的思路。Yin 等[13]提出了一种深度强化学习中考虑注意力的分散式多任务分配框架用于分配应用。Li 等[14]提出了一种基于强化学习的并发任务和安全应急任务分配方法,解决了并发任务分配问题。Oroojlooy 等[15]把 MARL(multi-agent reinforcement learning)方法分成了 5 个类别(独立学习者、完全可观察的批评、价值函数分解、共识和学习沟通)用于解决多智能体合

收稿日期:2024-11-20;**修订日期**:2024-12-21

基金项目:国家重点研发计划项目(2021YFB3301702);陕西省科技重大专项(2018zdzx01-01-01)

作者简介:张富强(1984—),男,山西运城人,长安大学副教授,博士,主要从事数字化与智能物流研究,E-mail:fqzhang@chd.edu.cn。

引用本文:张富强,张焱锐,丁凯,等. 基于多智能体强化学习的 AMR 协作任务分配方法[J]. 郑州大学学报(工学版),2025,46(3):26-33. (ZHANG F Q, ZHANG Y R, DING K, et al. AMRs autonomous collaboration task assignment method based on multi-agent reinforcement learning[J]. Journal of Zhengzhou University (Engineering Science),2025,46(3):26-33.)

作强化学习问题。Wang 等[16]基于适应度提出了一种分层强化学习方法,用于复杂地形条件下的多人机任务分配。Xiao 等[17]提出了一种改进的粒子群优化算法,解决了 AGV 任务的多模式资源受限调度问题。

综上所述,强化学习方法因其自适应性强、可以自我调节、适用于复杂环境、能够探索未知领域和实时决策能力强等特点,在柔性生产协同决策方面具有重要的应用价值。本文基于多智能体强化学习算法,提出了一种改进多智能体强化学习算法-多智能体深度确定性策略梯度算法(multi-agent deep deterministic policy gradient,MADDPG)。

1　AMR 任务分配问题的描述

柔性车间中 AMR 任务分配问题可描述如下:有 n 个工件需要在 m 台机器上进行加工,运输由 N 台 AMR 完成。每个工件都需要有 s 道工序,$s \geqslant 1$。每道工序都有一个对应的加工时间和加工机器集合。

根据以上描述进行假设:①在 0 时刻,所有机器处于空闲可用状态,所有 AMR 和工件材料都在装卸货点准备就绪;②不考虑机器和 AMR 故障等因素,AMR 在各个工位的装载/卸载时间均已知,机器间运输时间固定;③每台加工机器旁边均设有一定容量的缓存区,用于存放待加工工件;④每个 AMR 能够实时感知和更新自身状态,还可以获取其他 AMR 状态;⑤AGV 获得的任务下发顺序即为工件加工顺序,随着加工的不断进行,任务在不间断下发;⑥每个 AGV 的物料装载时间和卸载时间相同。

2　基于 MADDPG 算法的 AMR 任务分配

2.1　算法框架

在多 AMR 协作环境当中,每一个 AMR 都是独立的 Agent,算法需要不断学习来达到改进策略的目的[18];基于单个 AMR 视角,环境从静态转变为动态。这与传统强化学习收敛的条件大不相同,在一定程度上导致无法仅仅通过改变单个 Agent 自身的策略来适应不稳定的环境,并且在实际应用中,传统策略梯度算法中的误差会随着 Agent 数量的增加而增大。而 MADDPG 算法基于 Actor-Critic 思想[19]和 DDPG(deep deterministic policy gradient)[20]进行了一系列的改进,Agent 通过学习每个状态下的最优策略来优化自己的策略网络(Actor 网络)和价值网络(Critic 网络),并考虑其他 Agent 的策略和行动来提高效率。

MADDPG 算法采用了中心式训练和分散式执行(centralized training and decentralized execution,CTDE)[21]的框架,如图 1 所示。在训练阶段,每个 AMR 将得到的环境信息输入到自己的策略网络中,然后策略网络输出动作信息,并将这些动作应用于环境。环境的响应导致状态的变化和奖励值的产生,每个 AMR 不断与环境交互,不断获得新环境的状态和奖励值,最后将这些信息存储到各自的经验池中。

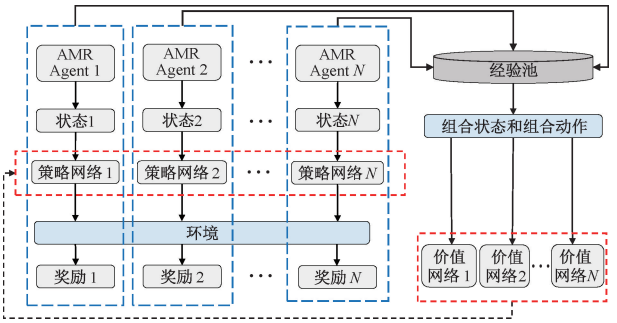


图 1　MADDPG 算法框架

Figure 1　MADDPG algorithm framework

2.1.1　中心式训练分散式执行框架分析

中心训练分散执行具体实现过程:每个 AMR 根据自身策略得到当前状态执行的动作,并与环境交互得到经验存入自身的经验缓存池。待所有 AMR 与环境交互后,每个 AMR 从经验池中随机抽取经验训练各自的神经网络。如图 2 所示,在训练过程中,根据现有的本地局部观察信息 o_i,Actor 网络会采取一个动作 a_i,将其作为 Critic 网络的输入,得到一个 Q 值,并反馈给 Actor 网络,指挥 Actor 网络的策略更新,因此,在训练过程中 Actor 网络与 Critic 网络都参与;而测试过程仅由 Actor 网络根据局部信息生成动作,无须 Critic 的反馈。

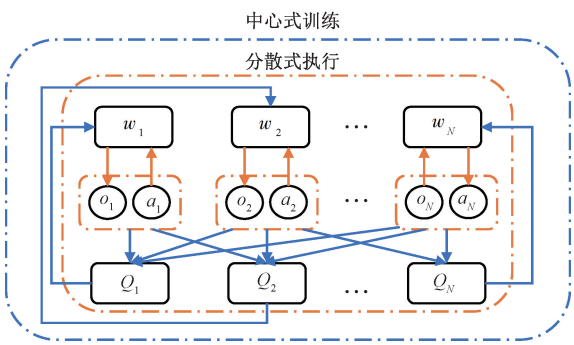


图 2　中心式训练分散式执行

Figure 2　Centralized training and decentralized execution

集中训练:采用联合状态价值动作值函数对策略进行评估,同时加入其他额外信息,能够更优地评估联合策略。该模式改善了算法的收敛效果且有助于解决多智能体强化学习中的非平稳性问题。在训练阶段,AMR 的 Actor 首先基于当前状态选择一个动

作,然后 Critic 计算基于状态-动作对应的 Q 值,反馈给 Actor,促使其优化动作策略。而在测试阶段,仅需要 Actor 来执行任务,无须 Critic 的反馈。

分散执行:完成强化学习的交互训练后,AMR 能够根据局部观测在不同状态下做出最优决策。这种方法考虑到了实际场景中的观测限制,每个 Actor 都可以根据自身状态采取合适的动作,不需要其他 AMR 的状态或动作。

综上,集中训练和分散执行是考虑了强化学习算法训练和实际场景限制的最佳训练方式。在训练过程中充分考虑了环境和 AMR 的状态,在执行阶段只需 AMR 的局部观测即可做出决策,更符合实际场景。因此,采用集中训练分散执行的框架可确保多 AMR 协作环境的稳定性。

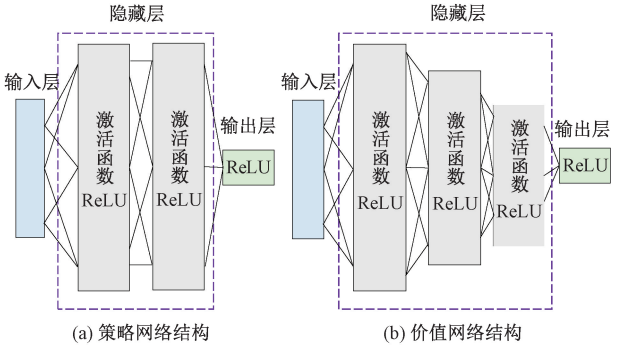
2.1.2 基于注意力机制的 MADDPG 算法

在神经网络的训练过程中,通过从 AMR 的经验池中随机抽取相同时刻的数据,构建一组经验样本(o,a,o',r)。其中 o 为所有 AMR 的状态信息集合;a 为 AMR 的动作集合;o'为 AMR 在做出动作后的环境状态信息集合;r 为 AMR 的奖励值。最后将 o'输入到 A_i(第 i 个 AMR)的目标策略网络中得到动作 a',随后将所有 AMR 的状态信息和动作信息组成联合状态和联合动作,输送到 A_j(第 j 个 AMR)的 Critic 网络中,计算下一时刻的目标 Q 值,计算公式为

$$r_i = \gamma Q'(s_{i+1}, u'(s_{i+1} \mid Q^{u'}) \mid \theta^{Q'})。 \tag{1}$$

式中:Q'为 AMR 的决策结果;$Q^{u'}$为 AMR 目标策略的决策结果;r_i 为奖励值;γ 为相关系数;s_{i+1} 为第 i+1 个 AMR 的状态;u'为目标策略;$\theta^{Q'}$为具有滞后更新的策略参数。

实际的 Q 值是通过一个评价网络得到的,这个评价网络会根据时间差异误差(TD-error)进行对应的更新。算法使用策略梯度来更新策略网络,策略网络与价值网络的具体结构如图 3 所示。



(a) 策略网络结构 (b) 价值网络结构

图 3 网络结构

Figure 3 Network structure

环境中 AMR 用 A 表示,其策略集合为 $w = \{w_1, w_2, \cdots, w_N\}$,参数集合 $\theta = \{\theta_1, \theta_2, \cdots, \theta_N\}$,联合动作 $a = (a_1, a_2, \cdots, a_N)$,联合状态 $o = (o_1, o_2, \cdots, o_N)$。$A_i$ 的策略集合为 w_i,策略参数为 θ_i,联合动作为 a_i,联合状态为 o_i,每一个样本数据组成为$(x, x', a_1, a_2, \cdots, a_N, r_1, r_2, \cdots, r_N)$,$A_i$ 的期望收益梯度为

$$\nabla_{\theta_i} J(\theta_i) = E_{s\sim p^w, a_i \sim w_i}[\nabla_{\theta_i} \ln w_i(a_i \mid o_i) Q_i^w(o,a)]。 \tag{2}$$

式中:p^w 为状态分布;$Q_i^w(o,a)$ 为联合动作值函数,即集中式的状态-动作函数,每一个样本数据不仅包含了自身观测到的状态和所执行的动作,而且包含了其他 AMR 的动作。Q_i^w 针对每一个 AMR 建立了值函数,极大地弥补了传统强化学习算法在多 Agent 领域的不足。因此,每一个 AMR 的 Critic 网络不仅清楚自身 AMR 的变化,也知道其他所有 AMR 的动作策略,即使策略在不断地更新变化,环境还是稳定的。$Q_i^w(o,a)$ 的更新公式为

$$L(\theta_i) = E_{o,a,r,o'}[(Q_i^w(x, a_1, a_2, \cdots, a_M) - \bar{x})^2]。 \tag{3}$$

MADDPG 算法解决了 Agent 之间不可观测性的问题,但随着 Agent 数量的增加,问题的复杂度也随之增大。在多 AMR 环境中,为了应对这一挑战,引入了注意力机制,这使得每个 AMR 能够观察其他 AMR 的信息,并根据信息的重要性将其整合到自身的动作值函数估计中。这种机制允许每个 AMR 专注于某些信息,这些信息对于获取更大的奖励至关重要,而不是无目标地学习其他 AMR 提供的全部信息。在 Critic 模块中引入注意力机制可以实现这一目标。注意力机制的结构如图 4 所示。

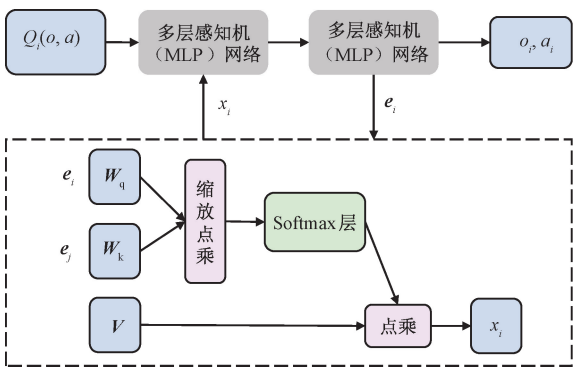


图 4 注意力机制结构

Figure 4 Structure of attention mechanism

加入注意力机制后,算法发生改变,此时需要对 Q 值函数进行改进,A_i 的 Q 值函数 $Q_i(o,a)$ 为

$$Q_i(o,a) = f_i(g_i(o_i, a_i), x_i)。 \tag{4}$$

式中:a_i 表示策略 u_i 与环境不断进行交互后做出的动作。多层感知机(multi-layer perception,MLP)[22-23]

是由多个全连接层构成的神经网络,每个全连接层都由线性变换和非线性变换组成。式(4)中包含一个两层的 MLP 网络 f_i 和一个单层 MLP 嵌入式网络函数 g_i。x_i 为其他 AMR 对 A_i 价值的加权和,计算式为

$$x_i = \sum_{j \neq i} (\alpha_i v_j) = \sum_{j \neq i} (\alpha_i h(\boldsymbol{V} g_j(o_j, a_j)))。\quad (5)$$

式中:v_j 为 A_j 的嵌入式编码函数;$h(\cdot)$ 为 ReLU 激活函数;$\boldsymbol{V}$ 为共享矩阵;α_j 为注意力权重,通过查询值-键值系统比较 $\boldsymbol{e}_j = g_j(o_j, a_j)$ 与 $\boldsymbol{e}_i = g_i(o_i, a_i)$,将 A_j 和其相似值传达给 Softmax 网络,得到:

$$\alpha_j = \exp\left(\frac{\boldsymbol{e}_j^{\mathrm{T}} \boldsymbol{W}_{\mathrm{k}}^{\mathrm{T}} \boldsymbol{W}_{\mathrm{q}} \boldsymbol{e}_i}{\sqrt{D_{\mathrm{k}}}}\right)。\quad (6)$$

式中:$\boldsymbol{e}_i$ 和 $\boldsymbol{e}_j$ 分别表示 A_i 和 A_j 的状态编码;D_{k} 表示维度;$\boldsymbol{W}_{\mathrm{k}}$ 和 $\boldsymbol{W}_{\mathrm{q}}$ 分别代表两个线性映射的矩阵,$\boldsymbol{W}_{\mathrm{k}}$ 将 $\boldsymbol{e}_j$ 转化为键值,$\boldsymbol{W}_{\mathrm{q}}$ 将 $\boldsymbol{e}_i$ 转化为查询值;每个参数($\boldsymbol{W}_{\mathrm{q}}$,$\boldsymbol{W}_{\mathrm{k}}$,$\boldsymbol{V}$)都是独立的。

2.2　动作设置

AMR 的动作空间是一个连续二维空间,每个 AMR 的速度主要包括两个方向:一个是 X 方向上的速度 $\boldsymbol{V}_x$,一个是 Y 方向上的速度 $\boldsymbol{V}_y$,将这两个方向上的速度进行矢量合成,根据 t 时刻的状态输入,输出一个确定速度。$L^t_{\mathrm{AGV}_i,x}$ 表示在 X 方向上 t 时刻所有 AMR 采取的接收策略;$L^t_{\mathrm{AGV}_i,y}$ 表示在 Y 方向上 t 时刻所有 AMR 采取的接收策略;$(L^t_{\mathrm{AGV}_i,x}, L^t_{\mathrm{AGV}_i,y})$ 为联合动作。则 AMR 在 $t + \Delta t$ 时刻的位置动作为 $(L^{t+\Delta t}_{\mathrm{AGV}_i,x}, L^{t+\Delta t}_{\mathrm{AGV}_i,y})$。

$$\begin{cases} L^{t+\Delta t}_{\mathrm{AGV}_i,x} = L^t_{\mathrm{AGV}_i,x} + \boldsymbol{V}_x \cdot \Delta t; \\ L^{t+\Delta t}_{\mathrm{AGV}_i,y} = L^t_{\mathrm{AGV}_i,y} + \boldsymbol{V}_y \cdot \Delta t。 \end{cases} \quad (7)$$

2.3　状态设置

2.3.1　工件任务状态

任务 T_i 在 t 时刻的状态用函数 $StateT_i(t) = (pos_i(t), c_i, call_i(t), Reward_i(t))$ 表示,其中 $pos_i(t)$ 表示任务 T_i 的位置信息,$pos_i(t) = (x_i, y_i)$;$c_{i,1}$ 表示任务节点 i 对能力 1 的需求,$c_i = \{c_{i,1}, c_{i,2}, \cdots, c_{i,n}\}$;$call_i(t) = 1$ 和 $call_i(t) = 0$ 分别表示生产过程中 t 时刻 T_i 对 AMR 有任务需求和无任务需求。

2.3.2　AMR 状态

在 AMR 自主调度过程中,AMR 在时刻 t 的状态用函数 $StateA_j(t) = (pos_j(t), c_j, b_j, \boldsymbol{q}(t))$ 表示。其中,$pos_j(t)$ 为 A_j 在 t 时刻的位置信息,$pos_j(t) = (x_j, y_j)$;$c_j = \{c_{j,1}, c_{j,2}, \cdots, c_{j,n}\}$,$c_{j,1}$ 表示 A_j 在能力 1 上的取值,n 为能力类型的数量;b_j 为 A_j 的速度值;$\boldsymbol{q}(t)$ 为 t 时刻的 A_j 任务分配状态矩阵。$\boldsymbol{q}_{i,j}(t) = 1$ 表示 t 时刻将目标任务 T_i 分配给 A_j;$\boldsymbol{q}_{i,j}(t) = 0$ 表示未将任务 T_i 分配给 A_j,即 A_j 处于空闲状态。通过计算位置坐标,得到 AMR 与任务节点间的距离,见式(8),二者能力匹配值 e_{ij} 的计算公式见式(9),其中,c_{ij} 表示能力匹配系数,所能提供的能力值与任务节点所需能力值之间差别越大,说明采用该 AMR 来完成该任务的效果越好。c_{ij} 等于 2 表示 AMR 能力值大于该任务节点需求值,等于 0.5 表示 AMR 能力值小于任务节点需求值,见式(10)。

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2};\quad (8)$$

$$e_{ij} = \left(\frac{c_{i,1}}{c_{j,1}} + \frac{c_{i,2}}{c_{j,2}} + \cdots + \frac{c_{i,n}}{c_{j,n}}\right) c_{ij};\quad (9)$$

$$c_{ij} = \begin{cases} 2, 1 \leqslant k \leqslant n, c_{j,k} > c_{i,k}; \\ 0.5, \text{其他}。 \end{cases} \quad (10)$$

多 AMR 各自进行决策,输出的结果组成一个完整的 AMR 分布式协同调度方案 $Q = \{Q_1, Q_2, \cdots, Q_A\}$,其中,$Q_j$ 表示对应 A_j 所获得的决策结果。

2.4　奖励函数设计

奖励函数设置如式(11)和式(12)所示,R 为总奖励函数。

$$R = R_1 + R_2 + R_3 - R_4;\quad (11)$$

$$R_i^t = R_{1i}^t + R_{2i}^t。\quad (12)$$

式中:R_{1i}^t 为碰撞奖励,当两个 AMR 之间距离低于安全距离时,给予一个负的奖励,$R_{1i}^t = -1$ 表示惩罚;R_{2i}^t 为距离奖励,将 AMR 和任务工件距离作为奖励值,距离越近,奖励值越多,$R_{2i}^t = -d_{ij}$;R_i^t 表示 t 时刻任务 T_i 被某 AMR 选择执行后所能得到的奖励;$R_1 = \sum R_{1i}^t$ 为所有 AMR 碰撞奖励之和;$R_2 = \sum R_{2i}^t$ 为所有 AMR 距离奖励之和;$R_3 = \sum e_{ij}$ 为各智能体与任务节点的能力匹配之和;R_4 为没有被分配对应的 AMR 的任务节点的数量。

在多 AMR 分布协同调度场景中,目标是车间里所有任务都被 AMR 自主分配,且各个 AMR 的能力都得到了匹配,达到群体最优。在车间 AMR 调度场景中将各 AMR 视为多个决策 Agent,“AMR Agent-配送任务”的匹配可以视为一个多智能体强化学习任务训练过程,AMR 需要不断地学习来提高自己和群体的智慧。AMR 的协同调度优化问题的目标函数即作为强化学习中的奖励信号,奖励函数的设计可以根据任务节点的覆盖程度以及完成效果来进行设计。

算法 1　多 AMR 协作任务分配强化学习算法。

输入:初始化各智能体策略网络的参数 $\theta = (\theta_1,$

$\theta_2,\cdots,\theta_N)$,价值网络参数 $\varphi=(\varphi_1,\varphi_2,\cdots,\varphi_N)$;

输出:训练后的最优参数 θ^*、φ^*。

① For $i_{ep}\leftarrow 1,2,\cdots,\max\ eposide$ do

② 重置环境,得到每个智能体的联合状态 $o=(o_1,o_2,\cdots,o_N)$;

③ 根据各智能体的策略网络得到各智能体的动作 $a_i\sim w_i(\cdot\mid o_i)$,

④ 将联合动作 $a=(a_1,a_2,\cdots,a_N)$ 输入到环境中,得到反馈的奖励值 $r=(r_1,r_2,\cdots,r_N)$;

⑤ 将各智能体的数据元组 (o_i,a_i,r_i) 存储到数据池 datapool 中,

⑥ If i_{ep}>最小参数更新间隔,

⑦ 从 datapool 中采样出一个批次的数据样本 dataset,

⑧ for agent $i=1$ to N:

⑨ 计算各智能体的策略网络和价值网络的梯度值

$$\mathrm{d}\theta_i\leftarrow\frac{1}{n}\sum_{k=1}^{n}E_{o\sim\text{datapool},a\sim w}\nabla_{\theta_i w_i}(o_i^k)\nabla_{\theta_i}Q_i^w(o^k,a);$$

$$\mathrm{d}\varphi_i\leftarrow\frac{1}{n}\sum_{k=1}^{n}E_{(o,a,r)\text{datapool}}(r_i^k-Q_i^{\varphi}(o,a))^2;$$

⑩ for Agent $i=1$ to N:

⑪ 更新各智能体的策略网络和价值网络的参数

$\theta_i\leftarrow(1-\delta)\theta_i+\delta\cdot Adam(\mathrm{d}\theta_i)$

$\varphi_i\leftarrow(1-\delta)\varphi_i+\delta\cdot Adam(\mathrm{d}\varphi_i)$

⑫ end for。

3 仿真测试

选取某智能离散机加车间作为案例仿真场景,AMR 从当前位置出发接受任务,并对相应的工位进行物料配送。在该场景中,有 3 个 AMR 参与了调度,设计了两组实验进行分析,如图 5 所示。在第 1 组实验中,任务分布情况如图 5(a)所示;在第 2 组实验中,下发了新的随机任务,任务分布情况如图 5(b)所示。实验使用基于 MADDPG 的任务动态分配算法与现有的 DQN 算法以及 BicNet 算法进行训练和比较。

3.1 参数设置

在 Pycharm 上进行仿真,采用的核心工具包的版本为 python3.6.0、tensorflow1.8.0、numpy1.14.5、gym0.10.5。神经网络隐藏层使用 ReLU 函数作为激活函数,策略和价值网络均采用了 Adam Optimizer 优化器进行学习,MADDPG 算法、DQN 算法和 BicNet 算法的网络结构参数分别如表 1、表 2 和表 3 所示。经过测试,模型在经过 1 000 次训练后基本收敛,因此训练迭代次数设定为 1 000 次,为平衡模型更新速度与训练性能,网络参数的批次大小被设置为 64,经验池的大小为 1 000 000。

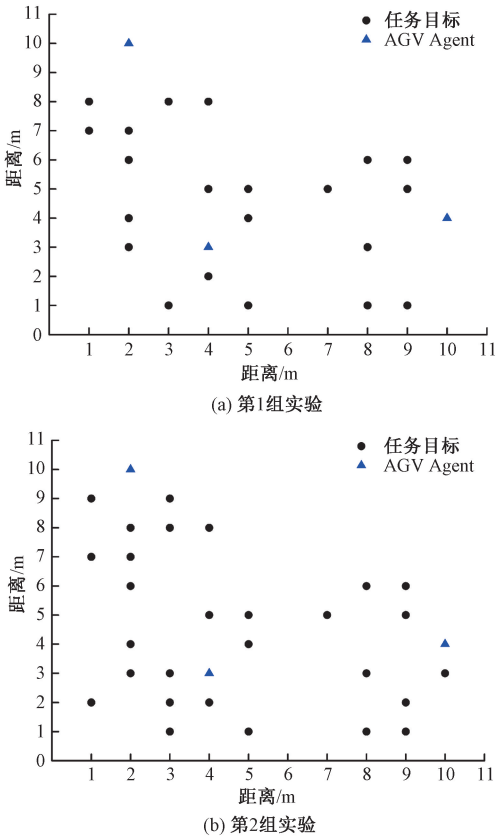


图 5 实验任务布局

Figure 5 Layout of experimental tasks

表 1 MADDPG 网络结构参数

Table 1 MADDPG network structure parameters

参数	取值	参数	取值
经验池大小	10^6	策略网络层数	3
隐藏层神经元	64	价值网络层数	3
更新批次大小	64	价值网络学习率	0.01
训练次数	1 000	策略网络学习率	0.01

表 2 DQN 网络结构参数

Table 2 DQN network structure parameters

参数	取值	参数	取值
经验池大小	10^6	折扣因子	0.9
学习率	0.001	ε-贪心策略参数	0.01
更新批次大小	64	目标网络更新频率	次/(200 步)
训练次数	1 000		

MADDPG 算法的训练过程可分为两个主要阶段。第 1 阶段为数据收集,在每个 *episode* 中,各自的 Actor 网络根据当前状态选择动作并执行;然后,观察下一个状态、奖励和终止信号,并将这些经验存

表 3　BicNet 网络结构参数

Table 3　BicNet network structure parameters

参数	取值	参数	取值
经验池大小	10^6	ε-贪心策略参数	0.01
学习率	0.001	训练次数	1 000
更新批次大小	64	目标网络更新频率	次/(200 步)

储到经验回放缓冲区中。第 2 阶段为网络更新，当经验回放缓冲区的样本数量达到一定值时，从中随机采样一批经验；对于每个 AMR，根据当前状态和动作计算目标 Q 值，并使用这些值更新 Actor 网络和 Critic 网络的参数。

3.2　结果分析

通过 MADDPG 算法训练，第 1 组实验和第 2 组实验的 3 个 AMR 调度结果如图 6 所示，每个 AMR 都选择了最合适的配送任务，使得 AMR 的运动范围之和最小，整体完成时间达到最小。

本案例验证算法在 AMR 执行任务中对任务开始时间和完成时间进行了设定，即考虑任务在一定时间约束下的完成度。

第 1 组实验不同算法的各个 AMR 的奖励曲线对比如图 7 所示，横坐标是训练次数，纵坐标是奖励值。每个 AMR 在 MADDPG 算法训练下的奖励值是最高的且能较快达到收敛。同时随着训练次数的增多，奖励逐渐增大，在模型训练初期，AMR 由于训练刚开始，还未学会与其他 AMR 进行协同，没有考虑群体目标，导致任务完成效率较低，AMR 获得的奖励值有限，且波动较后期有较大浮动，但随训练时间的增加，AMR 不断“试错”，逐步地学会根据自身情况和外界信息与其他 AMR 进行分布式协同，从而完成群体任务的分配。在训练次数达到 1 000 次后，算法的奖励曲线都趋于平缓，呈收敛趋势。

第 2 组实验下系统在前期随机下发新任务后不同算法奖励值随训练次数变化的情况如图 8 所示。新任务刚下发时，AMR 未执行过此新任务，因此奖励值变化不大，且有一定的波动，但随着训练次数的增加，整体奖励值逐渐恢复上升趋势，MADDPG 算法在执行过程中展现出了更高的学习速度。相比其他算法，MADDPG 算法的任务完成度位于前列，MADDPG 算法的平均奖励值较其他算法增幅为 3，训练次数减少了 300 次，学习速度更快，且能很快达到收敛。随着任务增多，时间约束更加严格。采用动态决策策略，有助于完成任务。因此，提出的 MADDPG 算法在下发新任务的突发情况下，能够有效地进行实时分析，实现群体决策，最终获得最优任务分配结果，提高整体任务完成效率。

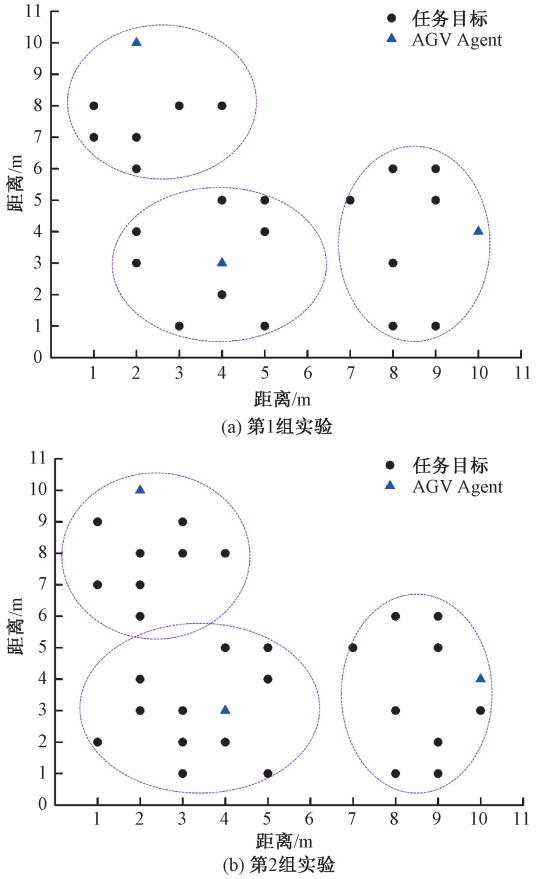


图 6　AMR 调度结果

Figure 6　AMR scheduling results

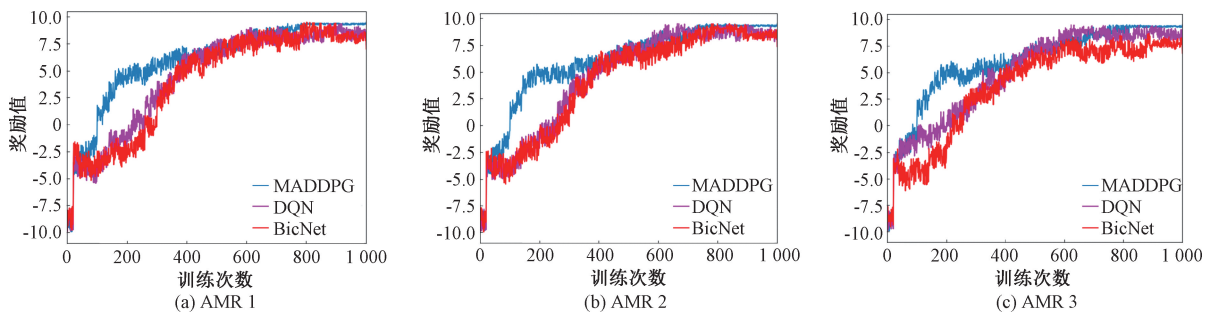


图 7　第 1 组实验 AMR 的算法奖励曲线图

Figure 7　Algorithm reward curve of the first group of experiment AMR

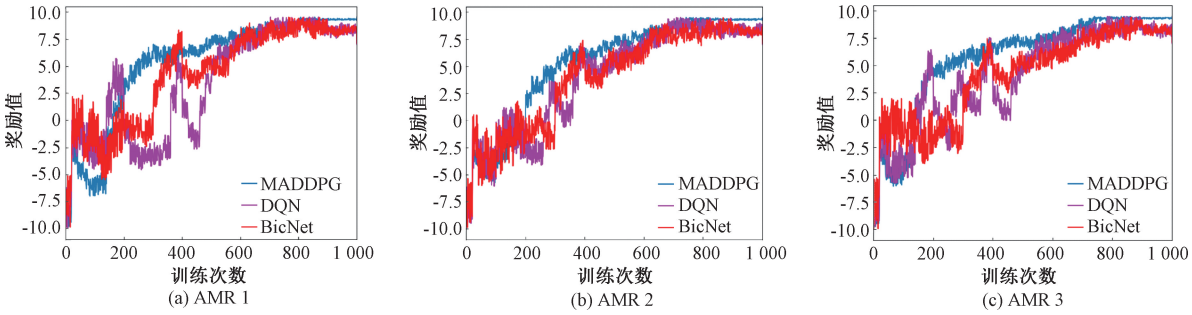


图 8 第 2 组实验 AMR 的算法奖励曲线图

Figure 8 Algorithm reward curve of the second group of experiment AMR

4 结论

(1)设计 MADDPG 算法来实现 AMR 的任务分配,并引入注意力机制,对 AMR 的动作以及状态进行了设置,通过奖励值的大小提高了任务节点的覆盖程度以及任务的完成效果,AMR 通过与环境的不断交互训练,与其余 AMR 信息共享,最终各自独立选择了任务,提高了 AMR 任务分配效率。

(2)通过仿真测试对比,MADDPG 算法中的任务完成效率比 DQN 和 BicNet 算法高,MADDPG 算法的平均奖励值较其他算法增幅为 3,同时在下发新任务后,在同一奖励值下,MADDPG 算法的训练次数比其他算法少 300 次。因此,MADDPG 算法在保证求解任务分配完成度的基础上,具有更快的学习速度和稳定的收敛过程。

参考文献:

[1] SHOJAEINASAB A, CHARTER T, JALAYER M, et al. Intelligent manufacturing execution systems: a systematic review[J]. Journal of Manufacturing Systems, 2022, 62: 503-522.

[2] 李腾, 冯珊. 面向“货到人”拣选系统的一种随机调度策略[J]. 工业工程, 2020, 23(2): 59-66.
LI T, FENG S. A research on a random scheduling strategy of “rack to picker” picking system[J]. Industrial Engineering Journal, 2020, 23(2): 59-66.

[3] FRAGAPANE G, IVANOV D, PERON M, et al. Increasing flexibility and productivity in Industry 4.0 production networks with autonomous mobile robots and smart intralogistics[J]. Annals of Operations Research, 2022, 308(1): 125-143.

[4] HERCIK R, BYRTUS R, JAROS R, et al. Implementation of autonomous mobile robot in SmartFactory[J]. Applied Sciences, 2022, 12(17): 8912.

[5] WANG X, WANG L, WANG S Y, et al. Recommending-and-grabbing: a crowdsourcing-based order allocation pattern for on-demand food delivery[J]. IEEE Transactions on Intelligent Transportation Systems, 2023, 24(1): 838-853.

[6] 刘广瑞, 王庆海, 姚冬艳. 基于改进人工蜂群算法的多无人机协同任务规划[J]. 郑州大学学报(工学版), 2018, 39(3): 51-55.
LIU G R, WANG Q H, YAO D Y. Multi-UAV cooperative mission planning based on improved artificial bee colony algorithm[J]. Journal of Zhengzhou University (Engineering Science), 2018, 39(3): 51-55.

[7] WU X W, XIAO B, CAO L, et al. Optimal transport and model predictive control-based simultaneous task assignment and trajectory planning for unmanned system swarm [J]. Journal of Intelligent & Robotic Systems, 2024, 110(1): 28.

[8] 王俊英, 颜芬芬, 陈鹏, 等. 基于概率自适应蚁群算法的云任务调度方法[J]. 郑州大学学报(工学版), 2017, 38(4): 51-56.
WANG J Y, YAN F F, CHEN P, et al. Task scheduling method based on probability adaptive ant colony optimization in cloud computing[J]. Journal of Zhengzhou University (Engineering Science), 2017, 38(4): 51-56.

[9] 吴蔚楠, 关英姿, 郭继峰, 等. 基于 SEAD 任务特性约束的协同任务分配方法[J]. 控制与决策, 2017, 32(9): 1574-1582.
WU W N, GUAN Y Z, GUO J F, et al. Research on cooperative task assignment method used to the mission SEAD with real constraints[J]. Control and Decision, 2017, 32(9): 1574-1582.

[10] 鞠锴, 冒泽慧, 姜斌, 等. 基于势博弈的异构多智能体系统任务分配和重分配[J]. 自动化学报, 2022, 48(10): 2416-2428.
JU K, MAO Z H, JIANG B, et al. Task allocation and reallocation for heterogeneous multiagent systems based on potential game[J]. Acta Automatica Sinica, 2022, 48(10): 2416-2428.

[11] 施伟, 冯旸赫, 程光权, 等. 基于深度强化学习的多机协同空战方法研究[J]. 自动化学报, 2021, 47(7): 1610-1623.
SHI W, FENG Y H, CHENG G Q, et al. Research on

multi-aircraft cooperative air combat method based on deep reinforcement learning [J]. ACTA Automatica Sinica, 2021, 47(7):1610-1623.

[12] MOTES J, SANDSTRÖM R, LEE H, et al. Multi-robot task and motion planning with subtask dependencies[J]. IEEE Robotics and Automation Letters, 2020, 5(2): 3338-3345.

[13] YIN Z Z, LIU J H, WANG D P. Multi-AGV task allocation with attention based on deep reinforcement learning [J]. International Journal of Pattern Recognition and Artificial Intelligence, 2022, 36(9): 1-20.

[14] LI M G, MA M, WANG L, et al. Multitask-oriented collaborative crowdsensing based on reinforcement learning and blockchain for intelligent transportation system[J]. IEEE Transactions on Industrial Informatics, 2023, 19(9): 9503-9514.

[15] OROOJLOOY A, HAJINEZHAD D. A review of cooperative multi-agent deep reinforcement learning[J]. Applied Intelligence, 2023, 53(11): 13677-13722.

[16] WANG H P, LI S Q, JI H C. Fitness-based hierarchical reinforcement learning for multi-human-robot task allocation in complex terrain conditions[J]. Arabian Journal for Science and Engineering, 2023, 48(5): 7031-7041.

[17] XIAO X J, PAN Y H, LV L L, et al. Scheduling multi-mode resource-constrained tasks of automated guided vehicles with an improved particle swarm optimization algorithm[J]. IET Collaborative Intelligent Manufacturing, 2021, 3(2): 93-104.

[18] 王乐,齐尧,何滨兵,等. 机器人自主探索算法综述[J]. 计算机应用,2023,43(A1):314-322.
WANG L, QI Y, HE B B, et al. Survey of autonomous exploration algorithms for robots[J]. Journal of Computer Applications, 2023,43(A1):314-322.

[19] VU Q T, DUONG V T, NGUYEN H H, et al. Optimization of swimming mode for elongated undulating fin using multi-agent deep deterministic policy gradient[J]. Engineering Science and Technology, an International Journal, 2024, 56: 101783.

[20] SUMIEA E H, ABDULKADIR S J, ALHUSSIAN H S, et al. Deep deterministic policy gradient algorithm: a systematic review[J]. Heliyon, 2024, 10(9): e30697.

[21] CHAI J J, LI W F, ZHU Y H, et al. UNMAS: multiagent reinforcement learning for unshaped cooperative scenarios[J]. IEEE Transactions on Neural Networks and Learning Systems, 2023, 34(4): 2093-2104.

[22] MENG X Q, JIANG J H, WANG H. AGWO: advanced GWO in multi-layer perception optimization[J]. Expert Systems with Applications, 2021, 173: 114676.

[23] 敬超,全育涛,陈艳. 基于多层感知机-注意力模型的功耗预测算法[J/OL]. 计算机应用,2024:1-10(2024-11-13)[2024-11-15]. http://kns.cnki.net/kcms/detail/51.1307.TP.20241112.1237.004.html.
JING C, QUAN Y T, CHEN Y. Improved multi-layer perceptron and attention model-based power consumption prediction algorithm[J/OL]. Journal of Computer Applications, 2024:1-10(2024-11-13)[2024-11-15]. http://kns.cnki.net/kcms/detail/51.1307.TP.20241112.1237.004.html.

AMRs Autonomous Collaboration Task Assignment Method Based on Multi-agent Reinforcement Learning

ZHANG Fuqiang[1,2], ZHANG Yanrui[1,2], DING Kai[1,2], CHANG Fengtian[1,2]

(1. Key Laboratory of Road Construction Technology and Equipment of MOE, Chang'an University, Xi'an 710064, China; 2. Institute of Smart Manufacturing Systems, Chang'an University, Xi'an 710064, China)

Abstract: In order to solve the task autonomy assignment problem of AMR in flexible production, a multi-agent deep deterministic policy gradient (MADDPG) algorithm based on improved multi-agent reinforcement learning algorithm was adopted. The attention mechanism was introduced to improve the algorithm. Firstly, the framework of centralized training decentralized execution was adopted, and then the action and state of AMR were set. Secondly, according to the size of the reward value, the coverage degree of the task node and the completion effect of the task were determined. The simulation results showed that the average reward value of MADDPG algorithm increase 3 than other algorithms, and the training times were reduced by 300 times. It could have faster learning speed and more stable convergence process while ensuring the completion of task allocation.

Keywords: autonomous mobile robot; multi-agent; reinforcement learning; collaboration; task assignment