计算机研究与发展
Journal of Computer Research and Development
ISSN 1000-1239/CN 11-1777/TP
45(9): 1470-1476, 2008

基于路径匹配的在线分层强化学习方法

石 川[1,2] 史忠植[2] 王茂光[2]
[1](北京邮电大学北京市智能软件与多媒体重点实验室 北京 100876)
[2](中国科学院计算技术研究所智能信息处理重点实验室 北京 100190)
(shic@ics.ict.ac.cn)

Online Hierarchical Reinforcement Learning Based on Path-matching

Shi Chuan[1,2], Shi Zhongzhi[2], and Wang Maoguang[2]
[1](*Smart Software and Multimedia of Beijing Key Laboratory, Beijing University of Posts and Telecommunications, Beijing* 100876)
[2](*Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Beijing* 100190)

Abstract Although reinforcement learning (RL) is an effective approach for building autonomous agents that improve their performance with experiences, a fundamental problem of the standard RL algorithm is that in practice they are not solvable in reasonable time. The hierarchical reinforcement learning (HRL) is a successful solution which decomposes the learning task into simpler subtasks and learns each of them independently. As a promising HRL, option is introduced as closed-loop policies for sequences of actions to enable HRL. A key problem for HRL based on options is to discover the correct subgoals online. Through analyzing the actions of agents in subgoals, two useful properties are found: (1) the subgoals have more possibility to be passed through and (2) the effective actions in subgoals are restricted. As a consequence, subgoals can be regarded as the most matching action-restricted states in the paths. Considering the grid environment, the concept of unique-direction value is proposed to denote the action-restricted property, and the option discovering algorithm based on unique-direction value is introduced. The experiments show that the options discovered by the unique-direction value method can speed up the primitive Q learning significantly. Moreover, the experiments further analyze how the size and generating time of options affects the performance of Q learning.

Key words reinforcement learning; hierarchical reinforcement learning; option; subgoal; path-matching

摘 要 如何在线找到正确的子目标是基于option的分层强化学习的关键问题.通过分析学习主体在子目标处的动作,发现了子目标的有效动作受限的特性,进而将寻找子目标的问题转化为寻找路径中最匹配的动作受限状态.针对网格学习环境,提出了单向值方法表示子目标的有效动作受限特性和基于此方法的option自动发现算法.实验表明,基于单向值方法产生的option能够显著加快Q学习算法,也进一步分析了option产生的时机和大小对Q学习算法性能的影响.

关键词 强化学习;分层强化学习;option;子目标;路径匹配

中图法分类号 TP18

为了解决强化学习中的维数灾难问题,很多研究者提出了分层强化学习方法(hierarchical reinforcement learning, HRL)[1].它的基本思想是引入抽象(abstraction)机制实现状态空间降维,将强化学习任务分解到抽象内部和抽象间的不同层次上分别实现,从而每层上的学习任务仅需要在低维

收稿日期:2007-02-05;**修回日期**:2008-02-15
基金项目:国家自然科学基金项目(60402011, 90604017, 60675010);国家"十一五"科技支撑计划基金项目(2006BAH03B05)

空间中进行.目前典型的方法有 option[2],HAM[3]和 MAXQ[4],它们分别从动作、策略和任务的角度考察分层机制.option 方法是对 MDP 中的元动作进行扩展,将若干个动作合并形成一个宏动作,这些 option 构成了主体的动作选择集.每个 option 都有自己的动作策略,使得 option 内部的初始状态有效地到达对应的子目标(subgoal).对较大规模的问题,合适的 option 能够显著提升学习主体的效率[5].产生合适的 option 的关键在于如何在学习过程中自动地找到正确的子目标.研究者已经提出了许多自动找到子目标的方法.本文在已有的工作基础上提出了一种新颖的在线自动找到子目标的方法.

1 相关工作

在分层强化学习环境下,运用 option 算法已经被证明不仅可以大大加速当前任务的学习速度,而且已学到的知识还可以应用在其他类似的学习任务之中[6-8].构造 option 的关键在于自动地产生子目标.不同的方法对子目标的定义也并不相同,但是都认为子目标是主体必须经过的有用状态.目前已经提出了很多自动产生子目标的方法,这些方法都是根据主体过去的经验找到子目标.一类方法是根据状态的访问频率的统计特征找到子目标,苏畅和高阳等人用访问次数的变化率寻找子目标[9];Solle 和 Precup 等人将访问频率最高的结点作为子目标[10];McGovern 等人提出子目标是那些在成功到达终点的路径上面经常访问的结点,而不在那些没有成功到达终点的路径上面[11];Simsek 和 Barto 等人提出 relative novelty 的概念记录邻近状态的访问频率的比率的方法找到子目标.另一类方法是利用状态转移图构造子目标,Simsek 和 Wolfe 等人通过划分由最近经验构成的局部状态转移图找到子目标[12];Menache 和 Mannor 等人提出利用主体访问的历史图通过 Max-Flow/Min-cut 算法找到子目标[13].这些产生子目标的方法又可以分为离线方法和在线方法.离线方法事先产生 option,然后应用于不同的学习任务.这种方法虽然能够加快一些学习任务,但是需要较多的预处理时间,并且对于新的任务要重新学习,因此实用性并不强,文献[9-10]就是这种方法.在线方法就是在运行过程中产生 option,能够自动加快学习过程,是一种更加主流的方法.机器学习方法也广泛应用于寻找子目标的任务中.文献[5]将寻找子目标作为一个多示例学习问题;一些研究者利用聚类的方法自动构造 option[9,12,14];还有一些利用图论中的划分算法找到子目标[12-13].

2 在线分层强化学习方法

首先我们通过一个实例观察学习主体在网格环境中的动作特性.图 1 显示了网格学习环境:黑色的格子表示禁止通行的墙壁,主体可以在白色格子中上下左右移动.我们假设已知图 1 中整个状态空间的策略,并从中随机选择一些初始和终结状态.根据已有的策略,可以找到从初始状态到终结状态的最短路径,并且记录学习主体在每个状态的动作.图 1 显示 4 个随机任务和它们在每个状态的动作.很明显,图中有 4 个有用的子目标,它们分别是 $s(2,5)$,$s(5,2)$,$s(5,7)$和 $s(7,5)$(图中灰色的格子).这里 $s(x,y)$表示网格中的第 x 行和第 y 列.每一条路径都通过两个子目标,并且这些路径在子目标处的动作要么是同向的(例如 $s(2,5)$和$s(5,2)$),要么是反向的(例如 $s(5,7)$和 $s(7,5)$).但是其他状态的动作不受这样的限制,例如子目标附近的状态($s(2,6)$和 $s(4,2)$)或者其他区域的状态($s(2,7)$).

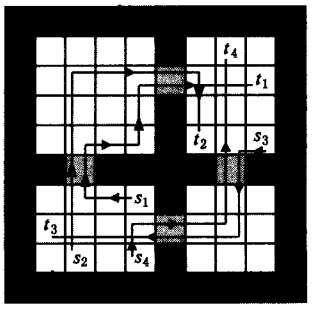


Fig. 1 Action property of subgoals in 10×10 grid.

图 1 10×10 网格环境中子目标的动作特性

通过观察主体在子目标处的动作,我们发现这些动作有下面两个特点.1)如果始末状态不在同一个区域,成功的路径必须要经过一些子目标.这个特性使得子目标有更大的访问频率.一些基于访问频率的找子目标的方法都是根据这个特性.例如,在 Stolle 和 Precup 的方法中,如果随机任务中一些状态有更高的访问频率,那么这些状态可能更重要[10].这个特性就是子目标的高频特性.2)主体在子目标处的有效动作是受限的.为了完成学习任务,主体必须沿着初始状态到终结状态的方向通过子目标.不同于其他状态,主体在子目标处的有效动作是受限的:动作的方向必须相同或者相反.这个特性

叫做子目标的动作受限特性.

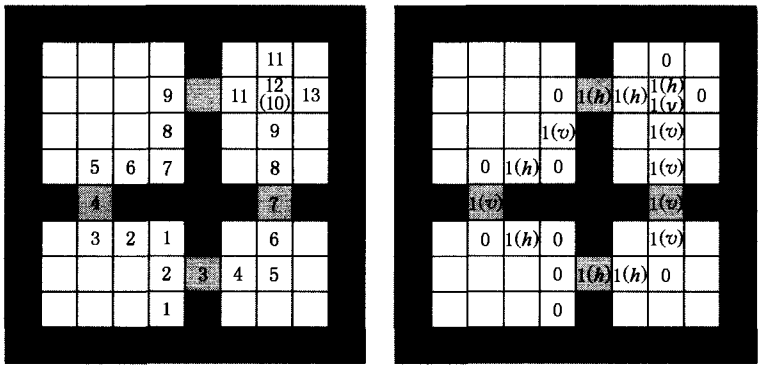
根据高频特性,不同路径在子目标处匹配次数应该更多,即子目标是高频匹配状态.但是高频匹配状态可能不是子目标,子目标附近的状态往往也是高频匹配状态.根据动作受限特性,不同路径在子目标处的动作被限制成同向或者反向.以图 1 为例,路径〈$s1$, $t1$〉和路径〈$s2$, $t2$〉在状态 $s(6,2)$, $s(5,2)$, $s(4,2)$, $s(2,4)$, $s(2,5)$和 $s(2,6)$处匹配,但只有状态 $s(2,5)$和 $s(5,2)$是动作受限的,因此只有这两个状态是潜在的子目标.通过上面的分析,我们认为路径中最匹配的动作受限状态是子目标.这样寻找子目标的问题转化为在学习路径中找到最匹配的动作受限状态.

2.1 单向值方法

寻找子目标也就是在学习路径中找到最匹配的动作受限状态.实际上,由于路径匹配有较高的时间复杂度,这个问题是比较难求的.对于网格环境的学习任务,我们提出用单向值的方法解决这个问题.单向值表示动作受限特性,它能够有效区分子目标和其他状态.这样寻找子目标就是找路径中具有最大单向值的状态.下面我们详细介绍该方法.

2.1.1 将路径映射到网格环境

具体的方法是路径中的状态按照它们的访问次序标号.如果一个状态被多次访问,只记录最后一次访问的次序.如图 2(a)所示,它将两条路径〈$s1$, $t1$〉和〈$s4$, $t4$〉映射到网格环境.



h/v in the bracket means horizontal/vertical unique-direction value respectively.

Fig. 2 Calculate the unique direction value. (a) Mapping path 〈$s1$, $t1$〉 and 〈$s4$, $t4$〉 in Fig. 1 into the grid and (b) The unique-direction value of states in Fig. 2(a).

图 2 单向值计算方法. (a) 将图 1 中的路径〈$s1$, $t1$〉和〈$s4$, $t4$〉映射到网格环境;(b) 图 2(a)中状态的单向值

2.1.2 计算路径的每个状态的单向值

由于子目标的有效动作是受限的,因此在网格环境中子目标处的有效动作是上下或者左右方向,而其他状态的动作是上下左右 4 个方向.我们使用状态的单向值表示动作的受限特性.路径中的每一个状态有水平单向值或者竖直单向值,分别被记为 $IsHorDir(s)$和 $IsVerDir(s)$.它们被定义为

$$IsHorDir(s(x,y)) = \begin{cases} 0, & 2Seq(s(x,y)) \neq Seq(s(x,y-1)) + Seq(s(x,y+1)), \\ 1, & 2Seq(s(x,y)) = Seq(s(x,y-1)) + Seq(s(x,y+1)), \end{cases} \quad (1)$$

$$IsVerDir(s(x,y)) = \begin{cases} 0, & 2Seq(s(x,y)) \neq Seq(s(x+1,y)) + Seq(s(x-1,y)), \\ 1, & Seq(s(x,y)) = Seq(s(x+1,y)) + Seq(s(x-1,y)), \end{cases} \quad (2)$$

$s(x, y)$表示在 x 行 y 列的状态. $Seq(s(x,y))$表示状态 $s(x,y)$处的访问顺序,例如 $Seq(s(4,2))=5$.由于路径必须水平或者竖直地通过子目标.因此对于必须水平通过的子目标(如图 1 中的 $s(2,5)$和 $s(7,5)$),所有通过该状态 s 的路径有 $IsHorDir(s)=1$ 且 $IsVerDir(s)=0$.对于必须竖直通过的子目标(如图 1 中的 $s(5,2)$和 $s(5,7)$),所有通过该状态 s 的路径有 $IsVecDir(s)=1$ 且 $IsHorDir(s)=0$.对于路径中的其他非子目标状态各种可能都会出现.因此子目标的单向值和其他状态的单向值是不同的.图 2(b)显示了图 2(a)中状态的单向值.状态的单向值直观表示了路径在该状态是否转弯.状态的单向值为 0 表示路径在该状态转弯,1 表示不转弯.如图 1 所示,路径不能在子目标处转弯,并且必须水平或者竖直地通过子目标,否则将会撞到墙,因此子目标的水平或竖直单向值必定为 1.主体在非子

目标状态可以随意转弯,因此路径中非子目标状态的水平和竖直单向值可以都为 0. 不同的路径可以从不同的方向通过非子目标状态,因此它的水平或竖直单向值可以都为 1. 因此子目标和其他状态的单向值是不同的.

2.1.3 找到子目标

对于所有的路径计算每一个状态的单向值. $HorDirVal(s)$和$VerDirVal(s)$分别表示状态 s 处的水平和垂直单向值.

$$HorDirVal(s)=\sum_{l\in L} IsHorDir(s_l), \quad (3)$$

$$VerDirVal(s)=\sum_{l\in L} IsVerDir(s_l), \quad (4)$$

L 是路径集合,l 是一条路径,s_l 是路径 l 中的状态 s.

状态 s 在所有路径中的单向值被记为 $UniDirVal(s)$,定义如下:

$$UniDirVal(s)=|VerDirVal(s)-HorDirVal(s)|. \quad (5)$$

因为每条路径中子目标的单向值总是 1,而且它只有水平或者垂直的单向值. 因此 $UniDirVal(s)$ 可以更加明显地区分子目标和其他状态. 因为子目标有更大的访问频率,子目标的单向值比其他状态的单向值更大. 我们选择具有最大单向值的状态作为子目标. 如果有 N_{sg} 个子目标我们选择前 N_{sg} 个最大单向值的状态作为子目标.

在单向值方法中用 $UniDirVal$ 表示状态的单向值,这样有效区分了子目标和它的相邻状态. 对于临近子目标的状态 s,如果 s 在通过子目标的路径上被水平(竖直)通过,则 $IsHorDir(s)=1$ ($IsVerDir(s)=1$). 那么 s 在同侧的路径上必然是竖直(水平)通过,则 $IsVerDir(s)=1$($IsHorDir(s)=1$). 对于所有路径 $UniDirVal(s)$将变小,因此子目标和相邻状态的 $UniDirVal$ 有显著的不同.

2.2 构造 option

一旦主体找到子目标就可以在线创建 option 集合. 我们的目标就是要在主体不断试错学习的过程中自动找到 option,从而加快学习过程. 学习过程中有很多成功的路径,利用前面提到的性质,分析这些路径就可以找到合适的子目标,进而产生 option.

首先我们描述一下基于 option 的 Q 学习算法过程:

1) 和环境交互,利用 Q-learning 方法进行学习;

2) 在学习的过程中,如果满足产生 option 的条件,开始记录学习路径,完成路径记录后:

① 利用学习路径产生 option;

② 将 option 加入到现有的 option 集合;

3) 转入到基于 option 的 Q-learning 方法进行学习.

上面的算法中将原子动作作为单步的 option. 相对于经典的 Q 学习方法,上面的学习算法加入了自动产生 option 的过程. 在学习过程中如果条件合适记录一些学习路径;通过这些路径产生 option. 产生 option 的时机需要认真权衡:太早产生 option 则很难找到正确的子目标;太晚开始记录路径,学习系统可能已经得到较好的性能,采用 option 的作用不大. 在我们的算法中定义了两个参数,$lower\_boundary$:开始记录学习路径的时机;$up\_boundary$:结束记录路径的时机. 它们都是运行代数的百分比. 在后面的实验中,我们将进一步的分析这两个参数对算法的影响. 开始产生 option 的时机一般和问题相关的,而且需要进一步的研究,文献[9]讨论了类似的问题.

下面考虑如何构造 option. 根据 option 的定义,我们必须找到输入集合 I、内部策略 π 和终结状态 β. option 发现算法描述如下:

1) 对每条路径

① 将路径映射到网格环境;

② 对路径上的每个状态 s,计算 $IsVerDir(s)$ 和 $IsHorDir(s)$.

2) 找到子目标 β

① 对每个状态 s,对所有的路径计算 $VerDirVal(s)$,$HorDirVal(s)$和 $UniDirVal(s)$;

② 选择 N_{sg} 个 $UniDirVal(s)$值最大的状态作为子目标.

3) 找到输入状态集 I

① 对于每条路径和路径中的每个子目标,将该路径中该子目标的前一个子目标和后一个子目标之间的状态并且与该子目标之间的距离小于 λ 的状态作为该子目标的输入状态;

② 对于每个 option O,它的终结状态是子目标 s. 对所有的路径,合并 s 的所有输入集合作为 O 的输入状态集,并保证输入状态集中没有重复的状态.

4) 学习内部策略 π

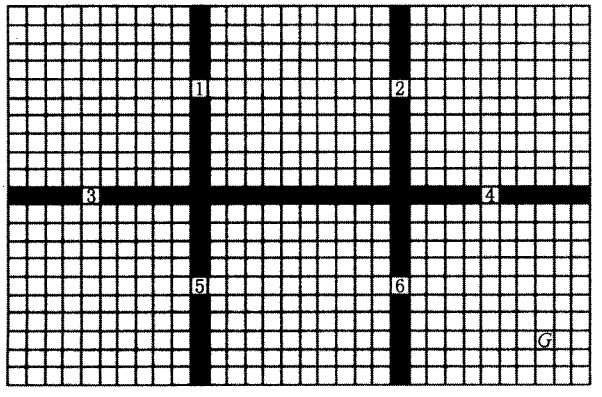
对每个 option 学习它的内部策略. 给子目标设定奖励,其他状态没有奖励. 在初始状态集 I 中随机选择状态作为开始状态,子目标作为结束状态,运行 Q 学习方法,得到内部策略.

算法中的 N_{sg} 是产生子目标的个数. 在第 2.1 节中我们使用了 λ 参数,设定该参数的目的是为了控制 option 的规模. 过大的 option 可能对强化学习过程产生负作用. 另外,如果产生的子目标有错误也可以减少 option 产生的负面影响. 过小的 option 可能也减少 option 的作用,选择合适的 λ 与具体问题相关. 后面的实验分析了 λ 对 option 的性能的影响.

本文提出的基于单向值的 option 产生方法是一种在线分层强化学习算法,它在学习过程中自动产生 option. 本文定义子目标为路径中最匹配的动作受限状态. 这个定义和文献 [5,9-10]中的定义类似,这些方法都是基于访问频率. 本文中提出的子目标不仅有较大的访问频率,而且它的有效动作是受限的. 因此相比他们的方法,本文方法能够更加准确地区分子目标和其他状态. 为了找到子目标,我们将它转化成一个路径匹配问题. McGovern 等人将它转化成一个多事例学习问题[5],Mannor 等人把它作为一个聚类问题[11]. 我们的方法也和状态转移图的方法类似[13],但是我们运用了有效动作受限的特性.

3 实验结果和分析

下面用如图 3 所示的有 6 个子目标的 21×32 网格环境(目标状态是图 3 中的 G 状态)的探索任务验证提出的算法. 子目标是网格环境中的"走廊"(也就是环境中的瓶颈状态),因为从一个房间到另一个房间的路径必须通过这些走廊. 好的 option 的初始状态应该尽可能包括子目标两边的子空间中的状态. 学习任务是要找到主体从各个状态到目标状态的最优动作策略. 状态是图中的格子位置. 主体有 4 个确定的基本动作:上下左右. 如果主体朝墙移动,它将留在原地,并没有惩罚. 折扣因子为 $\gamma=0.9$ 并且没有立即回报. 主体只有达到目标状态才有回报,回报值为 100. 算法独立运行 20 次,结果是多次运行的平均值. 在每次运行中,option 发现算法找到和子目标相同数目的 option. 成功到达终结状态后,随机选择初始状态继续学习. 两个算法中学习率为 $\alpha=0.1$,并且使用 ε-贪婪策略,$\varepsilon=0.1$. 在实验中,我们把所有状态的平均 Q 值作为时间步的函数,其中每次与环境的交互称为一个时间步. 实验比较使用基本动作的学习算法和使用 option 的学习算法中所有状态的平均 Q 值的变化情况.



1-6 is the subgoals in the environment, and G is the goal state.

Fig. 3 Rooms environment in 21×32 grid world.

图 3 21×32 网格的学习任务

图 4 显示了在不同的 option 产生时机所有状态的平均 Q 值随时间步的变化关系. 实验中 λ 为 $ROW/3$(ROW 为 21×32 网格环境的宽度,即为 32). 除了 Primitive Q 学习外,其他算法都使用了 option 方法. 图中 Cand 表示不同的 option 的产生时机. Cand1 表示[0.1,0.2],Cand2 表示[0,0.1],Cand3 表示[0.2,0.3],括号中的数分别表示 *lower_boundary* 和 *up_boundary*. 从图中我们发现,在启用 option 之前 4 种算法差别不大,option 启动之后,学习算法要比没有采用 option 的 Q 学习方法要好,它们更快达到了最大 Q 值,但是不同的[*lower_boundary*, *up_boundary*]对算法性能影响比较大. 在条件 Cand1 时提升性能最明显. 由于学习了一些路径后才开始记录路径,这样记录的路径更能产生好的 option. 好的 option 对 Q 学习的性能提升明显. 在条件 Cand2 时从一开始就记录路径. 这些路径包含的有用信息比较少,产生 option 的质量也不高,因此虽然 option 对 Q 学习的性能有提升,但是

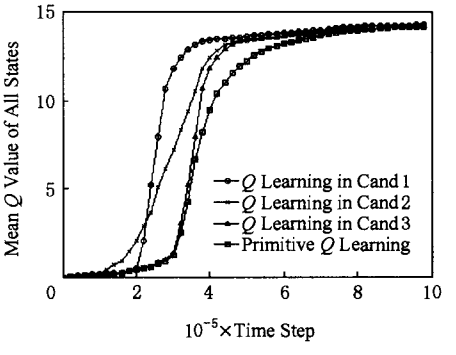


Fig. 4 The relation between the mean Q-value of all states and time step in different option-generating time.

图 4 不同 option 产生时机情况下所有状态的平均 Q 值随时间步的变化关系

提升速度比较慢.在条件 Cand3 时,开始产生 option 的时间比较晚.虽然产生的 option 对 Q 学习性能有提升,但是由于此时 Q 学习算法已经开始快速收敛,所以这种提升并不显著.通过实验,我们发现产生 option 的时机对算法性能有比较大的影响.利用经典的 Q 学习算法,学习主体在开始阶段性能的改善十分缓慢,当运行到一定阶段后性能会突然得到极大的提高,然后又缓慢地收敛到最优解.我们要根据这个特点选择合适的 option 产生的时机,这样才能提升算法性能.对于本算法而言,应该运行一段时间以后开始记录学习路径,但是要在 Q 学习快速收敛之前产生 option.

图 5 显示了选择不同的 λ 的情况下不同算法的所有状态的平均 Q 值随时间步的变化关系.除了 Primitive Q 学习外,其余的都采用了 option 方法,图中 Cand 表示不同的 option 大小.Cand1 表示 $\lambda=ROW/3$,Cand2 表示 $\lambda=ROW$,Cand3 表示 $\lambda=ROW/6$,ROW 表示网格环境的宽度(即为 32).实验中 *lower_boundary* 为 0.1,*up_boundary* 为 0.2.通过图 5 发现在产生 option 之前 4 种算法性能都差不多;当产生 option 之后采用 option 的算法性能大多比没有采用 option 的算法性能要好.在 Cand1 情况下($\lambda=ROW/3$)算法性能最好,在产生的 option 帮助下,算法很快收敛到最优值.当 $\lambda=ROW/3$ 时,option 的初始状态刚好是子目标左右两边的子空间之中的状态,这样 option 大小正合适,因此算法性能也最好.在 Cand2 情况下($\lambda=ROW$),开始算法的性能比没有采用 option 的算法性能要好,但是后期性能比没有用 option 的算法性能反而还要差.当 $\lambda=ROW$ 时产生的 option 比较大,它对算法性能可能有副作用.在 Cand3 情况下($\lambda=ROW/6$)算法性能也有较快的提升,但是比 Cand1 情况下的性能要差一些.当 $\lambda=ROW/6$ 时产生的 option 比较小,虽然它对算法有所提升,但是提升不够显著.选择合适的 λ 对算法性能也有比较大的影响.λ 的选择和具体问题相关,合适的 λ 应该尽可能包含子目标两边的子空间中的状态.

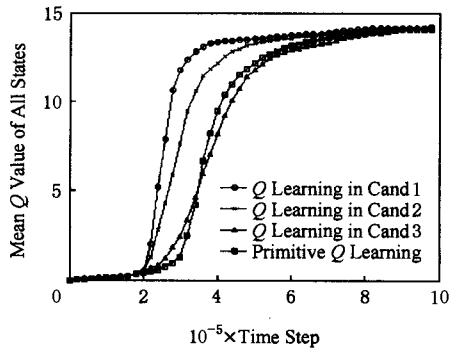


Fig. 5 The relation between the mean Q-value of all states and time step in different option size.

图 5 不同 λ 情况下所有状态的平均 Q 值随时间步的变化关系

4 结 论

基于 option 的分层强化学习的关键问题在于学习过程中如何自动找到子目标.本文提出了子目标的有效动作受限的启发式策略.根据这个策略,寻找子目标的问题转化为寻找路径中最匹配的动作受限状态.针对网格环境的学习任务,提出了单向值来定量表示动作受限特性,进而设计了单向值方法找到子目标.测试实验表明单向值方法相比基于频率的方法更能够准确地找到子目标.实验表明,本文提出的学习方法比基于原始动作的 Q 学习方法要快得多.实验还分析了产生 option 的时机和大小对算法性能的影响.在主体学习一段时间之后,并且在 Q 学习算法性能开始大幅提高之前产生的 option 对 Q 学习算法的性能提升比较显著.太大或者太小的 option 对算法性能的提升都不太显著,合适的 option 应该尽量包括子目标两边子空间的状态.产生 option 的时机和大小与具体问题相关,如何自适应地设定这些参数是我们下一步需要继续研究的问题.

参 考 文 献

[1] Gao Yang, Chen Shifu, Lu Xin. Research on reinforcement learning technology: A review [J]. Acta Automatica Sinica, 2004, 30(1): 86-100 (in Chinese)
(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)

[2] 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 (1-2): 181-211

[3] Parr R. Hierarchical control and learning for Markov decision process [D]. Berkeley, USA: University of California, 1998

[4] Dietterich T G. Hierarchical reinforcement learning with the MAXQ value function decomposition [J]. Journal of Artificial Intelligence Research, 2000, 13: 227-303

[5] McGovern A, Barto A G. Automatic discovery of subgoals in reinforcement learning using diverse density [C] //Proc of Int Conf on Machine Learning. New York: ACM Press, 2001: 325-333

[6] Bernstein D S. Reusing old policies to accelerate learning on new MDPs, UM-CS-1999-026 [R]. Amherst: University of Massachusetts, 1999

[7] Iba G A. A heuristic approach to the discovery of macro-operators [J]. Machine Learning, 1989, 3(4): 285-317

[8] Precup D. Temporal abstraction in reinforcement learning [D]. Amherstm: University of Massachusetts, 2000

[9] Su Chang, Gao Yang, Chen Shifu, *et al*. The study of recognizing Options based on SMDP [J]. Pattern Recognition and Artificial Intelligence, 2005, 18(6): 679-684 (in Chinese)
(苏畅, 高阳, 陈世福, 等. SMDP 环境下自主生成 Options 的算法研究[J]. 模式识别与人工智能, 2005, 18(6): 679-684)

[10] Stolle M, Precup D. Learning options in reinforcement learning [C] //Proc of the 5th Int Symp on Abstraction, Refromulation and Approximation. Berlin: Springer, 2002: 569-677

[11] Mannor S, Menache I, Hoze I, *et al*. Dynamic abstraction in reinforcement learning via clustering [C] //Proc of the 21st Int Conf on Machine Learning. New York: ACM Press, 2004: 560-567

[12] Simsek Ö, Wolfe A P, Barto A G. Identifying useful subgoals in reinforcement learning by local graph partitioning [C] //Proc of the Int Conf on Machine Learning. New York: ACM Press, 2005: 248-256

[13] Menache I, Mannor S, Shimkin N. Q-cut-dynamic discovery of subgoals in reinforcement learning [C] //Proc of the 13th European Conf on Machine Learning. Berlin: Springer, 2002: 295-306

[14] Wang Bennian, Gao Yang, Chen Zhaoqian, *et al*. K-Cluster subgoal discovery algorithm for option [J]. Journal of Computer Research and Development, 2006, 43(5): 851-855 (in Chinese)
(王本年, 高阳, 陈兆乾, 等. 面向 Option 的 *k*-聚类 subgoal 算法[J]. 计算机研究与发展, 2006, 43(5): 851-855)

[15] Stolle M, Precup D. Learning options in reinforcement learning [C] //Proc of the 5th Int Symp on Abstraction, Refromulation and Approximation. Berlin: Springer, 2002: 569-677

Shi Chuan, born in 1978. Ph. D. and lectuer. member of CCF, His main research interests include evolutionary computation, machine learning and data mining.

石 川, 1978 年生, 博士, 讲师, 中国计算机学会会员, 主要研究方向为进化计算、机器学习、数据挖掘.

Shi Zhongzhi, born in 1941. He is a professor and Ph. D. supervisor in the Key Laboratory of Intelligent Information Processing, the Institute of Computing Technology, CAS. Senior member of CCF. His main research interests include intelligence science, multi-agent systems, semantic Web, machine learning and neural computing. He is a senior member of IEEE, a member of AAAI and ACM. He serves as vice president for the Chinese Association of Artificial Intelligence.

史忠植, 1941 年生, 研究员, 博士生导师, 计算机学会高级会员, 主要研究方向为智能科学、多主体系统、语义 Web、机器学习和神经计算等.

Wang Maoguang, born in 1974. Ph. D. and assistant professor. His main research interests include artifical intelligence, software engineering, and autonomic computing.

王茂光, 1974 年生, 博士, 副教授, 主要研究方向为人工智能、软件工程、自治计算等.

Research Background

A fundamental problem of the standard reinforcement learning algorithm is that in practice they are not solvable in reasonable time due to the size of the state space and the lack of immediate reinforcement signal. The hierarchical reinforcement learning (HRL) is an effective solution which decomposes the learning task into simpler subtasks and learns each of them independently. As a promising approach automatically defining the required decomposition, option is introduced as closed-loop policies for sequences of actions to enable HRL. The key problem that develops appropriate options automatically is to identify subgoals and learn options for these subgoals. Through analyzing the actions of agents in subgoals, this paper discovers that the effective actions in subgoals are restricted. As a consequence, subgoals can be regarded as the most matching action-restricted states in the paths. Considering the grid environment, this paper proposes the concept of unique-direction value to denote the action-restricted property, and introduce the options discovering algorithm based on unique-direction value further. The experiments show that the options discovered by the unique-direction value method can speed up the primitive *Q* learning significantly. This work is supported by the National Natural Science Foundation of China (No. 60402011, 90604017, and 60675010).