多 Agent 系统的协作及强化学习算法研究*

郑淑丽 韩江洪 骆祥峰 蒋建文

(合肥工业大学 计算机学院 合肥 230009)

摘 要 研究了多 Agent 环境下的协作与学习.对多 Agent 系统中的协作问题提出了协作模型 MACM,该模型通过提供灵活协调机制支持多 Agent 之间的协作及协作过程中的学习.系统中的学习 Agent 采用分布式强化学习算法.该算法通过映射减少 Q 值表的存储空间,降低对系统资源的要求,同时能够保证收敛到最优解.

关键词 多 Agent 系统,协作与协调,强化学习
中图法分类号 TP18

1 引 言

多 Agent 协作是分布式人工智能领域的重要研究课题,同时也是多 Agent 系统技术的重要问题,目前,对多 Agent 系统的研究主要集中在 Agent 之间的交互上.在多 Agent 系统中,每个 Agent 可以被视为是一个自治的实体(如软件程序或机器人).Agent 之间的交互可以是协作关系(即多个 Agent 拥有共同的目标)也可以是自私关系(每个 Agent 追求自身利益),协作是区分多 Agent 系统与分布式计算,面向对象系统以及专家系统的关键概念,在多 Agent 系统中如何进行有效的协作与协调是一个难以实现的问题[1].理论分析表明,如果在协作多 Agent 系统中引入学习机制,使得每个 Agent 通过学习协调自身的行为,则能有效的完成共同目标.因此,协作过程中的学习得到了越来越多研究人员的重视.

对策论的研究者对学习在协作过程中的作用进行了广泛的研究,例如 fictitious play[5] 以及贝叶斯最佳响应方法[6]在简单的对策环境下取得了较好的协调效果.但是,这些模型假设 Agent 在交互过程中能够完全观测到其他 Agent 所采取的动作,而在实际应用中,由于环境的动态性以及动作的随机性,该假设条件通常是不成立的.

本文研究了动态环境下的多 Agent 协作、协调与学习问题.在协作多 Agent 系统中,通常存在三种协调方法:基于通讯的协调;基于社会法则的协调以及基于学习的协调[2].前两种协调方式都存在着一定的局限性.本文将基于学习的协调方法与规划机制相结合,建立了规划协调-学习协调共同作用的多 Agent 协作模型 MACM.在该模型中,Agent 以强化学习(本文以 Q-学习为例)作为在线学习机制进行协作与协调.由于在大规模复杂问题求解过程中,随着参与协作的 Agent 数目以及环境状态的增加,Q 值表的大小将呈指数级递增,为了降低了 Q-学习算法对系统计算资源的要求,本文采用了分布式强化学习算法,在假设 Agent 是乐观的条件下,将系统的联合 Q 值表空间映射到维数小得多的单个 Agent 的 q 值表空间,并结合一定的协调机制和策略选择机制,确保算法以较快的速度收敛到最优平衡点.

2 协作模型 MACM

多 Agent 协作模型通常是对单个 Agent 马尔可夫决策过程(MDP)的扩展,与 MDP 所不同的是,在多 Agent 协作模型中,系统的整体性能受到所有参与协作的多个 Agent 联合动作的影响,并且 Agent 动作选择不仅取决于系统的当前状态,同时还取决于其它 Agent 的策略.因此,多 Agent 协作模型可以

* 国家"十五"科技攻关计划(2001BA104C)、安徽省自然科学基金(00043115)资助项目
收稿日期:2001-12-14;修回日期:2002-08-15

被视为分布式学习和决策制定过程[10]. 由于参与协作的 Agent 都是自治实体,故在协作过程中,必须引入协调机制,以协调多个 Agent 的动作,避免冲突,达到共同目标. 基于上述思想,定义了多 Agent 协作模型 MACM. MACM 的协作过程可以定义为一个七元组 $\langle M, \{A_i\}_{i\in M}, S, G, T, R, CM\rangle$

- M:参与协作的 n 个Agent组成的有限集合.
- $\{A_i\}_{i\in M}$:对于每个 agent $i \in M$,都有一个有限动作集合 A_i. n 个 Agent 采取的联合动作 $\langle a_1, a_2, \cdots, a_n\rangle$, $a_i \in A_i$ 构成了联合动作空间 $A = \times A_i$ 中的元素.
- S:系统的状态空间.
- G:协作 Agent 的共同目标.
- R:奖励函数,$S \times A \to R$.
- T:在随机环境下,$T: S \times A \times S \to \Delta$,$\Delta$ 为环境状态空间 S 上的概率分布,满足 $\sum_{s'\in S} T(s, a, s') = 1$. 在确定环境下,$T: S \times A \to S$ 为确定函数,即对于 s 状态的某个后续状态(如 s'_1), $T(s, a, s'_1) = 1$, 对于其它后续状态,$T(s, a, s') = 0, s' \neq s'_1$,简记为 $T(s, a)$.
- CM:协调机制(Coordination Mechanism)是参与协作的 Agent 必须共同遵守的协议. 可以定义为三元组 $\langle State, Rule, PIO - actions\rangle$

State:总结概括 Agent 历史经验与外界环境的相关信息.

Rule:可以表示为 *State* 的函数. 用于从集合 *PIO – actions* 中选择要执行的动作.

PIO – actions: *PIO – actions* 的定义如下[4]:

定义 在联合动作空间中,如果对于除联合动作 a 之外的其它任意联合动作 a',都有 $R(a) > R(a')$,则称 a 是最优联合动作. 如果最优联合动作 a 包含 $a_i \in A_i$,则称 a_i 是 Agent i 的 PIO – actions(Potentially Individually Optimal Actions).

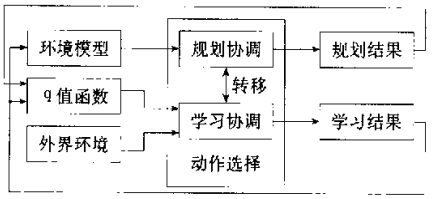

图1 规划—学习协调机制

本文提出了规划—学习协调机制,该机制是对学习协调机制的扩展,参见图1. 规划—学习协调机制可以工作在两种模式:学习协调模式和规划协调模式. 在学习协调模式下,Agent 随机的选择一个 PIO – actions 执行,直到最优联合解产生为止,通过学习协调,可以对环境变化及其它 Agent 的策略做出快速的反应;在规划协调模式下,Agent 在其环境模型(包括转移概率 T 和奖励函数 R)的基础上选择执行动作,通过规划协调,可以避免多 Agent 系统因协调失败而产生的严重后果[7]. 在多 Agent 系统协作过程中,系统可以交替工作在两种协调模式下.

3 协作过程中的学习

3.1 多 Agent 强化学习

在协调机制的约束下,多个 Agent 通过协调自身的动作,从而达到协作的目的. 通常,Agent 参与协作的方式有两种[3]:

- Agent 不仅学习和改善自身的策略,同时也学习参与协作的其它 Agent 的策略,作为选择动作的基础. 文献[4]称这样的 Agent 为联合动作学习者.
- Agent 仅仅学习和改善自身的策略来选择动作,而不考虑其它 Agent 的策略. 以这种方式参与协作的 Agent 称为独立学习者. 由于环境状态的转移取决于所有协作 Agent 的联合动作,因此,对于独立学习者,环境的下一个状态是无法预测的.

在实际应用中,针对 Agent 知识的不完备性以及所处环境的动态性,我们采用了强化学习中无模型 Q- 学习算法作为 Agent 的在线学习机制,并且,为了减少 Q 表的存储空间,每个 Agent 作为独立学习者参与协作.

强化学习是一种通过 Agent 与动态环境之间的交互作用进行决策学习的一种学习机制[8],强化学习的原理是 Agent 对环境执行某种动作,改变环境的状态并获得环境给予的报酬信号来强化某一状态与最优动作策略之间的映射关系,反复执行这一过程,Agent 可获得在任意环境状态下给出最优动作策略的能力. 在基于强化学习的多 Agent 协作系统中,Agent_i 的目标就是寻求最优策略 π_i,以使未来期望折扣总收益 $E(\sum_{t=0}^{\infty} \gamma^t R(s^t \mid \prod^*))$ 达到最大[9]. 其中,$0 < \gamma \leqslant 1$ 为折扣因子,$\prod^* = (\pi_1(s), \pi_2(s), \cdots, \pi_i(s), \cdots, \pi_n(s))$ 为联合策略.

如果将多 Agent 联合动作视为一个整体,则多 Agent 的 Q- 学习算法与单 Agent 类似,也是通过对 Q 值的更新进行的,即

$$Q_{t+1}(s,a)=\begin{cases}Qt(s,a) \ if\ s\neq s_t\ or\ a\neq a_t\\ \gamma(s,a)+\sum_{s'\in S}(T(s,a,s')*\\ \max_{a'\in A_t}Q(s',a')\ s=s_t\ and\ a=a_t\end{cases}$$

Watkins 证明了对于有限 *MDP*，*Q* 学习算法最终可以收敛到最优解[11].

本文的协作多 Agent *Q*- 学习算法采用了分布式最优强化学习算法，在假设 Agent 是乐观的条件下，将系统的联合 *Q* 表空间(用大写字母 *Q* 表示)映射到维数小得多的单 Agent 的 q 表空间(用小写字母 q 表示)，Agent 作为独立学习者参与协作. 即

$$q^i(s,u)=\max_{u=(a^{(1)},\cdots,a^{(n)}),a^{(i)}=u}Q(s,a). \tag{1}$$

根据 Bollzmann 策略选择机制，即 $p(u)=\frac{e^{q(u)/T}}{\sum_{u'\in A_i}e^{q(u')/T}}$，Agent$_i$ 将以较大的概率选择动作 $\hat{u}\in A_i$，使得

$$\begin{aligned}q^i(s,\hat{u})&=\max_{u\in A^i}q^i(s,u)\\&=\max_{u\in A}\max_{u=(a^{(1)},\cdots,a^{(n)}),a^{(i)}=u}Q(s,a)\\&=\max_{a=(a^{(1)},\cdots,a^{(n)}),a^{(i)}=\hat{u}}Q(s,a).\end{aligned}\tag{2}$$

下面举例说明联合 Q 表空间到单 Agent 的 q 表空间的映射过程. 图 2 所示为一个简单的多 Agent 协作系统，该系统包括两个 Agent(Agent1 和 Agent2)，每个 Agent 在状态 S_0 可以采取上移(a)，下移(b)，左移(c) 和右移(d) 四个动作，假设 Agent 所处的环境是确定性的. 采用集中式 *Q* 学习算法，系统最终可以收敛到 $Q^*=7$，其最优联合动作为$\langle b,c\rangle$，如表 1 所示. 随着参与协作的 Agent 数目以及环境状态的增加，*Q* 值表的大小将呈指数级递增. 采用分布式强化学习算法，每个 Agent 只需维护自身的一维 q 值表，如表 2 所示，它满足等式(1). 显然，每个 Agent 根据等式(2)选择动作，其结果必将收敛到最优联合动作$\langle b,c\rangle$. 类似的，在有多个 Agent 参与协作的情况下，通过将多维联合 *Q* 值表映射到单 Agent 的一维 q 表空间，可以显著降低 *Q* 学习算法对系统计算资源的要求.

等式(2)为选择最优联合策略的必要条件，而非充分条件. 在有多个最优联合策略同时存在的条件下(例如在上例中将状态 S_2 的奖励值改为 7，则存在$\langle a,a\rangle$，$\langle c,a\rangle$，$\langle b,c\rangle$ 等多个最优联合动作)，虽然每个 Agent 根据自身的 q 表选择最优策略，结果所得的联合策略很可能并非是最优解(如联合动作$\langle a,c\rangle$ 将导致奖励值为 - 10). 因此必须将分布式强化算法和协调机制相结合，以协调 Agent 之间的动作选择，确保协作结果达到最优.

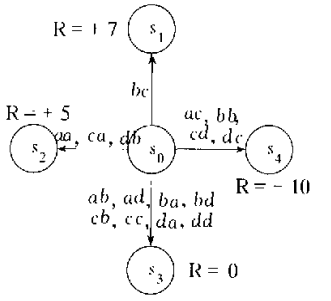

图 2　多 Agent 协作系统

表 1　联合 Q 值表

Agent2 \ Agent1	a	b	c	d
a	5	0	5	0
b	0	- 10	0	5
c	- 10	7	0	- 10
d	0	0	- 10	0

表 2　单 Agent 的 q 值表

q 值表	a	b	c	d
Agent1	5	7	5	5
Agent2	5	5	7	0

3.2　Agent 强化学习模型

在协作多 Agent 系统中，应用 *Q*- 学习算法的 Agent 通过与环境交互获得奖励值，根据奖励值对 q 值进行更新. Agent 强化学习模型中的几个主要模块如图 3 所示.

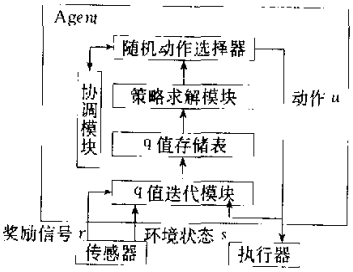

图 3　Agent 强化学习模型

q 值迭代模块：根据环境状态 s，采取的动作 u 以及反馈的瞬时奖励值对 q 值进行更新.

q 值存储表：存储 q 值，每次 q 值迭代后对 q 值存储表进行更新.

策略求解模块：根据当前状态输入 s 和该状态下的 q 值估计，求解最优策略.

随机动作选择器：根据 Bollzmann 策略选择机制，Agent 除以较大概率选择最优策略外，也能以较小概率选择其它非最优策略.

协调模块：根据其工作模式，与随机动作选择器一起协调 Agent 的动作选择.

3.3 分布式多 Agent 强化学习算法

前面一节简单讨论了 Agent 协作过程中的学习以及多 Agent 强化学习模型. 本节给出在确定环境下，分布式最优多 Agent 强化学习算法. 以 Agent_i 为例：

Step1：初始化 $q_0^i(s,u)$

Step2：根据 Bollzmann 策略选择机制从初始 q 表中选择动作 u，$u \in A_i$

Step3：执行 u，观测后续状态 s' 和奖励值 r，修改 q 表. 修改规则如下：

$$q_{t+1}^i(s,u)=\begin{cases} q_t^i(s,u) & s \neq s_t \text{ or } u \neq u_t \\ \max\{q_t^i(s,u), r(s_t,u_t)\} & \\ \gamma \max\limits_{u' \in A_i} q_t^i(T(s_t,u_t),u')\} & \end{cases}$$

Step4：进行协调和动作选择. 如果 $s \neq s_t$ 或 $|\max\limits_{u \in A'} q_{t+1}^i(s,u) - \max\limits_{u \in A'} q_t^i(s,u)| \leqslant \delta$（$\delta$ 为任意小的正整数），则 $\pi_{t+1}^i(s)=\pi_t^i(s)$. 否则按照 Bollzmann 策略选择机制重新选择动作.

Step5：$t=t+1$，转 Step3.

按照上面的算法，我们可以获得最优联合策略，满足 $Q_t(s,\prod)=\max\limits_{a \in A} Q_t(s,a)$，其中 $\prod=(\pi_t^1(s),\cdots,\pi_t^i(s)\cdots\pi_t^n(s))$. 可以证明，该算法最终能够收敛到最优联合策略.

4 结论和展望

本文针对多 Agent 系统中的协作问题提出了协作模型，该模型通过提供灵活的规划 - 学习协调机制支持多 Agent 之间的协作和学习. 由于强化学习可以不需要环境模型，是一种无监督学习方法并且具有较好的收敛性，因此在规划和控制领域得到了广泛的应用. 强化学习已经成为人工智能学习机制研究的一大热点. 针对多 Agent 协作系统，许多研究人员设计了可以收敛到 Nash 平衡解的多 Agent 强化学习算法，这些算法和经典的 Q- 学习算法一样，由于 Q 值表的存储空间过大，因而对系统的计算资源要求过高，不适用于复杂的大规模问题求解. 本文采用分布式多 Agent 强化学习，将系统的联合 Q 值表空间映射到维数小得多的单 Agent 的 q 值表空间，降低对系统资源的要求，并结合规划 — 学习协调机制，从而提高算法的实用性.

我们在 MMDP（移动 Agent 开发平台）系统[12]中实现了该协作模型及其算法. 实验例为两个 Agent 之间的协作，每个 Agent 可以采取 4 个动作. 仿真实验采用两台 PC 机模拟 Agent，用 IBM 服务器模拟环境的变化以及环境对动作的奖励值. 实验分为两组进行，一组采用学习协调机制，一组采用规划 — 学习协调机制. 初步的实验结果证明了本文协作模型及分布式强化学习算法相结合能够确保 Agent 以较快的速度收敛到最优解.

我们的工作虽然取得了一些阶段性成果，但仍然有许多问题需要作进一步的研究：1、深入研究多 Agent 协作系统的工作机理及多 Agent 强化学习算法复杂度；2、研究随机环境下多 Agent 强化学习算法以及算法的收敛性；3、研究概率聚类方法，如函数逼近和神经网络学习以降低算法对系统资源的要求.

参 考 文 献

[1] Jennings N R, Sycara K, Wooldridge M. A Roadmap of Agent Research and Development. International Journal of Autonomous Agent and Multi-Agent System, 1998, 1(1): 275 - 306

[2] Buffet O, Dutech A, Charpillet F. Incremental Reinforcement Learning for Designing Multi-Agents Systems. In: Proce of the 5th International Conference on Autonomous Agents. Montreal, 2001, 31 - 32

[3] Tan M. Multi-Agent Reinforcement Learning: Independent vs. Cooperative Agents. In: Proce of the 10th International Conference on Machine Learning, Amberst, MA, 1993, 330 - 337

[4] Boutilier C. Sequential Optimality and Coordination in Multi-agent Systems. In: Proce of the 16th International Joint Conferences on Artificial Intelligence, San Francisco, 1999, 478 - 485

[5] Claus C, Boutilier C. The Dynamics of Reinforcement Learning in Cooperative Multi-Agent Systems. In: Proce of 15th National Conference on Artificial Intelligence, Cambridge, MA, 1998, 235 - 262

[6] Ehud K, Ehud L. Rational Learning Leads to Nash Equibrium. Econometric, 1993, 61(5): 1019 - 1045

[7] Boutilier B C. Planning, Learning and Coordination in Multi-Agent Decision Processes. In: Proce of the 6th Conference on Artificial Intelligence, Amsterdam, 1996, 195 - 210

[8] Sutton R S, Andrew G. Reinforcement Learning: An Introduction. Cambridge, MIT Press, 1998

[9] Kaelbling L P, Littman M, Moore A. Reinforcement Learning: A Survey. Journal of Artificial Intelligence Research, 1996, 4: 237 - 285

[10] Martin L, Martin R. An Algorithm for Distributed Reinforcement

Learning in Cooperative Multi-Agent Systems. In: Proce of International Conference on Machine Learning, Stanford CA, 2000, 535 - 542

[11] Watkins C, Dayan P. Q-Learning. Machine Learning, 1992, 8 (3): 279 292

[12] 骆正虎,等.基于移动 Agent 的分布式计算模型研究.小型微型计算机系统.2002, 23(3): 300 - 304

RESEARCH ON COOPERATION AND REINFORCEMENT LEARNING ALGORITHM IN MULTI-AGENT SYSTEMS

Zheng Shuli, Han Jianghong, Lou Xiangfeng, Jiang Jianwen

(*Computer Science Department, Hefei University of Technology, Hefei* 230009)

ABSTRACT

A cooperation model called MACM is presented which provides a flexible coordination mechanism to support cooperation and learning in multi-agent system. Learning agent adopts model-free distributed Q-learning since reinforcement learning can provide a robust and natural means for agents to learn how to coordinate their action choices. With projection the distributed Q-learning algorithm needs less storage space for Q-table than the classical Q-learning. Also it can be proved to find optimal polices in deteministic environments.

Key Words Multi-Agent System, Cooperation and Coordination, Reinforcement Learning