

第十五章 非合作关系设定下的多智能体强化学习

上一章研究了多智能体强化学习 (MARL) 中最简单的设定——完全合作关系，在这种设定下，所有的智能体有相同的奖励、回报、价值、目标函数。本章研究非合作关系，那么不同智能体各自有不同的奖励、回报、价值、目标函数。本章中采用的符号如图 15.1 所示。

第 15.1 节定义非合作关系设定下的策略学习、策略梯度方法、以及收敛判别。第 15.2 节推导非合作关系下的 A2C 方法，本书称之为 Multi-Agent Noncooperative A2C，缩写 MAN-A2C，可以用于离散控制问题。第 15.3 节用三种架构实现 MAN-A2C：完全去中心化、完全中心化、中心化训练 + 去中心化决策。第 15.4 介绍多智能体确定策略梯度方法，缩写 MADDPG，可以用于连续控制问题。

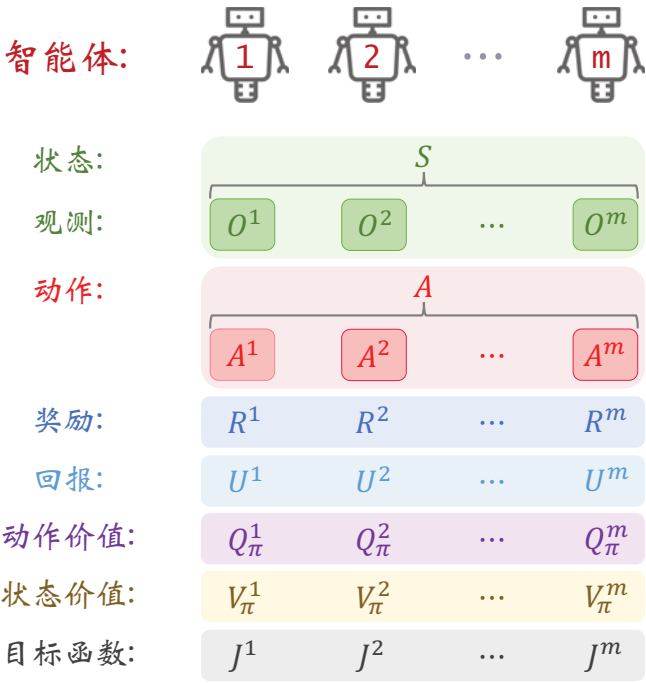


图 15.1: 多智能体强化学习 (MARL) 的符号。

《深度强化学习》2021-02-19 尚未校对，仅供预览。
如发现错误，请告知作者 shusen.wang@stevens.edu

15.1 非合作关系设定下的策略学习

上一章研究合作关系的 MARL，即所有智能体的奖励都相等： $R^1 = \dots = R^m$ 。在这种设定下，所有智能体有相同的状态价值函数 $V_\pi(s)$ 和目标函数

$$J(\theta^1, \dots, \theta^m) = \mathbb{E}_S[V_\pi(s)].$$

目标函数可以衡量策略网络参数 $\theta^1, \dots, \theta^m$ 的好坏。策略学习的目的是改进 $\theta^1, \dots, \theta^m$ 使得 J 变大。合作关系的设定下，策略学习的收敛标准很明确：如果找不到更好的 $\theta^1, \dots, \theta^m$ 使得 J 变大，那么当前的 $\theta^1, \dots, \theta^m$ 就是最优解。

非合作关系设定下的目标函数：如果是非合作关系，那么不存在这样的关系： $R^1 = \dots = R^m$ 。两个智能体的奖励不相等 ($R^i \neq R^j$)，那么它们的回报也不相等 ($U^i \neq U^j$)，回报的期望（价值函数）也不相等。把状态价值记作：

$$V^1(s), V^2(s), \dots, V^m(s).$$

第 i 个智能体的目标函数是状态价值的期望：

$$J^i(\theta^1, \dots, \theta^m) = \mathbb{E}_S[V_\pi^i(s)].$$

J^i 的意义是回报 U^i 的期望，所以能反映出第 i 个智能体的表现好坏。

注 目标函数 J^1, J^2, \dots, J^m 是各不相同的，也就是说智能体没有共同的目标（除非是完全合作关系）。举个例子，在 Predator-Prey（捕食者—猎物）的游戏中，捕食者的目标函数 J^1 与猎物的目标函数 J^2 负相关： $J^1 = -J^2$ 。

注 第 i 个智能体的目标函数 J^i 依赖于所有智能体的策略网络参数 $\theta^1, \dots, \theta^m$ 。为什么一个智能体的目标函数依赖于其他智能体的策略呢？举个例子，捕食者改进自己的策略 θ^1 ，而猎物没有改变策略 θ^2 。虽然猎物的策略 θ^2 没有变化，但是它的目标函数 J^2 会减小。

非合作关系设定下的策略学习：在多智能体的策略学习中，第 i 个智能体的目标是改进自己的策略参数 θ^i ，使得 J^i 尽量大。多智能体的策略学习可以描述为这样的问题：

$$\text{第 1 个智能体求解：} \quad \max_{\theta^1} J^1(\theta^1, \dots, \theta^m),$$

$$\text{第 2 个智能体求解：} \quad \max_{\theta^2} J^2(\theta^1, \dots, \theta^m),$$

$$\vdots \quad \quad \quad \vdots$$

$$\text{第 } m \text{ 个智能体求解：} \quad \max_{\theta^m} J^m(\theta^1, \dots, \theta^m).$$

注意，目标函数 J^1, J^2, \dots, J^m 是各不相同的，也就是说智能体没有共同的目标（除非是完全合作关系）。策略学习的基本思想是让每个智能体各自做策略梯度上升：

$$\text{第 1 号智能体执行：} \quad \theta^1 \leftarrow \theta^1 + \alpha^1 \cdot \nabla_{\theta^1} J^1(\theta^1, \dots, \theta^m),$$

$$\text{第 2 号智能体执行：} \quad \theta^2 \leftarrow \theta^2 + \alpha^2 \cdot \nabla_{\theta^2} J^2(\theta^1, \dots, \theta^m),$$

$$\vdots \quad \quad \quad \vdots$$

$$\text{第 } m \text{ 号智能体执行：} \quad \theta^m \leftarrow \theta^m + \alpha^m \cdot \nabla_{\theta^m} J^m(\theta^1, \dots, \theta^m).$$

公式中的 $\alpha^1, \alpha^2, \dots, \alpha^m$ 是学习率。由于无法直接计算策略梯度 $\nabla_{\theta^i} J^i$ ，我们需要对其做

《深度强化学习》2021-02-19 尚未校对，仅供预览。
如发现错误，请告知作者 shusen.wang@stevens.edu

近似。各种策略学习方法的区别就在于如何对策略梯度做近似。

收敛的判别：在合作关系设定下，所有智能体有相同的目标函数 ($J^1 = \dots = J^m$)，那么判断收敛的标准就是目标函数值不再增长。也就是说改变任何智能体的策略都无法让团队的回报增长。

在非合作关系设定下，智能体的利益是不一致的、甚至是冲突的，智能体各有各的目标函数。该如何判断策略学习的收敛呢？不能用 $J^1 + J^2 + \dots + J^m$ 作为判断收敛的标准。比如在 Predator-Prey（捕食者—猎物）的游戏中，双方的目标函数是冲突的： $J^1 = -J^2$ 。如果捕食者改进策略，那么 J^1 会增长，而 J^2 会下降。自始至终， $J^1 + J^2$ 一直等于零，不论策略学习有没有收敛。

在非合作关系设定下，收敛标准是纳什均衡。一个智能体在制定策略的时候，要考虑到其他各方的策略。在纳什均衡的情况下，每一个智能体都在以最优的方式来应对其他各方的策略。在纳什均衡的情况下，谁也没有动机去单独改变自己的策略，因为改变策略不会增加自己的收益。这样就达到了一种平衡状态，所有智能体都找不到更好的策略。这种平衡状态就被认为是收敛。在实验中，如果所有智能体的平均回报都不再变化，就可以认为达到了纳什均衡。

定义 15.1. 纳什均衡

在多智能体系统中，当其余所有智能体都不改变策略的情况下，一个智能体 i 单独改变策略 θ^i ，无法让其期望回报 $J^i(\theta^1, \dots, \theta^m)$ 变大。

评价策略的优劣：有两种策略学习的方法 \mathcal{M}_+ 和 \mathcal{M}_- ，把它们训练出的策略网络参数分别记作 $\theta_+^1, \dots, \theta_+^m$ 和 $\theta_-^1, \dots, \theta_-^m$ 。该如何评价 \mathcal{M}_+ 和 \mathcal{M}_- 的优劣呢？在合作关系设定下，很容易评价两种方法的好坏。在收敛之后，把两种策略的平均回报记作 J_+ 和 J_- 。如果 $J_+ > J_-$ ，就说明 \mathcal{M}_+ 比 \mathcal{M}_- 好；反之亦然。

在非合作关系的设定下，不能直接用平均回报评价策略的优劣。以捕食者—猎物的游戏为例，我们用两种方法 \mathcal{M}_+ 和 \mathcal{M}_- 训练策略网络，把它们训练出的策略网络记作：

$$\begin{aligned} \pi(a | s, \theta_+^{\text{predator}}), & \quad \pi(a | s, \theta_+^{\text{prey}}), \\ \pi(a | s, \theta_-^{\text{predator}}), & \quad \pi(a | s, \theta_-^{\text{prey}}). \end{aligned}$$

设收敛时的平均回报为：

$$\begin{aligned} J_+^{\text{predator}} &= 0.8, & J_+^{\text{prey}} &= -0.8, \\ J_-^{\text{predator}} &= 0.1, & J_-^{\text{prey}} &= -0.1. \end{aligned}$$

请问 \mathcal{M}_+ 和 \mathcal{M}_- 孰优孰劣呢？假如我们的目标是学习捕食者 (Predator)，能否说明 \mathcal{M}_+ 比 \mathcal{M}_- 好呢？答案是否定的。 $J_+^{\text{predator}} > J_-^{\text{predator}}$ 可能是由于方法 \mathcal{M}_+ 没有训练好猎物 (Prey) 的策略 θ_+^{prey} ，导致捕食者 (Predator) 相对有优势。 $J_+^{\text{predator}} > J_-^{\text{predator}}$ 不能说明策略 $\theta_+^{\text{predator}}$ 优于 $\theta_-^{\text{predator}}$ 。

在非合作关系的设定下，该如何评价两种方法 \mathcal{M}_+ 和 \mathcal{M}_- 的优劣呢？以捕食者—

猎物的游戏为例，我们让一种方法训练出的捕食者与另一种方法训练出的猎物对决：

$$\begin{aligned} \pi(a \mid s, \theta_+^{\text{predator}}) & \quad \text{对决} \quad \pi(a \mid s, \theta_-^{\text{prey}}), \\ \pi(a \mid s, \theta_-^{\text{predator}}) & \quad \text{对决} \quad \pi(a \mid s, \theta_+^{\text{prey}}). \end{aligned}$$

记录下两方捕食者的平均回报，记作 J_+^{predator} 、 J_-^{predator} 。两者的大小可以反映出 \mathcal{M}_+ 和 \mathcal{M}_- 的优劣。

15.2 非合作设定下的多智能体 A2C

本节研究“非合作关系”设定下的多智能体 A2C 方法 (Multi-Agent Non-cooperative A2C), 缩写 MAN-A2C。

15.2.1 策略网络和价值网络

MAN-A2C 中, 每个智能体有自己的策略网络和价值网络, 记作:

$$\pi(a^i | s; \theta^i) \quad \text{和} \quad v(s; w^i).$$

第 i 个策略网络需要把所有智能体的观测 $s = [o^1, \dots, o^m]$ 作为输入, 并输出一个概率分布; 第 i 个智能体依据该概率分布抽样得到动作 A^i 。两类神经网络的结构与上一章的 MAC-A2C 完全相同。请注意上一章 MAC-A2C 与本章 MAN-A2C 的区别:

- 上一章的 MAC-A2C 用于完全合作关系, 所有智能体有相同的状态价值函数 $V_\pi(s)$, 所以只用一个神经网络近似 $V_\pi(s)$, 记作 $v(s; w)$ 。
- 本章的 MAN-A2C 用于非合作关系, 每个智能体各有一个状态价值函数 $V_\pi^i(s)$, 所以每个智能体各自对应一个价值网络 $v(s; w^i)$ 。

MAN-A2C 属于 Actor-Critic 方法: 策略网络 $\pi(a^i | s; \theta^i)$ 相当于第 i 个运动员, 负责做决策; 每个运动员都有一个专属的评委 $v(s; w^i)$, 对运动员 i 的表现予以评价。请注意, 虽然评委 $v(s; w^i)$ 是对运动员 i 个人做出评价, 但是评委委员会考虑到全局的状态 $s = [o^1, \dots, o^m]$ 。举个例子, 在足球比赛中, 评委 i 只对运动员 i 做评价, 目的在于改进运动员 i 的技术。在比赛中, 想要评价运动员 i 的跑位、传球的好坏, 还需要要考虑到队友、对手的位置, 所以评委 i 会考虑到场上所有球员的表现 $s = [o^1, \dots, o^m]$ 。注意与上一章中 MAC-A2C 的区别: MAC-A2C 中只有一位评委, 他会点评整个团队的表现, 而不会给每位运动员单独一个评分。

15.2.2 算法推导

在非合作关系设定下, 第 i 号智能体的动作价值函数记作 $Q_\pi^i(s, a)$, 策略网络记作 $\pi(A^i | S; \theta^i)$ 。不难证明下面的策略梯度定理:

定理 15.1. 非合作关系 MARL 的策略梯度定理

设基线 b 为不依赖于 $A = [A^1, \dots, A^m]$ 的函数。那么有

$$\nabla_{\theta^i} J^i(\theta^1, \dots, \theta^m) = \mathbb{E}_{S, A} \left[\left(Q_\pi^i(S, A) - b \right) \cdot \nabla_{\theta^i} \ln \pi(A^i | S; \theta^i) \right].$$

期望中的动作 A 的概率质量函数为

$$\pi(A | S; \theta^1, \dots, \theta^m) \triangleq \pi(A^1 | S; \theta^1) \times \dots \times \pi(A^m | S; \theta^m).$$

我们用 $b = V_\pi^i(s)$ 作为定理中的基线, 并且用价值网络 $v(s; w^i)$ 近似 $V_\pi^i(s)$ 。按照上

一章的算法推导，我们可以把策略梯度 $\nabla_{\theta^i} J^i(\theta^1, \dots, \theta^m)$ 近似成：

$$\tilde{g}^i(s_t, a_t^i; \theta^i) \triangleq \left(r_t^i + \gamma \cdot v(s_{t+1}; \mathbf{w}^i) - v(s_t; \mathbf{w}^i) \right) \cdot \nabla_{\theta^i} \pi(a_t^i | s_t; \theta^i).$$

观测到状态 s_t 、 s_{t+1} 、动作 a_t^i 、奖励 r_t^i ，这样更新策略网络参数：

$$\theta^i \leftarrow \theta^i + \beta \cdot \tilde{g}^i(s_t, a_t^i; \theta^i).$$

更新价值网络 $v(s; \mathbf{w}^i)$ 的方法与 A2C 基本一样。在观测到状态 s_t 、 s_{t+1} 、奖励 r_t^i 之后，计算 TD 目标：

$$\hat{y}_t^i = r_t^i + \gamma \cdot v(s_{t+1}; \mathbf{w}^i).$$

更新参数 \mathbf{w}^i ，使得 $v(s_t; \mathbf{w}^i)$ 更接近 \hat{y}_t^i 。

15.2.3 训练和决策

训练： 实现 MAN-A2C 的时候，应当使用目标网络缓解自举造成的偏差。第 i 号智能体的目标网络记作 $v(s; \mathbf{w}^{i-})$ ，它的结构与 $v(s; \mathbf{w}^i)$ 相同，但是参数不同。设第 i 号智能体策略网络、价值网络、目标网络当前的参数是 θ_{now}^i 、 $\mathbf{w}_{\text{now}}^i$ 、 $\mathbf{w}_{\text{now}}^{i-}$ 。MAN-A2C 重复下面的步骤更新参数：

1. 观测到当前状态 $s_t = [o_t^1, \dots, o_t^m]$ ，让每一个智能体独立做随机抽样：

$$a_t^i \sim \pi(\cdot | s_t; \theta_{\text{now}}^i), \quad \forall i = 1, \dots, m,$$

并执行选中的动作。

2. 从环境中观测到奖励 r_t^1, \dots, r_t^m 与下一时刻状态 $s_{t+1} = [o_{t+1}^1, \dots, o_{t+1}^m]$ 。
3. 让价值网络做预测：

$$\hat{v}_t^i = v(s_t; \mathbf{w}_{\text{now}}^i), \quad \forall i = 1, \dots, m.$$

4. 让目标网络做预测：

$$\hat{v}_{t+1}^{i-} = v(s_{t+1}; \mathbf{w}_{\text{now}}^{i-}), \quad \forall i = 1, \dots, m.$$

5. 计算 TD 目标与 TD 误差：

$$\hat{y}_t^i = r_t^i + \gamma \cdot \hat{v}_{t+1}^{i-}, \quad \delta_t^i = \hat{v}_t^i - \hat{y}_t^i, \quad \forall i = 1, \dots, m.$$

6. 更新价值网络参数：

$$\mathbf{w}_{\text{new}}^i \leftarrow \mathbf{w}_{\text{now}}^i - \alpha \cdot \delta_t^i \cdot \nabla_{\mathbf{w}^i} v(s_t; \mathbf{w}_{\text{now}}^i), \quad \forall i = 1, \dots, m.$$

7. 更新目标网络参数：

$$\mathbf{w}_{\text{new}}^{i-} \leftarrow \tau \cdot \mathbf{w}_{\text{new}}^i + (1 - \tau) \cdot \mathbf{w}_{\text{now}}^{i-}, \quad \forall i = 1, \dots, m.$$

8. 更新策略网络参数：

$$\theta_{\text{new}}^i \leftarrow \theta_{\text{now}}^i - \beta \cdot \delta_t^i \cdot \nabla_{\theta^i} \ln \pi(a_t^i | s_t; \theta_{\text{now}}^i), \quad \forall i = 1, \dots, m.$$

MAN-A2C 属于同策略 (On-policy)，不能使用经验回放。

决策：在完成训练之后，不再需要价值网络 $v(s; \mathbf{w}^1), \dots, v(s; \mathbf{w}^m)$ 。每个智能体可以用它自己的策略网络做决策。在时刻 t 观测到全局状态 $s_t = [o_t^1, \dots, o_t^m]$ ，然后做随机抽样得到动作：

$$a_t^i \sim \pi(\cdot \mid s_t; \boldsymbol{\theta}^i),$$

并执行动作 a_t^i 。智能体并不能独立做决策，因为策略网络需要知道所有的观测 $s_t = [o_t^1, \dots, o_t^m]$ 。

15.3 三种架构

本节介绍 MAN-A2C 的三种实现方法：“中心化训练 + 中心化决策”、“去中心化训练 + 去中心化决策”、“中心化训练 + 去中心化决策”。

15.3.1 中心化训练 + 中心化决策

可以用完全中心化 (Fully Centralized) 的方式实现 MAN-A2C 的训练和决策。图 15.2 描述了系统的架构。最上面是中央控制器 (Central Controller)，里面部署了所有 m 个价值网络和策略网络：

$$v(s | \mathbf{w}^1), \quad v(s | \mathbf{w}^2), \quad \dots, \quad v(s | \mathbf{w}^m), \\ \pi(a^1 | \theta^1), \quad \pi(a^2 | \theta^2), \quad \dots, \quad \pi(a^m | \theta^m).$$

训练和决策全部由中央控制器完成。智能体负责与环境交互，执行中央控制器的决策 a^i ，并把观测到的 o^i 和 r^i 汇报给中央控制器。这种中心化的方式严格实现了上一节的算法。

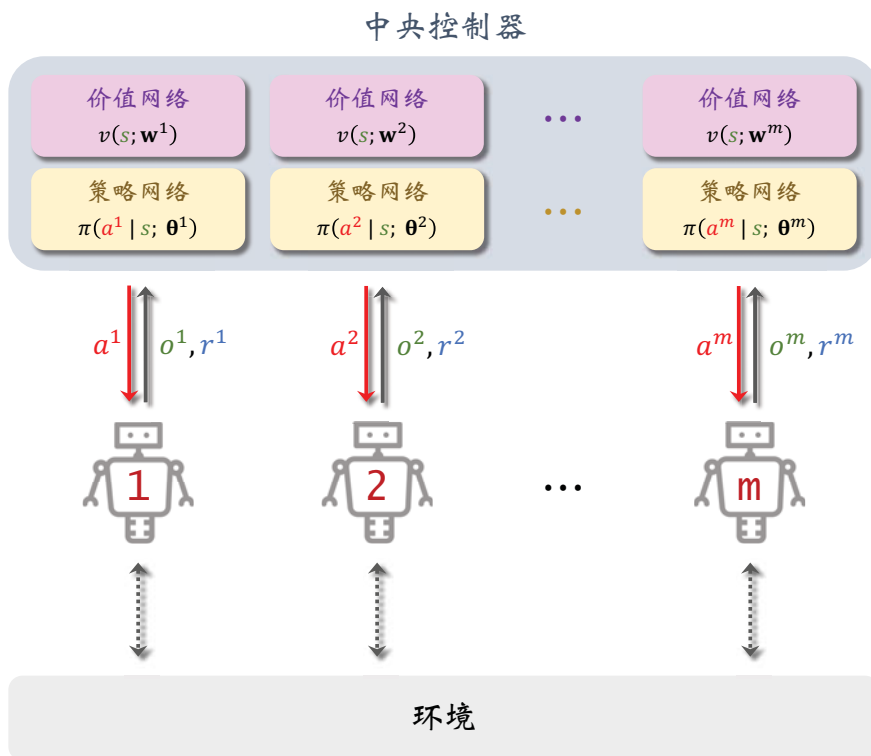


图 15.2: 中心化训练 + 中心化决策的系统架构。

在上一章中，我们用完全中心化的方式实现了 MAC-A2C（见图 14.5）。请注意 MAC-A2C 与此处的 MAN-A2C 的区别。第一，MAC-A2C 的中央控制器上只有一个价值网络，而此处 MAN-A2C 则有 m 个价值网络。第二，MAC-A2C 的每一轮只有一个全局的奖励 r ，而 MAN-A2C 的每个智能体都有自己的奖励 r^i 。

《深度学习》2021-02-19 尚未校对，仅供预览。
如发现错误，请告知作者 shusen.wang@stevens.edu

15.3.2 去中心化训练 + 去中心化决策

为了避免“完全中心化”中的通信，可以对策略网络和价值网络做近似，做到“完全去中心化”。把 MAN-A2C 中的策略网络和价值网络做近似：

$$\begin{aligned}\pi(a^i | s; \theta^i) &\implies \pi(a^i | o^i; \theta^i), \\ v(s; w^i) &\implies v(o^i; w^i).\end{aligned}$$

图 15.3 描述了“完全去中心化”的系统架构。每个智能体上部署一个策略网络和一个价值网络，它们的参数记作 θ^i 和 w^i ；智能体之间不共享参数。这样一来，训练就可以在智能体本地完成，无需中央控制器的参与，也无需通信。这种实现的本质是单智能体强化学习，而非多智能体强化学习。

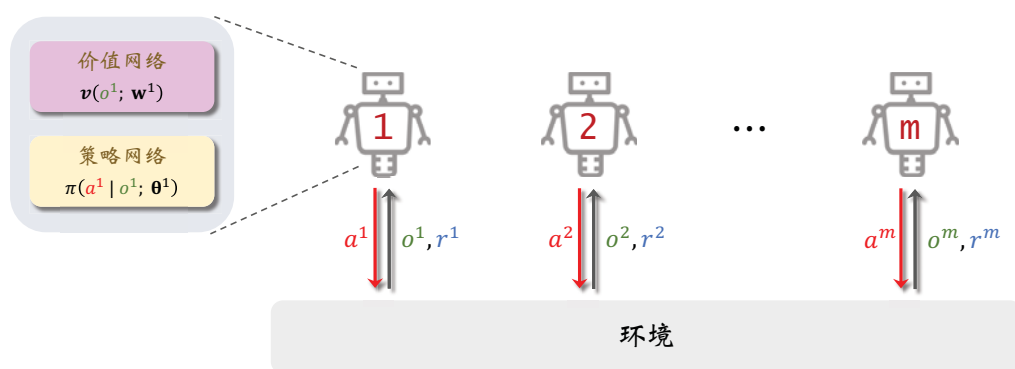


图 15.3: 去中心化训练 + 去中心化决策的系统架构。这种方法也叫做 Independent Actor-Critic。

此处的实现与上一章“完全合作关系”设定下的“完全去中心化”几乎完全相同（见图 14.6）。唯一的区别在于此处每个智能体获得的奖励 r^i 是不同的，而上一章“完全合作关系”设定下的奖励是相同的 $r^1 = \dots = r^m = r$ 。

15.3.3 中心化训练 + 去中心化决策

第三种实现方式是“中心化训练 + 去中心化决策”。与“完全中心化”的 MAN-A2C 相比，唯一的区别在于对策略网络做近似：

$$\pi(a^i | s; \theta^i) \implies \pi(a^i | o^i; \theta^i), \quad \forall i = 1, \dots, m.$$

由于用智能体局部观测 o^i 替换了全局状态 $s = [o^1, \dots, o^m]$ ，策略网络可以部署到每个智能体上。而价值网络仍然是 $v(s; w^i)$ ，没有做近似。

图 15.4 描述了“中心化训练 + 去中心化决策”的系统架构。中央控制器上有所有的价值网络及其目标网络（图中没有画出目标网络）：

$$\begin{aligned}v(s; w^1), \quad v(s; w^2), \quad \dots, \quad v(s; w^m), \\ v(s; w^{1-}), \quad v(s; w^{2-}), \quad \dots, \quad v(s; w^{m-}).\end{aligned}$$

中央控制器用智能体发来的观测 $[o^1, \dots, o^m]$ 和奖励 $[r^1, \dots, r^m]$ 训练这些价值网络。中央控制器把 TD 误差 $\delta^1, \dots, \delta^m$ 反馈给智能体；第 i 号智能体用 δ^i 以及本地的 o^i 、 a^i 来

训练自己的策略网络。

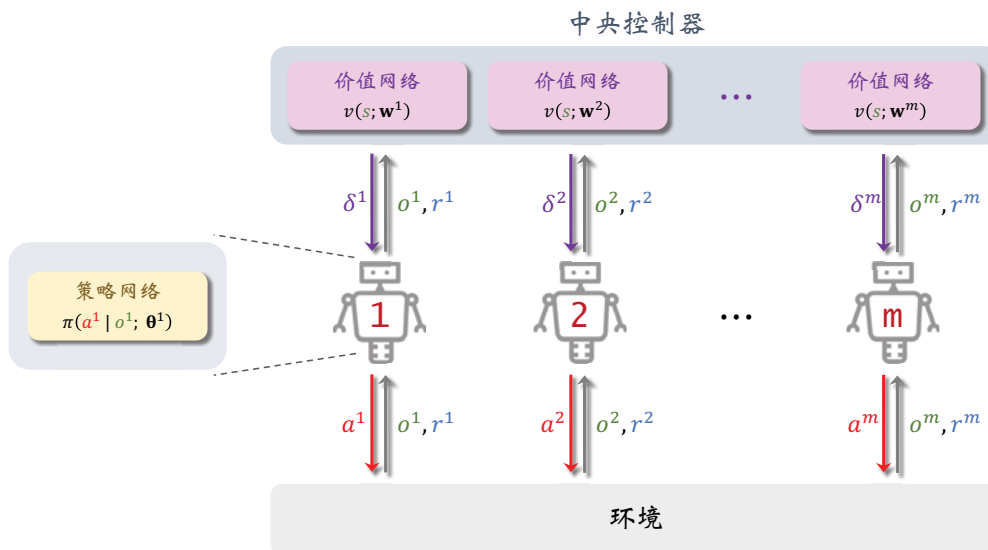


图 15.4: 中心化训练的系统架构。所有 m 个价值网络部署到中央控制器上，策略网络部署到每个智能体上。

上一章“完全合作关系”设定下的“中心化训练”只在中央控制器上部署一个价值网络 $v(s; \mathbf{w})$ 。而此处中央控制器上有 m 个价值网络，每个价值网络对应一个智能体。这是因为此处是“非合作关系”，每个智能体各自对应一个状态价值函数 $V_{\pi}^i(s)$ ，而非有共用的 V_{π} 。

中心化训练： 训练的过程需要所有 m 个智能体共同参与，共同改进策略网络参数 $\theta^1, \dots, \theta^m$ 与价值网络参数 $\mathbf{w}^1, \dots, \mathbf{w}^m$ 。设第 i 号智能体的策略网络、价值网络、目标网络当前的参数分别是 θ_{now}^i 、 $\mathbf{w}_{\text{now}}^i$ 和 $\mathbf{w}_{\text{now}}^{i-}$ 。训练的流程如下：

1. 每个智能体 i 与环境交互，获取当前观测 o_t^i ，独立做随机抽样：

$$a_t^i \sim \pi(\cdot | o_t^i; \theta_{\text{now}}^i), \quad \forall i = 1, \dots, m, \quad (15.1)$$

并执行选中的动作。

2. 下一时刻，每个智能体 i 都观测到 o_{t+1}^i 和收到奖励 r_t^i 。
3. 每个智能体 i 向中央控制器传输观测 o_t^i 、 o_{t+1}^i 、 r_t^i ；中央控制器得到状态

$$s_t = [o_t^1, \dots, o_t^m] \quad \text{和} \quad s_{t+1} = [o_{t+1}^1, \dots, o_{t+1}^m].$$

4. 让价值网络做预测：

$$\hat{v}_t^i = v(s_t; \mathbf{w}_{\text{now}}^i), \quad \forall i = 1, \dots, m.$$

5. 让目标网络做预测：

$$\hat{v}_{t+1}^{i-} = v(s_{t+1}; \mathbf{w}_{\text{now}}^{i-}), \quad \forall i = 1, \dots, m.$$

6. 计算 TD 目标与 TD 误差：

$$\hat{y}_t^i = r_t^i + \gamma \cdot \hat{v}_{t+1}^{i-}, \quad \delta_t^i = \hat{v}_t^i - \hat{y}_t^i, \quad \forall i = 1, \dots, m.$$

7. 更新价值网络参数:

$$\mathbf{w}_{\text{new}}^i \leftarrow \mathbf{w}_{\text{now}}^i - \alpha \cdot \delta_t^i \cdot \nabla_{\mathbf{w}^i} v(s_t; \mathbf{w}_{\text{now}}^i), \quad \forall i = 1, \dots, m.$$

8. 更新目标网络参数:

$$\mathbf{w}_{\text{new}}^{i-} \leftarrow \tau \cdot \mathbf{w}_{\text{new}}^i + (1 - \tau) \cdot \mathbf{w}_{\text{now}}^{i-}, \quad \forall i = 1, \dots, m.$$

9. 更新策略网络参数:

$$\boldsymbol{\theta}_{\text{new}}^i \leftarrow \boldsymbol{\theta}_{\text{now}}^i - \beta \cdot \delta_t^i \cdot \nabla_{\boldsymbol{\theta}^i} \ln \pi(a_t^i | o_t^i; \boldsymbol{\theta}_{\text{now}}^i), \quad \forall i = 1, \dots, m.$$

去中心化决策: 在完成训练之后, 不再需要价值网络 $v(s; \mathbf{w}^1), \dots, v(s; \mathbf{w}^m)$ 。智能体只需要用其本地部署的策略网络 $\pi(a^i | o^i; \boldsymbol{\theta}^i)$ 做决策, 决策过程无需通信, 因此决策速度很快。

15.4 连续控制与 MADDPG

前两节的 MAN-A2C 仅限于离散控制。本节研究连续控制问题，即动作空间 $\mathcal{A}^1, \mathcal{A}^2, \dots, \mathcal{A}^m$ 都是连续集合，动作 $\mathbf{a}^i \in \mathcal{A}^i$ 是向量。本节介绍一种适用于连续控制的多智能体强化学习 (MARL) 方法。多智能体深度确定策略梯度 (Multi-Agent Deep Deterministic Policy Gradient, 缩写 MADDPG) 是一种很有名的 MARL 方法，它的架构是“中心化训练 + 去中心化决策”。

15.4.1 策略网络和价值网络

设系统里有 m 个智能体。每个智能体对应一个策略网络和一个价值网络：

$$\mu(o^i; \theta^i) \quad \text{和} \quad q(s, \mathbf{a}; \mathbf{w}^i).$$

策略网络是确定性的：对于确定的输入 o^i ，输出的动作 $\mathbf{a}^i = \mu(o^i; \theta^i)$ 是确定的。价值网络的输入是全局状态 $s = [o^1, \dots, o^m]$ 与所有智能体的动作 $\mathbf{a} = [\mathbf{a}^1, \dots, \mathbf{a}^m]$ ，输出是一个实数，表示“基于状态 s 执行动作 \mathbf{a} ”的好坏程度。第 i 号策略网络 $\mu(o^i; \theta^i)$ 用于控制第 i 号智能体，而价值网络 $q(s, \mathbf{a}; \mathbf{w}^i)$ 则用于评价所有动作 \mathbf{a} ，给出的分数可以指导第 i 号策略网络做出改进；见图 15.5。MADDPG 因此可以看做一种 Actor-Critic 方法。

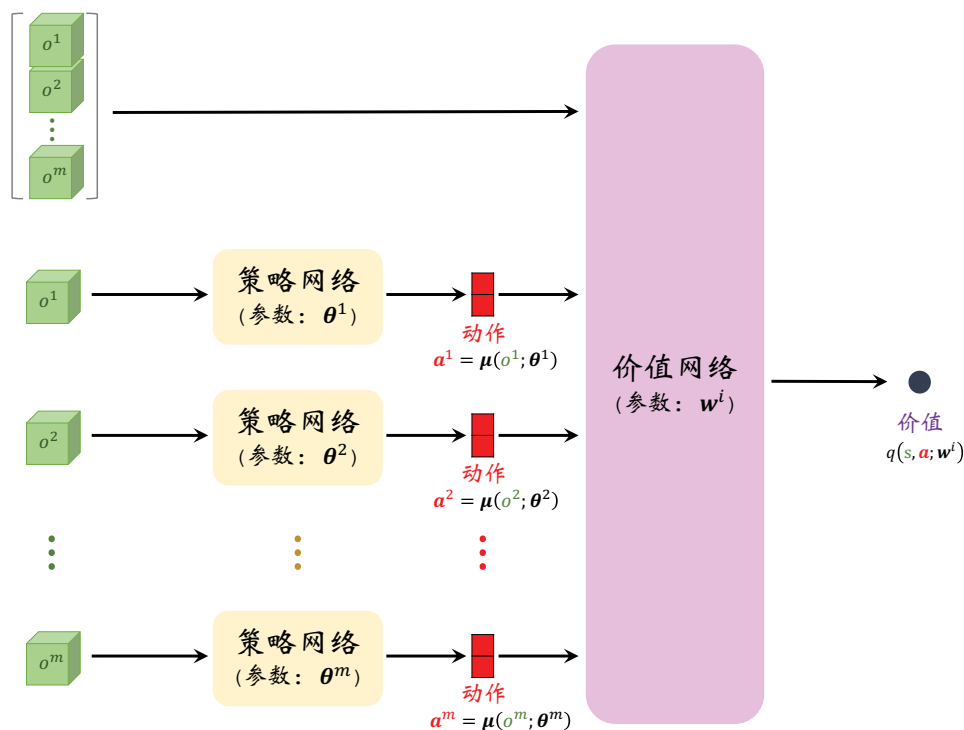


图 15.5: 所有智能体的策略网络与第 i 号智能体的价值网络。

15.4.2 算法推导

训练策略网络和价值网络的算法与第 10.2 节的单智能体 DPG 非常类似：用确定策略梯度更新策略网络，用 TD 算法更新价值网络。MADDPG 是异策略 (Off-policy)，我们可以使用经验回放，重复利用过去的经验。用一个经验回放数组存储收集到的经验，每一条经验都是 $(s_t, \mathbf{a}_t, r_t, s_{t+1})$ 这样一个四元组，其中

$$\begin{aligned} s_t &= [o_t^1, \dots, o_t^m], \\ \mathbf{a}_t &= [\mathbf{a}_t^1, \dots, \mathbf{a}_t^m], \\ s_{t+1} &= [o_{t+1}^1, \dots, o_{t+1}^m], \\ r_t &= [r_t^1, \dots, r_t^m]. \end{aligned}$$

训练策略网络：训练第 i 号策略网络 $\mu(o^i; \theta^i)$ 的目标是改进 θ^i ，增大第 i 号价值网络的平均打分。所以目标函数是：

$$\hat{J}^i(\theta^1, \dots, \theta^m) = \mathbb{E}_S \left[q \left(S, [\mu(O^1; \theta^1), \dots, \mu(O^i; \theta^i), \dots, \mu(O^m; \theta^m)]; \mathbf{w}^i \right) \right].$$

公式中的期望是关于状态 $S = [O^1, \dots, O^m]$ 求的。目标函数的梯度等于：

$$\nabla_{\theta^i} \hat{J}^i(\theta^1, \dots, \theta^m) = \mathbb{E}_S \left[\nabla_{\theta^i} q \left(S, [\mu(O^1; \theta^1), \dots, \mu(O^i; \theta^i), \dots, \mu(O^m; \theta^m)]; \mathbf{w}^i \right) \right].$$

接下来用蒙特卡洛近似公式中的期望。从经验回放数组中随机抽取一个状态：¹

$$s_t = [o_t^1, \dots, o_t^m],$$

它可以看做是随机变量 S 的一个观测值。用所有 m 个策略网络计算动作

$$\hat{\mathbf{a}}_t^1 = \mu(o_t^1; \theta^1), \quad \dots, \quad \hat{\mathbf{a}}_t^m = \mu(o_t^m; \theta^m).$$

那么目标函数的梯度 $\nabla_{\theta^i} \hat{J}^i(\theta^1, \dots, \theta^m)$ 可以近似成为：

$$\begin{aligned} \mathbf{g}_{\theta^i}^i &= \nabla_{\theta^i} q \left(s_t, [\mu(o_t^1; \theta^1), \dots, \mu(o_t^i; \theta^i), \dots, \mu(o_t^m; \theta^m)]; \mathbf{w}^i \right) \\ &= \nabla_{\theta^i} q \left(s_t, [\hat{\mathbf{a}}_t^1, \dots, \hat{\mathbf{a}}_t^m]; \mathbf{w}^i \right). \end{aligned}$$

由于 $\hat{\mathbf{a}}_t^i = \mu(o_t^i; \theta^i)$ ，用链式法则可得：

$$\mathbf{g}_{\theta^i}^i = \nabla_{\theta^i} \mu(o_t^i; \theta^i) \cdot \nabla_{\hat{\mathbf{a}}^i} q \left(s_t, [\hat{\mathbf{a}}_t^1, \dots, \hat{\mathbf{a}}_t^m]; \mathbf{w}^i \right).$$

做梯度上升更新参数 θ^i ：

$$\theta^i \leftarrow \theta^i + \beta \cdot \mathbf{g}_{\theta^i}^i.$$

注意，在更新第 i 号策略网络的时候，除了用到全局状态 s_t ，还需要用到所有智能体的策略网络，以及第 i 号价值网络 $q(s, \mathbf{a}; \mathbf{w}^i)$ 。

训练价值网络：可以用 TD 算法训练第 i 号价值网络 $q(s, \mathbf{a}; \mathbf{w}^i)$ ，让价值网络更好拟

¹更新策略网络只需要四元组 $(s_t, \mathbf{a}_t, r_t, s_{t+1})$ 中的 s_t ，没有用其余三个。

合价值函数 $Q_{\pi}^i(s, \mathbf{a})$ 。给定四元组 $(s_t, \mathbf{a}_t, r_t, s_{t+1})$ ，用所有 m 个策略网络计算动作

$$\hat{\mathbf{a}}_{t+1}^1 = \mu(o_{t+1}^1; \theta^1), \quad \dots, \quad \hat{\mathbf{a}}_{t+1}^m = \mu(o_{t+1}^m; \theta^m).$$

设 $\hat{\mathbf{a}}_{t+1} = [\hat{\mathbf{a}}_{t+1}^1, \dots, \hat{\mathbf{a}}_{t+1}^m]$ 。然后计算 TD 目标：

$$\hat{y}_t^i = r_t^i + \gamma \cdot q(s_{t+1}, \hat{\mathbf{a}}_{t+1}; \mathbf{w}^i).$$

再计算 TD 误差：

$$\delta_t^i = q(s_t, \mathbf{a}_t; \mathbf{w}^i) - \hat{y}_t^i.$$

最后做梯度下降更新参数 \mathbf{w}^i ：

$$\mathbf{w}^i \leftarrow \mathbf{w}^i - \alpha \cdot \delta_t^i \cdot \nabla_{\mathbf{w}^i} q(s_t, \mathbf{a}_t; \mathbf{w}^i).$$

这样可以让价值网络的预测 $q(s_t, \mathbf{a}_t; \mathbf{w}^i)$ 更接近 TD 目标 \hat{y}_t^i 。

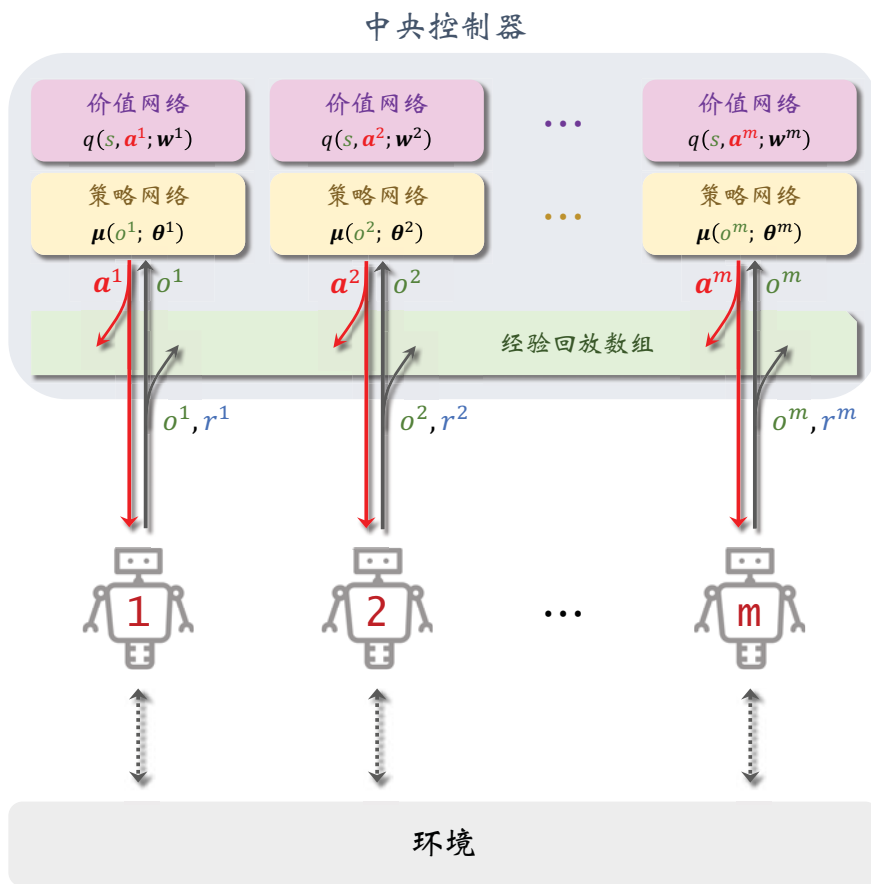


图 15.6: MADDPG 的 centralized 训练。

15.4.3 中心化训练

为了训练第 i 号策略网络和第 i 号价值网络，我们需要用到如下信息：从经验回放数组中取出的 s_t 、 \mathbf{a}_t 、 s_{t+1} 、 r_t^i 、所有 m 个策略网络、以及第 i 号价值网络。很显然，一

个智能体不可能有所有这些信息，因此 MADDPG 需要“中心化训练”。

中心化训练的系统架构如图 15.6 所示。有一个中央控制器，上面部署所有的策略网络和价值网络。训练过程中，策略网络部署到中央控制器上，所以智能体不能自主做决策，智能体只是执行中央控制器发来的指令。由于训练使用异策略，可以把收集经验和更新神经网络参数分开做。

用行为策略收集经验。行为策略 (Behavior Policy) 可以不同于目标策略 (Target Policy, 即 μ)。行为策略是什么都无所谓，比如第 i 个智能体的行为策略可以是

$$\mathbf{a}^i = \mu(o^i; \theta_{\text{old}}^i) + \epsilon,$$

其中 ϵ 是与 \mathbf{a}^i 维度相同的向量，每个元素都是从正态分布中独立抽取的，相当于随机噪声。具体实现的时候，智能体把其观测 o^i 发送给中央控制器。控制器往第 i 号策略网络输出的动作 $\mu(o^i; \theta^i)$ 中加入随机噪声 ϵ ，把动作 \mathbf{a}^i 发送给第 i 号智能体，智能体执行 \mathbf{a}^i 。随后智能体观测到奖励 r^i ，发送给控制器。控制器把每一轮的 o^i, \mathbf{a}^i, r^i 依次存入经验回放数组。

中央控制器更新策略网络和价值网络：实际实现的时候，中央控制器上还需要有如下目标网络（图 15.6 中没有画出）：

$$\begin{aligned} \pi(\mathbf{a}^1 | o^1; \theta^{1-}), \quad \pi(\mathbf{a}^2 | o^2; \theta^{2-}), \quad \dots, \quad \pi(\mathbf{a}^m | o^m; \theta^{m-}); \\ q(s, \mathbf{a}; \mathbf{w}^{1-}), \quad q(s, \mathbf{a}; \mathbf{w}^{2-}), \quad \dots, \quad q(s, \mathbf{a}; \mathbf{w}^{m-}). \end{aligned}$$

设第 i 号智能体当前的参数为：

$$\theta_{\text{now}}^i, \quad \theta_{\text{now}}^{i-}, \quad \mathbf{w}_{\text{now}}^i, \quad \mathbf{w}_{\text{now}}^{i-}.$$

中央控制器每次从经验回放数组中随机抽取一个四元组 $(s_t, \mathbf{a}_t, r_t, s_{t+1})$ ，然后按照下面的步骤更新所有策略网络和价值网络：

1. 让所有 m 个目标策略网络做预测：

$$\hat{\mathbf{a}}_{t+1}^{i-} = \mu(o_{t+1}^i; \theta_{\text{now}}^{i-}), \quad \forall i = 1, \dots, m.$$

把预测汇总成 $\hat{\mathbf{a}}_{t+1}^- = [\hat{\mathbf{a}}_{t+1}^{1-}, \dots, \hat{\mathbf{a}}_{t+1}^{m-}]$ 。

2. 让所有 m 个目标价值网络做出预测：

$$\hat{q}_{t+1}^{i-} = q(s_{t+1}, \hat{\mathbf{a}}_{t+1}^-; \mathbf{w}_{\text{now}}^{i-}), \quad \forall i = 1, \dots, m.$$

3. 计算 TD 目标：

$$\hat{y}_t^i = r_t^i + \gamma \cdot \hat{q}_{t+1}^{i-}, \quad \forall i = 1, \dots, m.$$

4. 让所有 m 个价值网络做预测：

$$\hat{q}_t^i = q(s_t, \mathbf{a}_t; \mathbf{w}_{\text{now}}^i), \quad \forall i = 1, \dots, m.$$

5. 计算 TD 误差：

$$\delta_t^i = \hat{q}_t^i - \hat{y}_t^i, \quad \forall i = 1, \dots, m.$$

6. 更新所有 m 个价值网络:

$$\mathbf{w}_{\text{new}}^i \leftarrow \mathbf{w}_{\text{now}}^i - \alpha \cdot \delta_t^i \cdot \nabla_{\mathbf{w}^i} q(s_t, \mathbf{a}_t; \mathbf{w}_{\text{now}}^i), \quad \forall i = 1, \dots, m.$$

7. 让所有 m 个策略网络做预测:

$$\hat{\mathbf{a}}_t^i = \mu(o_t^i; \theta_{\text{now}}^i), \quad \forall i = 1, \dots, m.$$

把预测汇总成 $\hat{\mathbf{a}}_t = [\hat{\mathbf{a}}_t^1, \dots, \hat{\mathbf{a}}_t^m]$ 。请区别 $\hat{\mathbf{a}}_t$ 与经验回放数组中抽出的 \mathbf{a}_t 。

8. 更新所有 m 个策略网络: $\forall i = 1, \dots, m$,

$$\theta_{\text{new}}^i \leftarrow \theta_{\text{now}}^i - \beta \cdot \nabla_{\theta^i} \mu(s_t; \theta_{\text{now}}^i) \cdot \nabla_{\mathbf{a}_t^i} q(s_t, \hat{\mathbf{a}}_t; \mathbf{w}_{\text{now}}^i).$$

9. 更新所有 $2m$ 个目标网络: $\forall i = 1, \dots, m$,

$$\begin{aligned} \theta_{\text{new}}^{i-} &\leftarrow \tau \cdot \theta_{\text{new}}^i + (1 - \tau) \cdot \theta_{\text{now}}^{i-}, \\ \mathbf{w}_{\text{new}}^{i-} &\leftarrow \tau \cdot \mathbf{w}_{\text{new}}^i + (1 - \tau) \cdot \mathbf{w}_{\text{now}}^{i-}. \end{aligned}$$

改进方法: 可以用三种方法改进 MADDPG。第一, 用第 10.3 节中 TD3 的三种技巧改进训练的算法:

- 用截断双 Q 学习 (Clipped Double Q-Learning) 训练价值网络 $q(s, \mathbf{a}; \mathbf{w}^i)$, $\forall i = 1, \dots, m$ 。
- 往训练算法第一步中的 $\hat{\mathbf{a}}_{t+1}^{i-}$ 加入噪声。
- 减小更新策略网络和目标网络的频率, 每更新 $k (> 1)$ 次价值网络, 更新一次策略网络和目标网络。

第二, 按照第 11 章中的方法, 在策略网络和价值网络中使用 RNN, 记忆历史观测。第三, 在价值网络的结构中使用注意力机制, 见下一章。

15.4.4 去中心化决策

在完成训练之后, 不再需要价值网络, 只需要策略网络做决策。如图 15.7 所示, 把策略网络部署到对应的智能体上。第 i 号智能体可以基于本地观测的 o^i , 在本地独立做决策: $\mathbf{a}^i = \mu(o^i; \theta^i)$ 。

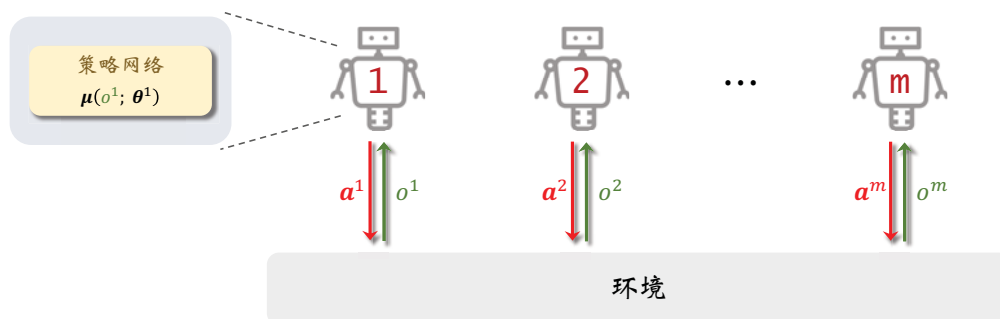


图 15.7: MADDPG 的去中心化决策。

第十五章 相关文献

MAN-A2C 是本书设计出来的简单方法，用于讲解非合作设定下的 MARL，方便读者理解。MAN-A2C 这个名字并没有出现在任何文献中。本章介绍的 MADDPG 由 Lowe 等人 2017 年的论文提出 [65]，它的改进版本叫做 MATD3。

《深度学习》2021-02-19 尚未校对，仅供预览。
如发现错误，请告知作者 shusen.wang@stevens.edu

《深度学习》2021-02-19 尚未校对，仅供预览。
如发现错误，请告知作者 shusen.wang@stevens.edu