一种批量最小二乘策略迭代方法

周 鑫[1] 刘 全[1,2] 傅启明[1] 肖 飞[1]
(苏州大学计算机科学与技术学院 苏州 215006)[1]
(符号计算与知识工程教育部重点实验室(吉林大学) 长春 130012)[2]

摘 要 策略迭代是一种迭代地评估和改进控制策略的强化学习方法。采用最小二乘的策略评估方法可以从经验数据中提取出更多有用信息,提高数据有效性。针对在线的最小二乘策略迭代方法对样本数据的利用不充分、每个样本仅使用一次就被丢弃的问题,提出一种批量最小二乘策略迭代算法(BLSPI),并从理论上证明其收敛性。BLSPI算法将批量更新方法与在线最小二乘策略迭代方法相结合,在线保存生成的样本数据,多次重复使用这些样本数据并结合最小二乘方法来更新控制策略。将BLSPI算法用于倒立摆实验平台,实验结果表明,该算法可以有效利用之前的经验知识,提高经验利用率,加快收敛速度。

关键词 强化学习,批量更新,最小二乘,策略迭代
中图法分类号 TP181 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2014.09.044

Batch Least-squares Policy Iteration

ZHOU Xin[1] LIU Quan[1,2] FU Qi-ming[1] XIAO Fei[1]
(Institute of Computer Science and Technology, Soochow University, Suzhou 215006, China)[1]
(Key Laboratory of Symbolic Computation and Knowledge Engineering (Jilin University), Ministry of Education, Changchun 130012, China)[2]

Abstract Policy iteration is a reinforcement learning method which evaluates and improves the control policy iteratively. Policy evaluation with the least-square method can extract more useful information from the empirical data and improve the data validity. For the low empirical utilization rate of online least-squares policy iteration method which uses each sample only once, a batch least-squares policy iteration (BLSPI) method was proposed and its convergence was proved in theory. BLSPI method combines online least-squares policy iteration method and batch updating method, stores the generated samplesonline and reuses these samples with least-squares methods to update the control policy. We applied the BLSPI method to the inverted pendulum system, and the experiment results show that the method can effectively utilize the previous experience and knowledge, improve the empirical utilization rate, and accelerate the convergence speed.

Keywords Reinforcement learning, Batch updating, Least-squares, Policy iteration

1 引言

强化学习(Reinforcement Learning,RL)是一种通过与环境的交互,将场景映射到动作,以获取最大累积奖赏的机器学习方法。学习者通过不断地尝试来发现能获得最大奖赏的动作。在强化学习实例中,动作不仅会影响立即奖赏,而且会影响下一状态及所有的后续奖赏。试错搜索和延迟奖赏是强化学习的两个最重要的特征[1,2]。强化学习在大空间、复杂非线性系统中具有良好的学习性能,在过程控制、任务调度、机器人设计和游戏等领域有着广泛的应用[3-5]。

在强化学习中,用值函数反映每个状态或状态动作对的长期累积奖赏,反映状态的称为V值函数,反映状态动作对的称为Q值函数。强化学习中的策略为状态空间到动作空间的映射,学习者最终通过值函数确定到达目标的最优策略[6]。策略迭代(Policy Iteration,PI)是一种重要的强化学习方法。策略迭代算法通过计算更新当前的值函数来评估策略,并采用贪心方法来改进策略。最小二乘方法可以有效利用到强化学习算法中。Bradtke,Barto等人提出了基于V值函数的最小二乘时间差分(Least-Squares Temporal Difference,LSTD)算法[7,8]。与传统的时间差分算法相比,尽管每个时间步需要更大的计算量,但能从经验数据中提取更多有用信息,提高数据有效性。然而,由于无法在无模型的情况下

到稿日期:2013-11-14 返修日期:2014-03-22 本文受国家自然科学基金项目(61070223,61103045,61070122,61272005,61303108),江苏省自然科学基金(BK2012616),江苏省高校自然科学研究项目(09KJA520002,09KJB520012,13KJB520020),吉林大学符号计算与知识工程教育部重点实验室资助项目(93K172012K04)资助。

周 鑫(1989-),男,硕士生,主要研究方向为强化学习,E-mail:jaljaljal.love@163.com;刘 全(1969-),男,博士后,教授,博士生导师,CCF高级会员,主要研究方向为强化学习、智能信息处理、自动推理;傅启明(1985-),男,博士生,CCF学生会员,主要研究方向为强化学习、贝叶斯推理、遗传算法;肖 飞(1988-),男,硕士,主要研究方向为强化学习、支持向量机。

选择动作，LSTD只能用于解决预测问题，而无法解决控制问题[9-11]。Lagoudakis等人将其扩展到基于Q值函数的最小二乘时间差分(Least-Squares Temporal Difference for Q-functions，LSTD-Q)学习中[12,13]，并提出了可用于解决控制问题的完整的最小二乘策略迭代(Least-Squares Policy Iteration，LSPI)算法。LSPI的策略评估部分采用LSTD-Q，用提前采集的所有样本数据准确评估当前策略，是一个典型的离线(offline)算法。然而，强化学习的一个主要目标是研究与环境交互的在线学习算法。Busoniu等人将离线的最小二乘策略迭代算法扩展到在线的情况[14]，提出了在线最小二乘策略迭代(online Least-Squares Policy Iteration，online LSPI)算法。该算法利用具有探索性的策略在线生成样本数据，并且每产生少量样本后就改进策略。但其经验利用率不高，对经验数据仅利用一次，利用过后就丢弃。因此，在算法执行初期，可用样本数据很少，难以得到较好的控制策略，导致算法的初始性能较差，收敛速度较慢。

批量更新方法可以重复利用过去的经验数据，从而减少所需样本数据的数量[15]。该方法将在线生成的经验数据存储起来，并多次重复使用这些经验数据来更新学到的策略。本文基于线性最小二乘函数逼近原理，将批量更新方法与online LSPI相结合，提出批量最小二乘策略迭代(Batch Least-Squares Policy Iteration，BLSPI)算法。该算法可以多次利用经验数据，从中提取出更多信息，提高经验利用率，从而提高算法的收敛速度。

2 相关理论

2.1 马尔可夫决策过程

强化学习问题是基于马尔可夫决策过程(Markov Decision Process，MDP)的。一个成功保留所有相关信息的状态信号称为马尔可夫的，或者说是具有马尔可夫性质。在强化学习中，如果状态信号具有马尔可夫性质，那么环境在$t+1$时刻的响应只取决于在t时刻的状态和动作的表示。满足马尔可夫性质的强化学习任务被称为是MDP。MDP可以由以下4个因素来定义：状态空间X、动作空间U、转移概率函数f、奖赏函数ρ。在确定性情况下，状态转移函数为$f:X\times U\to[0,1]$，奖赏函数为$\rho:X\times U\to R$。在时间步k给定当前状态x_k，控制器根据控制策略$h:X\to U$采取动作u_k，转移到下一个状态$x_{k+1}=f(x_k,u_k)$，得到的立即奖赏为$r_{k+1}=\rho(x_k,u_k)$。在随机情况下，状态转移函数为$\tilde{f}:X\times U\times X\to[0,\infty)$，奖赏函数为$\tilde{\rho}:X\times U\times X\to R$。在时间步$k$给定当前状态$x_k$，控制器根据控制策略$h:X\to U$采取动作$u_k$，转移到下一个状态$x_{k+1}$($x_{k+1}\in X_{k+1}$，$X_{k+1}\subseteq X$)的概率为$P(x_{k+1}\in X_{k+1}\mid x_k,u_k)=\int_{X_{k+1}}\tilde{f}(x_k,u_k,x')\mathrm{d}x'$，得到的立即奖赏为$r_{k+1}=\tilde{\rho}(x_k,u_k,x_{k+1})$。如果状态空间是可数的，状态转移函数可写为$\bar{f}:X\times U\times X\to[0,1]$，转移到下一个状态$x'$的概率为$P(x_{k+1}=x'\mid x_k,u_k)=\bar{f}(x_k,u_k,x')$。给定$f$和$\rho$，以及当前状态$x_k$和当前采取的动作$u_k$，足以确定下一个状态$x_{k+1}$及立即奖赏$r_{k+1}$，这就是马尔可夫性。它是强化学习算法的理论依据。

有些MDP有终止状态，一旦到达此状态，就不能再转移出去，这类任务称为情节式任务；反之，没有终止态的称为连续式任务。特别地，没有不可到达的状态，每个状态被访问到的概率都不为0的MDP称为是可遍历的。

2.2 在线最小二乘策略迭代算法

PI算法从任意初始策略h_0开始，在每轮迭代$l(l\geqslant 0)$中，算法评估当前策略，计算Q值函数Q^{h_l}，然后用$h_{l+1}(x)=\arg\max_u Q^{h_l}(x,u)$改进策略。PI算法最终收敛到最优策略$h^*$。

假设状态空间X和动作空间U中包含有限个元素，其中$X=\{x_1,\cdots,x_{\bar{N}}\}$，$U=\{u_1,\cdots,u_{\bar{M}}\}$。定义基函数矩阵$\phi\in\mathbb{R}^{\bar{N}\bar{M}\times n}$，如式(1)所示：

$$\phi_{[i,j],l}=\phi_l(x_i,u_j)(l=0,1,2,\cdots,n) \tag{1}$$

式中，n为基函数维数，取$n=\bar{N}\ \bar{M}$。在线性参数条件下，根据基函数ϕ，对于给定的策略参数$\theta(\theta\in\mathbb{R}^n)$，近似$Q$值函数可用式(2)表示：

$$\hat{Q}=\phi^{\mathrm{T}}\theta \tag{2}$$

PI算法通过迭代更新策略参数θ来获取最优策略。在LSPI算法中，基于提前采集的所有样本通过LSTD-Q来计算策略参数θ，计算等式如式(3)所示：

$$\frac{1}{n_s}\Gamma\theta=\gamma\frac{1}{n_s}\Lambda\theta+\frac{1}{n_s}z \tag{3}$$

式中，n_s为样本数量，矩阵$\Gamma,\Lambda\in\mathbb{R}^{n\times n}$，向量$z\in\mathbb{R}^n$，$\Gamma,\Lambda,z$的定义如式(4)所示：

$$\begin{cases}\Gamma=\sum_{l_s=1}^{n_s}\phi(x_{l_s},u_{l_s})\phi^{\mathrm{T}}(x_{l_s},u_{l_s})\\ \Lambda=\sum_{l_s=1}^{n_s}\phi(x_{l_s},u_{l_s})\phi^{\mathrm{T}}(x'_{l_s},h(x'_{l_s}))\\ z=\sum_{l_s=1}^{n_s}\phi(x_{l_s},u_{l_s})r_{l_s}\end{cases} \tag{4}$$

LSPI是离线算法，每次对θ的更新都需要采用提前采集的所有n_s个样本数据。在online LSPI中，仍用LSTD-Q计算策略参数θ，不同的是每经过规定的少量时间步数(连续式任务)或情节数(情节式任务)就用当前已收集到的样本数据更新θ，然后用更新后的θ继续在线产生新的样本，用于评估计算下一轮的θ。

3 批量最小二乘策略迭代算法

3.1 批量最小二乘策略迭代算法

在Online LSPI算法中，在线产生的样本数据仅被使用一次，而后不会再次使用，这使得算法经验利用率不高，尤其是在产生样本比较困难的情况下，如何提高经验的利用率显得尤为重要。本文使用批量更新算法，将在线产生的样本数据保存下来，以备重复利用。样本数据以四元组(x,u,r,x')的形式进行存储，其中，x表示当前状态，u表示当前策略下所选择的动作，r表示得到的立即奖赏，x'表示转移到的下一个状态。保存的样本数据构成了经验，连同当前策略h组成五元组$(x,u,r,x',h(x'))$来更新策略(策略h与策略参数θ相关，不同的θ对应于不同的h，$h(x')$表示在状态x'下应该采取的动作)。策略的更新不是在每个时间步都进行，而是在规定的k个情节(情节式任务)或T个时间步(连续式任务)后进行。每次都用当前采集到的所有样本更新策略参数θ，这样，相应的策略h被更新，再利用这些样本数据组成新的五元组，在新的策略下再次更新θ，如此重复执行d次，实现经验的重复利用。每次更新后重新开始收集样本数据，以防止

样本数据过多而导致计算量过大。

这里采用 LSTD-Q 评估策略，由式(3)计算 θ 需要完成矩阵求逆的运算，矩阵求逆运算计算量大，计算复杂度高，因此用 Sherman-Morrison 公式增量式计算求解 θ。Sherman-Morrison 公式如式(5)所示：

$$(A+uv^{\mathrm{T}})^{-1}=A^{-1}-\frac{A^{-1}uv^{\mathrm{T}}A^{-1}}{1+v^{\mathrm{T}}A^{-1}u} \tag{5}$$

式中，A 为方阵，u,v 为列向量，$1+v^{\mathrm{T}}A^{-1}u\neq 0$。

假设当前正在处理第 l_s 个样本数据，由式(3)、式(4)得：

$$\begin{aligned}&(\Gamma_{l_s}-\gamma\Lambda_{l_s})^{-1}\\&=(\Gamma_{l_s-1}+\phi(x_{l_s},u_{l_s})\phi^{\mathrm{T}}(x_{l_s},u_{l_s})-\gamma\Lambda_{l_s-1}-\gamma\phi(x_{l_s},u_{l_s})\phi^{\mathrm{T}}(x'_{l_s},h(x'_{l_s})))^{-1}\\&=(\Gamma_{l_s-1}-\gamma\Lambda_{l_s-1}+\phi(x_{l_s},u_{l_s})\phi^{\mathrm{T}}(x_{l_s},u_{l_s})-\gamma\phi(x_{l_s},u_{l_s})\phi^{\mathrm{T}}(x'_{l_s},h(x'_{l_s}))^{-1}\\&=(\Gamma_{l_s-1}-\gamma\Lambda_{l_s-1}+\phi(x_{l_s},u_{l_s})(\phi(x_{l_s},u_{l_s})-\gamma\phi(x'_{l_s},h(x'_{l_s}))^{\mathrm{T}})^{-1}\end{aligned}$$

将其代入 Sherman-Morrison 公式，令 B_{l_s} 表示 $(\Gamma_{l_s}-\gamma\Lambda_{l_s})$ 的逆矩阵，得：

$$B_{l_s}=B_{l_s-1}-\frac{B_{l_s-1}\phi(x_{l_s},u_{l_s})(\phi(x_{l_s},u_{l_s})-\gamma\phi(x'_{l_s},h(x'_{l_s}))^{\mathrm{T}}B_{l_s-1}}{1+(\phi(x_{l_s},u_{l_s})-\gamma\phi(x'_{l_s},h(x'_{l_s}))^{\mathrm{T}}B_{l_s-1}\phi(x_{l_s},u_{l_s})} \tag{6}$$

其中，$1+(\phi(x_{l_s},u_{l_s})-\gamma\phi(x'_{l_s},h(x'_{l_s}))^{\mathrm{T}}B_{l_s-1}\phi(x_{l_s},u_{l_s})\neq 0$。

这样，通过递推更新矩阵 B，省去了矩阵求逆的过程，减小了计算量。采用 LSTD-Q 评估策略的 BLSPI 算法如算法 1 所示。

算法 1 采用 LSTD-Q 评估策略的 BLSPI 算法

1. $B=\beta_\Gamma I$(β_Γ 为一较小的常数)，$z=0$，$n_s=0$(n_s 为当前样本数)，$i=0$(i 为计数器)；
2. repeat
3. 在当前状态 x_{l_s} 下，根据一定策略(如 ε-greedy)选择动作 u_{l_s}($l_s=1,2,3,\cdots$)；
4. 执行动作 u_{l_s}，保存样本数据(x_{l_s}，u_{l_s}，r_{l_s}，x'_{l_s})；
5. $l_s=l_s+1$，$n_s=n_s+1$；
6. 若 x_{l_s} 是第 k 个情节的终止状态(情节式任务)或 n_s 等于规定的批量更新标准 T 的整数倍(连续式任务)
7. repeat
8. $l_s=1$；
9. repeat
10. $h(x'_{l_s})=\arg\max_u\phi^{\mathrm{T}}(x_{l_s},u)\theta$；
11. $B=B-\frac{B\phi(x_{l_s},u_{l_s})(\phi(x_{l_s},u_{l_s})-\gamma\phi(x'_{l_s},h(x'_{l_s}))^{\mathrm{T}}B}{1+(\phi(x_{l_s},u_{l_s})-\gamma\phi(x'_{l_s},h(x'_{l_s}))^{\mathrm{T}}B\phi(x_{l_s},u_{l_s})}$；
12. $z=z+\phi(x_{l_s},u_{l_s})r_{l_s}$；
13. $l_s=l_s+1$；
14. until $l_s=n_s$；
15. $\theta=n_sB\frac{1}{n_s}z$；
16. $i=i+1$；
17. until $i=d$
18. $l_s=0,n_s=0,i=0$；
19. 更新当前状态 $x_{l_s}=x'_{l_s-1}$；
20. until 满足设定的终止条件。

Jung，Polani 等人提出的关于 Q 值函数的最小二乘策略评估算法(Least-Squares Policy Evaluation for Q-functions，LSPE-Q)也是一种利用类似式(3)处理样本数据的策略评估算法[16,17]。算法的初始参数向量 θ_0 可以取任意值，更新公式如式(7)所示：

$$\theta_{l_s}=\theta_{l_s-1}+\alpha(\theta^+_{l_s}-\theta_{l_s-1}) \tag{7}$$

其中，$\theta^+_{l_s}=(\frac{1}{l_s}\Gamma_{l_s})^{-1}(\gamma\frac{1}{l_s}\Lambda_{l_s}\theta_{l_s-1}+\frac{1}{l_s}z_{l_s})$，$\alpha$ 为步长参数，l_s 为正在处理的样本数据编号。同样采用 Sherman-Morrison 公式消去矩阵求逆过程，由式(4)得：

$$\begin{aligned}\Gamma^{-1}_{l_s}&=(\Gamma_{l_s-1}+\phi(x_{l_s},u_{l_s})\phi^{\mathrm{T}}(x_{l_s},u_{l_s}))^{-1}\\&=(\Gamma^{-1}_{l_s-1}-\frac{\Gamma^{-1}_{l_s-1}\phi(x_{l_s},u_{l_s})\phi^{\mathrm{T}}(x_{l_s},u_{l_s})\Gamma^{-1}_{l_s-1}}{1+\phi^{\mathrm{T}}(x_{l_s},u_{l_s})\Gamma^{-1}_{l_s-1}\phi(x_{l_s},u_{l_s})})\end{aligned}$$

令 $B_{l_s}=\Gamma^{-1}_{l_s}$，得：

$$B_{l_s}=(B_{l_s-1}-\frac{B_{l_s-1}\phi(x_{l_s},u_{l_s})\phi^{\mathrm{T}}(x_{l_s},u_{l_s})B_{l_s-1}}{1+\phi^{\mathrm{T}}(x_{l_s},u_{l_s})B_{l_s-1}\phi(x_{l_s},u_{l_s})}) \tag{8}$$

不同于采用 LSTD-Q 评估策略的 BLSPI 算法，采用 LSPE-Q 评估策略的 BLSPI 算法处理每个样本数据后都要更新策略参数 θ，θ 的更新依赖于上一轮的结果，因此 θ 值与处理样本次序有关；并且，θ 的更新还受步长参数 α 的影响，算法稳定性不如每次用当前采集的所有样本更新 θ 且消去了步长参数的采用 LSTD-Q 评估策略的 BLSPI 算法。但是，如果给定的初始值较好，通过逐步减小步长参数 α，算法往往能取得较好的性能。采用 LSPE-Q 评估策略的 BLSPI 算法如算法 2 所示。

算法 2 采用 LSPE-Q 评估策略的 BLSPI 算法

1. $B=\beta_\Gamma I$(β_Γ 为一较小的常数)，$z=0$，$n_s=0$(n_s 为当前样本数)，$i=0$(i 为计数器)；
2. repeat
3. 在当前状态 x_{l_s} 下，根据一定策略(如 ε-greedy)选择动作 u_{l_s}($l_s=1,2,3,\cdots$)；
4. 执行动作 u_{l_s}，保存样本数据(x_{l_s}，u_{l_s}，r_{l_s}，x'_{l_s})；
5. $l_s=l_s+1$，$n_s=n_s+1$；
6. 若 x_{l_s} 是第 k 个情节的终止状态(情节式任务)或 n_s 等于规定的批量更新标准 T 的整数倍(连续式任务)
7. repeat
8. $l_s=1$；
9. repeat
10. $h(x'_{l_s})=\arg\max_u\phi^{\mathrm{T}}(x_{l_s},u)\theta_{l_s}$；
11. $B_{l_s}=(B_{l_s-1}-\frac{B_{l_s-1}\phi(x_{l_s},u_{l_s})\phi^{\mathrm{T}}(x_{l_s},u_{l_s})B_{l_s-1}}{1+\phi^{\mathrm{T}}(x_{l_s},u_{l_s})B_{l_s-1}\phi(x_{l_s},u_{l_s})})$；
12. $\Lambda_{l_s}=\Lambda_{l_s-1}+\phi(x_{l_s},u_{l_s})\phi^{\mathrm{T}}(x_{l_s},h(x_{l_s}))$；
13. $z=z+\phi(x_{l_s},u_{l_s})r_{l_s}$；
14. $\theta_{l_s}=\theta_{l_s-1}+\alpha(\theta^+_{l_s}-\theta_{l_s-1})$，其中，$\theta^+_{l_s}=(l_sB_{l_s})(\gamma\frac{1}{l_s}\Lambda_{l_s}\theta_{l_s-1}+\frac{1}{l_s}z_{l_s})$；
15. $l_s=l_s+1$；
16. until $l_s=n_s$；
17. $i=i+1$；
18. until $i=d$
19. $l_s=0,n_s=0,i=0$；
20. 更新当前状态 $x_{l_s}=x'_{l_s-1}$；
21. until 满足设定的终止条件。

3.2 算法分析

为了保证算法的收敛，必须用有探索性的策略生成样本数据。如果仅能得到形如$(x,h(x))$的状态动作对，那么当

$u \neq h(x)$时，状态动作(x,u)的信息就无法获得(即相应的权重$w(x,u)$的值为0)。因此，无法对相应的状态动作对的Q值函数进行估计，而且也无法根据近似Q值函数进行策略改进。为了避免这个问题，在一定的条件下，不属于$h(x)$的动作u也应该被选中，比如，按照一定的随机概率选择这些动作。对于给定的稳定探索策略，算法可以用于评估新的探索策略。

引理1 对任意马尔可夫链，若满足：(1)θ_t是通过BLSPI算法得到的；(2)每个状态动作对(x,u)都可以被无限次访问到；(3)从长远来看，每个状态动作对(x,u)被访问到的概率以概率1趋近$w(x,u)$；(4)$[\boldsymbol{\phi}^{\mathrm{T}}\boldsymbol{w}(I-\gamma\bar{\boldsymbol{f}}\boldsymbol{h})\boldsymbol{\phi}]$可逆，那么，等式

$$\theta_{BLSPI}=[\boldsymbol{\phi}^{\mathrm{T}}\boldsymbol{w}(I-\gamma\bar{\boldsymbol{f}}\boldsymbol{h})\boldsymbol{\phi}]^{-1}[\boldsymbol{\phi}^{\mathrm{T}}\boldsymbol{w}\boldsymbol{r}]$$

以概率1成立。其中，$\boldsymbol{\phi}\in\mathbb{R}^{N M\times N M}$为第$x$列为$\phi(x,u)$的基函数矩阵；$\boldsymbol{w}\in\mathbb{R}^{N M\times N M}$为对角矩阵$(w(x,u))$，$w(x,u)$表示状态动作对$(x,u)$的权重，且$\sum_{x\in X}\sum_{u\in U}w(x,u)=1$；$\bar{\boldsymbol{f}}\in\mathbb{R}^{N M\times N}$为状态转移概率矩阵，$\bar{\boldsymbol{f}}_{[i,j],i'}=\bar{f}(x_i,u_j,x_{i'})$；$\boldsymbol{h}\in h^{N\times N M}$为关于策略的矩阵，$\boldsymbol{h}_{i',[i,j]}=h(x_{i'},x_i,u_j)$，当$i'=i,h(x_i)=u_j$时，$\boldsymbol{h}_{i',[i,j]}=1$，否则$\boldsymbol{h}_{i',[i,j]}=0$。

证明：在每个时间步都进行更新的在线LSPI算法中，在时间步t，由式(3)可得：

$$\theta_t=[\frac{1}{t}\sum_{k=1}^{t}\phi_k(\phi_k-\gamma\phi_{k'})^{\mathrm{T}}]^{-1}[\frac{1}{t}\sum_{k=1}^{t}\phi_k r_k]$$

在经验重复利用次数为d的BLSPI算法中，上式可进一步变形为：

$$\theta_t=[\frac{1}{dt}\sum_{k=1}^{t}\sum_{i=1}^{d}\phi_k(\phi_k-\gamma\phi_{k_i'})^{\mathrm{T}}]^{-1}[\frac{1}{t}\sum_{k=1}^{t}\phi_k r_k]$$

当$t\to\infty$时，由条件(2)，每个状态动作对(x,u)都可以被无限次访问到，并且状态动作对间的转移概率以概率1逼近其真实转移概率$\bar{f}$。同时，由条件(3)，每个状态动作对(x,u)被访问到的概率以概率1趋近$w(x,u)$。因此，给定条件(4)，BLSPI算法的θ_{BLSPI}可以评估为：

$$\begin{aligned}\theta_{BLSPI}&=\lim_{t\to\infty}\theta_t\\&=\lim_{t\to\infty}[\frac{1}{nt}\sum_{k=1}^{t}\sum_{i=1}^{d}\phi_k(\phi_k-\gamma\phi_{k_i'})^{\mathrm{T}}]^{-1}[\frac{1}{t}\sum_{k=1}^{t}\phi_k r_k]\\&=[\lim_{t\to\infty}\frac{1}{nt}\sum_{k=1}^{t}\sum_{i=1}^{d}\phi_k(\phi_k-\gamma\phi_{k_i'})^{\mathrm{T}}]^{-1}[\lim_{t\to\infty}\frac{1}{t}\sum_{k=1}^{t}\phi_k r_k]\\&=[\sum_{x\in X}\sum_{u\in U}w(x,u)\sum_{x'\in X}\bar{f}(x,u,x')h(x',x',u')\phi(x,u)(\phi(x,u)-\gamma\phi(x',u'))^{\mathrm{T}}]^{-1}[\sum_{x\in X}\sum_{u\in U}w(x,u)\phi(x,u)\sum_{x'\in X}\bar{f}(x,u,x')r(x,u,x')]\\&=[\boldsymbol{\phi}^{\mathrm{T}}\boldsymbol{w}(I-\gamma\bar{\boldsymbol{f}}\boldsymbol{h})\boldsymbol{\phi}]^{-1}[\boldsymbol{\phi}^{\mathrm{T}}\boldsymbol{w}\boldsymbol{r}]\end{aligned}$$

定理1 对遍历马尔可夫链(ergodic Markov chains)，若满足：(1)代表状态动作的特征向量集合$\{\phi(x,u)\mid x\in X,u\in U\}$是线性无关的；(2)每个特征向量$\phi(x,u)$的维数$n=|X|\times|U|$；(3)折扣因子$0\leqslant\gamma<1$，那么，当状态转移次数趋近于无穷时，通过BLSPI算法得出的渐近参数θ_{BLSPI}以概率1收敛到其最优值θ^*。

证明：由于是遍历马尔可夫链，当时间步t趋于无穷时，每个状态动作对$(x,u)\in(X\times U)$被访问到的次数以概率1趋近于无穷。同时，每个状态动作对被访问到的概率以概率1趋近于$w(x,u)$，并且对任意$(x,u)\in(X\times U)$，都有$w(x,u)>0$。因此，矩阵$\boldsymbol{w}$可逆。由条件(2)，$\boldsymbol{\phi}$为$n\times n$的方阵，又由条件(1)，$\boldsymbol{\phi}$的秩为n，所以$\boldsymbol{\phi}$可逆。由条件(3)，$0\leqslant\gamma<1$，有$(I-\gamma\bar{\boldsymbol{f}}\boldsymbol{h})$可逆。因此，由引理1，等式$\theta_{BLSPI}=[\boldsymbol{\phi}^{\mathrm{T}}\boldsymbol{w}(I-\gamma\bar{\boldsymbol{f}}\boldsymbol{h})\boldsymbol{\phi}]^{-1}[\boldsymbol{\phi}^{\mathrm{T}}\boldsymbol{w}\boldsymbol{r}]$以概率1成立。

立即奖赏$r\in\mathbb{R}^{N M}$可以计算如下($r(x,u,x',u')$表示从状态动作对(x,u)到(x',u')的立即奖赏)：

$$\begin{aligned}&r(x,u,x',u')\\&=Q(x,u)-\gamma\sum_{x\in X}\sum_{u\in U}\bar{f}(x,u,x')h(x',x',u')Q(x',u')\\&=\phi^{\mathrm{T}}(x,u)\theta^*-\gamma\sum_{x\in X}\sum_{u\in U}\bar{f}(x,u,x')h(x',x',u')\phi^{\mathrm{T}}(x',u')\theta^*\\&=(\phi(x,u)-\gamma\sum_{x\in X}\sum_{u\in U}\bar{f}(x,u,x')h(x',x',u')\phi(x',u'))^{\mathrm{T}}\theta^*\end{aligned}$$

写成矩阵形式为：

$$r=(I-\gamma\bar{\boldsymbol{f}}\boldsymbol{h})\boldsymbol{\phi}\theta^*$$

代入θ_{BLSPI}的计算公式，得：

$$\begin{aligned}\theta_{BLSPI}&=[\boldsymbol{\phi}^{\mathrm{T}}\boldsymbol{w}(I-\gamma\bar{\boldsymbol{f}}\boldsymbol{h})\boldsymbol{\phi}]^{-1}[\boldsymbol{\phi}^{\mathrm{T}}\boldsymbol{w}\boldsymbol{r}]\\&=[\boldsymbol{\phi}^{\mathrm{T}}\boldsymbol{w}(I-\gamma\bar{\boldsymbol{f}}\boldsymbol{h})\boldsymbol{\phi}]^{-1})[\boldsymbol{\phi}^{\mathrm{T}}\boldsymbol{w}(1-\gamma\bar{\boldsymbol{f}}\boldsymbol{h})\boldsymbol{\phi}]\theta^*\\&=\theta^*\end{aligned}$$

因此，θ_{BLSPI}以概率1收敛到θ^*。

由于采用有探索的策略产生样本数据，当$t\to\infty$时，每个状态动作对(x,u)都可以被无限次访问到，并且状态动作对间的转移概率以概率1逼近其真实转移概率f。同时，每个状态动作对(x,u)被访问到的概率以概率1趋近$w(x,u)$，且$w(x,u)>0$，满足遍历马尔可夫链的性质。因此，采用维数$n=|X|\times|U|$的线性无关的高斯径向基函数$\phi(x,u)(x\in X,u\in U)$编码，并取折扣因子$0\leqslant\gamma<1$，由定理1可知，BLSPI算法收敛。

从计算的角度来看，直接由式(3)计算θ需要完成矩阵求逆的运算，矩阵求逆计算复杂度是$O(n^3)$(n为径向基函数的维数)。对于情节式任务，设批量更新间隔为k，样本数据重复利用次数为d，一个更新间隔内收集到的样本数为m，在采用LSTD-Q评估策略的BLSPI算法中，每个更新间隔要计算θd次，平均每个情节的计算复杂度为$O(\frac{d}{k}n^3)$；在采用LSPE-Q评估策略的BLSPI算法中，处理每个样本时都要更新θ，其计算复杂度为$O(m\frac{d}{k}n^3)$。通过Sherman-Morrison公式增量式计算求解θ，可消除矩阵求逆的过程，将每个情节的计算复杂度降低至$O(m\frac{d}{k}n^2)$，减小计算量。计算复杂度与d成正比，与k成反比。d越大，经验重复利用次数越多，越能更快学习到最优策略，计算量也越大。适量增大k的值可以减少策略更新的次数，从而在不减少样本数据的情况下减小计算量。Online LSPI算法与采用LSTD-Q、LSPE-Q评估策略的BLSPI算法的计算复杂度如表1所列。

表1 online LSPI算法与采用LSTD-Q、LSPE-Q评估策略的BLSPI算法的计算复杂度

一个情节内的平均计算复杂度	online LSPI	采用LSTD-Q评估策略	采用LSPE-Q评估策略
不使用Sherman-Morrison公式	$O(n^3)$	$O(\frac{d}{k}n^3)$	$O(m\frac{d}{k}n^3)$
使用Sherman-Morrison公式	$O(mn^2)$	$O(m\frac{d}{k}n^2)$	$O(m\frac{d}{k}n^2)$

4 实验及结果分析

4.1 实验描述

为了验证所提算法的性能，本文对倒立摆系统进行仿真实验。倒立摆系统是在水平轨道放置一辆可左右移动的带有倒立摆的小车，倒立摆可绕轴在轨道平面内自由转动。实验目的是通过给小车施加不同的力使倒立摆保持不倒。

参照文献[13]，建立实验的模型。系统的状态空间是连续的，由倒立摆与竖直方向的夹角 χ 和当前的角速度 $\dot{\chi}$ 表示。动作为施加在小车上的力，共有 3 种不同的动作：向左的力($-50N$)、向右的力($50N$)、不施加力($0N$)。3 个动作都是有噪声的，总是有[-10,10]上的均匀分布的扰动叠加在所选择的动作上，记小车实际受到的力为 F，则系统的状态转移模型可由下式表示：

$$\ddot{\chi}=\frac{g\sin(\chi)-\eta al(\dot{\chi})^2/2-\eta\cos(\chi)F}{4l/3-\eta al\cos^2(\chi)}$$

其中，g 为重力加速度($g=9.8\mathrm{m/s^2}$)，a 为倒立摆的质量($a=2.0\mathrm{kg}$)，b 为小车的质量($b=8.0\mathrm{kg}$)，l 为倒立摆的长度($l=0.5\mathrm{m}$)，η 为常数($\eta=1/(a+b)$)。系统仿真时间步为 0.1s。这是一个情节式任务，只要倒立摆与竖直方向的夹角的绝对值不超过 $\pi/2$，奖赏就为 0；超过 $\pi/2$，意味着一个情节的结束并得到 -1 的奖赏。实验的折扣因子为 0.95。

4.2 实验设置

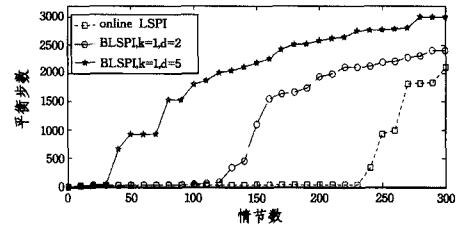
在本实验中，对每个动作采用一组 10 维(3 个动作共 30 维)的基函数去逼近值函数。这 10 个基函数包括 1 个常数和 9 个在 2 维状态空间上的高斯径向基函数。对状态 $x(\chi,\dot{\chi})$ 和动作 u 来说，对应于其余动作的基函数全为 0，对应于动作 u 的基函数如下：

$$(1,\mathrm{e}^{-\frac{\|x-\mu_1\|^2}{2\sigma^2}},\mathrm{e}^{-\frac{\|x-\mu_2\|^2}{2\sigma^2}},\mathrm{e}^{-\frac{\|x-\mu_3\|^2}{2\sigma^2}},\cdots,\mathrm{e}^{-\frac{\|x-\mu_9\|^2}{2\sigma^2}})^{\mathrm{T}}$$

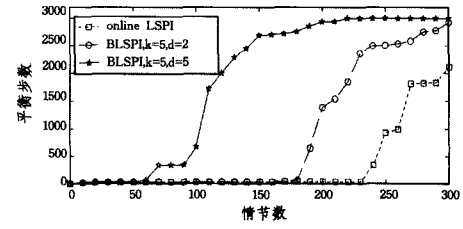
其中，μ_i ($i=1,2,3,\cdots,9$)为二维状态空间上的网格点$\{-\pi/4,0,+\pi/4\}\times\{-1,0,+1\}$，$\sigma$ 为方差。最小平衡步数设置为 3000，即仿真运行 3000 个时间步后倒立摆仍保持不倒就认为本次仿真成功，并开始下一个情节。实验的最大情节数设置为 300，通过观察 300 个情节内平衡步数的变化来分析算法的性能。实验中行为策略为 ε-greedy，ε 取 0.005。

4.3 实验结果及分析

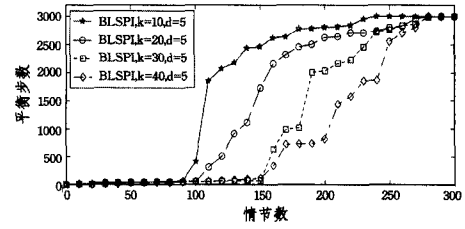
使用 LSTD-Q 评估策略的不同参数下的 BLSPI 方法与 online LSPI 方法的实验效果如图 1 所示。图中横轴表示实验的情节数，纵轴表示每个情节倒立摆的平衡次数。d 表示经验重复利用的次数，k 表示经过多少个情节后进行一次更新，如 BLSPI，$k=5$，$d=2$，表示每 5 个情节进行一次更新，每次更新经验重复利用 2 次。图 1(a)比较了在 $k=1$ 的情况下，BLSPI($d=2$)、BLSPI($d=5$)和 online LSPI 的实验效果，实验数据为运行 10 次后的平均结果。从图中可以看出，online LSPI 方法在前 200 多个情节内的平衡步数一直处于较低的水平，直到大约 230 个情节后才有所改善。而 BLSPI 方法的平衡步数明显高于 online LSPI 方法，并且 d 越大，效果越明显。这是经验被多次重复利用的结果，d 越大，经验重复利用次数越多，越能更快学习到最优策略，但计算量也越大。适量增大 k 的值可以减少策略更新的次数，从而在不减少样本数据的情况下减小计算量。图 1(b)给出了 $k=5$ 的情况下的实验效果图，可以看出 BLSPI 方法依然有很好的效果，只是初始性能略低于 $k=1$ 的情况，但增长速度更快，并略早于 $k=1$ 的情况先到达最优。然而，继续增大 k，实验效果会慢慢变差，如图 1(c)所示。可见，应当选择合适的参数，以确保在合适的计算量下得到较好的实验效果。



(a) $k=1$



(b) $k=5$



(c) $k>5$

图 1 使用 LSTD-Q 评估策略的不同参数下的 BLSPI 方法与 online LSPI 方法的性能比较

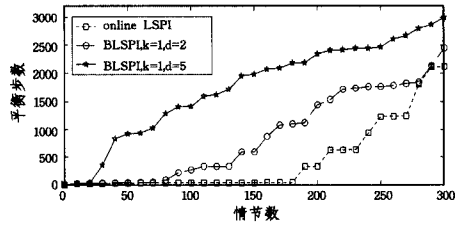
表 2 给出了使用 LSTD-Q 评估策略的不同参数下的 BLSPI 方法与 online LSPI 方法在前 p($p=50,100,150,200,250,300$)个情节内的平衡步数。表中数据表明，online LSPI 方法在前 200 个情节内平衡步数一直低于 50 步，在 300 个情节内最终达到 2112 个时间步，而在 $d=5$ 的 BLSPI 方法中平衡步数在几十个情节内就上升到数百步，并最终在 300 个情节内达到了 3000 步平衡。可见，重复利用以前的经验可以明显改善算法的初始性能，改进算法的收敛速度。当 $k>5$ 时，随着 k 的增大，计算量越来越少，平衡步数也越来越少，但仍能在 300 个情节内达到 3000 步平衡。

表 2 使用 LSTD-Q 评估策略的不同参数下的 BLSPI 方法与 online LSPI 方法的平均平衡步数

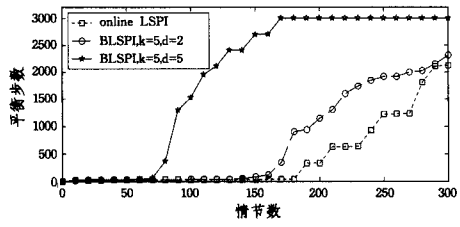
情节数	50	100	150	200	250	300
online LSPI	27	35	36	37	928	2112
BLSPI,k=1,d=2	30	45	1096	1934	2199	2407
BLSPI,k=1,d=5	921	1800	2187	2576	2771	3000
BLSPI,k=5,d=2	30	34	40	1386	2501	2930
BLSPI,k=5,d=5	39	671	2688	2932	3000	3000
BLSPI,k=10,d=5	37	401	2463	2803	3000	3000
BLSPI,k=20,d=5	37	56	1726	2632	2801	3000
BLSPI,k=30,d=5	36	49	75	2032	2761	3000
BLSPI,k=40,d=5	34	51	114	813	2555	3000

使用 LSPE-Q 评估策略的不同参数下的 BLSPI 方法与

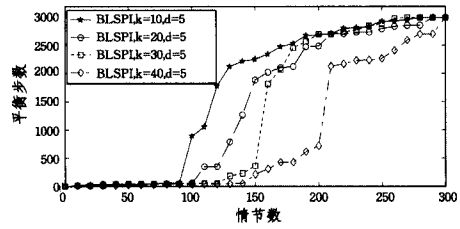
online LSPI 方法的实验效果如图 2 所示。取步长参数 $\alpha=0.005$。图 2(a)比较了在 $k=1$ 的情况下，BLSPI($d=2$)、BLSPI($d=5$)和 LSPI 的实验效果，实验数据为运行 10 次后的平均结果。从图中可以看出，与使用 LSTD-Q 评估策略的 BLSPI 方法类似，使用 LSPE-Q 评估策略的 online LSPI 方法在前近 200 个情节内的平衡步数一直处于较低的水平，直到大约 180 个情节后才有所改善。BLSPI 方法的平衡步数明显高于 online LSPI 方法，并且 d 越大，效果越明显。这也是经验被多次重复利用的结果，d 越大，经验重复利用次数越多，越能更快学习到最优策略，但计算量也越大。适量增大 k 的值可以减少策略更新的次数，从而在不减少样本数据的情况下减小计算量。图 2(b)给出了 $k=5$ 的情况下的实验效果图，可以看出 BLSPI 方法依然有很好的效果，只是初始性能略低于 $k=1$ 的情况，但增长速度更快，并早于 $k=1$ 的情况先到达最优。然而，继续增大 k，实验效果会慢慢变差，如图 2(c)所示。可见，应当选择合适的参数，以确保在合适的计算量下得到较好的实验效果。



(a) $k=1$



(b) $k=5$



(c) $k>5$

图 2　使用 LSPE-Q 评估策略的不同参数下的 BLSPI 方法与 online LSPI 方法的性能比较

表 3 给出了使用 LSPE-Q 评估策略的不同参数下的 BLSPI 方法与 online LSPI 方法在前 p($p=50,100,150,200,250,300$)个情节内的平衡步数。表中数据表明，online LSPI 方法在前 150 个情节内平衡步数一直低于 50 步，在 300 个情节内最终达到 2112 个时间步，而在 $d=5$ 的 BLSPI 方法中平衡步数在几十个情节内就上升到数百步，并最终在 300 个情节内达到了 3000 步平衡。可见，重复利用以前的经验可以明显改善算法的初始性能，提高算法的收敛速度。当 $k>5$ 时，随着 k 的增大，计算量越来越少，平衡步数也越来越少，但仍能在 300 个情节内达到 3000 步平衡。

表 3　不同参数下的 BLSPI 方法与 online LSPI 方法的平均平衡步数

情节数	50	100	150	200	250	300
online LSPI	28	33	35	332	1223	2124
BLSPI,k=1,d=2	35	265	593	1442	1773	2451
BLSPI,k=1,d=5	922	1413	1986	2342	2467	3000
BLSPI,k=5,d=2	30	36	82	1148	1922	2317
BLSPI,k=5,d=5	34	1532	2701	3000	3000	3000
BLSPI,k=10,d=5	33	897	2248	2690	2936	3000
BLSPI,k=20,d=5	36	61	1887	2486	2801	3000
BLSPI,k=30,d=5	42	47	364	2701	2887	3000
BLSPI,k=40,d=5	25	31	214	718	2266	3000

BLSPI 算法与 online LSPI、offline LSPI、Batch TD 算法的性能比较如图 3 所示，所有批量算法每 5 个情节更新一次，样本重复利用 5 次，步长参数 $\alpha=0.005$。Batch TD 算法在 300 个情节内的平衡步数一直维持在几十步左右，远低于其他最小二乘方法，这是由于最小二乘方法将样本数据累加到矩阵中，对样本数据的利用更加充分。online LSPI 算法对样本只使用一次，初始性能较差，但在大约 230 个情节后平衡步数开始较快增长，而 offline LSPI 算法初始性能较好，因为其重复利用当前所有样本直至策略稳定，样本利用次数最多，然而离线算法样本需要事先采集，并且无法在线补充新的更好的样本，后期增长速度缓慢。BLSPI 算法在线采集样本，并多次重复利用样本数据，在保证了一定的初始性能的条件下，能更快地收敛到最优策略，达到 3000 步平衡。

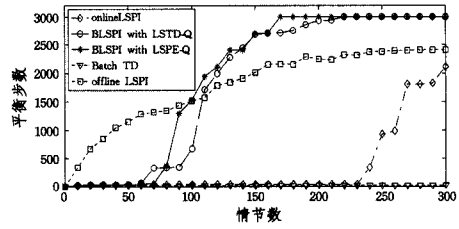


图 3　不同算法的平衡步数比较

定义策略参数 θ 的均方误差(Mean Squared Error，MSE)为：

$$MSE(\theta)=\sum_{i=1}^{N}(\theta(i)-\theta^{*}(i))^{2}$$

其中，θ^{*} 为最优策略参数，N 为 θ 维数，则 MSE 表示当前策略与最优策略的距离，MSE 越小，距离越近，策略的精确度越高。BLSPI 算法与 online LSPI、offline LSPI、Batch TD 算法的 MSE 如图 4 所示。Batch TD 算法在 300 个情节内无法找到较好的策略，MSE 较大，策略的精确度较低。online LSPI 算法能较明显地改进策略精度，但改进速度缓慢，offline LSPI 算法开始改进速度很快，但大约 120 个情节后趋于平缓，未能在 300 个情节内找到最优策略。BLSPI 算法在 200 个情节后基本找到了最优策略，并维持较高的策略精度，可见，BLSPI 算法能在相同的情节数下提高策略精确度，尽早找到最优策略，提高经验利用率，加快收敛速度。

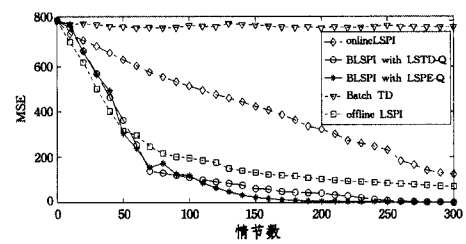


图 4　不同算法的 MSE 比较

结束语 传统的在线最小二乘策略迭代方法对在线生成的样本仅利用一次，用完后就丢弃，其经验利用率低、收敛速度慢。本文将批量更新方法与在线最小二乘策略迭代方法相结合，提出了一种批量最小二乘策略迭代算法。该算法以四元组(x,u,r,x')的形式保存在线生成的样本数据，并多次重复利用这批样本。重复利用的过程中，尽管用的都是同一批四元组形式的样本(x,u,r,x')，但更新策略参数用的是五元组$(x,u,r,x',h(x'))$，而每次更新用的策略h并不相同，因此同一个样本可以用来多次更新策略参数，提高了经验利用率。使用LSTD-Q评估策略的BLSPI算法在处理完一批样本后才更新策略参数，性能较稳定；而采用LSPE-Q评估策略的BLSPI算法每处理一个样本都更新策略参数，并且依赖于上一轮的评估结果，是一个增量式算法，通过给定较好的初始值并逐步减小步长参数也可以取得较好的性能。由于求解策略参数需要进行矩阵求逆运算，其计算复杂度较高，计算量较大，在算法中应用了Sherman-Morrison公式，消去了矩阵求逆运算，降低了计算复杂度，减少了计算量。倒立摆实验表明，重复利用以前的经验，可以提高经验利用率，加快收敛速度，并且重复次数越多，效果越明显，但计算量也越大，适量增大批量更新间隔可以在保证算法一定性能的基础上降低计算量，因此应根据具体情况选择合适的算法参数。

当状态划分比较细致、径向基函数维度比较大时，最小二乘方法将会带来巨大的计算量。如何减少计算量，提高计算效率及如何更加高效地筛选、利用经验知识是下一步研究的方向。另外，本文考虑的是离散动作空间的情况，对连续动作空间的研究也是下一步研究的课题。

参 考 文 献

[1] Sutton R S, Barto A G. Reinforcement learning: An introduction [M]. Cambridge: MIT Press, 1998

[2] 刘全，闫其粹，伏玉琛，等. 一种基于启发式奖赏函数的分层强化学习方法 [J]. 计算机研究与发展，2011，48(12)：2352-2358

[3] Kaelbing L P, Littman M L, Moore A W. Reinforcement learning: A survey [J]. Journal of Artificial Intelligence Research, 1996, 4(2): 237-285

[4] 刘全，傅启明，龚声蓉，等. 最小状态变元平均奖赏的强化学习方法 [J]. 通信学报，2011，32(1)：66-71

[5] Gao Yang, Chen Shi-fu, Lu Xin. Research on reinforcement learning technology: A review [J]. Journal of Acta Automatica Sinica, 2004, 30(1): 86-100

[6] Geist M, Pietquin O. Parametric value function approximation: A unified view [C]// Proc of the IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning. NJ: IEEE, 2011: 9-16

[7] Bradtke S J, Barto A G. Linear least-squares algorithms for temporal difference learning[J]. Journal of Machine Learning, 1996, 22: 33-57

[8] Boyan J. Technical update Least-squares temporal difference learning [J]. Journal of Machine Learning, 2002, 49: 233-246

[9] Maei H R, Szepesvari C, Bhatnagar S, et al. Toward off-policy learning control with function approximation [C]// Proc of the 27th International Conference on Machine Learning. Haifa: Omnipress, 2010: 719-726

[10] Sutton R S. Learning to predict by the method of temporal differences [J]. Journal of Machine Learning, 1988, 22: 33-57

[11] Sutton R S, Szepesvari Cs, Maei H R. A convergent O(n) algorithm for off-policy temporal-difference learning with Linear function approximation [C]// Proc of the 25th Annual Conference on Neural Information Processing Systems. Granada, 2008: 1609-1616

[12] Lagoudakis M, Parr R, Littman M. Least-squares methods in reinforcement learning for control [J]. Methods and Applications of Artificial Intelligence, 2002, 2308: 249-260

[13] Lagoudakis M, Parr R. Least squares policy iteration [J]. Journal of Machine Learning Research, 2003(4): 1107-1149

[14] Busoniu L, Babuska R, Schutter B D, et al. Reinforcement Learning and Dynamic Programming using Function Approximators [M]. New York: CRC Press, 2010

[15] Kalyanakrishnan S, Stone P. Batch reinforcement learning in a complex domain[C]// Proc of the 6th International Conference on Autonomous Agents and Multiagent Systems. New York, 2007: 650-657

[16] Jung T, Polani D. Kernelizing LSPE (λ) [C]// Proc of the IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning. NJ: IEEE, 2007

[17] Jung T, Polani D. Least squares SVM for least squares TD learning[C]// Proc of the 17th European Conference on Artificial Intelligence. Riva del Garda, 2006: 499-503

(上接第 214 页)

[7] Lin F, Reiter R. State constraints revisited[J]. Journal of Logic and Computation, 1994, 4(5): 655-677

[8] Lin F. Embracing causality in specifying the indirect effects of actions[C]// Proceedings of the Thirteenth National Conference on Artificial Intelligence. Oregon, Portland: AAAI Press, 1996 (1): 670-676

[9] Baader F, Lippmann M, Liu H. Using causal relationships to deal with the ramification problem in action formalisms based on description logics[C]// Proceedings of the 17th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning. Berlin, Heidelberg: Springer-Verlag, 2010: 82-96

[10] Henriksen J G, Thiagarajan P S. Dynamic linear time temporal logic[J]. Annals of Pure and Applied Logic, 1999, 96(1-3): 187- 207

[11] Giordano L, Martelli A, Schwind C. Reasoning about actions in dynamic linear time temporal logic[J]. Logic Journal of the IGPL, 2001, 9(2): 273-288

[12] Giordano L, Martelli A, Schwind C. Specifying and verifying interaction protocols in a temporal action logic[J]. Journal of Applied Logic, 2007, 5(2): 214-234

[13] Baader F, Liu H, Mehdi A. Verifying properties of infinite sequences of description logic actions [C]// ECAI 2010: 19th European Conference on Artificial Intelligence Proccedings. Lisbon, Portugal: IOS Press, 215, 2010: 53-58

[14] 常亮，史忠植，古天龙，等. 可判定的时序动态描述逻辑[J]. 软件学报，2011，22(7)：1524-1537

[15] Wang Xiao-feng, Chang Lang, Li Zhi-xin, el al. A dynamic description logic based system for video event detection[J]. Frontiers of Electrical and Electronic Engineering in China, 2010, 5 (2): 137-142