软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2013,24(1):25−36 [doi: 10.3724/SP.J.1001.2013.04258]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

基于点的 POMDPs 在线值迭代算法*

仵 博[1,2,3], 吴 敏[1,2], 佘锦华[4]

[1](中南大学 信息科学与工程学院,湖南 长沙 410083)

[2](先进控制与智能自动化湖南省工程实验室,湖南 长沙 410083)

[3](深圳职业技术学院 教育技术与信息中心,广东 深圳 518055)

[4](School of Computer Science, Tokyo University of Technology, Tokyo 192-0982, Japan)

通讯作者: 仵博, E-mail: wubo@szpt.edu.cn, http://www.szpt.edu.cn

摘 要: 部分可观察马尔可夫决策过程(partially observable Markov decision processes,简称 POMDPs)是动态不确定环境下序贯决策的理想模型,但是现有离线算法陷入信念状态"维数灾"和"历史灾"问题,而现有在线算法无法同时满足低误差与高实时性的要求,造成理想的 POMDPs 模型无法在实际工程中得到应用.对此,提出一种基于点的 POMDPs 在线值迭代算法(point-based online value iteration,简称 PBOVI).该算法在给定的可达信念状态点上进行更新操作,避免对整个信念状态空间单纯体进行求解,加速问题求解;采用分支界限裁剪方法对信念状态与或树进行在线裁剪;提出信念状态结点重用思想,重用上一时刻已求解出的信念状态点,避免重复计算.实验结果表明,该算法具有较低误差率、较快收敛性,满足系统实时性的要求.

关键词: 部分可观察马尔可夫决策过程;信念状态;基于点的算法;在线算法;与或树

中图法分类号: TP301 **文献标识码**: A

中文引用格式: 仵博,吴敏,佘锦华.基于点的 POMDPs 在线值迭代算法.软件学报,2013,24(1):25−36. http://www.jos.org.cn/1000-9825/4258.htm

英文引用格式: Wu B, Wu M, She JH. Point-Based online value iteration algorithm for POMDPs. Ruanjian Xuebao/Journal of Software, 2013,24(1):25−36 (in Chinese). http://www.jos.org.cn/1000-9825/4258.htm

Point-Based Online Value Iteration Algorithm for POMDPs

WU Bo[1,2,3], WU Min[1,2], SHE Jin-Hua[4]

[1](School of Information Science and Engineering, Central South University, Changsha 410083, China)

[2](Hu'nan Engineering Laboratory for Advanced Control and Intelligent Automation, Changsha 410083, China)

[3](Education Technology and Information Center, Shenzhen Polytechnic, Shenzhen 518055, China)

[4](School of Computer Science, Tokyo University of Technology, Tokyo 192-0982, Japan)

Corresponding author: WU Bo, E-mail: wubo@szpt.edu.cn, http://www.szpt.edu.cn

Abstract: Partially observable Markov decision processes (POMDPs) provide a rich framework for sequential decision-making in stochastic domains of uncertainty. However, solving POMDPs is typically computationally intractable because the belief states of POMDPs have two curses: Dimensionality and history, and online algorithms that can not simultaneously satisfy the requirement of low errors and high timeliness. In order to address these problems, this paper proposes a point-based online value iteration (PBOVI) algorithm for POMDPs. This algorithm for speeding up POMDPs solving involves performing value backup at specific reachable belief points, rather than over the entire a belief simplex. The paper exploits branch-and-bound pruning approach to prune the AND/OR tree of belief states online and proposes a novel idea to reuse the belief states that have been computed last time to avoid repeated computation. The experiment and simulation results show that the proposed algorithm has its effectiveness in reducing the cost of computing policies and retaining the quality of the policies, so it can meet the requirement of a real-time system.

* 基金项目: 国家自然科学基金(61074058, 60874042); 国家教育部博士点基金(20090162120068)
收稿时间: 2012-02-03; 修改时间: 2012-04-18; 定稿时间: 2012-05-18

Key words: POMDPs; belief state; point-based algorithm; online algorithm; AND/OR tree

部分可观察马尔可夫决策过程(partially observable Markov decision processes,简称 POMDPs)[1]能够客观、准确地描述真实世界,是随机决策过程研究的重要分支,最近成为计算机、控制和管理等学科研究的热点.综述现有算法,按照年代可大致分为两个阶段:第 1 阶段(20 世纪末),主要是精确求解算法,代表算法为 Witness 算法[1];第 2 阶段(21 世纪初),由于精确求解 POMDPs 过程存在信念状态空间维数灾和迭代更新历史灾问题,该阶段主要是近似求解算法,代表算法为基于点的值迭代算法[2].基于点的值迭代算法的主要思想是:根据误差判定条件,给出固定的有限可达信念状态集合,在其上进行更新操作,避免对整个信念状态空间单纯体进行求解,从而在有限的误差范围内快速求解.更进一步地,卞爱华等人提出基于点的 POMDPs 算法的预处理方法[3],该算法对于解决 POMDPs 维数灾问题有效,但在序贯决策中,信念状态空间会随着时间的推移而呈指数形式爆炸增长.近几年,为了使理想的 POMDPs 模型能够满足实时系统的要求,普遍采用在线近似算法来求解[4].在线近似算法主要是信念状态与或树查找算法,此类算法将 POMDPs 看成智能体与环境之间的博弈,在每一个信念状态结点上,智能体必须选择一个动作,并随机选择下一时刻的观察,在给定的深度内,通过遍历与或树获得当前时刻最优动作.信念状态与或树查找算法可分为蒙特卡罗采样算法、分支界限裁剪算法和启发式搜索算法.

Paquet 提出一种 RTBSS 算法[5],使用分支界限裁剪思想对动作进行在线裁剪.RTBSS 的效率很大程度上依赖于离线求解值函数上下界的精确度,当上下界很紧密的时候,查找算法很高效.但该算法忽略了观察集合,当观察集合很大时,为了在有限时间内求解问题,遍历深度应该很小,但这又将增加最优策略的误差.Wolf 等人在 RTBSS 算法的基础上提出一种基于蒙特卡罗采样的 MC-RTBSS 算法[6],采用粒子滤波来处理连续的信念状态空间.蒙特卡罗采样算法只对观察集合进行裁剪,而忽略了动作集合.当动作集合很大时,该算法的效果并不理想.Ross 等人提出一种 AEMS2 算法[7],它是基于启发式的误差最小化查找算法.启发式查找算法避免对观察或者动作分支进行裁剪,通过使用启发式方法来选择最佳的扩展边缘结点,从而查找出与决策相关性最高的可达信念状态点.在启发式查找算法中,每一个边缘结点都对应着一个启发式值,启发式值决定着是否扩展该边缘结点.每次迭代的目标是在所有边缘节点中寻找能够最大化启发式值的结点,该最佳边缘结点的参照是其子树的父亲结点,利用存储在父亲结点中的参照和启发式值,采用动态规划算法,可以高效地对最佳边缘结点进行扩展和更新.但是启发式查找算法需要计算出需要扩展的边缘结点,并且还要在每次迭代中更新父亲结点的值.因此,它比一般的深度优先和广度优先算法更为耗时.

本文针对现有在线算法和离线算法的优缺点,提出一种基于点的在线值迭代算法(point-based online value iteration,简称 PBOVI).该算法采用基于点的值迭代方法对可达信念状态结点进行更新,避免对所有的信念状态空间进行遍历,极大地降低了问题的求解规模;然后,采用分支界限裁剪方法对信念状态与或树进行在线裁剪,并提出信念状态结点重用思想,重用上一时刻已求解出的信念状态点,避免对已计算过的信念结点的重复计算.

本文中,除特殊说明外,上标代表时间,下标代表集合中的具体实例.例如,s_i 代表状态集合中的第 i 个状态,s^t 代表 t 时刻的状态,$P(s^t=s_i)$代表 t 时刻状态为 s_i 时的概率.有时,根据描述问题的需要,没有上下标,则表示当前时刻的变量,上标是“′”的表示下一时刻的变量.例如,s 表示当前时刻的状态,而 s'表示下一时刻的状态.

本文第 1 节介绍 POMDPs 模型.第 2 节介绍基于点的离线算法.第 3 节介绍 PBOVI 算法.第 4 节是实验和分析.第 5 节是本文结论部分.

1 POMDPs

POMDPs 通常用一个六元组$\langle S,A,T,R,Z,O\rangle$描述[8,9],其序贯决策示意图如图 1 所示.状态集合 $S=\{s_1,s_2,s_3,\ldots,s_n\}$,动作集合 $A=\{a_1,a_2,a_3,\ldots,a_m\}$,观察集合 $Z=\{z_1,z_2,z_3,\ldots,z_l\}$,分别表示所有可能的状态、动作和观察.状态转移函数集合 $T(s_i,a,s_j)=P(s_j|s_i,a)$,是在状态 s_i 下采用动作 a,可能转移到状态 s_j 的概率.观察函数集合 $O(s_i,a_j,z_k)=P(z_k|s_i,a_j)$,是在状态 s_i 下采用动作 a_j,观察为 z_k 的概率.报酬函数集合 $R:S\times A\times Z\mapsto\mathbb{R}$,$R$ 为负值时表示损失,为正值时表示报酬.t 时刻的信念状态 B 由公式(1)来描述,由公式(2)来计算,其中,η为归一化因子.

$$b^t=P(s^t|a^t,z^t,a^{t-1},z^{t-1},a^{t-2},z^{t-2},\ldots,a^0,z^0,s^0) \tag{1}$$

$$b(s')=b^t(s_i)=P(s_i\mid z^t,a^t,s^{t-1})=\eta O(s_i,a^t,z^t)\sum_{j=1}^{|S|}T(s_j,a^t,s_i)P(s^{t-1}=s_j) \tag{2}$$

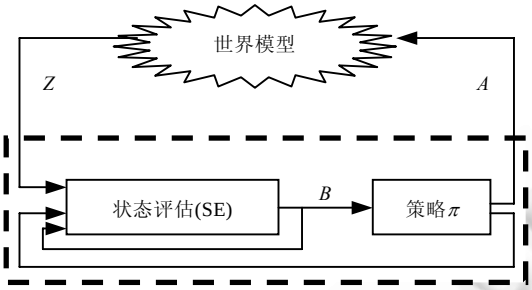


Fig.1 Model of POMDPs

图 1 POMDPs 模型

b^t 可以用 b^{t-1} 来递归求解.根据当前的信念状态和动作,就可以决定下一个周期的信念状态和动作.信念状态转移函数集合$\Gamma(b,a,b')=P(b'|a,b)$,计算如下:

$$P(b_j\mid b_i,a)=\sum_{k=1}^{|Z_{b_j}|}\sum_{l=1}^{|S|}O(s_l,a,z_k)\sum_{m=1}^{|S|}T(s_m,a,s_l)b_j(s_m) \tag{3}$$

公式(3)描述的是,在 b_i 状态下采用动作 a,转移到 b_j 的概率,其中,Z_{b_j} 是在 b_j 状态下的观察集合,$Z_{b_j}\subseteq Z$,$z_k\subseteq Z_{b_j}$. 信念状态报酬函数 $\rho(b,a)=\sum_{s\in S}b(s)R(s,a)$. 使用 Bellman 公式计算 POMDPs 的值函数 V 和策略π,计算如下:

$$V^{t+1}(b)=\max_{a\in A}\left[\sum_{s\in S}b(s)R(s,a)+\gamma\sum_{z\in Z}P(z\mid b,a)V^t(b')\right] \tag{4}$$

$$\pi^{t+1}(b)=\arg\max_{a\in A}\left[\sum_{s\in S}b(s)R(s,a)+\gamma\sum_{z\in Z}P(z\mid b,a)V^t(b')\right] \tag{5}$$

POMDPs 决策的策略是将信念状态映射到动作,即$\pi(b)\to a$,同时满足使智能体选择的动作能够获得环境报酬的累计值最大,即如公式(6)所示:

$$E\left[\sum_{t=0}^{T}\gamma^t\rho_t\right],\gamma\in(0,1] \tag{6}$$

其中,γ为折扣因子,其目标是让期望值收敛.最优策略π^*满足折扣报酬值和的期望值最大,即如公式(7)所示:

$$\pi^*=\arg\max_{\pi\in\Gamma}E\left[\sum_{t=0}^{\infty}\gamma^t\sum_{s\in S}b^t(s)R(s,\pi(b^t))\right] \tag{7}$$

其中,γ^t 为指数,$\pi(b')$表示在 t 时刻的动作,Γ为 α-*vector* 集合.

在 POMDPs 中,最优策略是从所有的信念状态中选取一个动作使得期望值最大,信念状态空间为$\mathbb{R}^{|S|-1}$,其中,$\mathbb{R}$的大小为$|S||A||Z|$.因此,求解信念状态空间是一个“维数灾”问题.Sondik[10]证明,任何 t 时间段的值函数可以用α-*vector* 集合$\Gamma^t=\{\alpha_0,\alpha_1,\ldots,\alpha_m\}$来表示,每一个$\alpha$-*vector* 代表一个$|S|$维的超平面.在每个信念状态上的值函数定义如下:

$$V^t(b)=\max_{\alpha\in\Gamma^t}\sum_{s\in S}\alpha(s)b(s) \tag{8}$$

每一个α-*vector* 对应着一个最优策略,每一个策略对应一个动作.t 时间段的α-*vector* 集合Γ^t 可以如下定义:

$$\Gamma^t=\max_{a\in A}\left[\sum_{s\in S}R(s,a)b(s)+\gamma\sum_{z\in Z}\max_{\alpha\in\Gamma^{t-1}}\sum_{s\in S}\sum_{s'\in S}T(s,a,s')O(s',a,z)\alpha(s')b(s)\right] \tag{9}$$

由于信念状态空间很大,值函数 $V^t(b)$无法直接计算,但与之相对应的Γ^t可以由Γ^{t-1}来迭代计算.精确求解算法步骤如下:

步骤 1. 产生中间集合 $\Gamma^t_{a,*}$ 和 $\Gamma^t_{a,z}$,它们都是$|S|$维的超平面:

$$\Gamma^t_{a,*}\leftarrow\alpha^{a,*}(s)=R(s,a) \tag{10}$$

$$\Gamma^t_{a,z}\leftarrow\alpha^{a,z}_i(s)=\gamma\sum_{s'\in S}T(s,a,s')O(s',a,z)\alpha_i(s') \tag{11}$$

步骤 2. 通过交叉和操作,计算 Γ^a_t.

如果集合 $A=\{a_1,a_2,\ldots,a_m\}$和集合 $B=\{b_1,b_2,\ldots,b_n\}$,$C=A\oplus B=\{a_1+b_1,a_2+b_2,\ldots,a_m+b_n\}$,则称⊕为交叉和操作.

$$\Gamma^t_a=\Gamma_{a,*}\oplus\Gamma^t_{a,z_1}\oplus\Gamma^t_{a,z_2}\oplus\ldots\oplus\Gamma^t_{a,z_{|Z|}} \tag{12}$$

步骤 3. 对 Γ^t_a 进行并操作.

$$\Gamma^t=\bigcup_{a\in A}\Gamma^t_a \tag{13}$$

令$|S|$为状态个数,$|A|$为动作个数,$|Z|$为观察个数,$|\Gamma^{t-1}|$为 $t-1$ 时刻α-*vector* 个数.整个计算过程在最坏情况下,步骤 1 将产生 $O(|A||Z||\Gamma^{t-1}|)$个α-*vector*.然后,算法采用交叉和操作,将会产生 $O(|A||Z||\Gamma^{t-1}|^{|Z|})$个$\alpha$-*vector*.最后,在整个状态空间$|S|$上裁剪被支配向量,此过程为线性规划,时间复杂度为 $O(|S|^2)$.因此,从Γ^{t-1}到Γ^t过程的复杂度为 $O(|S|^2|A||Z||\Gamma^{t-1}|^{|Z|})$,其中,$|\Gamma^{t-1}|=|A|^{\frac{|Z|^{T-1}-1}{|Z|-1}}$,$T$ 为从 0 时刻到 t 时刻的时间长度.因此,求解 POMDPs 策略树还是一个“历史灾”问题.

2 基于点的离线算法

基于点的值迭代算法的主要思想是:根据误差判定条件,给出固定的有限可达信念状态集合 $B=\{b_0,b_1,\ldots,b_m\}$,在 B 上进行更新操作,避免对整个信念状态空间单纯体Δ进行求解,从而在有限的误差范围内快速求解.其中,$B\subset\Delta$.对于每一个信念状态点,值函数至多包含一个α-*vector*.因此,基于点的值函数可以用与之相对应的$\{\alpha_0,\alpha_1,\ldots,\alpha_m\}$集合来表示.其主要步骤如下[11]:

步骤 1. 与精确求解算法一样,求解中间集合 $\Gamma^t_{a,*}$ 和 $\Gamma^t_{a,z}$,$\forall a\in A,\forall z\in Z$.

步骤 2. 由于在有限信念状态点上进行更新操作,因此可以使用简单的加法操作代替交叉和操作.

$$\Gamma^t_a\leftarrow\alpha^a_b=\Gamma^t_{a,*}+\sum_{z\in Z}\arg\max_{\alpha\in\Gamma^t_{a,z}}\left(\sum_{s\in S}\alpha(s)b(s)\right),\forall b\in B \tag{14}$$

步骤 3. 计算α-*vector* 集合Γ^t,并求解每一个信念状态点上的最优动作:

$$\alpha_b=\arg\max_{\Gamma^t_a,\forall a\in A}\left(\sum_{s\in S}\Gamma^t_a(s)b(s)\right),\forall b\in B \tag{15}$$

$$\Gamma^t=\bigcup_{b\in B}\alpha_b \tag{16}$$

整个值函数更新过程的时间复杂度为 $O(|S||A||\Gamma^{t-1}||Z||B|)$.由于 B 为有限数据集合,并且由裁剪算法可知$|\Gamma^t|\leqslant|B|$,因此,$|\Gamma^{t-1}|$不会随着更新时间而呈指数增长,一般情况下,它是一个固定值.

初始化Γ^0 可以使用上界约束,也可以使用下界约束.为了更容易实现收敛,基于点的值迭代算法一般使用下界约束来初始化Γ^0.初始化过程是通过向Γ^0添加一个α-*vector*,它是最差情况下最佳的立即报酬值,以保证满足下界约束:

$$\alpha(s)=\frac{\max_{a\in A}[\min_{s\in S}R(s,a)]}{1-\gamma},\forall s\in S \tag{17}$$

终止条件可以是时间,可以是固定的迭代次数,也可以是期望获得的策略质量等,主要根据实际应用而定.

在终止条件出现之前,可达信念状态点的采集和信念状态点的更新交替进行.

由于在可达信念状态空间单纯体 $\overline{\Delta}$ 上求解值函数是一个 NP 问题,如图 2 所示,可达信念状态点随着规划时间的增加而呈指数形式增长.可达信念状态点的采集是选择一个信念状态子集 $B \subset \overline{\Delta}$,使得 B 既要克服计算难而充分小的问题,又要满足为获得好的近似值函数而充分大的条件.因而,如何采集可达信念状态点,使之满足以上条件,是基于点的 POMDPs 值迭代算法所面临的共同难题.

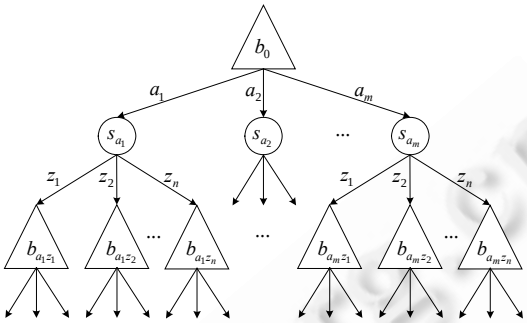


Fig.2 Reachable belief states tree

图 2 可达信念状态树

目前,基于点的值迭代算法的主要区别在于可达信念状态点的采集,信念状态点的更新操作基本相同.本节主要对现有的算法进行复杂度分析,具体见表 1.

Table 1 Analysis and comparison among some point-based algorithms

表 1 几种基于点的算法分析与比较

算法	信念点采集算法	信念点更新算法	采集算法时间复杂度
PBVI	L_1 norm	完全 *Backup* 操作	$O(\|N\|\|A\|(\|S\|\|B\|+\|Z\|))$
Perseus	随机采集	Perseus *Backup*	$O(\|N\|(\|S\|^2+\|A\|+\|Z\|))$
HSVI2	上下界不确定	最新点 *Backup*	$O(\|N\|(\|A\|+\|Z\|\|S\|(\|S\|+\|\Upsilon_{\overline{\Gamma}}\|+\|\Gamma_V\|)))$
SARSOP	错误最小化	完全 *Backup* 操作	$O(\|S\|^3\|A\|\|Z\|\|B\|^2\|N\|)$
FSVI	MDP 启发式	最新点 *Backup*	$O(\|N\|(\|S\|^2+\|A\|+\|Z\|))$

PBVI 和 SARSOP 是完全 *Backup* 操作,时间复杂度为 $O(|S||A||\Gamma^{t-1}||Z||B|)$,其他更新算法时间复杂度都要小于完全 *Backup* 操作,它们之间的主要区别为信念点采集算法.从采集算法时间复杂度分析可知,Perseus 算法和 FSVI 算法最优.

与其他 POMDPs 近似求解算法相比,基于点的值迭代算法具有较高的效率和较快的收敛.但是无论采用何种策略,基于点的值迭代算法都不可能在有限时间内遍历整个可达信念状态空间.因而,离线算法只能适用于较小规模的模拟问题,对于大规模实际问题无法实现算法收敛.

3 基于点的在线算法

3.1 算法思想

POMDPs 离线算法是求解全局最优策略或者全局近似最优策略,分为策略规划和策略执行两个阶段.在线算法是求解当前最优策略或者当前近似最优策略,它根据时间约束条件,将整个策略规划和策略执行分成若干个小的规划和执行.由于在线算法只考虑当前信念状态空间下的最优策略,因此可达信念状态空间 $\overline{\Delta}$ 很小.两类算法的比较如图 3 所示.

在线算法将整个决策周期分为若干个小的时间片,每个时间片内分别进行策略规划和策略执行.在规划阶段,根据当前信念状态计算出最优执行动作,可分为两个步骤:

第 1 步是建立可达信念状态与或树(AND-OR tree),如图 4 所示.在图 4 中,假设在线遍历深度 D=3,根结点为当前信念状态,孩子结点为可达信念状态点,可达信念状态点使用公式(2)来计算.信念结点为 OR 结点,其输出弧权重为 $R_B(b,a)$.两层信念点之间的结点为 AND 结点,其输出弧权重为 $P(z|b,a)$;

第 2 步是使用公式(3)计算当前信念状态的值函数,计算过程是通过边缘结点的向上传播算法加以实现[12].

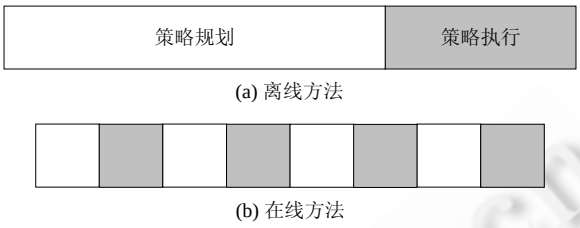


Fig.3 Comparison between offline and online algorithm

图 3 在线与离线方法比较

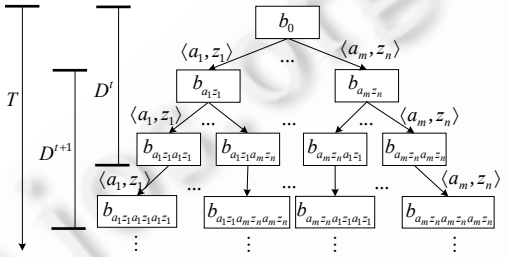


Fig.4 Thought of PBOVI presented by diagram

图 4 PBOVI 算法思想示意图

定理 1[13]. 令 $\hat{V}$ 为近似值函数,用于评估边缘结点的值, $\varepsilon=\sup_b|V^*(b)-\hat{V}(b)|$,$D$-step 的近似值函数 $\hat{V}^D(b)$ 的误差边界为

$$|V^*(b)-\hat{V}^D(b)|\leqslant\gamma^D\varepsilon \tag{18}$$

定理 1 的证明请参见文献[13].评估可达信念状态的时间复杂度为 $O((|A||Z|)^D|S|^2)$.由于$\gamma\in[0,1)$,当 $D\to\infty$时,定理 1 的误差将收敛到 0.因此,在线算法与离线算法的区别在于 D 的取值.在线算法对时间要求很苛刻,遍历所有的可达信念状态点是不可能的,需要在尽可能长地遍历信念状态点,使误差尽可能小与尽可能快地求解近似最优解之间做出权衡.

根据定理 1 可知,当 D 等于全局决策时间长度 T 时,在线算法求解结果与离线算法求解结果相同.因此,可以将在线算法看成离线算法的特例,除时间长度不一样外,其他都相同.在线算法经过对与或树进行 D 次迭代后,求出当前信念状态下最大的报酬值,并返回该报酬值对应的最佳动作,从而完成本时间片内的决策,其时间复杂度为 $O(|A||Z|)^D$.在下一时刻,目前的在线算法都是重新计算报酬值.其缺点是没有重用上一时刻计算的中间结果,造成重复计算,影响求解的速度.

本文提出的 PBOVI 算法利用历史计算结果,避免重复计算历史信念状态,达到快速求解的目的.

在图 4 中,假设在线遍历与或树的深度 D=3.

当处在时刻 t 时,需要计算的信念状态集合为 $b^t=\{b^t_{D=1},b^t_{D=2},b^t_{D=3}\}$,其中, $b^t_{D=1}=\{b_0\},b^t_{D=2}=\{b_{a_1z_1},...,b_{a_mz_n}\}$, $b^t_{D=3}=\{b_{a_1z_1a_1z_1},...,b_{a_mz_n\,a_mz_n}\}$.

当处在 t+1 时刻时, $b^{t+1}=\{b^{t+1}_{D=1},b^{t+1}_{D=2},b^{t+1}_{D=3}\}$,其中, $b^{t+1}_{D=1}=b^t_{D=2},b^{t+1}_{D=2}=b^t_{D=3}$.

因此,在 t+1 时刻可以重复使用时刻 t 已经计算出的信念状态 $b^t_{D=2}$ 和 $b^t_{D=3}$,则当前时刻可重用((D–1)/D)%上

一时刻信念结点.同时,这种求解思路与离线求解思路在理论上是统一的,经过较长时间的在线决策,其结果将与离线决策的结果一致.从以上分析可知,PBOVI 算法从短期来看,可获取当前信念状态下的最佳动作;从长期来看,满足全局最优近似解的求解过程.PBOVI 算法在信念状态更新和扩展操作上采用离线 PBVI 算法[14],在求解最佳动作上采用分支界限裁剪算法,在求解值函数最大上界上采用启发式查找算法.

3.2 PBOVI算法

为了将 Bellman 方程应用到在线算法,本文将值函数和最优策略进行重新定义:

$$V^d(b,d)=\begin{cases}\max\limits_{a\in A}\sum\limits_{s\in S}b(s)R(b,a), & d=0\\ \max\limits_{a\in A}\left[\sum\limits_{s\in S}b(s)R(b,a)+\gamma\sum\limits_{z\in Z}P(z\mid b,a)V^{d-1}(\tau(b,a,z),d-1)\right], & d>0\end{cases} \tag{19}$$

$$\pi^*(b,d)=\arg\max_{a\in A}\left[\sum_{s\in S}b(s)R(b,a)+\gamma\sum_{z\in Z}P(z\mid b,a)V^{d-1}(\tau(b,a,z),d-1)\right] \tag{20}$$

算法 1. PBOVI 算法.

输入为当前信念状态 b 和与或树 T 的深度 d.其中,D 为最大扩展深度,a_{best} 为最佳动作,$R_{\max}(b)$为最大报酬值,R_c 为当前报酬值,$U_T(b)$为报酬值上界,$Length_{AQ}$ 为动作队列的长度,b_c 为当前信念状态点;

输出为 $R_{\max}(b)$.

Step 1. 根据当前信念状态,对动作集合进行大根堆排序 HeapSort(b,A),满足 $U(b,a_i)\geqslant U(b,a_j)$,$i\leqslant j$;

Step 2. 对 $R_{\max}(b)$赋初值,即 $R_{\max}(b)\leftarrow-\infty$;

Step 3. 深度优先遍历与或树 T,即 DFS(T);

Step 4. 对信念状态采用基于点的 BACKUP 操作,即 BACKUP(B,Γ^{t-1});

Step 5. 如果 $b_{new}\notin$beliefQueue,将更新后的信念状态点存储在信念状态队列 beliefQueue 中;

Step 6. 如果 d=0,计算边缘结点的最大报酬值 $\max\limits_{a\in A}\sum\limits_{s\in S}b(s)R(b,a)$, 并将计算结果赋值给 R_c,$R_c\leftarrow R_B(b,a)$;

Step 7. 计算 $U_T(b)=R_c$+Heuristic(b,a,d),如果 $U_T(b)>R_{\max}(b)$,则 $R_c=R_c+\gamma P(z|a,b)$PBOVI($\tau(b,a,z)$,d−1);

Step 8. 如果 $R_c>R_{\max}(b)$,则 $R_{\max}(b)\leftarrow R_c$;

Step 9. 如果($d=D$)∪$\|V^*(b)-R_{\max}(b)\|<\varepsilon$成立,终止条件满足,则获得最佳动作 $a_{best}\leftarrow a$;

Step 10. 选择最小误差信念状态点作为 t+1 时刻的起始信念状态,b_c=EXPAND(B,Γ^t);

Step 11. 把队列中第 D 层的信念状态点删除,beliefQueue·delete(b^D);

Step 12. 输出 $R_{\max}(b)$.

3.3 算法分析

PBOVI 算法 Step 1 根据信念状态对动作发生概率进行从大到小的排序.在与或树 T 中,首先深度遍历概率值大的动作,其目的是可以加快与或树的裁剪速度.本文采用大根堆排序算法,其最坏时间复杂度为 $O(n\log n)$,辅助存储空间为 $O(1)$.Step 3 是深度优先遍历与或树 T,在最坏情况下,时间复杂度为 $O((|A||Z|)^D)$.由于 PBOVI 重用已计算的信念状态,当 t=0 时,其时间复杂度为 $O((|A||Z|)^D)$;当 t>0 时,每一个时间片时间复杂度为 $O(|A||Z|)$.因此,对于整个时间 T,其时间复杂度可以近似为 $O(|A||Z|)$.Step 4 执行 BACKUP 操作,如果更新的信念状态在信念状态队列已经存在,则不用更新,直接重用;如果信念状态队列不存在,则执行更新操作.这样,除了第 1 次需要遍历 D 层外,其他时间只需要遍历 1 层就可以求解当前信念状态的最大报酬值.在整个时间 T 内,其时间复杂度为 $O(|S||A||\Gamma||Z||B|)$.Step 6 计算边缘结点的最大报酬值.Step 7 计算当前信念状态的最大上界,本文使用最大报酬值启发式查找算法来求解最大上界.最大上界如果取值较小,将加速求解过程.但是有时为了计算方便,往往简化为 Heuristic(b,a,d)$=\sum_{i=1}^{d}\gamma^i R_{\max}$. Step 8 是分支界限裁剪条件,当值函数上界不大于当前最大值时,该动作的子树将被裁剪;当终止条件满足时,递归计算当前动作的报酬值.当迭代终止条件满足时,返回最佳动作.Step 10 采用 Pineau 等人[11]提出的 GER 算法,选择最小化误差信念状态点作为 t+1 时刻的起始信念状态.Step 11 把队列中第

D 层的信念状态点删除.综上分析可知,PBOVI 算法在最坏情况下的空间复杂度为 $O((|A|\cdot|Z|)^{D-1})$,最坏情况下的时间复杂度为 $O((|A|\cdot|Z|)^2|\Gamma||B|)$.由于采用基于点的值迭代算法,$|\Gamma|$和$|B|$不会随着更新时间而呈指数增长,一般情况下,它是一个固定值.

3.4 误差及收敛性分析

PBOVI 算法的误差可以从当前误差和长期误差两个方面来分析,当前误差描述的是在特定时间片内迭代 D 步的误差,长期误差是随着时间的推移 PBOVI 算法的误差.

定理 2[5]. 对于任何信念状态 b 和与或树深度 d,当效用函数 $U(b)=R_B(b)$时,PBOVI 算法的最大误差为

$$|\text{PBOVI}(b,d)-V^*(b)|\leqslant\gamma^d\max(|R_{\max}-V_{\min}|,|R_{\min}-V_{\max}|) \tag{21}$$

其中,$R_{\max}$ 和 $R_{\min}$ 分别为最大和最小报酬值,并满足:

$$V_{\max}=\sum_{i=0}^{\infty}\gamma^i R_{\max}=\frac{R_{\max}}{1-\gamma} \tag{22}$$

$$V_{\min}=\sum_{i=0}^{\infty}\gamma^i R_{\min}=\frac{R_{\min}}{1-\gamma} \tag{23}$$

证明:如果 $U(b)=R_B(b)$,则 PBOVI 求解的最优值函数为 $V_d^*(b)$, 即 $\text{PBOVI}(b,d)=V_d^*(b)$.

由三角不等式可知,$|\text{PBOVI}(b,d)-V^*(b)|\leqslant|\text{PBOVI}(b,d)-V_d^*(b)|+|V_d^*(b)-V^*(b)|$.

由已知条件,不等式右边的第 1 个绝对值的结果为 0.

根据定理 1 可知,$|V_d^*(b)-V^*(b)|\leqslant\gamma^d|V_0^*(b)-V^*(b)|=\gamma^d|R_B(b)-V^*(b)|\leqslant\gamma^d\max(|R_{\max}-V_{\min}|,|R_{\min}-V_{\max}|)$,公式(21)成立.

当 $R_B(b)=R_{\max}$ 并且 $V^*(b)=V_{\min}$ 时,或者 $R_B(b)=R_{\min}$ 并且 $V^*(b)=V_{\max}$ 时,$|R_B(b)-V^*(b)|$的差最大,两者之间的最大值即为 PBOVI 算法最大误差.当 $t\to\infty$时,PBOVI 算法值函数 V_B^t 与离线的最优值函数 V^*之间的误差为

$$\|V_B^t-V^*\|_\infty\leqslant\|V_B^t-V^{t,*}\|_\infty+\|V^{t,*}-V^*\|_\infty \tag{24}$$

根据文献[5]的结论,$\|V^{t,*}-V^*\|_\infty$收敛于 $\gamma^{t+D}\|V_0^*-V^*\|$,其中,$D$ 为迭代层次. □

定义 1. 假设 H 代表精确值更新,$\tilde{H}$ 代表 PBOVI 更新,对于 $b\in\Delta$,那么每步的迭代产生的误差为

$$\varepsilon(b)=|\tilde{H}V^B(b)-HV^B(b)|_\infty \tag{25}$$

定义 2. 每步迭代的最大误差ε为

$$\varepsilon=|\tilde{H}V(b)-HV(b)|_\infty=\max_{b\in\Delta}\varepsilon(b) \tag{26}$$

定义 3. 假设δ_B为 B 的密度,它是Δ中所有的信念点到 B 的最大距离,则

$$\delta_B=\max_{b'\in\Delta}\min_{b\in B}\|b-b'\|_1 \tag{27}$$

根据以上 3 个定义,可以得出引理 1,证明请参见文献[11].其中,$R_{\max}=\max_{s,a}R(s,a)$,$R_{\min}=\min_{s,a}R(s,a)$.

引理 1[11]. PBOVI 在 B 上的每步迭代更新产生的误差为

$$\varepsilon\leqslant\frac{(R_{\max}-R_{\min})\delta_B}{1-\gamma} \tag{28}$$

定义 4. t 时刻的 PBOVI 算法值函数 V_B^t 与该时刻最优值函数 $V^{t,*}$之间的误差为

$$\varepsilon_t=\|V_B^t-V^{t,*}\|_\infty \tag{29}$$

定理 3. 对于任意信念状态集合 B 和任意时间 t,PBOVI 算法误差 $\varepsilon_t=\|V_B^t-V^{t,*}\|_\infty$ 收敛于

$$\varepsilon_t\leqslant\frac{(R_{\max}-R_{\min})\delta_B}{(1-\gamma)(1-\gamma^D)} \tag{30}$$

证明:$\varepsilon_t=\|V_B^t-V^{t,*}\|_\infty$,根据 $\tilde{H}$ 的定义,$\|V_B^t-V^{t,*}\|_\infty=\|\tilde{H}V_B^{t-1}-HV^{t-1,*}\|_\infty$.

引入 HV_B^{t-1},由三角不等式可知,$\|\tilde{H}V_B^{t-1}-HV^{t-1,*}\|_\infty\leqslant\|\tilde{H}V_B^{t-1}-HV_B^{t-1}\|_\infty+\|HV_B^{t-1}-HV^{t-1,*}\|_\infty$.

由引理 1 可知,$\|\tilde{HV}_B^{t-1}-HV_B^{t-1}\|_\infty \leqslant \frac{(R_{\max}-R_{\min})\delta_B}{1-\gamma}$.

根据 BACPUP 操作可知,$\|HV_B^{t-1}-HV^{t-1,*}\|_\infty \leqslant \gamma^D \|V_B^{t-1}-V^{t-1,*}\|_\infty$.

再根据定义 4,$\gamma^D \|V_B^{t-1}-V^{t-1,*}\|_\infty = \gamma^D \varepsilon_{t-1}$.

根据数列和原理,$\frac{(R_{\max}-R_{\min})\delta_B}{1-\gamma}+\gamma^D\varepsilon_{t-1} \leqslant \frac{(R_{\max}-R_{\min})\delta_B}{(1-\gamma)(1-\gamma^D)}$,即公式(30)成立.定理 3 证毕. □

由定理 2 和定理 3 可知,在当前时间下的误差为局部误差,随着时间的推移,误差将逼近全局误差,值函数将逼近全局最优.

4 实验与仿真

本文将从 3 个角度评价 PBOVI 算法的有效性:第一是针对 RockSample 问题[14,15]评价不同的遍历深度 D 对 PBOVI 误差的影响,获取 D 的最优取值;第二是针对 RockSample 问题,评价不同在线 POMDPs 算法的计算时间及效果;第三是针对 RoboCup 机器人救援仿真比赛,进行算法有效性比较.本文的仿真平台为 Matlab R2010a,运行环境为 32 位 Windows7,Intel(R) Core(TM) i3 CPU:3.07GHZ,RAM:4GB.

4.1 RockSample问题

RockSample 问题由 Smith 和 Simmons 在 2004 年提出,它模拟的是机器人在火星上采集岩石,通过获取采集岩石动作的报酬值来判断采集的是否为好的岩石.在图 5 中,黑色点为岩石,A 为机器人.RockSample 问题的一般形式为 RockSample[n,k],表示有 $n\times n$ 个格子,k 个岩石.

- 状态集合 S.每一个状态由 k+1 个变量来描述:其中,X_P 为机器人所处的位置,取值范围为{(1,1),(1,2),…,(n,n)};k 个变量 $\{X_R^i\}_{i=1}^k$,每个 X_R^i 表示一个岩石,取值为{*Good*,*Bad*},终止状态为 EXIT.因此,对于 RockSample[n,k]问题,它有 $n^2\times 2^k$+1 个状态.
- 动作集合 A.机器人有 k+5 个动作,取值范围为{*North*,*South*,*East*,*West*,*Sample*,$Check_1$,…,$Check_k$}.前 4 个动作是确定的,描述的是机器人的朝向.*Sample* 动作描述的是在当前位置下采集岩石,当 X_R^c = *Good* 时,返回的报酬值 R=10,并将 X_R^c 赋值为 *Bad*;当 X_R^c = *Bad* 时,R=−10;当处在终止状态 EXIT 时,R=10;其他动作不会有报酬值.
- 观察集合 Z.$Check_i$ 动作是传感器对 X_R^i 进行{*Good*,*Bad*}评判,返回带有噪音的观察值.传感器越接近岩石,观察信息越准确.观察变量由效能因子 $\eta(X_P,i)$所决定.

$$\eta(X_P,i)=2^{-d(X_P,i)/d_0} \tag{31}$$

其中,$d(X_P,i)$为 X_P 与 X_R^i 之间的欧几里德距离,d_0 为可调频常量,称为半效能距离,当 d=d_0 时,η=1/2.当 η=1 时,传感器始终返回正确值;当 η=0 时,传感器返回 *Good* 和 *Bad* 值的几率都为 50%.将所有信念状态初始化状态设为相同的,对所有岩石 *Good* 或者 *Bad* 的概率都设为 0.5,折扣因子 γ=0.95.

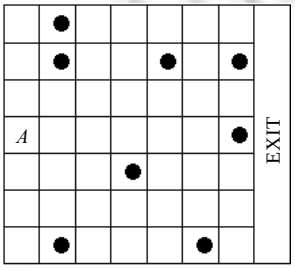


Fig.5 RockSample[7,8]

图 5 RockSample[7,8]

4.2 与或树遍历深度

与或树遍历深度 D 的取值对基于 POMDPs 模型的在线算法至关重要,D 取值过大,将影响算法的实时性;D 取值过小,则算法的误差将很大.图 6 是 PBOVI,RTBSS,MC-RTBSS 和 AEMS2 这 4 种算法在不同遍历深度下所得报酬值的分析比较结果,实验对象为 RockSample[7,8],其状态$|S|$=12545,动作$|A|$=13,观察$|Z|$=2.由图 6 可知,4 种不同算法在遍历深度 D=4 时,基本上收敛;当 D>4 时,无法满足实时系统的时间约束条件;当 D<4 时,算法的误差很大;在报酬值上,由于 PBOVI 算法和 AEMS2 算法都采用最大报酬值启发式查找方法,因此相差不大,但是 PBOVI 算法采用信念状态结点重用方法,遍历深度更深,因而具有更好的性能.

另一方面,还需对算法基本收敛时的计算时间进行分析,图 7 是裁剪与或树和未裁剪与或树的分析比较.由图 7 可知,在采用裁剪与或树算法的情况下,当 D=4 时,在 1s 之内可以计算出近似最优报酬值;在未采用裁剪与或树算法的情况下,当 D=4 时,耗时大约在 300s,无法在线求解近似最优策略.

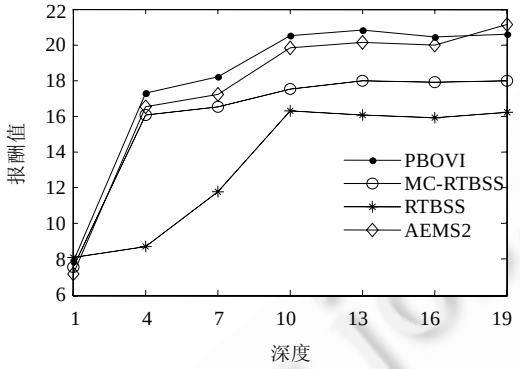


Fig.6 Comparison of convergence speed

图 6 收敛速度比较

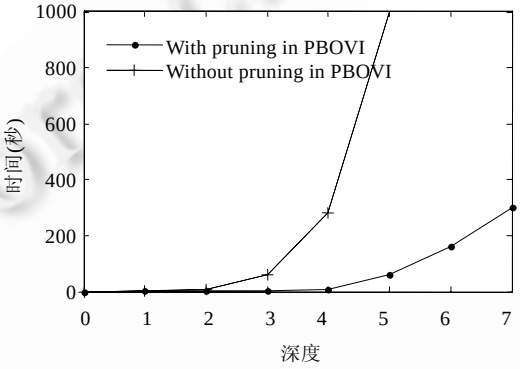


Fig.7 Analysis of time consuming in different depths

图 7 不同遍历深度的耗时分析

从以上实验数据分析可知,本文算法能够对 RockSample[7,8]这类中等规模问题进行实时求解,并能保证算法收敛.

4.3 实时性能分析

评价在线算法主要有两个指标:一个是要满足系统实时性的要求,尽可能快地求解近似最策略;另一个是满足误差最小化目标,尽可能地降低平均折扣报酬值与原始报酬值之间的误差.针对第 2 个评价指标,本文采用文献[4]的误差分析方法论,定义误差降低比率 EBR(b)和下界误差值 LBI(b).其中,在与或树 T 上定义 $V^*(b)$的上下界分别为 $U_T(b)$和 $L_T(b)$,定义边缘结点集合 $F(T)$的上下界分别为 $U(b)$和 $L(b)$.

$$\mathrm{EBR}(b)=1-\frac{U_T(b)-L_T(b)}{U(b)-L(b)} \tag{32}$$

$$\mathrm{LBI}(b)=L_T(b)-L(b) \tag{33}$$

本文的实验对象是 RockSample[7,8],时间约束为每秒执行一个动作,下界值计算统一采用 PBVI 方法,上界值计算统一采用 QMDP 方法,这样可以更加公平地评价算法本身的优劣.表 2 的结果是经过 20 次重复实验,取它们的平均值和标准差.

Table 2 Comparison of several parameters among different online algorithm

表 2 不同在线算法几种参数的比较

算法	报酬值	EBR (%)	LBI	信念状态结点数	结点重用率(%)	计算时间(ms)
AEMS2	21.37±0.22	57.7±0.2	3.08±0.02	2910±46	38.2±0.2	645±3
RTBSS	19.45±0.30	22.4±0.3	1.37±0.04	426±1	0.0±0.0	422±5
MC- RTBSS	21.36±0.22	49.5±0.2	2.73±0.02	2781±38	12.2±0.2	697±2
PBOVI	21.34±0.20	58.9±0.2	3.09±0.02	287±11	72.5±0.2	270±2

由表 2 可知,在报酬值指标上,PBOVI,AEMS2 和 MC-RTBSS 基本相同;在 EBR 和 LBI 指标上,PBOVI 和 AEMS2 基本相同.信念状态结点数主要用于衡量算法的空间复杂度,根据实验结果可知,PBOVI 算法最小,RTBSS 算法次之,其他算法都比较大.主要原因在于,PBOVI 和 RTBSS 在进行与或树裁剪之前,根据动作发生的概率大小进行了排序.PBOVI 更进一步,先采用基于点的信念状态点更新和扩展算法,然后在此基础上进行裁剪,因而具有更少的结点数.在结点重用率指标上,PBOVI 最优,因为在遍历深度 D=4 的情况下,除了根结点外,其他与或树的结点都被保存,并在下一时刻计算时重用.RTBSS 算法每次计算都是重新计算,没有重用已经计算过的信念状态.在计算时间指标上,PBOVI 和 RTBSS 两种算法耗时较少,因为它们计算的信念状态点较少.从标准差的角度来看,PBOVI 算法在主要指标上都小于其他算法,表明 PBOVI 算法具有更高的鲁棒性.

4.4 RoboCup机器人救援仿真实验

为了验证本文算法在实际应用中的可行性,将本文提出的方法应用在中南大学 RoboCup 机器人救援仿真队中,在不同的地图场景下进行大量比赛,取得了较好成绩.表 3 为采用和未采用本文提出算法的各种地图比赛数据.其中,未采用本文算法是指 2010 年中南大学 RoboCup 机器人救援仿真队(CSU_Yunlu)的框架,该框架采用子域适应度评估的合作协进化协作方法[16],在 2010 年中国机器人大赛暨 RoboCup 公开赛中取得 RoboCup 机器人救援仿真组冠军.在该框架的基础上,采用本文算法,CSU_Yunlu 队在 2011 年中国机器人大赛暨 RoboCup 公开赛中取得 RoboCup 机器人救援仿真组冠军.

Table 3 Simulation results of not using/using PBOVI respectively

表 3 未采用/采用本文算法的仿真结果

地图	得分	获救市民比例	灭火比例	死亡人数	房屋的剩余(%)
Foligno	71.53/96.75	0.100/0.471	1.000/1.000	36/3	0.997/0.999
FolignoFinal	68.21/73.49	0.333/0.400	0.876/0.936	13/8	0.517/0.550
Kobe	35.36/66.85	0.778/0.244	0.830/0.968	66/30	0.379/0.544
KobeFinal	76.44/77.24	0.313/0.373	1.000/1.000	20/21	0.993/0.990
randomLarge	67.47/84	0.089/0.467	0.850/0.870	48/18	0.755/0.667
randomSmall	66.98/97.1	0.175/0.425	0.824/0.956	35/11	0.582/0.783
VCFinal	53.64/75.67	0.213/0.365	0.863/0.900	49/28	0.630/0.730

由表 3 可以看出,采用本文提出的 PBOVI 算法后,在最后得分、获救市民的比例、灭火的比例、死亡人数和房屋剩余的百分比这些方面都有较大的提高.

5 结束语

部分可观察马尔可夫是求解动态不确定随机环境下的序贯决策的理想模型,然而由于其信念状态具有“维数灾”和“历史灾”问题,使其只能针对小规模离线问题进行求解.本文提出一种 PBOVI 算法,实验结果表明,本文算法不仅满足实时系统的要求,而且与离线算法在理论上具有一致性.采用基于点的值迭代方法对可达信念状态点进行更新和扩展,可保证 PBOVI 误差最小化.

在实际的 POMDPs 模型中,信念状态往往具有很大的稀疏性.如果在进行在线规划和执行前消除其稀疏性,将会极大地提高 PBOVI 的收敛速度.例如,在本文的实验对象 RockSample[7,8]中,由于观察变量仅依赖于已知机器人的位置,与其他岩石状态独立,即其他岩石不影响当前观察变量的取值.根据这一特点,可以将信念状态使用可分解描述来简化求解过程,并使用动态贝叶斯网络来消除其稀疏性,从而降低求解最优策略的计算量.

致谢 感谢 Poupart,Pineau 和 Ross 等人基于 Matlab 语言开发的 POMDPs 软件包,在本文的实验中,复用了其中的很多代码.

References:

[1] Kaelbling LP, Littman ML, Cassandra A. Planning and acting in partially observable stochastic domains. Artificial Intelligence, 1998,101(2):99−134. [doi: 10.1016/S0004-3702(98)00023-X]

[2] Kaplow R. Point-Based POMDP solvers: Survey and comparative analysis [MS. Thesis]. Montreal: McGill University, 2010.

[3] Bian AH, Wang CJ, Chen SF. Preprocessing for point-based algorithms of POMDP. Ruanjian Xuebao/Journal of Software, 2008,19(6):1309−1316. (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/19/1309.htm [doi: 10.3724/SP.J.1001.2008.01309]

[4] Ross S, Pineau J, Paquet S, Chaib-Draa B. Online planning algorithms for POMDPs. Journal of Artificial Intelligence Research, 2008,32(1):663−704. [doi: 10.1613/jair.2567]

[5] Paquet S. Distributed decision-making and task coordination in dynamic uncertain and real-time multi-agent environments [P.D. Thesis]. Québec: Laval University, 2006. 56−92.

[6] Wolf TB, Kochenderfer MJ. Aircraft collision avoidance using monte carlo real-time belief space search. Journal of Intelligent and Robotic Systems, 2011,64(2):277−298. [doi: 10.1007/s10846-010-9532-6]

[7] Ross S, Pineau J, Chaib-Draa B. Theoretical analysis of heuristic search methods for online POMDPs. In: Platt JC, Koller D, Singer Y, Roweis S, eds. Proc. of the NIPS. British Columbia: MIT Press, 2008. 1216−1225.

[8] Roy N, Gordon G. Finding approximate POMDPS solutions through belief compression. Journal of Artificial Intelligence Research, 2005,23(1):1−40. [doi: 10.1613/jair.1496]

[9] Sun Y, Wu B, Feng YP. Policy-Value iteration algorithm for POMDP. Journal of Computer Research and Development, 2008,45(10):1763−1768 (in Chinese with English abstract).

[10] Sondik E. The optimal control of partially observable Markov processes over the infinite horizon: Discounted costs. Operations Research, 1978,26(2):282−304. [doi: 10.1287/opre.26.2.282]

[11] Pineau J, Gordon G, Thrun S. Anytime point-based approximations for large POMDPs. Journal of Artificial Intelligence Research, 2006,27(1):335−380. [doi: 10.1613/jair.2078]

[12] Ross S, Pineau J, Chaib-Draa B, Kreitmann P. A Bayesian approach for learning and planning in partially observable Markov decision processes. Journal of Machine Learning Research, 2011,12(5):1729−1770. [doi: 10.1016/j.marpolbul.2011.08.045]

[13] Hauskrecht M. Value-Function approximations for partially observable Markov decision processes. Journal of Artificial Intelligence Research, 2000,13(1):33−94. [doi: 10.1613/jair.678]

[14] Smith T, Simmons R. Point-Based POMDP algorithms: Improved analysis and implementation. In: Bacchus F, Jaakkola T, eds. Proc. of the Int'l Conf. on Uncertainty in Artificial Intelligence. Arlington: AUAI Press, 2005. 542−547.

[15] Smith T, Simmons R. Heuristic search value iteration for POMDPs. In: Meek C, ed. Proc. of the 20th Conf. on Uncertainty in Artificial Intelligence. Arlington: AUAI Press, 2004. 520−527.

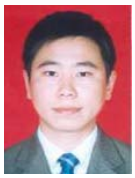
[16] Zhang XY, Peng J, Li ZQ. Cooperative co-evolutionary collaboration algorithm based on sub-domain fitness evaluation in multi-agent system. Journal of Central South University (Science and Technology), 2010,41(2):572−577 (in Chinese with English abstract). [doi: CNKI:SUN:ZNGD.0.2010-02-034]

附中文参考文献:

[3] 卞爱华,王崇骏,陈世福.基于点的 POMDP 算法的预处理方法.软件学报,2008,19(6):1309−1316. http://www.jos.org.cn/1000-9825/19/1309.htm [doi: 10.3724/SP.J.1001.2008.01309]

[9] 孙湧,仵博,冯延蓬.基于策略迭代和值迭代的 POMDP 算法.计算机研究与发展,2008,45(10):1763−1768.

[16] 张晓勇,彭军,李哲琴.多智能体系统中子域适应度评估的合作协进化协作.中南大学学报(自然科学版),2010,41(2):572−577. [doi: CNKI:SUN:ZNGD.0.2010-02-034]



仵博(1979－),男,河南桐柏人,博士生,副教授,CCF 会员,主要研究领域为序贯决策,强化学习,无线传感器网络.
E-mail: wubo@szpt.edu.cn

佘锦华(1963－),男,博士,教授,博士生导师,主要研究领域为鲁棒控制,机器人,智能控制.
E-mail: she@cc.teu.ac.jp

吴敏(1963－),男,博士,教授,博士生导师,主要研究领域为过程控制,鲁棒控制,智能系统.
E-mail: min@csu.edu.cn