文章编号: 1000−8152(2011)11−1595−06

基于概率模型的动态分层强化学习

戴朝晖, 袁姣红, 吴 敏, 陈 鑫

(中南大学 信息科学与工程学院, 湖南 长沙 410083)

摘要: 为解决大规模强化学习中的“维度灾难”问题, 克服以往学习算法的性能高度依赖于先验知识的局限性, 本文提出一种基于概率模型的动态分层强化学习方法. 首先基于贝叶斯学习对状态转移概率进行建模, 建立基于概率参数的关键状态识别方法, 进而通过聚类动态生成若干状态子空间和学习分层结构下的最优策略. 仿真结果表明该算法能显著提高复杂环境下智能体的学习效率, 适用于未知环境中的大规模学习.

关键词: 动态分层强化学习; 贝叶斯学习; 状态转移概率模型; 智能体

中图分类号: TP273 **文献标识码**: A

Dynamic hierarchical reinforcement learning based on probability model

DAI Zhao-hui, YUAN Jiao-hong, WU Min, CHEN Xin

(School of Information Science and Engineering, Central South University, Changsha Hunan 410083, China)

Abstract: To deal with the overwhelming dimensionality in the large-scale reinforcement-learning and the strong dependence on prior knowledge in existing learning algorithms, we propose the method of dynamic hierarchical reinforcement learning based on the probability model (DHRL−model). This method identifies some key states automatically based on probability parameters of the state-transition probability model established based on Bayesian learning, then generates some state-subspaces dynamically by clustering, and learns the optimal policy based on hierarchical structure. Simulation results show that DHRL−model algorithm improves the learning efficiency of the agent remarkably in the complex environment, and can be applied to learning in the unknown large-scale world.

Key words: dynamic hierarchical reinforcement-learning; Bayesian learning; state-transition probability model; agent

1 引言(Introduction)

强化学习因具有自学习和在线学习的良好特性, 使其成为机器学习领域的一个重要分支[1]. 它包括模型无关法和基于模型法[2]. 采用模型无关法(如Q−学习)时, 智能体未利用已有的经验盲目搜索, 学习效率低下. 而基于模型法因减少对环境的盲目探索而显著提高了学习效率. 因此, 后者引起了研究者们的关注, 并取得了一系列研究成果, 典型算法有Dyna、贝叶斯学习[3]和Prioritized Sweeping等.

在大规模高维度的决策环境中, 无论采用基于模型法, 还是模型无关法都难以克服“维度灾难”问题(学习参数个数随状态的维度成指数级增长). 目前, 分层强化学习(hierarchical reinforcement learning, HRL)[4, 5]是解决此问题的有效方法. 典型的HRL方法有Option[6], HAM[7], MAXQ[8], HEXQ[9]等. Option, HAM和MAXQ方法均不能自动分层, 而HEXQ尽管能自动分层, 但它只能用于可分解的马尔可夫问题. 由于在未知环境下学习时, 任务层次结构难以事先确定, 因此需要寻求一种新的对先验知识依赖程度很小的动态分层强化学习(dynamic hierarchical reinforcement learning, DHRL)方法.

利用基于模型法可以提高学习效率, 且学习过程中记录的大量信息能为DHRL提供数据支持, 而采用DHRL有利于解决“维度灾难”问题且对先验知识的依赖程度很小, 因此若能将二者有机结合起来, 则能既提高学习效率, 又有利于解决未知环境下的“维度灾难”问题. 目前, 在基于模型法和静态HRL(需利用先验知识预先分层)的融合方面取得了一定的成果[10~14], 但它们在具有极少先验知识的条件下, 其应用受到限制. 而将基于模型法和DHRL结合的研究目前存在很多难点, 但该研究对解决未知环境下的“维度灾难”问题具有重大的意义. 因此, 本文提出一种适用于未知环境下的基于概率模型的动态分层强化学习方法DHRL−model.

收稿日期: 2010−05−09; 收修改稿日期: 2011−01−21.

基金项目: 国家自然科学基金资助项目(60874042); 中国博士后科学基金一等资助项目(20080440177); 中国博士后科学基金特别资助项目(200902483); 教育部高等学校博士点基金新教师基金资助项目(20090162120068).

2 基于概率模型的动态分层(Dynamic hierarchy generation based on probability model)

假定环境是马尔可夫型的, 强化学习过程可以用Markov decision processes(MDPs)[15]表示, MDP用四元组$< S, A, p_{\rm T}, p_{\rm R} >$来定义, 其中$S$和$A$分别表示状态集和动作集, $p_{\rm T}(s,a,s')$表示在状态s执行动作a转移到状态s'的概率, $p_{\rm R}(s,a)$表示在状态s执行a获得的奖赏. 强化学习的目标是要获得一个最优策略, 使智能体选择的动作获得环境的最大奖赏. 然而, 在大规模高维度的决策环境下, 不可避免地出现“维度灾难”问题.

HRL是目前解决“维度灾难”问题的有效方法, 但它主要为静态分层方法. 因此, 针对大规模的未知环境, 需要提出一种对先验知识依赖程度很小的DHRL方法(指在学习过程中自动分层, 并在线学习子任务内部策略, 进而完成全局优化策略的搜索).

在Option学习中, 若能识别一些入口状态和出口状态, 便能自动划分子任务, 但其内部策略难以确定. MAXQ在线学习能力强, 但它的自动分层能力差. 因此, 若能将Option和MAXQ的定义方式融合起来形成一种新的子任务定义方式, 则能在利用一些关键状态实现动态分层的基础上, 较快地搜索到最优策略.

在分层Option中, 与值函数相应的Bellman方程[16]为

$$Q(s,o)= R(s,o)+\sum_{s'}P(s'|s,o)\max Q_{o'\in O_s}(s',o'), \quad (1)$$

其中:

$$\begin{cases} R(s,o)= \\ \mathrm{E}\{r_{\rm t}+\gamma r_{t+1}+\cdots+\gamma^{\tau-1}r_{t+\tau-1}|\varepsilon(o,s,t)\}, \\ P(s'|s,o)=\sum\limits_{\tau=1}^{\infty}\gamma^{\tau}p(s',\tau), \end{cases} \quad (2)$$

E表示在状态s下执行Option o所获得奖赏值的期望, γ为折扣因子, r为即时奖赏, τ为o持续的时间, O_s为所有Option的集合, $p(s',\tau)$为o从状态s开始经τ个时间步后在状态s'终止的概率, $\varepsilon(o,s,t)$表示t时刻在状态s下o被启动. 将式(2)代入式(1), 则与Option的值函数相应的Bellman方程可改为

$$Q(s,o)=\mathrm{E}\{\sum_{u=0}^{\tau-1}\gamma^u r_{t+u}+\sum_{u=\tau}^{\infty}\gamma^u r_{t+u}|s_{\rm t}=s,\mu\}. \quad (3)$$

MAXQ将给定任务W分解为子任务集$\{W_0,W_1,\cdots,W_n\}$以及将策略π分解为策略集合$\{\pi_0,\pi_1,\cdots,\pi_n\}$, 其中$\pi_i$是$W_i$的策略. 假设子任务$W_i$选择的第一个动作为$a$, 且$a$执行$N$步后在状态$s'$结束, 则根据值函数为无限折扣奖赏和之定义, $Q(i,s,a)$可表示为

$$Q(i,s,a)=\mathrm{E}\{\sum_{u=0}^{N-1}\gamma^u r_{t+u}+\sum_{u=N}^{\infty}\gamma^u r_{t+u}\}, \quad (4)$$

其中:

$$\mathrm{E}\{\sum_{u=0}^{N-1}\gamma^u r_{t+u}\}=V(a,s),$$
$$\mathrm{E}\{\sum_{u=N}^{\infty}\gamma^u r_{t+u}\}=C(i,s,a),$$

则$Q(i,s,a)=V(a,s)+C(i,s,a)$, 即通过MAXQ值函数分解, 将Q值划分为$V(a,s)$和$C(i,s,a)$两部分, $V(a,s)$表示在状态s下执行动作a的立即奖赏, $C(i,s,a)$表示从状态s'(动作a执行N步后结束时对应的状态)开始完成任务W_i的期望回报.

从式(3)和式(4)可知, Option和MAXQ的分层立意是相似的, 即将值函数Q值在某时刻拆分为两部分, 在不同层内分别计算这两部分的值, 最后再将二者相加起来.

因此, 本文结合Option和MAXQ的定义, 提出用六元组$< S_i, A_i, p_{{\rm T}i}, p_{{\rm R}i}, U_i, T_i >$来定义子任务$W_i$. 其中$S_i$, A_i, $p_{{\rm T}i}$, $p_{{\rm R}i}$, U_i和T_i分别为W_i的状态集、动作集、状态转移概率、奖赏函数、启动状态集和终止状态集. 此定义下, 在状态$s\in U_i$处执行子任务W_i, 在状态$s'\in T_i$处终止W_i, 并利用MAXQ的学习机理搜索递归最优策略. 从而可以将Option便于划分子任务和MAXQ在线学习能力强的优势结合起来实现DHRL.

DHRL的首要问题在于自动识别一些关键状态来自动分层. 而基于模型法不但提高了学习效率, 而且在学习过程中记录了大量信息, 如状态转移概率、每个状态的前导状态和后继状态等, 通过分析这些数据可以获得一些子目标点来实现自动分层. 当环境模型参数$p_{\rm T}$和$p_{\rm R}$已知时, 值函数的最优方程为

$$Q^*(s,a)=E_{p_{\rm R}}[r|s,a]+\sum_{s'\in S}\gamma p_{\rm T}(s,a,s')V^*(s'), \quad (5)$$
$$V^*(s)=\max_{a\in A}Q^*(s,a). \quad (6)$$

而当一个动作需要多个时间步才能完成时, 采用SMDP模型来描述值函数的最优方程, 如式(7)所示. 其中τ为在状态s执行动作a所需的时间步数.

$$Q^*(s,a)=E_{p_{\rm R}}[r|s,a]+\sum_{s',\tau}\gamma^{\tau}p_{\rm T}(s',\tau|s,a)V^*(s'), \quad (7)$$

综上所述, 基于模型的学习过程中记录的信息能为动态分层提供数据支持, 而由Option和MAXQ结合后的子系统定义方式便于实现动态分层, 且具有良好的在线学习能力. 因此, 在未知大规模环境下学习时, 将基于模型法和DHRL结合起来, 是提高学习

效率并解决“维度灾难”问题的一种有效途径. 然而, 要将这二种算法结合起来, 目前还存在较多的问题, 例如: 1) 如何自动识别出一些子目标点; 2) 识别子目标点后如何自动分层; 3) 如何动态调整系统的分层结构; 4) 如何保证算法的收敛性.

3 DHRL-model算法(DHRL-model algorithm)

本文提出一种新的基于模型的动态分层学习算法DHRL-model, 该算法在基于模型的学习过程中基于概率模型自动识别一些关键状态来实现自动分层, 进而学习分层结构下的最优策略, 并在以后的学习过程中动态调整分层结构. 因此, 为方便阐述动态分层算法, 需要首先定义基于模型的学习算法的一般表述形式.

3.1 基于模型的强化学习(Model-based reinforcement learning)

本文把贝叶斯学习引入到基于模型的强化学习中, 通过概率的方法来估计环境的动态变化, 在线建立环境模型.

设$\mathrm{suc}(s,a)=s'=\{s'_i,\ i=1,\cdots,N\}$($N$为可观测的后继状态数), 表示状态-动作对$(s,a)$的一步后继状态集, 则状态$s$的一步后继状态集

$$\mathrm{suc}(s)=\bigcup_{1\leqslant i\leqslant|A|}\mathrm{suc}(s,a_i),\ a_i\in A.$$

状态转移概率

$$p_{\mathrm{T}}(s,a,s')=[p_{\mathrm{t}}(s,a,s'_1),\cdots,p_{\mathrm{t}}(s,a,s'_N)],$$
$$p_{\mathrm{t}}(s,s'_i)=\sum_{j=1}^{|A|}p_{\mathrm{t}}(s,a_j,s'_i),\ a_j\in A,$$

其中$|A|$为动作集A中的元素个数. 并做如下假设:

假设 1　智能体所在状态空间中的状态离散且有界, 即$|S|<\infty,\|s\|\leqslant\infty$;

假设 2　智能体在有限动作集A中选择动作, 即A中元素的个数$|A|<\infty$;

假设 3　$p_{\mathrm{T}}(s,a,s')$满足狄利克雷分布, 即$f(p_{\mathrm{T}})\propto\prod\limits_{1\leqslant i\leqslant N}p_{\mathrm{t}}^{\alpha_i}(s,a,s'_i)$, 其中$\alpha_i$是一个超参数;

假设 4　执行(s,a)获得的奖赏$p_{\mathrm{R}}(s,a)$满足均值为μ, 标准差为σ的正态分布, 其先验分布$f(\mu)\propto N(\mu_0,\sigma_0^2)$, μ_0, σ_0根据先验知识预先设定.

贝叶斯学习算法描述如下: 分别从$p_{\mathrm{T}}(s,a,s')$和$p_{\mathrm{R}}(s,a)$的先验分布中采样获得状态转移概率和即时回报的假设值, 并根据式(5)和式(6)分别计算$Q^*(s,a)$和$V^*(s)$的假设值, 进而根据式(8)确定最优策略, 并执行选定的策略完成一次试验. 然后根据式(9)～(11)计算得到p_{T}和p_{R}的分布参数(即方差ψ和均值μ)的后验概率, 完成模型参数的更新[3]. 其中: $\psi=1/\sigma$, n为(s,a)的出现次数, $\bar{x}$为(s,a)的平均奖赏值, d^2为立即奖赏的方差, O为(s,a)的所有可观测的后继状态总数, o_i为(s,a)的某后继状态s'_i的计数.

$$a=\arg\max_{a\in A}Q^*(s,a),\tag{8}$$

$$f(p_{\mathrm{T}}|O)\propto\prod_{1\leqslant i\leqslant N}p_{\mathrm{t}}^{\alpha_i+o_i}(s'_i),\tag{9}$$

$$f(\psi|O)\propto\psi^{n-1}\exp(-\psi^2(nd^2+\sigma_0^2)/2),\tag{10}$$

$$f(\mu|O)\propto N(\bar{x},\sigma^2/n).\tag{11}$$

3.2 动态分层(Dynamic hierarchy generation)

动态分层学习中首要的问题是如何基于概率模型自动识别一些关键状态作为子目标状态点, 进而将整个状态空间通过聚类生成若干个状态子空间, 并在以后的学习过程中动态调整系统的分层结构.

动作 1　关键状态的识别.

定义 1　路径P_{ks}为在第k个试验周期(从起点到达终点的学习过程为一个试验周期)中, 智能体按照时间顺序探索过的所有状态构成的序列, 即P_{ks}=$\{s(1),s(2),\cdots,s(i),\cdots\}$, $s(i)=s_j,s_j\in S$.

则由$\bigcup\limits_{1\leqslant k\leqslant c}P_{ks}$中所有状态构成的空间为智能体已探明的状态空间, 其中c为学习的试验周期数.

定义 2　$D(s_i)$为状态s_i的所有一步后继状态构成的集合, 即$D(s_i)=\{s_j\,|s_j\in\mathrm{suc}(s_i)\}$.

智能体在有限动作集A中选择动作, 若$|D(s_i)|$比较少, 说明在s_i处执行动作受限, 即s_i为瓶颈状态, 而具有“瓶颈性”是关键状态的一个充分条件. 因此, 在完成一个试验周期的探索后, 统计路径P_{1s}中所有不同状态分别对应的$D(s_i)$, 并将满足$|D(s_i)|\leqslant\delta$的状态s_i加入集合S'_{t}中, 其中δ是预先给定的阈值.

由于在每次试验的路径中, 某关键状态s_i相当于桥梁来连通该路径中此关键状态前后访问次数较高(即状态转移概率较大) 的两个状态, 设分别为s_{i1}和s_{i2}, 则$D(s_{i1})\cap D(s_{i2})=\{s_i\}$是状态$s_i$为关键状态的另一个充分条件. 其中:

$$s_{i1}=\arg\max_{s_j\in D(s_i)}p_{\mathrm{t}}(s_i,s_j),\ s_i\in S'_{\mathrm{t}},$$
$$s_{i2}=\arg\max_{s_j\in D(s_i)/\{s_{i1}\}}p_{\mathrm{t}}(s_i,s_j),\ s_i\in S'_{\mathrm{t}}.$$

因此, 关键状态集(子目标集)S_{t}表示为

$$S_{\mathrm{t}}=\{s_i||D(s_i)|\leqslant\delta,\ D(s_{i1})\cap D(s_{i2})=\{s_i\}\},\tag{12}$$

其中s_{i1}和s_{i2}为$D(s_i)$中状态转移概率较大的前两位分别对应的状态.

动作 2　状态子空间的自动划分.

在一个试验周期结束后, 执行动作1获得了关键状态集S_{t}, 此时可利用S_{t}中的关键状态来对已探明的状态空间进行聚类生成初始的分层结构, 为后续的动态调整任务层次提供基础. 而为了划分状态空间, 本文首先采用改进的k-均值聚类算法对S_{t}中的

所有关键状态进行聚类.

Step 1 对所有关键状态进行聚类.

对$S_{\rm t}$中的所有关键状态, 按照其在本次试验的路径P_{ks}中出现的先后顺序, 依次将它们加入$P_{\rm key}$中, 从而获得关键状态序列

$$P_{\rm key}=\{s(1),s(2),\cdots,s(i),\cdots\},\ s(i)=s_{\rm t},s_{\rm t}\in S_{\rm t}.$$

例如, 若$P_{ks}=\{s_1,s_2,s_3,s_4,s_3,s_2,s_3,s_4,s_5,s_3,s_6\}$, $S_{\rm t}=\{s_2,s_4,s_5\}$, 则$P_{\rm key}=\{s_2,s_4,s_2,s_4,s_5\}$. 设关键状态序列$P_{\rm key}$中各状态的序号为$X=(x_1,x_2,\cdots,x_i,\cdots)$, 则聚类算法的具体步骤为:

1) 从X中任意选取m(如$m=0.8|S_{\rm t}|$)个样本作为初始聚类中心$\bar{x}_1(1),\bar{x}_2(1),\cdots,\bar{x}_m(1)$.

2) 在第b次迭代中, 按下述规则把全部样本分配到m个类别中. 对$\forall x_{\rm t}\in X$, 令

$$j=\arg\min_i\|x_{\rm t}-\bar{x}_i(b)\|,\ i\in\{1,2,\cdots,m\},$$

则$x_{\rm t}\in Z_j(b)$, 其中$Z_j(b)$为以$\bar{x}_j(b)$为中心的聚类中的样本集合.

3) 计算新的聚类中心$\bar{x}_i(b+1)=\sum\limits_{x_{\rm t}\in Z_i(b)}x_{\rm t}/n_i$, 其中$n_i$为$Z_i(b)$中的样本数.

4) 如果$\bar{x}_i(b+1)=\bar{x}_i(b),i=1,2,\cdots,m$, 则说明聚类中心不再发生变化, 转向5). 否则$b=b+1$, 转向2).

5) 若$\|\bar{x}_i(b)-\bar{x}_j(b)\|\geqslant\varepsilon,\ i\neq j,\forall i,j\in\{1,2,\cdots,m\}$($\varepsilon$根据先验知识预先设定), 则聚类结束. 否则将$Z_i(b)$和$Z_j(b)$合并为一类, 且令$m=m-1$, 再转向5).

X中的全部样本通过聚类生成了m个集合Z_i, $i=1,2,\cdots,m$. 根据X中元素与$P_{\rm key}$中元素的一一对应关系, 通过映射$g:Z_i\to F_i$, 可获得m个关键状态集合$F_i,i=1,2,\cdots,m$, 并用 $s_{i,n}\in F_i$表示F_i中的某关键状态.

Step 2 根据关键状态类生成状态子空间.

由于探索的随机性, 某关键状态$s_{i,n}$在一次试验中会出现多次. 假设状态$s_{i,n}$在试验中某次出现时, 它的后继N_b(预设的常数)步内探索的所有状态构成的集合为$R(s_{i,n},N_b)$, 它的前溯N_b步内对应的所有状态构成的集合为$P(s_{i,n},N_b)$, 令$J(s_{i,n},N_b)=R(s_{i,n},N_b)\cap P(s_{i,n},N_b)$, 当满足$|J(s_{i,n},N_b)|=0$, 表示智能体已暂时走出某区域, 即此刻的$s_{i,n}$可作为临界状态来划分状态空间.

从某F_i中任意选取两个关键状态$s_{i,u}$和$s_{i,n}$, 利用下述的方法自动形成一个状态子集W_{un}. 本次试验中, 当满足$|J(s_{i,n},N_b)|=0$时, 设$s_{i,n}$在路径P_{ks}中对应的序号为t_n, 状态$s_{i,u}$在P_{ks}中对应的序号为$t_j,j=1,2,\cdots,b$(b为$s_{i,u}$在P_{ks}中出现的次数), 令

$$u=\arg\min_j(t_n-t_j),\text{s.t. }t_j<t_n,j=1,2,\cdots,b,$$

则将P_{ks}中序号介于t_u和t_n间的所有状态总和起来作为一个状态子集W_{un}. 由于关键状态$s_{i,u}$和$s_{i,n}$是任意选取的, 所以根据关键状态类F_i获得的状态子集

$$W_i=\{s_i\,|s_i\in W_{un},\forall s_{i,u},s_{i,n}\in F_i\}.\qquad(13)$$

按照这样的方式, 依次对每个关键状态类F_i, 生成相应的状态子空间$W_i,i=1,2,\cdots,m$. 此外, 对路径P_{ks}中某些仍未归入任一子空间W_i的状态s_r, 将其归入$D(s_r)$中元素所在的子空间内, 从而将本次试验中已探明的所有状态归入相应的子空间内, 实现整个状态空间的自动划分.

为了获得任意子系统W_i的启动状态集U_i和终止状态集T_i, 假设$\forall s_{i,n}\in F_i$, 其首次出现于$P_{\rm key}$中时, 对应的序号为$x^1_{i,n}$, 若满足:

$$\max\{x^1_{i,o}\}\geqslant x^1_{i,n},\min\{x^1_{i,p}\}\leqslant x^1_{i,n},$$

$$s_{i,o},s_{i,p}\in F_i,\forall s_{i,n}\in F_i,s_{i,o}\neq s_{i,n},s_{i,p}\neq s_{i,n},$$

其中$s_{i,o}$和$s_{i,p}$为$x^1_{i,o}$和$x^1_{i,p}$分别对应的状态, 则可以令启动状态集$U_i=\{s_i|s_i\in F_i/\{s_{i,o}\}\}$, 终止状态集$T_i=\{s_i|s_i\in F_i/\{s_{i,p}\}\}$. 根据此规则依次获得所有子系统的启动状态集$U_i$和终止状态集$T_i$, $i=1,2,\cdots,m$.

动作 3 分层结构的动态调整.

初次分层后, 每次试验结束时, 需要对已有的分层结构进行动态调整, 降低由于探索不充分导致分层不合理的局限性.

首先, 对探索到的新状态s_i, 利用式(12)判断它是否为关键状态. 若s_i是关键状态, 则先利用改进的k-均值聚类方法将s_i归入某关键状态类F_n中, 并拆分原状态子空间W_n, 然后按照Step 2中状态子空间的形成方法重新生成对应的状态子空间. 否则, 将s_i归入$D(s_i)$中元素所在的子空间内.

其次, 重新判断子目标集$S_{\rm t}$中的所有状态是否仍满足式(12), 若因已探明状态空间的变化, 某状态s_i不再满足式(12), 则应从$S_{\rm t}$中删除s_i, 并拆分原分层结构中与s_i相关的所有关键状态类及其对应的子空间, 然后, 将被拆分的关键状态采用改进的k-均值聚类方法重新生成新的关键状态类, 并按照Step 2中子空间形成的方法获得相应的状态子空间.

为了保证至少存在一条从起点至终点的连续路径, 则最少需要一个包含起点和终点在内的紧的状态集合, 所以必须限定任意一子系统的某个启动状态为另一个子系统的终止状态.

3.3 算法流程(Flow of the algorithm)

在DHRL-model算法中, 将任务W分解为子任务集$\{W_0, W_1, \cdots, W_n\}$(初始时$n=1$), 子任务$W_i$用六元组$< S_i, A_i, p_{\mathrm{T}i}, p_{\mathrm{R}i}, U_i, T_i >$定义, 该算法的具体流程如下:

1) 在第1个试验周期内采用贝叶斯学习算法对环境进行探索和学习，同时记录概率模型等信息;

2) 第1次试验结束时, 根据式(12)获得关键状态集S_{t}, 然后采用改进的k-均值聚类算法对S_{t}进行聚类，再根据式(13)依次生成若干状态子空间, 完成初次分层, 并根据式(9)～(11)更新模型, 并开始第2次试验;

3) 在第$k(k \geqslant 2)$个试验周期内, 利用Bayesian-MAXQ[13]算法学习分层结构下的最优策略, 本次试验结束时, 利用动作3调整分层结构, 然后判断是否达到预定义的步数, 若是, 转4), 否则循环执行3);

4) 本次训练结束.

3.4 算法的性能分析(Performance analysis of the algorithm)

本小节从收敛性、复杂性以及在未知大规模环境下的有效性方面来分析DHRL-model算法的性能.

该算法中，自动分层和基于模型的策略学习采用顺序执行的方式. 显然, 当基于改进的k-均值聚类的自动分层算法和基于MAXQ分层结构的递归最优策略学习分别收敛时, DHRL-model能够收敛. 一般来说, MAXQ算法能收敛到递归最优策略. 因此, 只要基于改进的k-均值聚类的自动分层算法能够收敛, DHRL-model算法就能实现动态分层条件下的最优策略自学习.

DHRL-model算法的时间复杂性为$O(n)$. 具体来说, 由于动作1中利用了已记录的信息, 计算代价可忽略不计. 执行动作2生成状态子空间过程的时间复杂度为$O(n)$. 而动作3只对新探索到的状态和发生变化的状态空间进行调整, 故此过程的复杂度$\leqslant O(n)$.

DHRL-model算法动态生成了类似于MAXQ分层结构, 对先验知识的依赖程度很小, 且能利用MAXQ值函数逼近学习递归最优策略, 有效提高了学习效率, 适用于分层结构未知的大规模环境.

4 仿真实验与分析(Simulation and analysis)

以图1所示的二维有障碍栅格的未知环境中两点间最短路径规划为任务背景, 对DHRL-model算法的有效性进行测试. 图1中: 黑色网格表示障碍物, 其他网格为可以通行的区域, 起点和终点位置随机选择. 学习前, 智能体对环境信息完全未知, 且可以执行上、下、左、右4个基本动作, 执行动作后以0.8的概率转移到预期方向, 分别以0.1的概率转移到与预期方向垂直的两个方向上. 学习过程中, 利用波尔兹曼策略(Boltzmann policy)获得探索动作, 即

$$\pi(s) = \exp(\max_{a \in A} Q(s,a)/T) / \sum_{a \in A} \exp(Q(s,a)/T),$$

并设到达目标点的奖励信号为40, 其他均为-1.

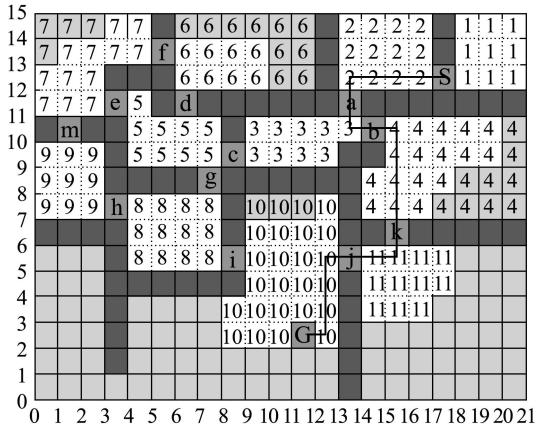

图 1 DHRL-model学习结果

Fig. 1 Learning results of DHRL-model

DHRL-model 算法识别的关键状态总数和动态生成的子空间数目随试验周期的变化情况如图2所示. 通过若干周期的反复试错和学习, 该算法最终的学习效果如图1所示, 其中: 黑色线段表示智能体学习获得的最优路径, 浅灰色网格表示没有探明的状态, 深灰色网格表示最终的所有关键状态点, 整个状态空间被分成11个分区, 每个分区用不同的数字表示.

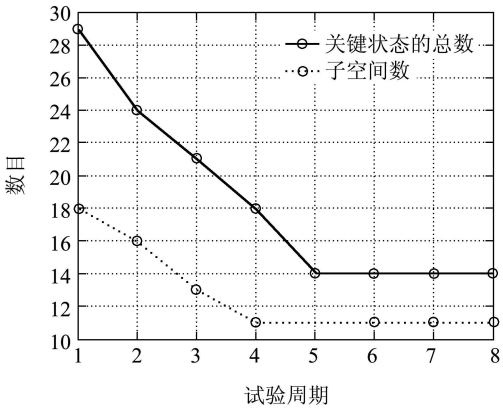

图 2 关键状态数目和子空间数目

Fig. 2 Number of key states and sub-spaces

为了比较Q-学习、Dyna-Q、一般的贝叶斯学习和DHRL-model 4种学习算法的性能, 分别采用上述算法(设置相同的学习参数, 如折扣因子为0.99, 温度系数为1000)在图1所示的环境下重复进行50次训练(每次训练至少完成200次试验). 图3和图4 给出了4种算法各自在每次试验中获得的平均奖赏值和完成一次试验所需的平均步数.

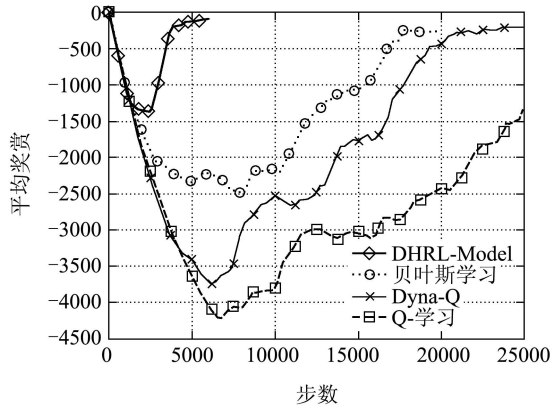

图 3 每次试验中获得的平均奖赏值

Fig. 3 Average reward obtained every trial

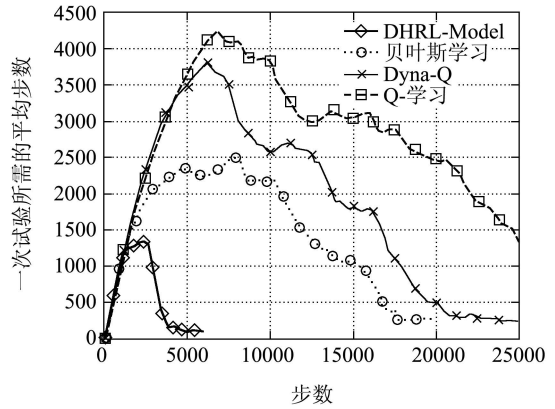

图 4 完成一次试验所需的平均步数

Fig. 4 Average steps required to complete a trial

此外, 为了在人工分层的条件下将DHRL−model算法与MAXQ算法进行对比, 本文分别利用这两种算法在图1所示的环境中来完成出租车的载客任务. 其中, 共有4个站台(18,13), (7,10), (12, 3)和(10,15), 乘客处于任意一个站台, 出租车的起点和乘客的终点位置随机选择, 智能体需要接载乘客并将其运送到目的地. 经过50次训练后, 两种算法完成一次试验所需的平均步数如图5所示.

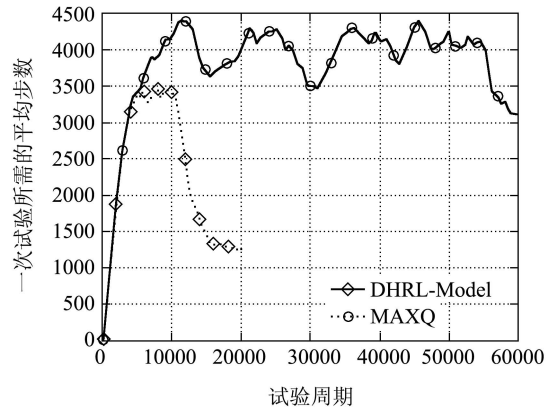

图 5 MAXQ和DHRL−model的性能比较

Fig. 5 Performance comparison between MAXQ and DHRL−model

由图3∼图5可知, DHRL−model的收敛速度明显快于Q−学习、Dyna−Q、一般的贝叶斯学习和MAXQ学习, 而且完成一次试验所需的平均步数远小于其它几种算法, 即DHRL−model算法能在较高收敛速度的情况下保证求得最优解, 显著提高了未知环境下智能体的自学习效率.

5 结论(Conclusion)

本文在基于模型的强化学习过程中, 提出了一种动态分层学习算法. 智能体利用已有的模型知识学习优化策略, 显著提高了学习效率, 且能在未知环境下实现动态分层, 有利于解决大规模学习时的“维度灾难”问题, 从而能在较高收敛速度的情况下保证求得最优解. 因此, DHRL−model算法有效降低了以往学习算法的性能高度依赖于先验知识的局限性, 适合于解决未知环境下的大规模强化学习问题.

参考文献(References):

[1] 高阳, 陈世福, 陆鑫. 强化学习研究综述[J]. 自动化学报, 2004, 30(1): 86 – 100.
(GAO Yang, CHEN Shifu, LU Xin. Research on reinforcement learning technology: a review[J]. *Acta Automatica Sinica*, 2004, 30(1) : 86 – 100.)

[2] KAELBLING L P, LITTMAN M L. Reinforcement learning: a survey[J]. *Journal of Artificial Intelligence Research*, 1996, 4(1): 237 – 285.

[3] STRENS M. A Bayesian framework for reinforcement learning[C] //*Proceeedings of the 17th International Conference on Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000: 943 –950.

[4] 沈晶, 顾国昌, 刘海波. 分层强化学习中的动态分层方法研究[J]. 小型微型计算机系统, 2007, 28(2): 287 – 291.
(SHEN Jing, GU Guochang, LIU Haibo. Dynamic hierarchies in hierarchical reinforcement learning[J]. *Journal of Chinese Computer Systems*, 2007, 28(2): 287 – 291.)

[5] 彭志平, 李绍平. 分层强化学习研究进展[J]. 计算机应用研究, 2008, 25(4): 974 – 978.
(PENG Zhiping, LI Shaoping. Research progress on hierarchical reinforcement learning[J]. *Application Research of Computers*, 2008, 25(4): 974 – 978.)

[6] SUTTON R S, PRECUP D, SINGH S. Between MDPs and Semi-MDPs: a framework for temporal abstraction in reinforcement learning[J]. *Artificial Intelligence*, 1999, 112(1): 181 – 211.

[7] PARR R E. *Hierarchical control and learning for Markov decision processes*[D]. Berkeley, CA: University of California, 1998.

[8] DIETTERICH T G. Hierarchical reinforcement learning with the MAXQ value function decomposition[J]. *Journal of Artificial Intelligence Research*, 2000, 13(1): 227 – 303.

[9] HENGST B. Discovering hierarchy in reinforcement learning with HEXQ[C] //*Proceedings of the Nineteenth International Conference on Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc, 2002: 243 – 250.

[10] JONG N K, STONE P. Hierarchical model-based reinforcement learning: R-MAX + MAXQ[C] //*Proceedings of the 25th International Conference on Machine Learning*. New York: ACM, 2008: 432 – 439.

(下转第1606页)