强化学习系统 知识获取 人工智能 算法 (13)

计算机科学1999Vol. 26№. 10

53-56

强化学习系统的结构及算法

The Architectures and Algorithm of Reinforcement Learning System

张汝波 顾国昌 张国印 TP18

(哈尔滨工程大学计算机系 哈尔滨150001)

Abstract The word „reinforcement learning„comes from behavior psychology. This subject takes learning as trial and error process so as to map world state to the actions. The architecture of reinforcement learning system is discussed and implement method of each function and learning algorithm are presented in this paper.

Keywords Reinforcement learning„Intelligent control system„Q-learning

1. 引言

学习是人类获取知识的主要形式，也是人类具有智能的显著标志，是人类提高智能水平的基本途径。建造具有类似人的智能机器(Agent)是智能控制、人工智能的研究目标。要使机器具有一定的智能，一种方式是靠人事先编程来建立知识库和推理机制，这具有明显的局限性。我们希望 Agent 具有向环境学习的能力，即自动获取知识、积累经验、不断更新和扩充知识，改善知识性能。强化学习机制为我们提供了一种方法，使得 Agent 具有自学习能力。

所谓强化学习就是从环境到行为映射的学习，以使结果信号(强化信号)最大。采用这种学习方式时，学习者不象大多数机器学习那样被告知采取何种行动，而是通过尝试而发现行为以产生最大的强化信号。强化信号可以告诉 Agent 某个动作是错误的，但却不能告诉正确的动作是什么，这样，按目标函数来说，梯度信息不存在了。所以，在强化学习网络中需要某种随机因素，我们才能研究可能输出的空间并找到正确的输出值。因此，在强化学习系统中一般都含有随机单元。

本文首先讨论了强化学习系统的结构，给出了形式化描述；然后讨论了各模块的实现方法。

2. 强化学习的结构模型

强化学习的基本框架如图1所示，系统主要由两部分组成：环境 World 和 Agent。我们可把环境看成一个动态系统，Agent 产生的动作使其状态发生变化。Agent 包括三个部分：输入模块 I、强化模块 R 及策略模块 P。输入模块把描述环境的状态变成 Agent 的输入 X；强化模块 R 把环境的每一个状态赋给一个值 r；策略模块更新 Agent 的知识，同时使 Agent 根据某种策略选择一个动作并作用于环境。上述的强化学习系统可用三元组描述：

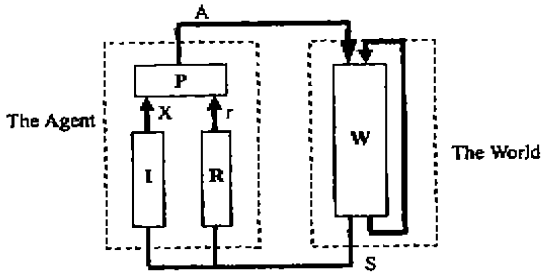

图1 强化学习系统的基本框架

$$\mathcal{RS}=(\mathbf{S},\mathbf{A},\mathbf{W}) \quad (1)$$

其中 $\mathbf{S}=(S_1,\cdots,S_N)$ 为环境的所有可能状态的集合；$\mathbf{A}=(a_1,\cdots,a_M)$ 是 Agent 可能产生的动作集合；W 是环境的状态转移集合，即：

$$\mathbf{W}:\mathbf{S}\times\mathbf{A}\to\mathbf{S} \quad (2)$$

Agent 可用一个四元组来描述：

$$\mathcal{A}=(\mathbf{X},\mathbf{I},\mathbf{R},\mathbf{P}) \quad (3)$$

其中 X 为 Agent 的所有输入的集合；I 是一种映射：

$$\mathbf{I}:\mathbf{S}\to\mathbf{X} \quad (4)$$

把环境的状态映射为 Agent 的输入，R 为 Agent 的强化模块，把状态映射为一个实数：

张汝波 副教授，在职博士生，研究方向为强化学习，智能控制及智能机器人。顾国昌 博导，教授，研究领域为智能机器人，机器人体系结构，行动决策和控制技术。

$$R: S \rightarrow \mathcal{R} \tag{5}$$

P 为 Agent 的策略模块，把一系列的输入变成相应的动作：

$$P: X \times R \rightarrow A \tag{6}$$

强化模块决定了 Agent 的目标。Agent 的目标是使它接受的长期奖励最大。强化模块定义了对 Agent 来说什么是好的事件、什么是坏的事件。强化信号是立即得到的，它体现了 Agent 所面临的问题的特征。这样，强化模块一定是固定不变的，它是改变策略的基础。例如，如果一个被选择动作得到一个较低的强化信号，那么在将来遇到相同情况下，策略模块应选择其他的动作。

强化学习的目的是构造一个控制策略，使得 Agent 的性能达到最大。因此，需要定义一个值函数(Value function)或目标函数来表明从长期的观点来确定什么是好的动作。状态的值函数或状态-动作对的值函数是强化学习重要的步骤，目标函数可用如下形式描述：

$$V = \sum_{k=0}^{\infty} \gamma^k r_{t+k} \tag{7}$$

r_t 是从 t 到 t+1 状态转移后 Agent 接受到的强化信号，$0 \leqslant \gamma \leqslant 1$ 为折扣因子。强化信号可以是正(奖)、负(惩)和零。

3. 强化学习系统各模块的实现方法

3.1 输入模块 I 的实现方法

输入模块的作用是把环境的状态转换成适应 Agent 所能接受的状态，为策略模块提供输入。当然，也可把状态信息直接提供给策略模块。输入模块可用线性函数、模糊方法及神经网络等方法实现。转换结果的取值范围可为$(-\infty, \infty)$、$\{0,1\}$、$\{-1,1\}$及模糊语言值等。实现输入模块化方法各不相同，有各自的特点及应用场合。

3.1.1 BOX 结构方法 BOX 算法的基本思想是经过量化，把状态空间分别确定为非重叠区域。每个区域称之为一个 BOX，每个 BOX 中还有一个局部精灵，当系统进入某个 BOX 时，该 BOX 中的局部精灵被激活，相应的状态设为1，设传感器可测得环境的 N 个信息 S_i，每个信息 S_i 可划分成 n_i 个状态，则 N 个信息可划分成 $N_s = \prod_{i=1}^{N} n_i$，也就是说 Agent 有 N_s 个输入状态。

BOX 算法简单地把状态空间量化为互不重叠的 BOX，使 BOX 之间没有任何的泛化，而且如何划分也需要更多的先验知识。当划分过细时，BOX 数目太大，不利于策略模块的学习和决策；而划分过粗，环境的状态就不能得到正确的描述。

3.1.2 Fuzzy 方法 利用模糊方法，可以改进 BOX 结构，使 BOX 之间允许有一定的重叠。设 n_i 表示第 i 个状态变量 S_i 所对应的语言变量值的个数，每个状态 S_i 都有对应的一个隶属函数，将状态 S_i 用模糊语言变量来描述。

3.1.3 神经网络方法 利用 BOX 结构及模糊方法量化时，都存在量化空间太大问题。状态空间较大时，不仅要占用大量的内存，而且计算量较大，不利于策略模块的决策，需要对状态空间进行压缩。

Kohonen 的神经网络是一个自组织神经网络，其学习的结果能体现出输入样本的分布情况，从而对输入样本实现数据压缩。由于 Kohonen 网络的这种特性，可用来对状态空间进行量化，把状态信息作为网络的输入，神经网络中每个神经元的状态用来表示一定的状态空间的压缩结果。这样可以减少学习算法的搜索空间，提高强化学习的学习速度。

3.2 强化模块的实现方法

在讨论强化模块实现方法之前，首先要明确什么是强化信号？这个概念在人们学习过程中实际上是很模糊的。从大量的文献来看，强化学习要有效地用于智能控制系统，强化信号的选取是比较关键的。例如，在倒立摆控制问题中，通常认为杆的倾斜带来了一个惩罚信号，强化信号也选为－1，其余时候都为0。这是典型的滞后评价问题。随着反复尝试，我们对杆的位置有了进一步的评价。比如，一旦倾斜角度大于某个值，杆必将面临无可挽回的倾倒结局。这些逐步建立起来的认识，事实上就变成后续学习的新的强化信号，而不是等到杆最后倾倒。从这个角度看，一个状态动作序列获得的反馈评价是随着知识和经验增加慢慢建立起来的，这个思想就是时差(Temporal difference)法的一个学习原则，人们在学习中总是试图建立准确的预测关系，对实验结果的评价的预测随着经验的增加是收敛的。在已有的强化学习范畴里，这种不断细化的"强化信号"实际上体现于 Agent 的预测器的输出。所以强化信号不是强化学习的标识符。

强化信号可以从环境的状态直接得到，例如：倒立摆的角度大于一定值时，就可以产生一个失败信号；当机器人与障碍物相撞时，即，传感器的距离信息小于给定值时，都可看作是一个失败信号。另一个方法，强化信号也可以从环境的状态信息中间接地得到，当环境的状态值达不到预定的要求时，也可以认为产生一个失败强化信号。强化信号不但来自环境的状态，而且和主观的目标紧密相关。因此，实现强化模块化方法有许多，如神经网络方法、模糊方法以及一些简单的函数方法等等。

强化信号可以是下列形式中的一种：1)二值，$r \in$

{-1,0},这里 r=-1表示失败;r=0表示成功。2)介于[-1,1]间的多个离散值,例如:∈{-1,-0.5,0,0.5,1}分段地表示成功和失败的程度。3)介于[-1,1]间的实数连续值,r∈[-1,1],它能细致表示成败的程度。

3.3 策略模块的实现方法

在强化学习系统中,最关键的部分就是策略模块,其主要功能是通过学习机制,更新内部知识,选择一个动作作用于环境。策略模块的框图如图2所示。策略模块包括三部分:内部状态 W_I、学习模块 L 和动作选择模块 Se。内部状态 W_I 概括地表述了 Agent 拥有关于环境的知识水平,它没有明确表明与环境的关系。在每个循环中,动作选择模块 Se 基于状态 W_I 及强化信号决定了 Agent 对输入 X 的响应 a:

$$Se: X \times W_I \times r \rightarrow a \tag{8}$$

被选择的动作使得环境产生了一个新的状态 X' 和一个强化值 r,Agent 的内部状态 W 则被学习模块 L 更新为:

$$L: X' \times W_I \times A \times r \rightarrow W_I \tag{9}$$

使得 Agent 对环境的知识有了进一步的更新。下面介绍策略模块的主要实现方法。

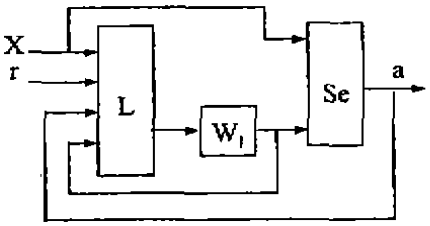

图2 策略模块的组成

3.3.1 概率矢量法 又称非联想算法。它没有利用环境的状态信息,只利用强化学习信号来进行学习和动作选择。在每一步,Agent 根据概率分布来选择动作。这个分布以概率矢量 $\pi=\{\pi_1,\pi_2,\cdots,\pi_M\}$ 来表示。这个分布把选择动作的概率 π_i 与每个动作 a_i 联系起来。就前面描述的策略模块而言,Agent 的内部状态 W_I 就是概率 π_i。动作选择模块 Se 唯一地根据这些概率来确定其动作。学习模块根据作用于环境的动作及返回的强化值来调整这些概率。这种 Agent 叫随机学习自动机。Sutron 给出了一些只有两个动作的概率矢量算法[1]:

$$\pi(t+1)=\pi(t)=\begin{cases} \alpha[a(t)-\pi(t)] & \text{if } r=1 \\ \alpha[1-a(t)-\pi(t)] & \text{if } r=-1 \end{cases} \tag{10}$$

其中,$a(t)$ 表示输出值,$a(t)\in\{0,1\}$;$\pi(t)$ 表示 t 时刻选择 $a(t)=1$ 的概率。上述算法就是非常著名的学习自动机算法,称为线性奖惩算法 L_{RP}。

3.3.2 联想算法 与概率矢量算法不同,联想算法不仅利用环境的强化信号而且利用通过输入模块 I 感知的环境状态信息。联想算法的任务就是学习系统根据某些机制把输入映射为相应的输出。也就是说,系统针对环境的状态能够选择出合适的动作。动作联想矢量(权系数)的更新及动作的选择都与输入状态有关。例如,动作输出可取为:

$$a(t)=\begin{cases} 1 & \text{if } s(t)+\eta(t)>0 \\ 0 & \text{其它} \end{cases} \tag{11}$$

而 $s(t)=\sum_{i=1}^{n} w_i(t)x_i(t)$;$\eta(t)$ 为零均值、方差为 σ_η 的正态分布随机变量,$x_i(t)$ 为系统的输入;$w_i(t)$ 联想矢量;其修改方法为:

$$w_i(t+1)=w_i(t)+\alpha\cdot[r(t+1)-r(t)]\cdot[a(t)-\pi(t)]x_i(t) \tag{12}$$

其中,$\pi(t)$ 为动作选择概率;式(12)的权值更新规则均为两部分的乘积:一部分取决于动作的选择,如 $a(t)-\pi(t)$;而另一部分取决于系统的输入状态 $x_i(t)$。这种乘积通常称为 $w_i(t)$ 的资格(eligibility)。

联想算法的典型实现方法是利用神经网络技术,网络的输出可作为动作概率分布参数。联想强化学习 Agent 的动作选择模块利用这些分布来产生动作。利用神经网络的分类能力,联想算法能够把最优动作与整个环境状态相联系。换句话说,它可以从过去与环境发生联系的经验中得出一般的结论,这种能力对具有时空复杂性的学习机来说是非常重要的。然而,和许多连接主义算法一样,在应用到比较复杂的任务时,其收敛很慢或不能收敛。

Agent 的内部状态就是网络的权值,学习模块根据输入矢量 X 及强化信号 r 来更新权值,其更新的过程可采用随机爬山法。

3.3.3 延时强化学习算法 上面讨论的算法都是在一定的假设条件下实现的,那就是 Agent 采取一个动作,环境立即产生一个强化值,但是,有许多应用任务都是在几个步骤以后 Agent 才收到行为的评价信号。这种类型的强化方法称之为延时强化。

动态规划 DP(Dynamic Programming)技术就是在给定完整的环境状态转移模型及每个转移的信度(强化信号)条件下来计算最优动作映射的。DP 算法的变形就是对于每一个〈状态-动作〉对除了状态值 V(s) 外还保持一个 $Q(s,a)$ 值,对于每一个状态动作对〈s,a〉及策略,$Q(s,a)$ 定义为 Agent 处于状态 S 时执行动作 a,且以后按此策略执行时的累积强化值。在此方法中,状态 S 的值 $V(s)$ 是此状态可能动作集合中最大的 $Q(s,a)$。Watkins 还证明了,如果缺乏完整环境模型

时，$Q(s,a)$按下式公式进行更新[4]：

$$Q_t(s_t,a_t)=\begin{cases}(1-\alpha_t)Q_{t-1}(s_t,a_t)+\\ \quad\alpha_t[r_t+\gamma V(s_{t+1})] & s=s_t;a=a_t\\ Q_{t-1}(s_t,a_t) & \text{其它}\end{cases}\quad(13)$$

只要每一个变化经过无穷多次，当$t\to\infty$时，$\alpha_t\to 0$，$Q(s,a)$就会收敛于最优值。在式(13)中，$V(s_{t+1})=\max_{a\in A}\{Q_{t-1}(s_{t+1},a)\}$是由动作$a$产生的状态值；$\alpha_t$为学习速率；$\gamma$为折扣系数；$r_t$立即强化信号。由于这个结果只允许对被选中动作的V值进行更新，且更新可以实时完成，这个算法被命名为Q-学习算法。

3.3.4 *基于模型的算法* 另一种在环境中学习动作的方法是建立环境内部表示，也就是建模。利用学会的模型来选择适当的动作。在这种方法中学习的目的是产生对环境行为描述最好的逼近模型，有时称为动作模型。由于环境可以看成一个自动机，在给定初始状态和执行动作后模型应能够正确预测环境的变化。

强化学习Agent的外部环境模型在某种程序上模仿了环境的行为。例如，给定一个状态和一个动作，模型可以预测下一个状态和下一个强化值。模型可能要占据较大的存储空间，如果有$|S|$个状态和$|A|$个动作，那么整个模型将占据的空间与$|S|\times|S|\times|A|$成比例。这是由于它把状态-动作对匹配成整个状态空间的概率分布。相反，强化值和值函数把状态匹配成一个实数，其维数是$|S|$；而随机策略的维数是$|S|\times|A|$。

并不是所有的强化学习Agent都利用模型，从未使用模型的方法叫无模型强化学习方法，无模型方法非常简单，也许能令人惊讶地找到最优行为。基于模型的方法能够较快地找到最优行为(利用较少的经验)，最令人感兴趣的情况是Agent事先没有完美的模型，而是利用学习的方法来确定它。

与联想算法类似，基于模型强化学习系统的输入信息包括环境状态，强化值由强化模块R来计算。由于合适动作的选择依赖于模型的正确性，所以强化值是动作模型预测环境行为真实程度的一种测量，策略模块的内部状态W_t是环境模型的一种编码。

结论 本文讨论了强化学习系统的一般组成结构，并对结构模型进行了描述，对组成系统的输入模块、强化模块特别是策略模块的实现方法进行了研究，重点讨论了策略模块的实现方法。强化学习存在学习速度慢的问题。所以，如何提高强化学习系统的速度，是今后研究的一个重要问题。

参考文献

1 Sutton R S. Temporal Credit Assignment in Reinforcement Learning: [PhD thesis]. University of Massachusetts, Amherst, MA, 1984

2 Sutton R S. Learning to Predict by the Methods of Temporal Difference. Machine Learning, 1988, 3: 9～44

3 Sutton R S. The Challenge of Reinforcement Learning. Machine Learning, 1992, 8: 225～227

4 Watkins J C H, Dayan P. Q-learning. Machine Learning, 1992, 8: 279～292

5 Peng J, Williams R J. Increment Multi-Step Q-Learning. Machine Learning, 22: 283～291

6 Thrum S, Mitchell T M. Lifelong Robot Learning. Robotics and Autonomous System, 1995, 15: 25～46

7 Ilg W, Berns K. A Learning Architecture Based on Reinforcement Learning for Adaptive Control of the Walking Machine LAURON. Robotics and Autonomous System, 1995, 15: 32～334

8 阎平凡. 再励学习—原理、算法及其在智能控制中的应用. 信息与控制, 1996(1): 28～34

9 Barto A G, et al. Neuronlike Adaptive elements that can solve difficult learning control problems. IEEE Transaction on System, Man, and Cybernetics, 1983, 13: 834～846

10 Beom H B. A Sensor-Based Navigation for a Mobile Robot Using Fuzzy Logic and Reinforcement Learning. IEEE Trans. on SMC, 1995, 25(3)

11 Moure A W, Atkson C G. Prioritized Sweeping: Reinforcement Learning with Less Data and Less Time. Machine Learning, 1993, 13: 103～130

(上接第91页)

3 Dewan D, Shen H. Controlling access in multiuser interface. ACM Transactions on Computer-Human Interaction, 1998, 5(1): 34～62

4 Sandhu R S, et al. Role-Based Access Control Models. IEEE Computer, 1996, 29(2), 38～47

5 Gavrila S I, Barkley J F. Formal Specification for Role Based Access Control User/Role and Role/Role Relationship Management. RBAC' 98. In: Proc. of the third ACM workshop on role-based access control. Fairfax, VA, 1998. 81～90

6 Ferraiolo D F, et al. Role-Based Access Control: Features and Motivations. In: Annual Computer Security Applications Conf. IEEE Computer Society Press, 1995

7 Giuri L, Iglio P. Role templates for content-based access control. RBAC' 97. In: Proc. of the second ACM workshop on Role-based access control. Fairfax, VA, 1997, 153～159

8 Radack B F. An Introduction to Role-Bases Access Control, NIST CSL Bulletin, 1995