强化学习算法研究

刘 忠[1]， 李海红[2]， 刘 全[1,3]
(1. 苏州大学 计算机科学与技术学院，江苏 苏州 215006; 2. 浙江工业大学 信息学院，浙江 杭州 310014;
3. 南京大学 软件新技术国家重点实验室，江苏 南京 210093)

摘 要：针对智能Agent运动中普遍存在的避障问题，结合强化学习具有的试错和环境交互获得在某状态下选择动作的策略以及无导师在线学习等特性。在介绍强化学习的原理、分类以及主要算法($TD(\lambda)$、Q_learning、Dyna、Prioritized Sweeping、Sarsa)的基础上，对$TD(\lambda)$、Q_learning的算法进行分析，并将其应用到实验中。实验结果表明，强化学习中的$TD(\lambda)$、Q_learning等算法在不同情况下都能高效地解决避障等问题。

关键词：强化学习；Q学习；Agent智能体；机器人控制；避障；搜索引擎
中图法分类号：TP18 **文献标识码**：A **文章编号**：1000-7024 (2008) 22-5805-05

Research on algorithm of reinforcement learning

LIU Zhong[1], LI Hai-hong[2], LIU Quan[1,3]
(1. College of Computer Science and Technology, Soochow University, Suzhou 215006, China; 2. College of Information Engineering, Zhejiang University of Technology, Hangzhou 310014, China; 3. State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China)

Abstract: Aiming to solve the problem of passing the block for the intelligent agents, the trial-and-error, the policy that obtained after agents communicate is combined with the environment in one state for choosing actions in RL learning and the unsupervised on_line learning feature. The principium, partition and the main algorithms ($TD(\lambda)$, Q_learning, Dyna, Prioritized Sweeping, Sarsa) of the RL is introduced with the analysis of $TD(\lambda)$ and Q_learning algorithm, which has been applied into the experiment. The experimental result proves that the algorithms solve this problem efficiently under different environments.

Key words: reinforcement learning; Q_learning; agent; robot control; obstacle avoidance; search engine

0 引 言

智能Agent的主要特征之一就是能够适应未知环境，而在这一过程中，主动学习是至关重要的。在机器学习领域，大致可以将学习分为监督学习、非监督学习和强化学习3大类。

监督学习，就是对于每一个输入，学习者都被提供一个目标，即环境或者“老师”来告诉学习者，对于每次的输入应该做出如何回应。辨别是何种乐器所发出的声音就是这一学习类型。

非监督学习，主要是建立一个模型，用其试着对输入的数据进行解释，并用于下次输入。采用隐马尔科夫模型(HMM)的语音识别系统，就是采用的这一学习类型，HMM通过在训练中记录下每一个语音之后，我们就可以用这些来识别输入的语音了。

强化学习作为一种在线的、无导师机器学习方法，把环境的反馈作为输入、通过学习选择能达到其目标的最优动作。自20世纪80年代末以来，由于数学理论上的突破而取得了长足的发展，现在已经是机器学习领域的热点方向。强化学习可被应用于任何涉及采取序列行为的任务，主要集中在有限资源调度、机器人控制、棋类游戏等应用领域。

1 强化学习

1.1 基本原理

强化学习就是能够感知环境的自治Agent，怎样通过学习来选择能达到其目标的最优动作[1]。当Agent在其环境中做每个动作时，环境都会提供一个反馈信号，即奖赏值。强化学习也可看成是从环境到动作的映射学习过程，其目的就是采用的某动作能够从环境中得到最大的累积奖赏值。

收稿日期：2007-11-17 **E-mail**：210513050@suda.edu.cn
基金项目：国家自然科学基金项目(60473003、60673092)；中国博士后科研基金项目(20060390919)；江苏省高校自然科学基金项目(06KJB520104)；江苏省博士后科研基金项目 (060211C)。
作者简介：刘忠 (1981－)，男，山东临沂人，硕士研究生，研究方向为强化学习、个性化搜索； 李海红 (1981－)，女，山东潍坊人，硕士研究生，研究方向为CIMS、智能算法； 刘全 (1970－)，博士，副教授，硕士生导师，研究方向为自动推理、机器学习等。

观察生物(特别是人)适应环境的学习过程可以发现，他们从来都不是静止的被动等待而是动态地主动对环境进行试探。从环境对试探动作提供的反馈信号来看，多数情况下的试探是评价性的(奖励或者惩罚)。生物在动作——评价的过程中不断地获得知识、改进行动方案，以达到预期的目的，这也是符合强化学习的特征[2]。

强化学习系统的结构如图1所示，它由 Environment 和 Agent 两部分组成。Agent 与 Environment 不断的进行试探交互，得到 sensor data，agent 在策略$\pi^*:S\rightarrow A$指导下，得到最大的累积奖赏值。而 Agent 的任务就是获得一个最优控制策略$\pi^*:S\rightarrow A$，其中S为状态集，A为动作集。

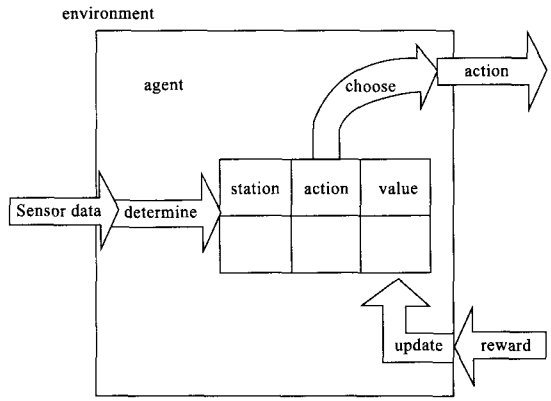


图1 强化学习系统的结构

在这里，需要考虑的是长期的累积奖赏，故就要考虑采用动作以后长期奖赏对现在的影响。所以，我们就需要定义一个目标函数来从长期的观点确定什么是最优的动作。通常有状态的值函数或动作函数对的值函数表达此目标函数，状态值函数可以有如下3种形式

$$V^{\pi}(s_t)=E(\sum_{t=0}^{h}r_t) \tag{1}$$

$$V^{\pi}(s_t)=\lim_{h\to\infty}E(\frac{1}{h}\sum_{t=0}^{h}r_t) \tag{2}$$

$$V^{\pi}(s_t)=E(\sum_{t=0}^{\infty}\gamma^t r_t) \tag{3}$$

式(1)为有限水平模型，Agent 只考虑未来 h 步内的奖赏值；式(2)为平均奖赏模型，Agent 只考虑平均的奖赏值；式(3)为无限水平折扣模型，Agent 考虑未来 h 步内的奖赏，并以某种形式折扣在值函数中，由于它富有数学技巧，所以也是应用最广泛的一种模型，其中r_t是 Agent 从s_t到s_{t+1}所得到的奖赏值，$\gamma(0\leqslant\gamma<1)$是折扣因子，确定了延迟奖赏与立即奖赏的相对比例。故最优策略就可以根据式(4)确定

$$\pi^*=\arg\max_{\pi}V^{\pi}(s),\forall s\in S \tag{4}$$

由于环境具有不确定性，因此在策略π的作用下，状态 s_t 的值也可写为

$$V^{\pi}(s_t)=r_t+\gamma\sum_{s_{t+1}\in S}P(s_{t+1}|s_t,a_t)V^{\pi}(s_{t+1}) \tag{5}$$

式中：$P(s_{t+1}|s_t,a_t)$——在状态s_t时采用a_t到达s_{t+1}的概率。

动态规划理论保证至少有一个π^*，满足 Bellman 最优方程

$$V^{\pi^*}(s_t)=\max_{a\in A}\{r_t+\gamma\sum_{s_{t+1}\in S}P(s_{t+1}|s_t,a_t)V^{\pi^*}(s_{t+1})\} \tag{6}$$

所以，学习的任务就是求解π^*。

通常假定环境是马尔可夫型的，强化学习过程可以使用一个马氏决策过程(Markov decision process，MDP)表示，MDP 由四元组<S, A, R, T>定义，其中 S 为状态集，A 为动作集，$R:S\times A=R$为奖赏函数，记$R(s,a,s_t)$为在状态 s 下采用动作 a 到达 s_t 后所得到的瞬间奖赏值；$T:S\times A=PD(S)$为状态转移函数，其中$T(s,a,s_t)$为 Agent 在状态 s 下采用动作 a 到达 s_t 的概率[1]。

马氏决策过程的本质是：当前状态向下一状态转移的概率和奖赏值只取决于当前状态和选择的动作，而与历史状态和历史动作无关。因此在已知状态转移概率函数 T 和奖赏函数 R 的环境模型知识下，可以采用动态规划技术求解最优策略，而强化学习着重研究在 T 函数和 R 函数未知的情况下，系统如何学习到最优行为策略。

1.2 强化学习的分类

如果 Agent 在学习过程中，无需学习 MDP 的模型知识，直接学习最优策略，我们称这类方法为模型无关(Model_free)；如果在学习过程中，先学习 MDP 的模型知识，然后再根据这些推导、学习出最优策略，我们称之为模型有关(Model_based)。其中前者是我们研究的重点，因为在实际中主要遇到的问题是如何在模型不知的情况下学习到最优策略。前者的算法主要有$TD(\lambda)$和 Q_learning 算法，后者主要有 Dyna、Prioritized Sweeping、Sarsa 算法等。

当然，现在由于强化学习的广泛应用以及研究的深入和细化，也可分为如下几种类型：逻辑强化学习、分层强化学习、多 Agent 强化学习、POMDP 强化学习等。

2 主要算法

2.1 $TD(\lambda)$算法

$TD(\lambda)$结合了蒙特卡罗和动态规划算法，是强化学习技术中最主要的算法之一。其中最简单的 TD 算法是 TD(0)算法，即它获得的奖赏值仅向前回馈一步，它的迭代公式如式(7)所示

$$V(s):=V(s)+\alpha(r+\gamma V(s)-V(s)) \tag{7}$$

式中：α——学习率，$V(s)$——Agent 在 t 时刻得到的估计状态值函数，$V(s)$——Agent 在 t+1 时刻得到的估计值，r 为瞬间奖赏值，γ——折扣因子，式(7)的关键是$r+\gamma V(s_{t+1})$是$V(s)$的参照值，并且它随着真实值 r 的迭代和更新，也会变成真实值。如果学习率α合适(即随着迭代的进行，逐渐变小)且策略固定，TD(0)算法就可以收敛到最优值函数[3]。

前面所讨论的 TD(0)算法，由于仅向前回馈一步，故具有收敛慢等不足，其实它只是$TD(\lambda)$的最初形式，即$\lambda=0$的情况。当 Agent 获得瞬时奖赏值可以向前回馈任意步时，我们称之为$TD(\lambda)$算法。它的迭代公式用式(8)表示

$$V(s):=V(s)+\alpha(r+\gamma V(s_{t+1})-V(s))e(s) \tag{8}$$

式中：$e(s)$——状态 s 的选举度[12]，它是状态 s 已经被访问的度量，在实际应用中可以通过下式得到

$$e(s)=\sum_{k=1}^{t}(\lambda\gamma)^{t-k},\text{where }\delta s,s_k=\begin{cases}1 & \text{if } s=s_k\\ 0 & \text{otherwise}\end{cases} \tag{9}$$

$$e(s)\equiv\begin{cases}\gamma\lambda e(s)+1 & \text{if } s=\text{current state}\\ \gamma\lambda e(s) & \text{otherwise}\end{cases} \tag{10}$$

在一个迭代过程中，状态 s 被访问的次数越多，它的 e(s) 越大，表示它对当前回馈值的贡献就越大，其中式(10)是对式(9)的改进，从计算的角度来说，λ越小执行的效率越高；但实际情况是λ越大，迭代收敛的越快[4]。

2.2 Q_learning 算法

Q_learning 算法是 Watkins 最早于 1989 提出来的[5]，它采用 Q(s,a)和状态—动作对的值来作为估计函数，而不是$TD(\lambda)$的状态值函数 V(s,a)。

$TD(\lambda)$算法的学习过程是减小 Agent 在不同时间做出估计间的差异，而 Q_learning 算法则是循环地减小对相邻状态的 Q 值的估计之间的差异[15]。它的基本形式如下

$$Q^*(s,a)=R(s,a)+\gamma\sum_{s_i\in s}T(s,a,s_i)\max_{a_i}Q^*(s_i,a_i) \tag{11}$$

式中：$Q^*(s,a)$——在状态 s 下采用动作 a 所得到的最优奖赏值的和，而在前面我们也定义$V^*(s)$为在状态 s 下的最优值函数，故

$$V^*(s)=\max_a Q^*(s,a) \tag{12}$$

即我们所采用的最优策略π^*是

$$\pi^*(s)=\arg\max_a Q^*(s,a) \tag{13}$$

这就是说只需对当前状态的 Q 的局部值重复做出反应，就可选择到全局最优化的动作序列，也即当前状态和动作的 Q 值在单个的数值中概括了所有需要的信息，以确定在状态 s 下选择动作 a 时在将来会获得的折算累积奖赏。

Q_learning 的迭代公式如(14)

$$Q(s,a):=Q(s,a)+\alpha(r+\gamma\max_a Q(s_i,a_i)-Q(s,a)) \tag{14}$$

算法在初始过程中初始每个$Q(s,a)$值（为了计算简便，大都设为 0），然后根据贪心策略选择最大的 Q 值，得到训练规则$<s,a,r,s>$，再通过式(14)得到实际迭代值，当到达目标状态时此次迭代过程结束，再继续从初始状态进行迭代，直至学习过程结束。

如果每个状态—动作对被无限频繁的访问，且α衰减合适，那么 Q 值就会最终收敛到Q^*，其收敛性已经得到证明[1]。

2.3 Dyna 算法

Dyna 算法是一种 Model_based 的强化学习算法，它利用经验来建立一个模型（即$\hat{T}$和$\hat{R}$），再利用经验来调整策略，同时也利用 model 来调整策略。它与环境交互的循环迭代过程如下：

(1) 更新 model，即在状态 s 下采用动作 a 到达s'并接收到 r，统计这些数据，更新$\hat{T}$和$\hat{R}$。

(2)根据上述得到的 model，利用式(15)来更新策略

$$Q(s,a)=\hat{R}(s,a)+\gamma\sum_{s'}\hat{T}(s,a,s')\max_{a'}Q(s',a') \tag{15}$$

(3) 随机选择 k 个状态—动作对，并按照式(16)进行值函数迭代

$$Q(s_k,a_k)=\hat{R}(s_k,a_k)+\gamma\sum_{s'}\hat{T}(s_k,a_k,s')\cdot\max_{a'}Q(s',a') \tag{16}$$

(4)根据当前 Q 值在状态s'下选择动作a'。转至(1)，继续进行迭代。

Dyna 算法每次大约需要 k 次 Q_learning 算法的计算量，但它依然比单纯的 Model_based 算法的计算量要少的多。其中 k 值是由计算量和选择动作的相对快慢决定的[4]。

2.4 Prioritized Sweeping 算法

尽管 Dyna 算法已经有了很大的进步，但它依然受困于相对盲目等缺点，特别是当 Agent 已处于 goal 或 dead 状态时。这些问题在 Prioritized Sweeping 算法中得到了解决。

Prioritized Sweeping 也是 Model_based 算法，除了下述的一点外很相近于 Dyna 算法，即更新不再是随机的选择，而 Value 值也通过值迭代与状态相联系。在这一算法中，为了有更合适的选择，必须在模型中增加额外的信息，即每一状态需记住它的前驱和每一状态都有一个优先权值（一般设初始值为 0）[6]。

它的思想即不再更新 k 个随机的状态—动作值，而是更新 k 个状态中优先权值最高的。对于每个权值最高的状态 s 来说，按照如下进行运算：

(1) 记下当前状态值：$V_{old}=V(s)$；

(2) 通过(17)来更新状态值

$$V(s)=\max_a(\hat{R}(s,a)+\gamma\sum_{s'}\hat{T}(s,a,s')V(s')) \tag{17}$$

设状态 s 的优先权值为 0。

(3) 计算 Value 值的相对差：$\Delta=|V_{old}-V(s)|$；

(4) 利用Δ来修改 s 前驱的优先权值。

如果我们更新了 s'的 V 值，并且有相对差值Δ，那么 s'的前驱也将受到这一更新的影响。对于任一状态 s(它存在一动作 a，并且$\hat{T}(s,a,s')\neq 0$)，都会将它的权值提高到$\Delta\cdot\hat{T}(s,a,s')$，除非它的优先权值已经超过它的 V 值。

从全局行为的角度来描述这一算法就是：当在真实的状态转换过程中达到了这一类状态，如已经达到了 goal 状态，那么这些计算将会传播到相应的前驱状态中；当转换到了另一类状态，如转换到的真实状态非常类似于我们预期的结果，那么这些计算将会继续在应得的状态空间中[8]。

2.5 Sarsa 算法

Sarsa 算法是 Rummery 和 Niranjan 于 1994 年提出的一种基于模型的算法，最初被称为改进的 Q_learning 算法，它采用实际的 Q 值进行迭代，而不是 Q_learning 所采用的值函数的最大值进行迭代。它的迭代公式如式(18)

$$Q(s_t,a_t)=Q(s_t,a_t)+\alpha(r_{t+1}+\gamma Q(s_{t+1},a_{t+1})-Q(s_t,a_t)) \tag{18}$$

其中，它的一步 Sarsa 算法已被 S.Singh 证明是收敛的。

3 强化学习的应用

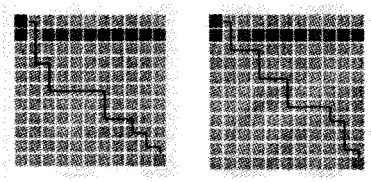
强化学习能够成为研究热点的重要原因是它可以作为研究 Agent 学习原理的理论工具。由于它具有的状态—动作特性和学习控制策略以使累积奖赏最大化等特点，它在生产优化问题，如选择一系列的动作，使工厂生产出的货物减去成本能得到最大的价值；在序列调度问题中，如在大型的公交车枢纽中心，如何调度公交车运行，以使车辆的整体运行最有效等方面都取得广泛的应用。其中，强化学习应用最广泛的方面就是棋类游戏和机器人控制等领域。

棋类游戏在人工智能研究领域始终占有重要的位置[7]。许多学者都试图将强化学习应用到这一领域，并取得一定的成就。最著名的是 Tesauro 把瞬时差分法应用于 Backgammon。Backgammon 大约有 10^{20} 个状态，把它做成基于表格的强化学习方法显然是不行的。Tesauro 采用三层 BP 神经网络把棋盘

上的棋子位置与棋手的获胜概率联系起来 (board position→probability of victory for current player)，通过训练取得在 40 盘比赛中负 1 盘的最好战绩[4]。

强化学习在机器人等智能控制领域也取得了长足的发展，Crites 和 Barto 将 Q_learning 算法应用到电梯调度中[10]；强化学习在智能控制应用的典型实例，就是倒摆控制系统，倒摆控制是一个非线性不稳定系统，许多强化学习的文章都把这一控制系统作为验证各种强化学习算法的实验系统，即当倒摆保持平衡时，得到奖励，倒摆失败时，得到惩罚，这样倒摆在一段时间时间之后就能达到平衡。Hee Rak Beem 利用模糊逻辑和强化学习实现陆上移动机器人导航系统，可以完成避碰和到达指定目标点两种行为。Winfried llg 采用强化学习来使 6 足昆虫机器人学会 6 条腿的协调动作[10]。Sebastian Thurn 采用神经网络结合强化学习方式使机器人通过学习能够到达室内环境中的目标[11]。

目前我们主要做了 Q_learning 算法的仿真实验平台。在这个平台中，我们分为确定性和非确定性两种情况，其界面设计有展示界面、参数设置、Q 值、收敛效果图等选项卡。在“界面展示”中，一个单元格代表一个状态，在一个状态中，共有可以向上、下、左、右 4 个方向的动作。在“参数设置”中，可以设置行数、列数，起点终点以及障碍数，其中我们设置蓝色为起点、橙色为终点、红色为障碍。在 Q 值选项卡中列出各个状态的 Q 值变化，在收敛效果图中以线状形式展示了在从起点到终点的收敛过程。在确定和非确定环境下的展示效果如图 2 所示。



(a)确定环境下　(b)非确定环境下

图 2　在确定和非确定环境下的展示效果

从上述实验中，我们可以分析出：确定性环境下的Q学习算法，经过训练，最终可以得到一个真实的最优策略(如图 3 所示)，而最确定性环境下最终不能完美收敛到最优策略(如图 4 所示)，在这一展示平台中，还可以显示出各个实验所需要的时间，上述两实验所需要的时间基本相当。

我们还对$TD(\lambda)$算法进行了比较分析：

从上述的$TD(\lambda)$算法中$\lambda=1$和 9 进行实验(如图 5~图 8 所示)，可以得出：$\lambda=1$和$\lambda=9$的情况下的TD算法均能经过训练收敛到真实的最优策略、在其余参数都相同的情况下，$\lambda=1$情况下的 TD 算法的收敛速度没有$\lambda=9$情况下的快。亦是在某些条件下，$TD(\lambda)$如果考虑更远的前瞻，训练会更加有效，得到了更好的验证。

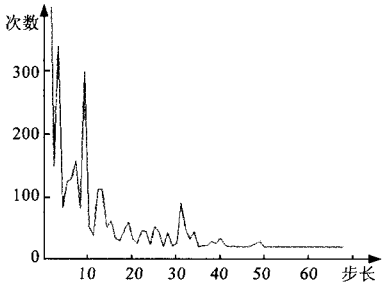


图 3　确定性 Q 学习算法训练的收敛仿真

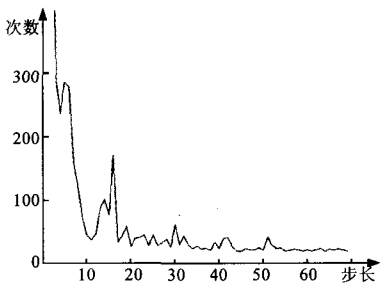


图 4　非确定性 Q 学习算法训练的收敛仿真

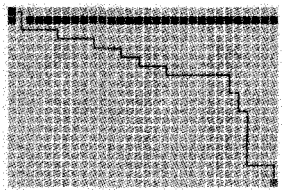


图 5　$\lambda=1$ 情况下的 TD 学习算法的界面

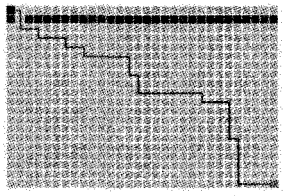


图 6　$\lambda=9$ 情况下的 TD 学习算法的界面

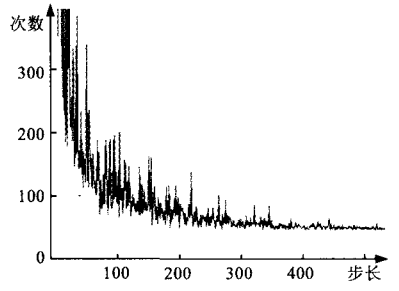


图 7　$\lambda=1$ 情况下 TD 算法训练的收敛仿真

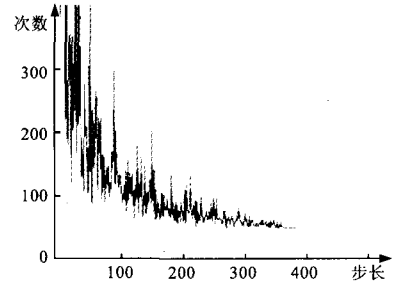


图 8　$\lambda=9$ 情况下 TD 算法训练的收敛仿真

4 结束语

强化学习是一种无导师的在线学习方法，它一般通过迭代来减小后继状态估计之间的差异来完成迭代过程。在马尔科夫环境下，Q_learning和$TD(\lambda)$等Model_free算法的收敛性已经得到证明，但在非MDP中的部分感知强化学习以及分层强化学习、逻辑强化学习等方面还有很多工作要做，这也是今后强化学习研究的重点。

在今后的研究中，我们将强化学习应用到智能搜索引擎中，即将搜索引擎应用于Splider技术中，在互联网中抓取网页时，将一个页面作为一个状态，网页中的链接作为动作，如果网页中的内容符合要求，我们就给一个正的奖赏值，如果不符合要求，那么奖赏值是0或者是负值，这也完全符合马尔科夫模型的要求，然后再用相关技术搭建出搜索引擎。

参考文献:

[1] Tom M Mitchell.Machine learning[M].Beijing,China:Machine Press,2004:263-280.

[2] 谭民,王硕,曹志强.多机器人系统[M].北京:清华大学出版社,2005:65-72.

[3] Dayan P.The convergence of TD (λ) for general λ[J].Machine Learning,1992(8):341-362.

[4] Kaelbling L P,Littman M L,Moore A W.Reinforcement learning: A survey[J].Journal of Artificial Intelligence Research,1996(4): 237-285.

[5] Watins P Dyna. Q_learning [J]. Machine Learning, 1992,8 (3): 279-292.

[6] Moor A W,Atkeson C G.Prioritized sweeping: Reinforcement learning with less data and less real time[J].Machine Learning, 1993,13:103-130.

[7] Hu J,Wellman M P.Nash Q-learning for general-sum stochastic games [J]. Journal of Machine Learning Research, 2003 (4): 1039-1069.

[8] Badtke S J,Barto R G.Linear least-squares algorithms for temporal difference learning [J]. Machine Learning, 1996,22 (1-3): 33-57.

[9] Bowling M.Convergence and no-regret in multiagent learning [C].Advances in Naural Information Processing Systems,2004.

[10] Winfried Ilg,Karsten Berns.A learning architecture based on for adaptive control of the walking machine LAURON[J].Robot and Autonomous System,1995,15:323-334.

[11] 马莉,蔡自兴.再励学习控制器结构与算法[J].模式识别与人工智能,1998,11(1):96-100.

[12] Lovejoy W S.A survey of algorithmic methods for partially observed Markov decision processes[C].Annals of Operations Research,1991,28:47-65.

[13] Precup D. Temporal abstraction in reinforcement learning [C]. University of Massachusetts, Amherstm:Doctoral Dissertation, 2000.

[14] Sutton R S,Barto A G.Reinforcement Learning[M].Cambridge, MA:MIT Press,1998.

[15] Crites R H,Barto A G.Elevator group control using multiple reinforcement learning agents [J]. Machine Learning, 1998,33 (2): 235-262.

[16] Roy N, Pineau J, Thrun S. Spoke dialogue managements using probabilistic reasoning[C].Hongkong: Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics,2000.

[17] 高阳,陈世福.强化学习研究综述[J].自动化学报,2004(1):86-98.

(上接第 5804 页)

参考文献:

[1] Eberhart R C,Shi Y H.Particle swarm optimization: Development,applications and resources[C].Proc Congress on Evolutionary Computation 2001. Piscataway, NJ: IEEE Press, 2001: 81-86.

[2] Kennedy J,Eberhart R C.Particle swarm optimizations[C]. Perth, Australia: Proceedings of IEEE International Conference on Neutral Networks,1995:1942-1948.

[3] 周驰,高海兵,高亮,等.粒子群优化算法[J].计算机应用研究,2003,20(12):7-11.

[4] 王俊伟,汪定伟.粒子群算法中惯性权重的实验与分析[J].系统工程学报,2005(4):194-198.

[5] 陈贵敏,贾建援,韩琪.粒子群优化算法的惯性权值递减策略研究[J].西安交通大学学报,2006(1):53-56.

[6] Clerc M.The swarm and queen: Towards a deterministic and adaptive particle swarm optimization[C]. Washington DC:Proceedings of Congress on Evolutionary Computation, 1999: 1951-1957.

[7] 高鹰.一种自适应扩展粒子群优化算法[J].计算机工程与应用,2006,42(15):12-15.

[8] Parsopoulos K E,Vrahatis M N.Parameter selection and adaptation in unified particle swarm optimization[J].Mathematical and Computer Modeling,2007,10:198-213.

[9] Srinivas Pasupuleti,Roberto Battiti.The gregarious particle swarm optimizer[C]. Seattle,Washington,USA:GECCO'06, 2006.

[10] Van den Bergh F, Engelbrecht A P. Using cooperative particle swarm optimization to train product unit neural networks [C]. Washington DC, USA: IEEE International Joint Conference on Neural Networks,2001.

[11] 杨维,李歧强.粒子群优化算法综述[J].中国工程科学,2004(5):87-93.

[12] Parsopoulos K E.Stretching technique for obtaining global minimizes through particle swarm optimizations[C].Proceedings of the Workshop on PSO,Indianapolis: Purdue School of Engineering and Technology,INPUI,2001.