第 十四 章　合作关系设定下的多智能体强化学习

本章只考虑最简单的设定——完全合作关系——并在这种设定下研究多智能体强化学习 (MARL)。第 14.1 节定义“完全合作关系”下的策略学习。第 14.2 节介绍“完全合作关系”下的多智能体 A2C 方法，本书称之为 MAC-A2C。第 14.3 节介绍 MARL 的三种常见架构——完全去中心化、完全中心化、中心化训练 + 去中心化决策——并在三种框架下实现 MAC-A2C。

本章与上一章对状态的定义有所区别。在多智能体系统中，一个智能体未必能观测到全局状态 S。设第 i 号智能体有一个局部观测，记作 O^i，它是 S 的一部分。不妨假设所有的局部观测的总和构成全局状态：

$$S = [O^1, O^2, \cdots, O^m],$$

MARL 的文献大多采用这种假设。本章中采用的符号如图 14.1 所示。

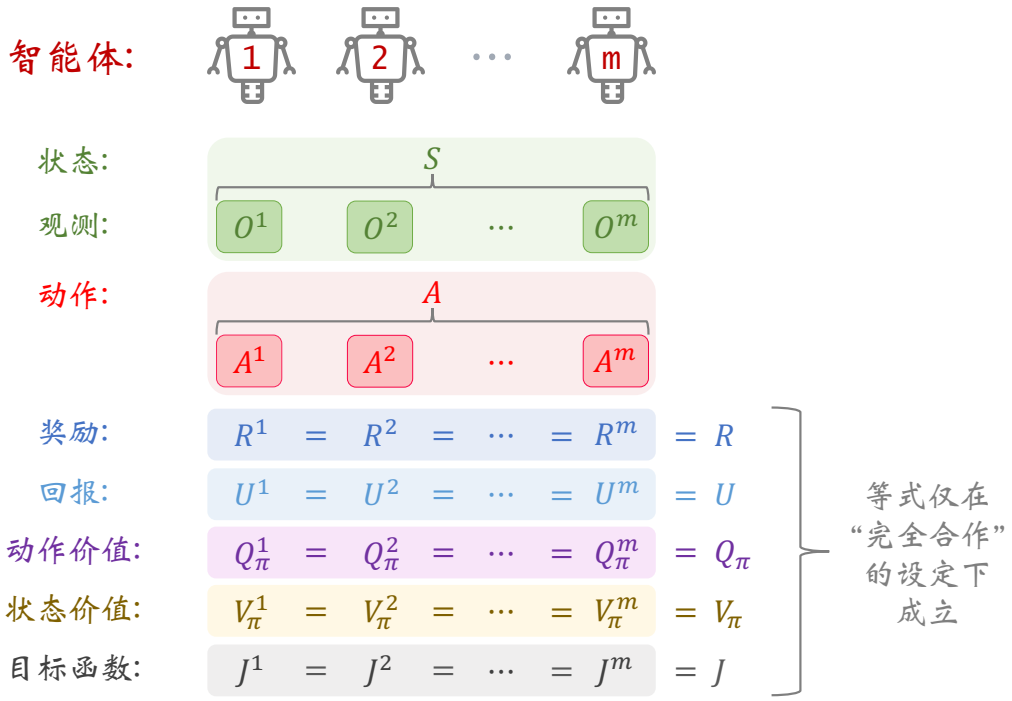

图 14.1: 多智能体强化学习 (MARL) 在“完全合作关系”设定下的符号。

14.1 合作关系设定下的策略学习

MARL 中的**完全合作关系** (Fully-Cooperative) 意思是所有智能体的利益是一致的，它们具有**相同的奖励**：

$$R^1 \;=\; R^2 \;=\; \cdots \;=\; R^m \;\triangleq\; R.$$

因此，所有的智能体都有**相同的回报**：

$$U^1 \;=\; U^2 \;=\; \cdots \;=\; U^m \;\triangleq\; U.$$

因为价值函数都是回报的期望，所以所有的智能体都有**相同的价值函数**。省略上标 i，把动作价值函数记作 $Q_\pi(S,A)$，把状态价值函数记作 $V_\pi(S)$。

注意，价值函数 Q_π 和 V_π 依赖于所有智能体的策略：

$$\pi\big(A^1 \,\big|\, S;\boldsymbol{\theta}^1\big), \qquad \pi\big(A^2 \,\big|\, S;\boldsymbol{\theta}^2\big), \qquad \cdots, \qquad \pi\big(A^m \,\big|\, S;\boldsymbol{\theta}^m\big).$$

举个例子，在某个竞技电游中，玩家组队打任务；每完成一个任务，团队成员（即智能体）获得相同的奖励。所以大家的 R, U, Q_π, V_π 全都是一样的。回报的期望——即价值函数 Q_π 与 V_π——显然与所有成员的策略相关：只要有一个猪队友（即策略差）拖后腿，就有可能导致任务失败。通常来说，团队成员有分工合作，所以每个成员的策略是不同的，即 $\boldsymbol{\theta}^i \neq \boldsymbol{\theta}^j$。

如果做策略学习（即学习策略网络参数 $\boldsymbol{\theta}^1, \cdots, \boldsymbol{\theta}^m$），那么所有智能体都有一个共同目标函数：

$$J\big(\boldsymbol{\theta}^1, \cdots, \boldsymbol{\theta}^m\big) \;=\; \mathbb{E}_S\Big[V_\pi(S)\Big].$$

所有智能体的目的是一致的，即改进自己的策略网络参数 $\boldsymbol{\theta}^i$，使得目标函数 J 增大。那么策略学习可以写作这样的优化问题：

$$\max_{\boldsymbol{\theta}^1, \cdots, \boldsymbol{\theta}^m} J\big(\boldsymbol{\theta}^1, \cdots, \boldsymbol{\theta}^m\big). \tag{14.1}$$

注意，只有“完全合作关系”这一种设定下，所有智能体才有共同的目标函数，其原因在于 $R^1 = \cdots = R^m$。对于其它设定——“竞争关系”、“混合关系”、“利己主义”——智能体的目标函数是各不相同的（见下一章）。

合作关系设定下的策略学习的原理很简单，即让每个智能体各自做策略梯度上升，使得目标函数 J 增长。

$$\begin{aligned}
&\text{第 1 号智能体执行：} && \boldsymbol{\theta}^1 \;\leftarrow\; \boldsymbol{\theta}^1 \;+\; \alpha^1 \cdot \nabla_{\boldsymbol{\theta}^1} J\big(\boldsymbol{\theta}^1, \cdots, \boldsymbol{\theta}^m\big),\\
&\text{第 2 号智能体执行：} && \boldsymbol{\theta}^2 \;\leftarrow\; \boldsymbol{\theta}^2 \;+\; \alpha^2 \cdot \nabla_{\boldsymbol{\theta}^2} J\big(\boldsymbol{\theta}^1, \cdots, \boldsymbol{\theta}^m\big),\\
&\quad\vdots && \quad\vdots\\
&\text{第 } m \text{ 号智能体执行：} && \boldsymbol{\theta}^m \;\leftarrow\; \boldsymbol{\theta}^m \;+\; \alpha^m \cdot \nabla_{\boldsymbol{\theta}^m} J\big(\boldsymbol{\theta}^1, \cdots, \boldsymbol{\theta}^m\big).
\end{aligned}$$

公式中的 $\alpha^1, \alpha^2, \cdots, \alpha^m$ 是学习率。判断策略学习收敛的标准是目标函数 $J\big(\boldsymbol{\theta}^1, \cdots, \boldsymbol{\theta}^m\big)$ 不再增长。在实践中，当平均回报不再增长，即可终止算法。由于无法直接计算策略梯度 $\nabla_{\boldsymbol{\theta}^i} J$，我们需要对其做近似。下一节用价值网络近似策略梯度，从而推导出一种实际可行的策略梯度方法。

14.2 合作设定下的多智能体 A2C

第 8 章介绍过 Advantage Actor-Critic (A2C) 方法。本节介绍“完全合作关系”设定下的多智能体 A2C 方法 (Multi-Agent Cooperative A2C)，缩写 MAC-A2C。注意，本节介绍的方法仅适用于“完全合作关系”，也就是要求所有智能体有相同的奖励：$R^1 = \cdots = R^m$。第 14.2.1 小节定义策略网络和价值网络。第 14.2.3 小节描述 MAC-A2C 训练和决策。第 14.2.4 小节讨论 MAC-A2C 实现中的难点。

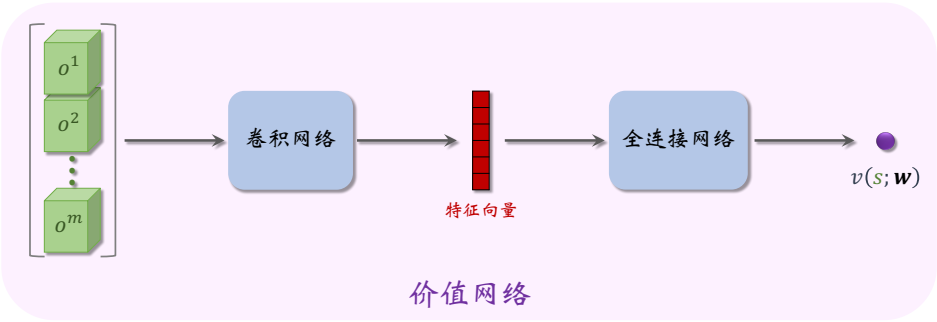

图 14.2: MAC-A2C 中的价值网络 $v(s;\boldsymbol{w})$；所有智能体共用这个价值网络。输入是所有智能体的观测：$s = [o^1, \cdots, o^m]$。输出是价值网络给 s 的评分。

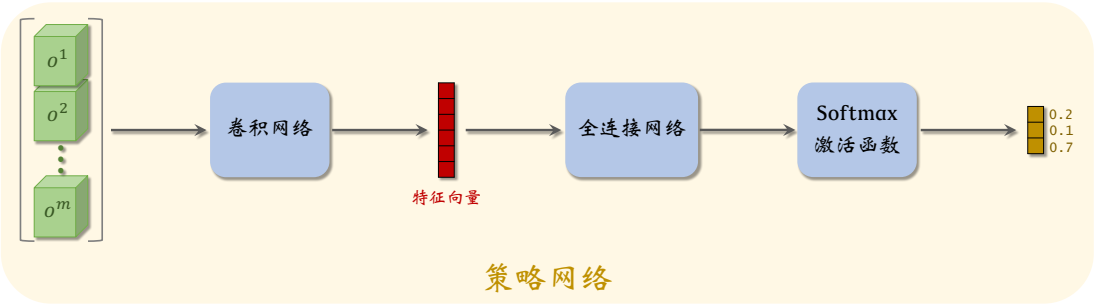

图 14.3: MAC-A2C 中第 i 号智能体的策略网络 $\pi(\cdot|s;\boldsymbol{\theta}^i)$。所有智能体的策略网络结构都一样，但是参数 $\boldsymbol{\theta}^1, \cdots, \boldsymbol{\theta}^m$ 可能不一样。输入是所有智能体的观测：$s = [o^1, \cdots, o^m]$。输出是在离散动作空间 $\mathcal{A}^i$ 上的概率分布。

14.2.1 策略网络和价值网络

本章只考虑离散控制问题，即动作空间 $\mathcal{A}^1, \cdots, \mathcal{A}^m$ 都是离散集合。MAC-A2C 使用**两类神经网络**：价值网络 v 与策略网络 π；见图 14.2、图 14.3。

所有智能体共用一个价值网络，记作 $v(s;\boldsymbol{w})$，它是对状态价值函数 $V_\pi(s)$ 的近似。它把所有观测 $s = [o^1, \cdots, o^m]$ 作为输入，并输出一个实数，作为对状态 s 的评分。

每个智能体有自己的策略网络。 把第 i 号策略网络记作 $\pi(a^i|s;\boldsymbol{\theta}^i)$。它的输入是所有智能体的观测 $s = [o^1, \cdots, o^m]$。它的输出是一个向量，表示动作空间 $\mathcal{A}^i$ 上的概率分布。比如，第 i 号智能体的动作空间是 $\mathcal{A}^i = \{\text{左}, \text{右}, \text{上}\}$；策略网络的输出是

$$\pi\big(\text{左}\,\big|\,s;\boldsymbol{\theta}^i\big) = 0.2, \qquad \pi\big(\text{右}\,\big|\,s;\boldsymbol{\theta}^i\big) = 0.1, \qquad \pi\big(\text{上}\,\big|\,s;\boldsymbol{\theta}^i\big) = 0.7.$$

第 i 号智能体依据该概率分布抽样得到动作 a^i。

MAC-A2C 属于 Actor-Critic 方法：策略网络 $\pi(A^i|S;\boldsymbol{\theta}^i)$ 相当于第 i 个运动员，负责做决策；价值网络 $v(S;\boldsymbol{w})$ 相当于评委，对运动员团队的整体表现予以评价，反馈给整个团队一个分数。

14.2.2 算法推导

训练价值网络： 我们用 TD 算法训练训练价值网络 $v(s;\boldsymbol{w})$。观测到状态 s_t、s_{t+1} 和奖励 r_t，计算 TD 目标：

$$\widehat{y}_t \;=\; r_t \;+\; \gamma \cdot v\big(s_{t+1};\boldsymbol{w}\big).$$

把 $\widehat{y}_t$ 视作常数，更新 $\boldsymbol{w}$ 使得 $v(s_t;\boldsymbol{w})$ 接近 $\widehat{y}_t$。定义损失函数：

$$L\big(\boldsymbol{w}\big) \;=\; \frac{1}{2}\Big[\,v\big(s_t;\boldsymbol{w}\big) \;-\; \widehat{y}_t\Big]^2.$$

损失函数的梯度等于：

$$\nabla_{\boldsymbol{w}} L\big(\boldsymbol{w}\big) \;=\; \delta_t \;\cdot\; \nabla_{\boldsymbol{w}}\, v\big(s_t;\boldsymbol{w}\big),$$

其中 $\delta_t = v(s_t;\boldsymbol{w}) - \widehat{y}_t$ 是 TD 误差。做一次梯度下降更新 $\boldsymbol{w}$：

$$\boldsymbol{w} \;\leftarrow\; \boldsymbol{w} \;-\; \alpha \;\cdot\; \delta_t \;\cdot\; \nabla_{\boldsymbol{w}}\, v\big(s_t;\boldsymbol{w}\big).$$

这样可以减小损失函数，也就是让 $v(s_t;\boldsymbol{w})$ 接近 $\widehat{y}_t$。上述 TD 算法与单智能体 A2C 的 TD 算法完全一样。

训练策略网络： 完全合作关系设定下的动作价值函数记作 $Q_\pi(s,a)$，第 i 号智能体的策略网络为 $\pi\big(A^i \,\big|\, S;\boldsymbol{\theta}^i\big)$。不难证明下面的策略梯度定理（见习题 1）：

定理 14.1. 合作关系 MARL 的策略梯度定理

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

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

期望中的动作 A 的概率密度函数为

$$\pi\big(A \,\big|\, S;\boldsymbol{\theta}^1,\cdots,\boldsymbol{\theta}^m\big) \;\triangleq\; \pi\big(A^1 \,\big|\, S;\boldsymbol{\theta}^1\big) \;\times\; \cdots \;\times\; \pi\big(A^m \,\big|\, S;\boldsymbol{\theta}^m\big).$$

♡

把基线设置为状态价值：$b = V_\pi(s)$。定义

$$\boldsymbol{g}^i\big(s,a;\,\boldsymbol{\theta}^i\big) \;\triangleq\; \Big(\,Q_\pi\big(s,a\big) \;-\; V_\pi\big(s\big)\,\Big) \;\cdot\; \nabla_{\boldsymbol{\theta}^i}\, \pi\big(a^i \,\big|\, s;\,\boldsymbol{\theta}^i\big).$$

定理 14.1 说明 $\boldsymbol{g}^i\big(s,a^i;\boldsymbol{\theta}^i\big)$ 是策略梯度的无偏估计：

$$\nabla_{\boldsymbol{\theta}^i} J\Big(\,\boldsymbol{\theta}^1,\,\cdots,\,\boldsymbol{\theta}^m\,\Big) \;=\; \mathbb{E}_{S,A}\Big[\,\boldsymbol{g}^i\big(S,A;\,\boldsymbol{\theta}^i\big)\,\Big].$$

因此 $\boldsymbol{g}^i\big(s,a^i;\boldsymbol{\theta}^i\big)$ 可以作为策略梯度的近似。但是我们不知道公式中的 Q_π、V_π，还需要进一步做近似。根据第 8.3 节 A2C 的推导，我们把 $Q_\pi(s_t,a_t)$ 近似成 $r_t+\gamma\cdot v(s_{t+1};\boldsymbol{w})$，

把 $V_\pi(s_t)$ 近似成 $v(s_t;\boldsymbol{w})$。那么近似策略梯度 $\boldsymbol{g}^i(s_t,a_t;\boldsymbol{\theta}^i)$ 可以进一步近似成：

$$\tilde{\boldsymbol{g}}^i\big(s_t,a_t^i;\boldsymbol{\theta}^i\big) \triangleq \Big(\underbrace{r_t+\gamma\cdot v\big(s_{t+1};\boldsymbol{w}\big)-v\big(s_t;\boldsymbol{w}\big)}_{\text{对 } Q_\pi(s_t,a_t)-V_\pi(s_t) \text{ 的近似}}\Big)\cdot\nabla_{\boldsymbol{\theta}^i}\pi\big(a_t^i\,\big|\,s_t;\boldsymbol{\theta}^i\big).$$

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

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

根据 TD 误差 δ_t 的定义，不难看出 $\tilde{\boldsymbol{g}}^i\big(s_t,a_t^i;\boldsymbol{\theta}^i\big)=-\delta_t\cdot\nabla_{\boldsymbol{\theta}^i}\pi\big(a_t^i\,\big|\,s_t;\boldsymbol{\theta}^i\big)$。因此，上面更新策略网络参数的公式可以写作：

$$\boldsymbol{\theta}^i \leftarrow \boldsymbol{\theta}^i-\beta\cdot\delta_t\cdot\nabla_{\boldsymbol{\theta}^i}\pi\big(a_t^i\,\big|\,s_t;\boldsymbol{\theta}^i\big).$$

14.2.3 训练和决策

训练： 实际实现的时候，应当使用目标网络缓解自举造成的偏差。目标网络记作 $v(s;\boldsymbol{w}^-)$，它的结构与 v 相同，但是参数不同。设当前价值网络和目标网络的参数分别是 $\boldsymbol{w}_{\text{now}}$ 和 $\boldsymbol{w}_{\text{now}}^-$。设当前 m 个策略网络的参数分别是 $\boldsymbol{\theta}_{\text{now}}^1,\cdots,\boldsymbol{\theta}_{\text{now}}^m$。MAC-A2C 重复下面的步骤更新参数：

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

$$a_t^i\sim\pi\big(\cdot\,\big|\,s_t;\boldsymbol{\theta}_{\text{now}}^i\big),\qquad\forall i=1,\cdots,m,$$

   并执行选中的动作。
2. 从环境中观测到奖励 r_t 与下一时刻状态 $s_{t+1}=\big[o_{t+1}^1,\cdots,o_{t+1}^m\big]$。
3. 让价值网络做预测：$\widehat{v}_t=v\big(s_t;\boldsymbol{w}_{\text{now}}\big)$。
4. 让目标网络做预测：$\widehat{v}_{t+1}^-=v\big(s_{t+1};\boldsymbol{w}_{\text{now}}^-\big)$。
5. 计算 TD 目标与 TD 误差：

$$\widehat{y}_t^-=r_t+\gamma\cdot\widehat{v}_{t+1}^-,\qquad\delta_t=\widehat{v}_t-\widehat{y}_t^-.$$

6. 更新价值网络参数：

$$\boldsymbol{w}_{\text{new}}\leftarrow\boldsymbol{w}_{\text{now}}-\alpha\cdot\delta_t\cdot\nabla_{\boldsymbol{w}}v\big(s_t;\boldsymbol{w}_{\text{now}}\big).$$

7. 更新目标网络参数：

$$\boldsymbol{w}_{\text{new}}^-\leftarrow\tau\cdot\boldsymbol{w}_{\text{new}}+(1-\tau)\cdot\boldsymbol{w}_{\text{now}}^-.$$

8. 更新策略网络参数：

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

MAC-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 $v(s;\boldsymbol{w})$。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 $s_t=\big[o_t^1,\cdots,o_t^m\big]$，然后做随机抽样得到动作：

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

并执行动作。注意，智能体并不能独立做决策，因为一个智能体的策略网络需要知道其

他所有智能体的观测。

14.2.4 实现中的难点

上述 MAC-A2C 的训练和决策貌似简单，然而实际的实现却不容易。在 MARL 的常见设定下，第 i 号智能体只知道 o^i，而观测不到全局状态:

$$s = [o^1, \cdots, o^m].$$

这会给决策和训练造成如下的困难:

- 每个智能体有自己的策略网络 $\pi(a^i|s;\boldsymbol{\theta}^i)$，可以依靠它做**决策**。但是它的决策需要全局状态 s。
- 在**训练**的过程中，价值网络 $v(s;\boldsymbol{w})$ 需要知道全局状态 s 才能计算 TD 误差 δ 与梯度 $\nabla_{\boldsymbol{w}} v(s;\boldsymbol{w})$。
- 在**训练**的过程中，每个策略网络都需要知道全局状态 s 来计算梯度 $\nabla_{\boldsymbol{\theta}^i} \ln \pi(a^i|s;\boldsymbol{\theta}^i)$。

综上所述，如果智能体之间不交换信息，那么智能体既无法做训练，也无法做决策。想要做训练和决策，有两种可行的途径:

- 一种办法是让智能体**共享观测**。这需要做通信，每个智能体把自己的 o^i 传输给其他智能体。这样每个智能体都有全局的状态 $s = [o^1, \cdots, o^m]$。
- 另一种办法是对策略网络和价值函数**做近似**。通常使用 $\pi(a^i|o^i;\boldsymbol{\theta}^i)$ 替代 $\pi(a^i|s;\boldsymbol{\theta}^i)$。甚至可以进一步用 $v(o^i;\boldsymbol{w}^i)$ 代替 $v(s;\boldsymbol{w})$。

共享观测的缺点在于通信会让训练和决策的速度变慢。而做近似的缺点在于不完全信息造成训练不收敛、做出错误决策。我们不得不在两种办法之间做出取舍，承受其造成的不良影响。

下一节介绍**中心化** (Centralized) 与**去中心化** (Decentralized) 的实现方法。中心化会是让智能体共享信息；其优点是训练和决策的效果好，缺点是需要通信，造成延时，影响速度。去中心化的意思是做近似，避免通信；其优点在于速度快，而缺点则是影响训练和决策的质量。

14.3 三种架构

本节介绍 MAC-A2C 的三种实现方法。第 14.3.1 节介绍 “**中心化训练 + 中心化决策**” (Centralized Training with Centralized Execution)，它是对 MAC-A2C 的忠实实现，训练和决策都需要通信。第 14.3.2 节介绍 “**去中心化训练 + 去中心化决策**” (Decentralized Training with Decentralized Execution)，它对策略网络和价值网络都做近似，以避免训练和决策的通信。第 14.3.3 节介绍 “**中心化训练 + 去中心化决策**” (Centralized Training with Decentralized Execution)，它只近似策略网络，以避免决策的通信。

图 14.4 对比了三种架构的策略网络和价值网络。用 “完全中心化” 作出的决策最好，但是速度最慢，在很多问题中不适用。“中心化训练 + 去中心化决策” 虽然训练有通信代价，但是实际决策的时候不需要通信，可以做到实时决策，因此是当前最常用的架构，

	价值网络	策略网络	训练	决策
中心化训练 + 中心化决策	$v(s;\mathbf{w})$	$\pi(a^i \mid s;\boldsymbol{\theta}^i)$	需要通信	需要通信
去中心化训练 + 去中心化决策	$v(o^i;\mathbf{w}^i)$	$\pi(a^i \mid o^i;\boldsymbol{\theta}^i)$	无需通信	无需通信
中心化训练 + 去中心化决策	$v(s;\mathbf{w})$	$\pi(a^i \mid o^i;\boldsymbol{\theta}^i)$	需要通信	无需通信

图 14.4: 三种架构的对比。

14.3.1 中心化训练 + 中心化决策

本节用完全中心化 (Fully Centralized) 的方式实现 MAC-A2C，没有做任何近似。这种实现的缺点在于通信造成延时，使得训练和决策速度变慢。图 14.5 描述了系统的架构。最上面是中央控制器 (Central Controller)，里面部署了价值网络 $v(s;\boldsymbol{w})$ 与所有 m 个策略网络

$$\pi(a^1 \,|\, \boldsymbol{\theta}^1), \qquad \pi(a^2 \,|\, \boldsymbol{\theta}^2), \qquad \cdots, \qquad \pi(a^m \,|\, \boldsymbol{\theta}^m).$$

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 汇报给中央控制器。(如果智能体观测到奖励 r^i，也发给中央控制器。)

中心化训练： 在时刻 t 和 $t+1$，中央控制器收集到所有智能体的观测值

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

在 “完全合作关系” 的设定下，所有智能体有相同的奖励：

$$r_t^1 = r_t^2 = \cdots = r_t^m \triangleq r_t.$$

r_t 可以是中央控制器直接从环境中观测到的，也可能是基于所有智能体本地观测的奖励 $\tilde{r}_t^i$ 的加和：

$$r_t = \tilde{r}_t^1 + \tilde{r}_t^2 + \cdots + \tilde{r}_t^m.$$

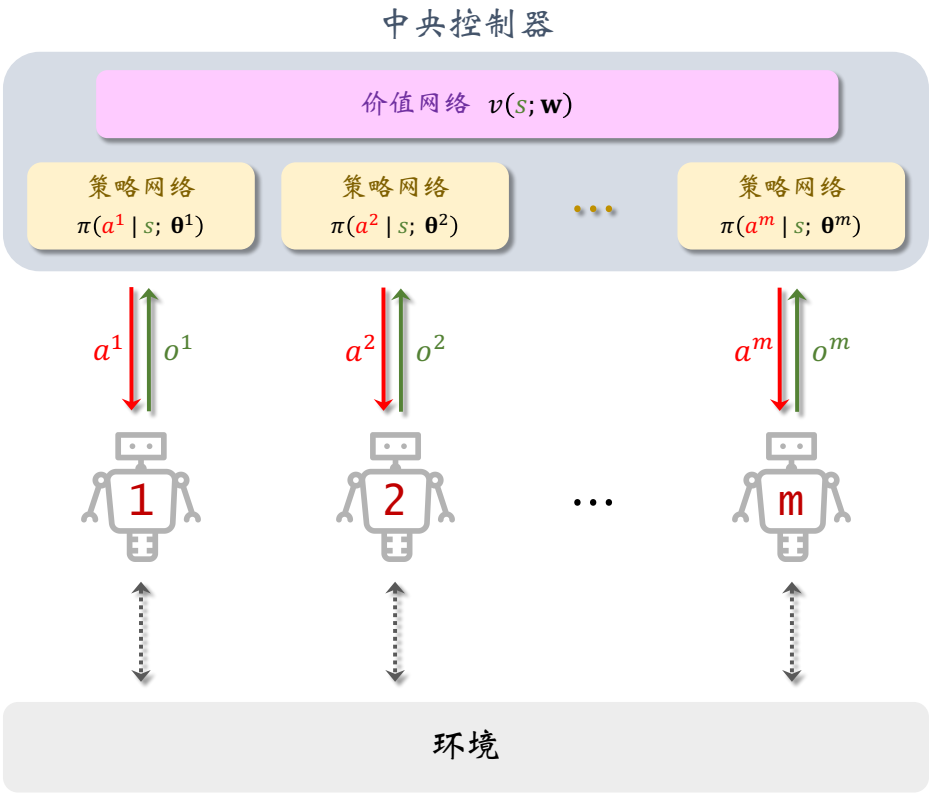

图 14.5: 中心化训练 + 中心化决策的系统架构。

决策是中央控制器上的策略网络做出的，中央控制器自然知道所有的动作：

$$a_t = [a_t^1, \cdots, a_t^m].$$

综上所述，中央控制器知道如下信息：

$$s_t,\ s_{t+1},\ a_t,\ r_t.$$

因此，中央控制器有足够的信息按照第 14.2.3 小节中的算法训练 MAC-A2C，更新价值网络和策略网络的参数：

$$\boldsymbol{w},\ \boldsymbol{\theta}^1,\ \boldsymbol{\theta}^2,\ \boldsymbol{\theta}^3,\ \cdots, \boldsymbol{\theta}^m.$$

中心化决策： 在 t 时刻，中央控制器收集到所有智能体的观测值 $s_t = [o_t^1, \cdots, o_t^m]$，然后用中央控制器上部署的策略网络做决策：

$$a_t^i \sim \pi\big(\cdot \,\big|\, s_t; \boldsymbol{\theta}^i\big), \qquad \forall\, i = 1, \cdots, m.$$

中央控制器把决策 a_t^i 传达给第 i 号智能体，该智能体执行 a_t^i。综上所述，智能体只需要执行按照中央下达的决策，而不需要自己“思考”。其原因在于策略函数 π 需要全局的状态 s_t 作为输入，而单个智能体不知道全局状态，没有能力单独做决策。

优缺点： 中心化训练 + 中心化决策的优点在于完全按照 MAC-A2C 的算法实现，没有做任何改动，因此可以确保正确性。中心化训练和决策的缺点在于延迟 (Latency) 很大，影响训练和决策的速度。在中心化执行的框架下，智能体与中央控制器要做通信。第 i 号

智能体要把 o_t^i 传输给中央控制器，而控制器要在收集到所有观测 $[o_t^1, \cdots, o_t^m]$ 之后才会做决策，做出的决策 a_t^i 还得传输给第 i 号智能体。这个过程通常比较慢，使得实时决策不可能做到。机器人、无人车、无人机等应用都需要实时决策，比如在几十毫秒内做出决策；如果出现几百毫秒、甚至几秒的延迟，可能会造成灾难性的后果。

14.3.2 去中心化训练 + 去中心化决策

上一小节的“中心化训练 + 中心化决策”严格按照 MAC-A2C 的算法实现，其缺点在于训练和决策都需要智能体与中央控制器之间通信，造成训练的决策的速度慢。想要避免通信代价，就不得不对策略网络和价值网络做近似。MAC-A2C 中的策略网络

$$\pi\big(a^1 \,\big|\, s; \boldsymbol{\theta}^1\big), \qquad \pi\big(a^2 \,\big|\, s; \boldsymbol{\theta}^2\big), \qquad \cdots, \qquad \pi\big(a^m \,\big|\, s; \boldsymbol{\theta}^m\big),$$

和价值网络 $v(s; \boldsymbol{w})$ 都需要全局的状态（观测）$s = [o^1, \cdots, o^m]$。“去中心化训练 + 去中心化决策”的基本思想是用局部观测 o^i 代替全局状态 s，把策略网络和价值网络近似成为：

$$\pi\big(a^i \,\big|\, o^i; \boldsymbol{\theta}^i\big) \qquad \text{和} \qquad v\big(o^i; \boldsymbol{w}^i\big).$$

在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 $\boldsymbol{\theta}^i$ 和 $\boldsymbol{w}^i$。智能体之间不共享参数，即 $\boldsymbol{\theta}^i \neq \boldsymbol{\theta}^j$，$\boldsymbol{w}^i \neq \boldsymbol{w}^j$。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，无需任何通信。见图 14.5 中的系统架构。

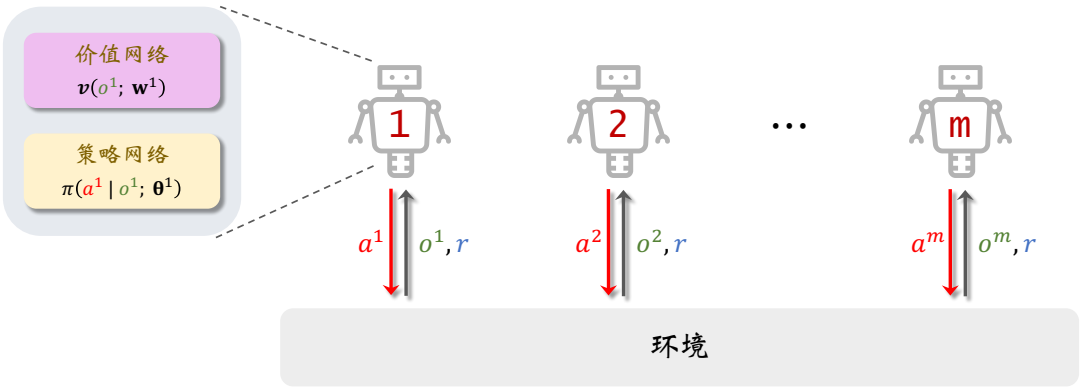

图 14.6: 去中心化训练 + 去中心化决策的系统架构。这种方法也叫做 Independent Actor-Critic。

去中心化训练： 假设所有智能体的奖励都是相同的，而且每个智能体都能观测到奖励 r。每个智能体独立做训练，智能体之间不做通信，不共享观测、动作、参数。这样一来，MAC-A2C 就变成了标准的 A2C，每个智能体独立学习自己的参数 $\boldsymbol{\theta}^i$ 与 $\boldsymbol{w}^i$。

实际实现的时候，每个智能体还需要一个目标网络，记作 $v(s; \boldsymbol{w}^{i-})$，它的结构与 $v(s; \boldsymbol{w}^i)$ 相同，但是参数不同。设第 i 号智能体的策略网络、价值网络、目标网络当前参数分别为 $\boldsymbol{\theta}^i_{\text{now}}$、$\boldsymbol{w}^i_{\text{now}}$、$\boldsymbol{w}^{i-}_{\text{now}}$。该智能体重复以下步骤更新参数：

1. 在 t 时刻，智能体 i 观测到 o_t^i，然后做随机抽样 $a_t^i \sim \pi(\cdot \,|\, o_t^i; \boldsymbol{\theta}^i)$，并执行选中的动作 a_t^i。
2. 环境反馈给智能体奖励 r_t 与新的观测 o_{t+1}^i。

3. 让价值网络做预测：$\widehat{v}_t^i = v\big(o_t^i; \boldsymbol{w}_{\text{now}}^i\big)$。
4. 让目标网络做预测：$\widehat{v}_{t+1}^i = v\big(o_{t+1}^i; \boldsymbol{w}_{\text{now}}^{i-}\big)$。
5. 计算 TD 目标与 TD 误差：

$$\widehat{y}_t^i \;=\; r_t + \gamma \cdot \widehat{v}_{t+1}^i, \qquad \delta_t^i \;=\; \widehat{v}_t^i - \widehat{y}_t^i.$$

6. 更新价值网络参数：

$$\boldsymbol{w}_{\text{new}}^i \;\leftarrow\; \boldsymbol{w}_{\text{now}}^i \;-\; \alpha \cdot \delta_t^i \cdot \nabla_{\boldsymbol{w}^i} v\big(o_t^i; \boldsymbol{w}_{\text{now}}^i\big).$$

7. 更新目标网络参数：

$$\boldsymbol{w}_{\text{new}}^{i-} \;\leftarrow\; \tau \cdot \boldsymbol{w}_{\text{new}}^i \;+\; (1-\tau) \cdot \boldsymbol{w}_{\text{now}}^{i-}.$$

8. 更新策略网络参数：

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

注 上述算法不是 MAC-A2C，而是单智能体的 A2C。去中心化训练的本质就是单智能体强化学习 (SARL)，而非多智能体强化学习 (MARL)。在 MARL 中，智能体之间会相互影响，而本节中的“去中心化训练”把智能体视为独立个体，忽视它们之间的关联，直接用 SARL 方法独立训练每个智能体。用上述 SARL 的方法解决 MARL 问题，在实践中效果往往不佳。

去中心化决策： 在完成训练之后，智能体 i 不再需要其价值网络 $v(o^i; \boldsymbol{w}^i)$。智能体只需要用其本地部署的策略网络 $\pi(a^i|o^i; \boldsymbol{\theta}^i)$ 做决策即可，决策过程无需通信。去中心化执行的速度很快，可以做到实时决策 (Real-Time Decision)。

14.3.3 中心化训练 + 去中心化决策

前面两节讨论了完全中心化与完全去中心化，两种实现各有优缺点。当前更流行的 MARL 架构是“中心化训练 + 去中心化决策”。训练的时候使用中央控制器，辅助智能体做训练；见图 14.7。训练结束之后，不再需要中央控制器，每个智能体独立根据局部观测 o^i 做决策；见图 14.8。

本小节与“完全中心化”使用相同的价值网络 $v(s; \boldsymbol{w})$ 及其目标网络 $v(s; \boldsymbol{w}^-)$；本节与“完全去中心化”使用相同的策略网络：

$$\pi\big(a^1 \,\big|\, o^1; \boldsymbol{\theta}^1\big), \qquad \cdots, \qquad \pi\big(a^m \,\big|\, o^m; \boldsymbol{\theta}^m\big).$$

第 i 号策略网络的输入是局部观测 o^i，因此可以将其部署到第 i 号智能体上。价值网络 $v(s; \boldsymbol{w})$ 的输入是全局状态 $s = [o^1, \cdots, o^m]$，因此需要将其部署到中央控制器上。

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 $\boldsymbol{\theta}^1, \cdots, \boldsymbol{\theta}^m$ 与价值网络参数 $\boldsymbol{w}$。设当前 m 个策略网络的参数为 $\boldsymbol{\theta}_{\text{now}}^1, \cdots, \boldsymbol{\theta}_{\text{now}}^m$. 设当前价值网络和目标网络的参数分别是 $\boldsymbol{w}_{\text{now}}$ 和 $\boldsymbol{w}_{\text{now}}^-$。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：

$$a_t^i \;\sim\; \pi\big(\cdot \,\big|\, o_t^i; \boldsymbol{\theta}_{\text{now}}^i\big), \qquad \forall\, i = 1, \cdots, m, \tag{14.2}$$

并执行选中的动作。

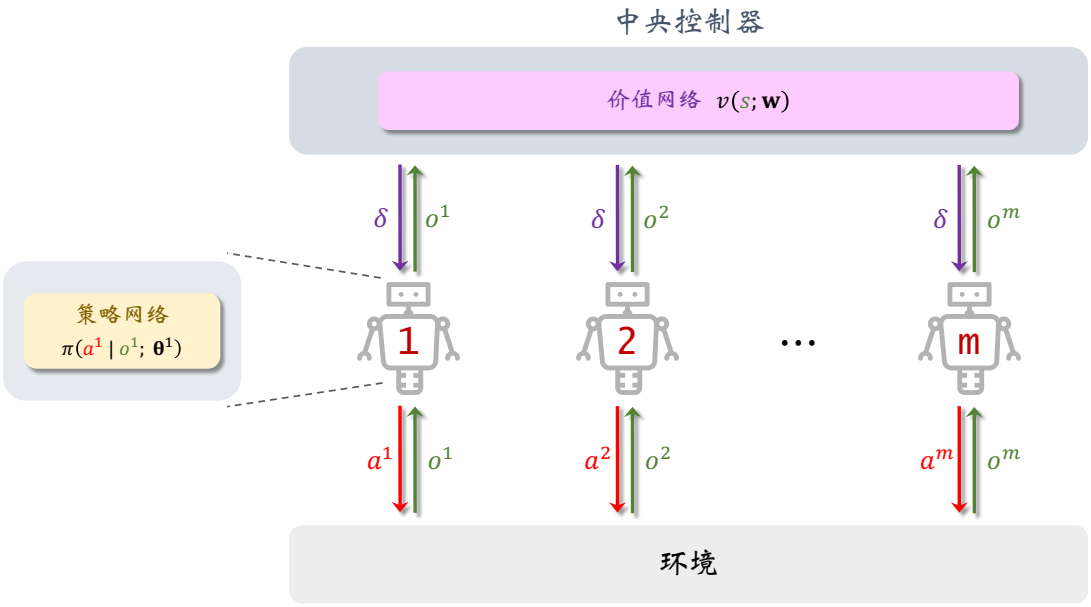

图 14.7: 中心化训练的系统架构。价值网络（以及没画出的目标网络）部署到中央控制器上，策略网络部署到每个智能体上。训练的时候，智能体 i 将观测 o^i 传输到控制器上，控制器将 TD 误差 δ 传回智能体。

2. 下一时刻，每个智能体 i 都观测到 o_{t+1}^i。假设中央控制器可以从环境获取奖励 r_t，或者向智能体询问奖励 r_t。
3. 每个智能体 i，向中央控制器传输观测 o_t^i 和 o_{t+1}^i；中央控制器得到状态

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

4. 中央控制器让价值网络做预测：$\widehat{v}_t = v(s_t; \boldsymbol{w}_{\text{now}})$。
5. 中央控制器让目标网络做预测：$\widehat{v}_{t+1}^- = v(s_{t+1}; \boldsymbol{w}_{\text{now}}^-)$。
6. 中央控制器计算 TD 目标和 TD 误差：

$$\widehat{y}_t^- = r_t + \gamma \cdot \widehat{v}_{t+1}^-, \qquad \delta_t = \widehat{v}_t - \widehat{y}_t^-,$$

并将 δ_t 广播到所有智能体。

7. 中央控制器更新价值网络参数：

$$\boldsymbol{w}_{\text{new}} \leftarrow \boldsymbol{w}_{\text{now}} - \alpha \cdot \delta_t \cdot \nabla_{\boldsymbol{w}} v(s_t; \boldsymbol{w}_{\text{now}}).$$

8. 中央控制器更新目标网络参数：

$$\boldsymbol{w}_{\text{new}}^- \leftarrow \tau \cdot \boldsymbol{w}_{\text{new}} + (1-\tau) \cdot \boldsymbol{w}_{\text{now}}^-.$$

9. 每个智能体 i 更新策略网络参数：

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

注 此处的算法并不等价于第 14.2 节的 MAC-A2C。区别在于此处用 $\pi(a^i|o^i;\boldsymbol{\theta}^i)$ 代替 MAC-A2C 中的 $\pi(a^i|s;\boldsymbol{\theta}^i)$.

去中心化决策： 在完成训练之后，不再需要价值网络 $v(s;\boldsymbol{w})$。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。去中心化执行的速度很快，

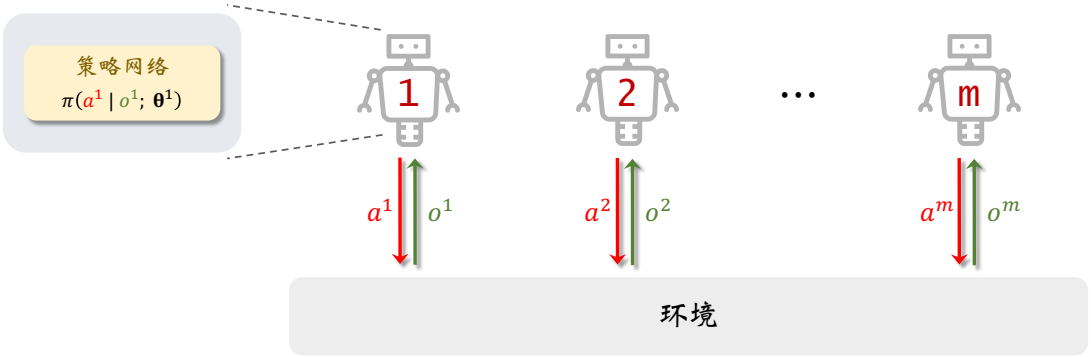

图 14.8: 去中心化决策的系统架构。在完成训练之后，智能体不再做通信，智能体用本地部署的策略网络做决策。

可以做到实时决策。

第十四章习题

1. 设动作 $A=[A^1,\cdots,A^m]$ 的概率密度函数为

$$\pi\big(A\,\big|\,S;\boldsymbol{\theta}^1,\cdots,\boldsymbol{\theta}^m\big)\ \triangleq\ \pi\big(A^1\,\big|\,S;\boldsymbol{\theta}^1\big)\ \times\ \cdots\ \times\ \pi\big(A^m\,\big|\,S;\boldsymbol{\theta}^m\big).$$

由第 8 章中带基线的策略梯度定理可得：

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

公式中动作 A 的概率密度函数为 $\pi\big(A\,\big|\,S;\boldsymbol{\theta}^1,\cdots,\boldsymbol{\theta}^m\big)$，公式中的 b 是任意不依赖于 A 的函数。请用上面两个公式证明下面的公式：

$$\nabla_{\boldsymbol{\theta}^i} J\big(\boldsymbol{\theta}^1,\cdots,\boldsymbol{\theta}^m\big)\ =\ \mathbb{E}_{S,A}\Big[\Big(Q_\pi(S,A)-V_\pi(S)\Big)\cdot\nabla_{\boldsymbol{\theta}^i}\ln\pi\big(A^i\,\big|\,S;\boldsymbol{\theta}^i\big)\Big].$$

相关文献

完全去中心化的架构早在 1993 年就被提出 [102]，在 2017 年被用在多智能体 DQN 上 [39, 101]。中心化训练 + 去中心化执行 (Centralized Training with Decentralized Execution) 在近年来很流行 [78, 43, 38, 66, 53]。

MAC-A2C 是本书设计出来的简单方法，用于讲解 MARL 的三种架构；MAC-A2C 这个名字并没有出现在任何文献中。MAC-A2C 本质是带基线的 Actor-Critic，其中的基线是状态价值

$$V_\pi(s) \triangleq \mathbb{E}_A\Big[Q_\pi(s, A)\Big],$$

期望是关于动作 $A = [A^1, \cdots, A^m]$ 求的。可以把基线换成

$$Q_\pi^{-i}(s, a^{-i}) \triangleq \mathbb{E}_{A^i}\Big[Q_\pi(s, A^i, a^{-i})\Big],$$

公式中 $a^{-i} = [a^1, \cdots, a^{i-1}, a^{i+1}, \cdots, a^m]$。公式中的期望是关于第 i 号智能体的动作 $A^i \sim \pi(\cdot\,|\,o^i, \boldsymbol{\theta}^i)$ 求的。用 $Q_\pi^{-i}(s, a^{-i})$ 作为基线，代替 $V_\pi(s)$，得到的方法叫做 COunterfactual Multi-Agent，缩写 COMA [38]。此外，COMA 还在策略网络中使用 RNN；其原理见第 11 章的解释。COMA 的表现略好于 MAC-A2C，但是 COMA 的实现很复杂，不建议读者自己实现。