基于 Tile Coding 编码和模型学习的 Actor-Critic 算法

金玉净 朱文文 伏玉琛 刘 全
(苏州大学计算机科学与技术学院 苏州 215006)

摘 要 Actor-Critic 是一类具有较好性能及收敛保证的强化学习方法,然而,Agent 在学习和改进策略的过程中并没有对环境的动态性进行学习,导致 Actor-Critic 方法的性能受到一定限制。此外,Actor-Critic 方法中需要近似地表示策略以及值函数,其中状态和动作的编码方法以及参数对 Actor-Critic 方法有重要的影响。Tile Coding 编码具有简单易用、计算时间复杂度较低等优点,因此,将 Tile Coding 编码与基于模型的 Actor-Critic 方法结合,并将所得算法应用于强化学习仿真实验。实验结果表明,所得算法具有较好的性能。

关键词 强化学习,Tile Coding,Actor-Critic,模型学习,函数逼近
中图法分类号 TP181 **文献标识码** A

Actor-Critic Algorithm Based on Tile Coding and Model Learning

JIN Yu-jing ZHU Wen-wen FU Yu-chen LIU Quan
(School of Computer Science and Technology, Soochow University, Suzhou 215006, China)

Abstract The Actor-Critic(AC) approach is a class of reinforcement learning method which has good performance and ensures convergence, but the Agent does not study the dynamic of environment in the process of learning and improving policy, which causes the performance of the AC method to be restricted to a certain extent. In addition, the AC method needs to represent the policy and value function approximately, and the encoding methods of state and action and parameters have important influence on AC method. Tile Coding has advantages of simple and low computing time complexity, so we combined the Tile Coding with Actor-Critic method based on model and applied the algorithm to the simulation experiment on reinforcement learning, and the results show that the algorithm has good performance.

Keywords Reinforcement learning, Tile Coding, Actor-Critic, Model learning, Function approximation

1 引言

很多工业过程通过控制算法进行优化,这些控制算法学习一个使得相关耗费最小或者回报最大的近似函数,然后通过调整该函数的相关参数来控制工业过程。强化学习(RL)就是一种基于经验、模拟或者搜索来学习和估计值函数的最优控制方法,在缺乏模型信息的情况下通过采样来解决马尔可夫决策过程,在优化和控制中具有广泛的应用[1,2]。

强化学习的缺点是收敛速度慢,因而学习时间长。近年来对强化学习方法的研究总体可以分为 3 类:Actor-only 算法,Critic-only 算法,Actor-Critic 算法。Actor-only 算法用一系列参数化的策略使得过程最优,将策略参数化的优势在于可以产生连续的动作范围,然后使用策略梯度估计,这种方法具有较好的收敛性,但估计方差过大且收敛速度慢,从而减慢学习速率。Critic-only 算法(例如 Q-学习)是基于值函数的学习算法,使用时间差分(TD)学习减小了期望回报估计的方差,在离散查找表参数化逼近方面有了很多成功的例子。然而在求解具有连续状态空间的马尔可夫策略过程(MDP)问题时,值函数方法需要借助函数逼近器来解决"维数灾"问题,从而不能保证收敛,而且算法根据 Bellman 方程寻找最优值函数,每次都选择使得值函数最大的那个动作。当动作空间为连续时计算量巨大,因此 Critic-only 算法通常将连续的动作空间离散化后枚举得到最优策略[3]。

最早的 Actor-Critic(AC)架构思想由 Barto[6] 提出,为了加快学习速度,Konda[4-9] 等人提出了具体的 AC 算法及其改进算法。AC 架构结合了 Actor-only 算法和 Critic-only 算法的优点,在连续的动作空间上应用参数化策略产生连续的动作范围而不需要寻找最优值函数对应的动作。在函数逼近的基础上 Critic 和 Actor 分别学习值函数和策略,Critic 可以提供小方差的值函数给 Actor,Actor 根据 Critic 对期望回报的估计向小方差方向梯度更新,加速学习过程。Actor-Critic 方法相对 Q-学习算法通常有更好的收敛性,并减小了期望回报的估计方差[7]。但 Agent 并没有对环境的动态性进行学习从而改进策略,这使得 AC 算法受到了一定的限制。对此,Grondman[10,11] 提出了基于模型学习的 AC 算法,通过学习环境模型来进一步提高算法学习速度和数据的利用率。

到稿日期:2013-08-16 返修日期:2013-11-13 本文受国家自然科学基金(61070122,61373094,61070223,61103045),江苏省自然科学基金(BK2009116),江苏省高校自然科学研究项目(09KJA520002)资助。

金玉净(1987—),女,硕士生,主要研究方向为强化学习与机器学习,E-mail:hyjttkl@126.com;朱文文(1988—),女,硕士生,主要研究方向为强化学习与机器学习;伏玉琛(1968—),男,博士,硕士生导师,主要研究方向为强化学习、机器学习及数据挖掘;刘 全(1969—),男,教授,博士生导师,主要研究方向为强化学习与人工智能。

另外，AC方法需要近似表示值函数和策略函数，其中状态和动作的编码方法以及参数的设置都对算法有重要的影响。传统的近似方法包括简单离散查找表、径向基函数和神经网络等，Sutton[12]在表现力、计算代价和易用性等方面对这些方法进行权衡，指出 Tile Coding 具有简单易用、计算时间复杂度较低等优点。

本文介绍的模型学习 AC 算法是基于样本和折扣回报模型[12,13]并运用策略更新的方法，在利用先验知识的基础上学习一个近似的过程模型，该模型以离散查找表的形式存储有效样本数据，从而提高了有效样本数据的利用率和学习速度。结合 Tile coding 线性函数逼近方法将问题空间映射到特征空间以便更好地适应逼近器，从而有效地解决连续状态动作空间问题。

2 强化学习及 Tile Coding 编码

2.1 强化学习与马尔可夫决策过程

强化学习(RL)是无模型的学习方法，通过试错方式学习一个有效策略，但是在缺乏信息数据的情况下，RL 学习速度很慢。模型学习是在 Agent 与环境的交互过程中学习一个过程模型，提高有效数据的利用率，从而提高学习速度。通常模型学习可以分为两大类，解析模型和模拟模型。模拟模型可以用离散查找表的方式存储有效样本数据，而解析模型则需要用函数来表示数据之间的映射，但在实际问题中，通常单个函数不足以表达数据之间的关系，在离散状态下，通常使用模拟模型来存储有限的有效数据，以提高运算速率。

RL 问题可以用一个四元组 $M(X,U,f,\rho)$ 的马尔可夫决策过程(MDP)来描述，X 和 U 分别表示状态和动作空间，整个控制过程用状态转移函数 $f:X\times U\mapsto X$ 描述，当环境处在状态 x_t 时，Agent 根据策略 $h:X\mapsto U$ 选择动作 u_t 并返回状态 x_{t+1}，每次转移后控制者依据奖赏函数 $\rho:X\times U\mapsto \mathbb{R}$ 得到一个立即奖赏 $r_{t+1}=\rho(x_t,u_t)$ 且 $r_{t+1}\in\mathbb{R}$。在 RL 中，Agent 为了完成任务，必须知道采取某个策略而导致出现的某个状态对该 Agent 所产生的长期回报，而不是立即奖赏，则目标是寻找一个策略使得长期获得的奖赏和最大，该和可以用值函数 $V^h:X\mapsto\mathbb{R}$ 来表示。假设情节能在有限时间步终止，由于未来奖赏对当前状态的影响逐渐减弱，则定义策略 h 下状态 x 的值函数 V^h 的折扣形式为：

$$V^h(x)=[r_{t+1}+\gamma r_{t+2}+\cdots+\gamma^{T-1}r_{t+T}\mid x_t=x]=\sum_{i=0}^{T}\gamma^i r_{t+i+1} \tag{1}$$

其中，x_t 为折扣因子，$V^h(x)$ 为折扣回报。Agent 的目标是找到一个策略 h^*，使得 $V^{h^*}(x)\geqslant V^h(x)$，$\forall h;\forall x\in X$，满足此条件的策略 h^* 称为最优策略。

式(1)满足 Bellman 方程：

$$V^h(x)=\rho(x,h(x))+\gamma V^h(x') \tag{2}$$

其中，x' 在策略 h 下由状态转移函数给出，即 $x'=f(x,h(x))$。但由于转移函数和奖赏函数并没有事先给出，因此本文的模型学习就是学习该转移函数，用 $\hat{f}(x,u)$ 表示。

2.2 Tile Coding 编码基本理论

Tile Coding 编码是一种简单、计算量较小的线性函数逼近器，通过设置给定的近似参数解决连续状态空间问题[2]。同时 Tile Coding 方法还可以进行状态抽取，计算速度快，在近来的 RL 复杂问题中应用很成功[1,14,16]。Tile Coding 方法在实际应用中的成功很大程度上取决于参数的选择和设置，在不同参数和不同 tiling 数目下性能也不同[15]。

Tile Coding 使用相同的划分方式将空间划分为许多重叠的 tiling，不同的 tiling 使用随机数偏移使得空间不同，每个 tiling 又被划分为多个 tile，tile 的形状一般以规则的矩形出现。如果使用像网格一样的 tiling，给定状态空间中一个点的坐标(x,y)，很容易计算出 tile 所在的索引。对于状态(x,y)，通过计算它在每个 tiling 中的坐标(m,n)就可以获得一组对应的特征向量，每个特征都有一个对应的参数 θ，特征向量对应的参数值的线性累加就是估计的函数值。一个状态对应的 T 个 tile 的参数的平均值可以被用来计算预测值 $y=\frac{1}{T}\sum_{i=1}^{T}\theta_i$。tiling 越密集，期望函数就可以逼近得越好、越精确，但是计算量开销也就越大。Tile Coding 从参数的初始值开始增量调整，从长远看，是一种比较好的函数逼近方法。

3 基于模型学习的 Actor-Critic 算法

3.1 模型学习 AC 算法

传统的 Actor-Critic 算法是基于策略梯度的在线策略迭代方法，通过与环境交互，在没有环境模型的情况下在线更新。在连续状态动作空间中，Actor 和 Critic 都用参数化函数表示，结合函数逼近器使用 TD 学习算法来估计值函数参数，根据随机梯度下降或者上升方向来更新策略参数，使得函数在连续空间中更容易操作[9]。同时 AC 学习还是在策的，即无论 Actor 当前选择了什么动作，Critic 都必须知道并对其进行评判。Critic 采取 TD 误差的形式，可以减小梯度估计的方差。

模型学习 AC 算法使用基于折扣回报模型且带有资格迹的常规梯度，Critic 根据 $TD(\lambda)$ 学习算法来更新值函数参数，Actor 在学习策略函数的同时还学习一个逼近的过程模型 $x'=\hat{f}(x,u)$，根据在当前状态 x 下输入的动作 u 预测下一个状态 x'，该过程模型用 $\zeta\in\mathbb{R}^{r\times n}$ 参数化表示，其中 n 为状态维，使用学习到的过程模型来简化 Actor 的更新，这意味着我们可以选择动作 u 使得 $V(x')$ 最优。下面分别介绍 Critic 和 Actor 如何学习近似值函数和策略。

3.1.1 利用 Tile Coding 线性逼近 Critic 值函数

Critic 利用 Tile Coding 学习近似值函数 $V^h(x)$，根据 $V^h(x)$ 利用 TD 误差更新 Actor 以进行策略改进。

确定地表示每一个状态 x 的值函数是不切实际的，用 Tile Coding 近似表示的值函数由一组权值决定，tile 编码使用独占的二值(0 或 1)特征，则每个状态 x 近似值函数为：$V^h(x)=\Phi(x)^{\mathrm{T}}\cdot\theta$，其中 $\Phi(x)=(\varphi_1(x),\varphi_2(x),\cdots,\varphi_n(x))^{\mathrm{T}}$ 为基函数，$\theta=(\theta_1,\theta_2,\cdots,\theta_n)^{\mathrm{T}}$ 是依赖于策略的参数。每个状态 x 在不同的 tiling 中所在 tile 的索引也不同，$\varphi_i(x)$ 表示与状态 x 相关的 tile 的二值特征，若状态 x 在第 i 个 tile 中，则 $\varphi_i(x)=1$，否则 $\varphi_i(x)=0$，因此有 $V(x)=\Phi(x)^{\mathrm{T}}\cdot\theta=\sum_{i=1}^{n}\varphi_i(x)\cdot\theta_i$，其中 n 为 tiling 的个数。在此用向量 θ 来参数化逼近值函数 $V(x,\theta)$，在每次选择动作之后，Critic 评估新状态的值函数以判断所选动作得到的奖赏与期望相比是好还是坏，则式(2)中 Bellman 方程的参数化 TD 误差 δ_t 为：$\delta_t=r_t+\gamma V(x_t,\theta_{t-1})-V(x_{t-1},\theta_{t-1})$。用 δ_t 来评估状态 x_t 下选择的动作 u_t，若 δ_t 为正，表明在未来选择 u_t 的趋势应该被加强；否则被减弱。目标是使得 Critic 的近似函数满足 Bellman 方程，

使得 δ_t 足够小。

Critic 参数向量 θ 的梯度下降更新公式为：

$$\theta_t=\theta_{t-1}+\alpha_c[r_t+\gamma V(x_t,\theta_{t-1})-V(x_{t-1},\theta_{t-1})]\nabla_{\theta_{t-1}}V(x_t) \tag{3}$$

其中，$\alpha_c\in[0,1]$，是 Critic 的学习速率。$\nabla_{\theta_{t-1}}$ 表示偏微分，则 $\nabla_{\theta_{t-1}}V(x_t)=\frac{\partial V(x_t,\theta_t)}{\partial\theta}$ 为 $\sum_{i=1}^{n}\varphi_i(x)$。

参数利用式(3)更新并求得近似值函数，从而使得在某一状态下估计值和真实值之间的差值最小。但利用式(3)只会导致一步更新，而奖赏通常是在经历很多步后才获得的，因此用资格迹的方式对先前已访问过的状态进行效用分配。特定状态 x 在时间 t 的资格迹用 $e_t(x)$ 表示：

$$e_t(x)=\begin{cases}1, & \text{if } x=x_t\\ \lambda\gamma e_{t-1}(x), & \text{else}\end{cases} \tag{4}$$

资格迹通常会随着时间的变化以延迟因子 $\gamma\lambda$ 衰减，其中 $\lambda\in[0,1)$ 为资格迹延迟参数，这样使得访问的状态越近则得到的效用越多。沿着资格迹，所有状态都能影响 θ 的更新，此时的更新公式为：

$$\theta_t=\theta_{t-1}+\alpha_c\delta_t\sum_{x\in\chi_v}\sum_{i=1}^{n}\varphi_i(x)e_t(x) \tag{5}$$

χ_v 表示在当前尝试的过程中已访问过的状态集，资格迹的使用加速了学习的速率。

3.1.2 利用 Tile Coding 逼近的 Actor 策略

Actor 是利用常规策略梯度更新算法来进行策略评估的。

与近似值函数类似，近似策略用 $h(x,\vartheta)$ 的 ϑ 参数化表示：$h(x_t,\vartheta_t)=\varphi(u)^{\mathrm{T}}\cdot\vartheta=\sum_{i=1}^{n}\varphi_i(u)\cdot\vartheta_i$，其中 m 为 tiling 的个数。RL 方法在面临某一个状态时可能需要探索新的更好的动作，则控制动作 u_t 可能与策略 $h(x_t,\vartheta_{t-1})$ 提供的动作不同。由 $x'=\hat{f}(x,u)$ 可知过程模型根据当前状态 x 下输入的动作 u 预测下一个状态 x'，然而我们假设动作空间是连续的，枚举所有可能的输入动作是不可能的，所以要用模型策略梯度。因为 Critic 的参数已知，则 Actor 参数的策略梯度可由下式估计：

$$\vartheta_t=\vartheta_{t-1}+\alpha_a\nabla_x V(x_t,\theta_{t-1})^{\mathrm{T}}\cdot\nabla_u\hat{f}_{\zeta_t}(x_t,u_t)\cdot\nabla_\vartheta h(x_t,\vartheta_{t-1}) \tag{6}$$

过程模型参数采用系统真实输出值和模型输出值之间的误差的梯度下降更新：

$$\zeta_{t+1}=\zeta_t+\alpha_P(x_{t+1}-\hat{f}_{\zeta_t}(x_t,u_t))\cdot\nabla_\zeta\hat{f}_{\zeta_t}(x_t,u_t) \tag{7}$$

其中，α_P 为学习速率。由式(6)可以看出模型学习 AC 不需要探索就可估计策略梯度，并沿梯度方向更新 Actor 参数从而得到更高的状态值，但当策略只能访问状态空间的一部分时，还是需要探索来完成对整个状态空间的值函数估计，而且探索会改进模型的过程动态性。比较传统的 AC 算法[4]，模型 AC 的过程图如图 1 所示。

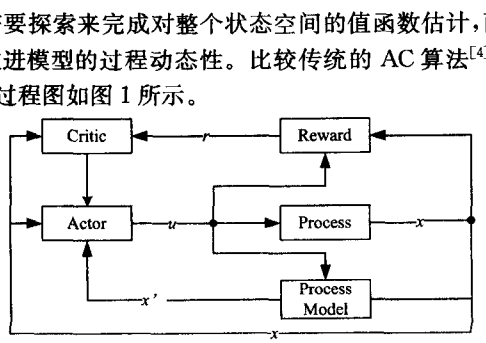


图 1 模型学习 AC 的过程图

则基于模型学习的 TCAC 算法如表 1 所列。

表 1 TCAC 算法

输入：$\gamma,\lambda,\alpha_a,\alpha_c,\alpha_P$

1. $e_0(x)=0,\forall x$
2. 初始化 $x_0,\theta_0,\vartheta_0,\zeta_0$，函数逼近器
3. Repeat(对每个情节)
4. $k\leftarrow1$
   随机输入 u_0，并采用 u_0
5. Repeat(对每一步)
6. 衡量 x_k,r_k//(x_1 是根据状态 x_0 采用 u_0 得到的)
7. $\delta_k\leftarrow r_k+\gamma V(x_k,\theta_{k-1})-V(x_{k-1},\theta_{k-1})$
8. $u_k\leftarrow h(x_k,\vartheta_{k-1})$
9. $\vartheta_k=\vartheta_{k-1}+\alpha_a\nabla_x V(x_k,\theta_{k-1})^{\mathrm{T}}\nabla_u\hat{f}_{\zeta_k}(x_k,u_k)\nabla_\vartheta h(x_k,\vartheta_{k-1})$
10. $\zeta_{k+1}=\zeta_k+\alpha_P\cdot(x_{k+1}-\hat{f}_{\zeta_k}(x_k,u_k))\cdot\nabla_\zeta\hat{f}_{\zeta_k}(x_k,u_k)$
11. $e_k(x)=\begin{cases}1, & \text{if } x=x_k\\ \lambda\gamma e_{k-1}(x), & \text{else}\end{cases}$
12. $\theta_k=\theta_{k-1}+\alpha_c\delta_k\sum_{x\in\chi_v}\sum_{i=1}^{n}\varphi_i(x)e_t(x)$
13. 选择探索因子 $\Delta u_k\sim N(0,\sigma^2)$
14. 采用 $u_k+\Delta u_k$
15. $k\leftarrow k+1$
16. until $|u_k-h(x_k,\vartheta_k)|\leqslant\delta$//$\delta$ 是一个很小的值
17. until x_k 是目标状态

3.2 收敛性分析

在连续状态动作空间中，我们无法得到精确的状态值函数 V 以及策略 h，所以用函数逼近的方法结合 AC 算法来实现近似的值函数和策略。AC 是基于梯度的，所以要证明全局最优策略的收敛性是很困难的，但是在实际中，只要证明 $\nabla\theta\to0$，局部 θ 的最小值收敛即可。Critic 将收敛到近似的值函数，相应的收敛结果比较接近真实值，只要保证 $\nabla\theta$ 足够小即可。梯度方法的收敛结果假设步长参数是不断减少的，引出一些相应的假设：

(1)对每个 $\theta\in X$，定义一个 $m\times m$ 的矩阵 $G(\theta)$，假设 $G(\theta)$ 是均匀的正定矩阵，$\exists\varepsilon_1>0$，对 $r\in\mathbb{R}^m$ 和 $\theta\in X$，$r^{\mathrm{T}}\cdot G(\theta)\cdot\gamma\geqslant\varepsilon_1\cdot\|\gamma\|_2$。

(2)假设 Critic 和 Actor 更新中的步长参数序列 $\{\alpha_c\}$ 和 $\{\alpha_a\}$ 是正的非递增的，$\exists d>0$，$\{\alpha_c\}$ 和 $\{\alpha_a\}$ 必须满足以下关系：

$$\delta_t>0,\forall t;\sum\delta_t=\infty;\sum\delta_t^2<\infty;\sum(\frac{\alpha_a}{\alpha_c})^d<\infty$$

δ_t 表示 $\{\alpha_c\}$ 和 $\{\alpha_a\}$，同时存在正的常量 C_1 和 C_2，假设函数 $\Gamma(\cdot)$ 是一个根据 α_c 控制步长 α_a 的标量，满足：$\frac{C_1}{1+|\alpha|}\leqslant\Gamma(\alpha_t)\leqslant\frac{C_2}{1+|\alpha|}$。假设 $\frac{\alpha_a}{\alpha_c}\to0$，表示某时间段内 Actor 的参数更新小于 Critic 的。如果它满足上式的标准随机逼近不断减少的条件，那么梯度下降方法能保证收敛到一个局部最优。

4 实验结果及分析

Tile Coding 是强大的逼近器，但是它在产生局部逼近上是否仍然很方便？并且将其应用在模型学习 AC 算法中是否能够达到快速的学习性能？对连续的状态和动作空间都使用 Tile Coding 线性函数逼近方法，将该模型 AC 算法称作 TCAC，使用著名的 Mountain Car[1] 实例来验证 TCAC 的可行性并与 $Q(\lambda)$ 作相应对比。

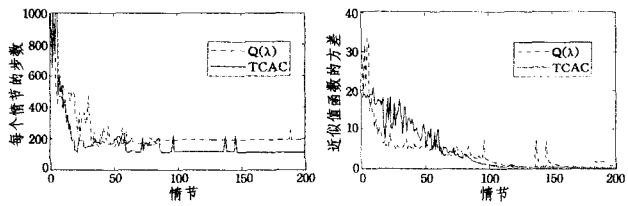
4.1 Mountain Car 实验描述

假设驾驶一辆动力不足的小车，想让它到达陡峭的山顶，

而由于重力大于引擎提供的动力，即使使用最大的油门也难以到达山顶，唯一的解决办法就是先向相反方向的斜坡后退，然后利用惯性和最大的油门向山顶冲。到达山顶之前的所有的时间步奖赏值均为-1，到达山顶时情节结束。状态由位置x_t和速度v_t两个变量构成，更新如下：$x_{t+1}=bound\{x_t+v_{t+1}\}$，$v_{t+1}=bound\{v_t+0.001a_t-0.0025\cos(3x_t)\}$，其中$a_t$为动作值且$a_t\in[-1,1]$。*bound*函数用来修正两个变量位置和速度的值，使其分别保持在范围$[-1.2,0.5]$和$[-0.07,0.07]$内。当x_{t+1}到达左边界时v_{t+1}被置为0；当到达右边界时，即到达山顶，情节结束[2]。

4.2 Mountain Car 实验结果及分析

为了将两个连续变量转换成二值特征，我们使用16个$7*7$tilings划分状态空间，每个使用tile宽度的随机数来偏移，而用10个tiling划分动作空间，每个tiling被分成3段以离散化动作。实验数据取前1000个情节，且每个情节限制在1000步数，如果1000步还没有到达目标，则情节结束；如果小于1000步，则到达目标时情节结束。我们通过在抽取的状态空间中的状态的数量来衡量该方法的有效性，通过RL算法需要的学习更新的数量来学习一个最优的策略。参数设置如下：$\lambda=0.9$，$\gamma=0.95$，$\alpha_c=0.1$，$\alpha_a=0.5$，$\alpha_P=0.5$，模型存储大小$M=3000$。图2(a)中横坐标表示情节数，纵坐标表示每个情节的步数，由图可以看出在相同参数的设置下TCAC学习速度更快，在第57个情节时收敛，且收敛时的步数小于$Q(\lambda)$。TCAC实验最终收敛步数为106步，前2000个情节平均值为115步，因为200个情节以后基本上使用的是模型里的数据，步数始终维持在106步，而$Q(\lambda)$有探索的存在，所以偶尔还有波动，但改变不大，则图中后800个情节的情况省略。从图2(b)中可以看到$Q(\lambda)$估计值函数的方差减小较快，但不稳定，在学习初期，模型学习TCAC表现没有$Q(\lambda)$好，但在长期运行下，方差逐渐减小并稳定在一个很小的范围中，且最终方差比$Q(\lambda)$好，则学习曲线也表现得更好。

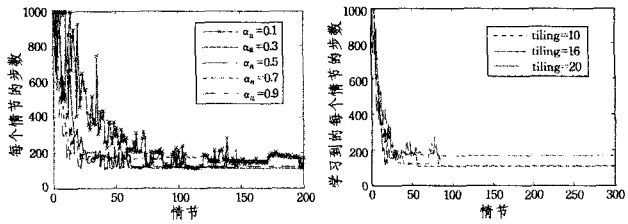


(a) 相同参数设置下TCAC与$Q(\lambda)$每个情节步数的比较　(b) 近似值函数方差的大小比较

图2　模型学习TCAC算法与$Q(\lambda)$学习的比较

4.3 参数的选择对AC算法的影响

模型学习AC算法和其他算法一样都对参数的选择十分敏感，尤其是步长参数。本文中的TCAC有4个参数：λ和3个步长参数α_a，α_P，α_c，λ的值取决于实际问题，在初始时用一个小值(通常不大于0.4)效果会更好。α_c通常设置成0.1是由特征向量$\Phi(s)$分离来的，一旦TCAC能够以$\alpha_c=0$稳定地学习值函数，则可以保持α_a的值较小。由图3(a)可以看出，α_a的增加可使Actor的策略改进，并且随着α_a的增加学习收敛速度也更快，当$\alpha_a=0.9$时，学习速度很快且在第47个情节就收敛了，并保持稳定。



(a) $\alpha_c=0.1$情况下，不同α_a对学习速度与收敛结果的影响　(b) TCAC状态空间tiling个数不同的曲线图

图3　不同参数对模型学习TCAC算法的影响

由图3(b)可以看出，模型学习AC方法在不同tiling个数状态下学习速度不同，在tiling=10和tiling=16时最终收敛结果比tiling=20好，而tiling=16学习速度比tiling=10快，说明随着tiling的个数增多，学习速度和收敛效果都有所提高，而当进一步增多时算法性能反而变差。这是因为Tile Coding只是简单地将连续状态空间离散化的线性方法，不能在多维的情况下实现，而且即使目标函数的复杂度不增加，Tile Coding自身的复杂度也会随着维数的增加而增加，tiling个数越多，需要计算的θ也就越多，计算量变大，而且状态空间更细化了，泛化能力变差，而模型在更新的时候存储空间是有限的，在此模型的存储大小设置为3000，一旦收敛便不再改变了，所以效果反而变差。

结束语　Tile Coding是一个强大的线性函数逼近器，给定适当的参数设置就可以有效地解决大空间问题。将Tile Coding应用在AC算法中可以学习到更精确的逼近函数以达到最优性能，在有Critic的基础上，算法最终会随着当前最优策略的改变而改变局部逼近，从而在一个给定的问题中最终适应Tile Coding的参数。Actor-Critic架构是无模型的强化学习方法，本文提出的方法在更新策略和值函数的同时还学习一个模型，以加快学习速度。鉴于Tile Coding的泛化能力强，使用Tile Coding进行特征提取，对特征参数进行线性累加，并根据梯度下降的方式调整参数，在维数少时用于连续状态空间上有很大的优势。通过实验证明了TCAC可以收敛到一个稳定的策略，并且最终的性能是比较好的，与$Q(\lambda)$相比它拥有更小的标准误差。

参 考 文 献

[1] Sutton R S, Barto A G. Reinforcement Learning: An Introduction[M]. MIT Press, 1998

[2] Busoniu L, Babuska R, DeSchutter B, et al. Reimforcement Learning and Dynamic Programming Using Function Approximators[M]. Boca Raton, FL: CRC Press, 2010

[3] Grondman I, Buşoniu L, et al. A Survey of Actor-Critic Reinforcement Learning: Standard and Natural Policy Gradients[J]. IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews, 2012, 42(6): 1291-1307

[4] Barto A G, Sutton R S, Anderson C W. Neuronlike Adaptive Element That Can Solve Difficult Learning Control Problems[J]. IEEE Trans Syst Man Cybern, 1983, 13: 834-846

[5] Konda V R, Tsitsiklis J N. Actor-Critic Algorithms [C]// Proceedings of Advances in Neural Information Processing Systems. 2000

(下转第249页)

Combinatorial Pattern Matching, CPM 96. Berlin, Germany: Springer-Verlag, 1996: 50-63

[19] Ma B, Tromp J, Li M, et al. PatternHunter: faster and more sensitive homology search[J]. Bioinformatics, 2002, 18(3): 440-445

[20] Egidi L, Manzini G. Better spaced seeds using Quadratic Residues[J]. Journal of Computer and System Sciences, 2013, 79 (7): 1144-1155

[21] Baeza-Yates R, Navarro G. Faster approximate string matching [J]. Algorithmica, 1999, 23(2): 127-158

[22] Myers E W. A sublinear algorithm for approximate keyword searching[J]. Algorithmica, 1994, 12(4/5): 345-374

[23] Sutinen E, Szpankowski W. On the collapse of the q-gram filtration[C]// Proceedings of the International Conference on FUN with Algorithms. Waterloo, Canada: Carleton Scientific, 1998: 178-193

[24] Puglisil S J, Smyth W F, Turpin A. Inverted files versus suffix arrays for locating patterns in primary memory[C]// Proceedings of the 13th Symposium on String Processing and Information Retrieval. Berlin, Germany: Springer Verlag, 2006: 122-133

[25] Karp R M, Rabin M O. Efficient randomized pattern-matching algorithms[J]. IBM Journal of Research and Development, 1987, 31(2): 249-260

[26] Smith T F, Waterman M S. Identification of common molecular subsequences[J]. Journal of Molecular Biology, 1981, 147(1): 195-197

[27] NCBI. UniGene Build #223, Homo sapiens[DB/OL]. ftp. ncbi. nih. gov//repository/ UniGene/Homo_sapiens/Hs. seq. uniq. gz, 2010-04-28

(上接第 238 页)

[7] Eusuff M M, Lansey K E. Optimization of water distribution network design using the shuffled frog-leaping algorithm[J]. Journal of Water Resources Planning and Management, 2003, 129(3): 210-225

[8] 贺毅朝, 曲文龙, 许冀伟. 一种改进的混合蛙跳算法及其收敛性分析[J]. 计算机工程与应用, 2011, 47(22): 37-40

[9] 李晓磊, 邵之江, 钱积新. 一种基于动物自治体的寻优模式: 鱼群算法[J]. 系统工程理论与实践, 2002(11): 32-38

[10] 李晓磊, 冯少辉, 钱积新, 等. 基于人工鱼群算法的鲁棒 PID 控制器参数整定方法研究[J]. 信息与控制, 2004, 33(1): 112-115

[11] Yang X S. Biology-derived algorithms in engineering optimization[M]// Olarius S, Zomaya A Y, eds. Handbook of Bioinspired Algorithms and Applications. Chapman & Hall /CRC, 2005

[12] Yang X S. Nature-Inspired Metaheuristic Algorithms[M]. UK: Luniver Press, 2008

[13] Engelbrecht A P. Computational intelligence: an introduction [M]. Wiley Publishing. Inc. , 2009

[14] Wolpert D H, Macready W G. No Free Lunch Theorems for Optimization [J]. IEEE Transactions on Evolutionary Computation, 1997, 1(1): 67-82

[15] Pringle J W S. A physiological analysis of cicada song[J]. J. Exp. Biol. , 1954, 32: 525-560

[16] Simmons P J. Periodical cicada: sound production and hearing [J]. Science, 1971, 171: 212-213

[17] 刘勇, 康立山, 陈毓屏. 非数值并行算法——遗传算法[M]. 北京: 科学出版社, 2003: 22-86

[18] Iosifescu M. Finite Markov processes and their applications [M]. Chichester: Wiley, 1980

[19] 江瑞, 罗予频, 胡东成, 等. 一种协调勘探和开采的遗传算法: 收敛性及性能分析[J]. 计算机学报, 2001, 24(12): 1233-1241

[20] Yao Xin, Liu Yong, Lin Guang-ming. Evolutionary programming made faster[J]. IEEE Transactions on Evolutionary Computation, 1999, 3(2): 82-102

[21] He S, Wu Q H, Saunders J R. Group search optimizer: An optimization algorithm inspired by animal searching behavior[J]. IEEE Transactions on Evolutionary Computation, 2009, 13(5): 973-990

(上接第 242 页)

[6] Rosenstein M T, Barto A G. Supervised Learning Combined with an Actor-Critic Architecture[J]. CMPSCI Technical Report 02-41. October 2002

[7] Peters J, Schaal S. Natural actor-critic [J]. Neurocomputing, 2008, 71(7-9): 1180-1190

[8] Bathnagar S, Sutton R S, Ghavamzadeh M, et al. Natural actor-critic algorithms[J]. Automatica, 2009, 45(11): 2471-2482

[9] Vamvoudakis K G, Lewis F L. Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem[J]. Automatica, 2010, 46(5): 878-888

[10] Grondman I, Vaandrager M, Busoniu L, et al. Efficient Model Learning Methods for Actor-Critic Control[J]. IEEE Transactions on Systems Man and Cybernetics Part B-Cybernetics, 2012, 42(3): 591-602

[11] Grondman I, Vaandrager M, Busoniu L, et al. Actor-Critic Control with Reference Model Learning[C]// Proceedings of the 18th IFAC World Congress. Milan, Italy, 2011: 14723-14728

[12] Kuvayev L, Sutton R. Model-Based Reinforcement Learning with an Approximate, Learned Model[C]// Proceedings of the Ninth Yale Workshop on Adaptive and Learning Systems. 1996: 101-105

[13] Goschin W S, Littman M. Integrating sample-based planning and model-based reinforcement learning[C]// Proc. Assoc. Adv. Artif. Intell. . Atlanta, GA, 2010: 612-617

[14] Santamaria J, Sutton R, Ram A. Experiments with reinforcement learning in problems with continuous state and action spaces [J]. Adaptive Bechavior, 1998, 6: 163-138

[15] Sherstov A A, Stone P. Function Approximation via Tile Coding: Automating parameter choice[C]// Zucker J-D, Saitta L, eds. SARA, volume 3607 of Lecture Notes in Computer Science. Springer, 2005: 194-205

[16] Lanzi P L, Loiacono D, Wilson S W, et al. Classifier Prediction based on Tile Coding[C]// Proceedings of the 2006 Geneticand Evolutionary Computation Conference Workshop Program (GECCO 2006). Seattle, Washington, 2006: 1497-1504