

一种基于 Monte Carlo 滤波的对 POMDPRS 系统性能的改进

李 响

(中兴通讯南京研发中心 南京 210012)

摘 要 规划是人工智能研究的一个重要方向,具有极其广泛的应用背景. POMDPRS 是一种结合了 PRS 的持续规划机制、POMDP 的概率分布信念模型和极大效用原理的持续规划系统. 它具有较强的对动态不确定性环境的适应能力. 但是在太状态空间下的信念更新是其作为实时系统的瓶颈. 该文试图将 Monte Carlo 滤波引入 POMDPRS, 从而达到降低信念更新的复杂度的目的,满足系统实时性的要求.

关键词 POMDPRS;信念更新; Monte Carlo 滤波

中图法分类号 TP18

Performance Promotion for POMDPRS Based on Monte Carlo Filter

LI Xiang

(Nanjing Research Center of ZTE Corporation, Nanjing 210012)

Abstract Planning is a main research direction in artificial intelligence and has widely application background. POMDPRS is a continual planning system which combines the continual planning mechanism of PRS, the probabilistic distribution belief model and the maximum utility principle of POMDP, so that it gains stronger abilities of adapting to dynamic nondeterministic environments. However, belief updating is the bottleneck of planning performance in big state space. This paper introduces the Monte Carlo filter into POMDPRS to reduce the complexity of its belief updating, so that it can meet the requirement as a real-time system.

Keywords POMDPRS; belief update; Monte Carlo filter

1 引 言

规划是人工智能研究的一个重要领域. 而动态不确定性环境下的行动规划是其中的热点. 其动态性和不确定性是在这种环境下进行行动规划的主要难点.

PRS(Procedural Reasoning System)^[1]是目前最有影响的规划系统之一. 它是一种基于 BDI 模型和理论的持续规划系统. PRS 的规划基于所谓过程性知识,即与领域相关的用于解决问题的局部计划.

系统根据即时信念、事件和目标,不断地交替实施规划和计划的执行. 这种方式使得 PRS 既能够较好地利用领域定性知识,又能够较好地适应环境的动态变化. PRS 在实际应用中获得了很大成功,已被 NASA 用于航天飞机的故障检测和处理. 基于 PRS 还衍生出了很多系统,如 dMARS^[2]和 UM-PRS^[3]等. 其它的系统还有 JACK^[4]和 JAM^[5]等,它们都具有跨平台性质. 这些系统统称为 PRS 类系统. 但是,现有 PRS 类系统及其形式语义仅仅支持关于环境的确定性描述(通常用一个一阶逻辑文字集或者一个关系数据库来表示). 这决定了主体对当前环境

状态和自己行动后果的估计可能是不准确的,甚至是完全错误的.另外,当有多个计划可以完成同一个目标时,现有 PRS 类系统都采用选择第一个计划或者随机选择的方法,带有很大的盲目性.而现有形式化研究也没有对这些缺陷提出解决办法.

决策论规划是当前不确定性环境中规划问题研究的主流方向,其理论基础是 POMDP (Partially Observable Markov Decision Process)^[6-8]. POMDP 为求解最优行动策略提供了一种数学模型.它将环境的变迁看作状态空间上的 Markov 链,用状态空间上的信念分布表示主体对当前状态的估计,并根据主体的行动和获取的观察加以更新.在此基础上, POMDP 依据效用最大化原则,根据回报函数和状态转换函数计算最优策略.经典算法是 Value-iteration^[7]. Cassandra^[9], Kaelbling^[6], Zhang^[10] 以及 Pineau^[11] 等人分别进行了不同的优化和改进,一定程度上提高了决策效率. Pineau^[12] 等通过对动作集进行分级划分和只对信念空间上一些重要的有限点进行计算来减小决策的复杂度.但是,目前 POMDP 仍然缺乏面向复杂环境的实用高效的策略生成方法,也没有提出利用领域定性和静态知识的有效手段.

POMDPRS^[13] 是结合了 PRS 和 POMDP 各自优势的一种实时规划系统.主要思想是:使用 POMDP 的状态空间上的可能性分布描述主体对当前环境的认知(即信念),以适应环境的不确定性;利用回报评估方法进行计划选择,消除 PRS 类系统在计划选择中的盲目性;保持 PRS 类系统的规划与执行交替进行的运行机制,以适应环境的动态性,提高决策效率,并方便地利用定性领域知识.

使用状态空间上的可能性分布描述主体的信念,虽然可以克服 PRS 对环境状态的单点估计的缺陷,但是 POMDPRS 作为一个实时规划系统,却存在信念更新的效率问题.本文以下提出了在 POMDPRS 的信念更新中引入 Monte Carlo 滤波来提高信念更新效率,阐述了在这种信念更新改进下的计划选择机制,并进行了系统的性能分析.

本文第 2 节简单介绍 POMDPRS 的机制和存在的问题;第 3 节介绍 Monte Carlo 滤波在 POMDPRS 上的应用,给出形式化描述和操作语义,进行时间性能分析和局限分析;第 4 节给出具体实验应用例子.

2 POMDPRS 介绍和问题提出

POMDPRS 的基本模型框架可以表示为一个

九元组 $\langle S, A, G, P, \Omega, T, O, R, \Theta \rangle$.

(1) S 是所有状态的一个有限集合, $|S| = N$.

(2) A 是所有原子动作的有限集合.原子动作是系统可以直接执行的行动,包括直接与外部环境发生关系的外部动作和仅仅影响系统内部状态的内部动作(如更新信念分布等).

(3) G 是目标的有限集,一个目标指定在一系列状态变化后为真的某一条条件.

(4) P 是局部计划的有限集合.一条计划由触发子、计划体和评估值表组成.触发子描述计划被激活的条件,满足条件的事件将激活计划.计划体描述了计划的具体流程,是一个树结构.其中结点是原子动作或者子目标.每个原子动作是原子动作集 A 中的一个元素,用一个原子公式表示,由唯一的动作名和参数构成.子目标是目标集合中的元素.计划体中的边是执行下个动作的条件,用一个文字集表示.计划体中“开始”为空结点,开始结点连接的唯一一条边表示计划的前提条件.评估值表中存储本计划在各状态下的先验的评估值.

(5) Ω 是观察的有限集合.

(6) $T: S \times (A \cup G) \rightarrow \Pi(S)$ 是状态转移函数. $T(s, a, s')$ 表示状态 s 下执行动作 a 到达状态 s' 的概率. $T(s, g, s')$ 表示状态 s 下完成了目标 g 后到达状态 s' 的概率.

(7) $O: S \rightarrow \Pi(\Omega)$ 为观察函数.这里假设观察只与当前状态相关. $O(o, s)$ 表示状态 s 下得到观察 o 的概率.

(8) $R: S \times (A \cup G) \rightarrow R$ 为过程的回报函数.其中 $R(s, a)$ 表示在状态 s 下执行动作 a 所得到的回报, $R(s, g)$ 表示在状态 s 下完成目标 g 所得到的回报.

(9) Θ 是参数集.包括可信度阈值 ϵ 和观察符合度阈值 λ .

POMDPRS 的系统决策循环包括两个过程:信念更新过程和计划选择过程.

系统的信念分布 B 根据主体执行的行动和获得的观察来更新:

(1) 根据行动的更新以 $B' = \tau_a(B, a)$ 表示信念分布 B 下执行动作 a 到达信念分布 B' . 设 $b(s) \in B$, $b'(s') \in B'$, 则

$$b'(s') = \sum_{s \in S} T(s, a, s') b(s).$$

(2) 根据观察的更新以 $B' = \tau_o(B, o)$ 表示信念分布 B 下得到观察 o 后信念分布变为 B' . 首先,定义观察 o 与当前信念分布 B 的符合度

$$W(B, o) = \sum_{s \in S} O(o, s) b(s).$$

假定 $O(o, s)$ 是可靠的, 则当 $W(B, o) \geq \lambda$ 时, 可以认为当前对环境状态的估计与得到的观察在一定程度上是一致的. 设 $b(s) \in B, b'(s) \in B'$, 则

$$b'(s) = \alpha O(o, s) b(s),$$

其中 α 为归一常量, 使得 $\sum_{i=1}^N b(s_i) = 1$. 当 $W(B, o) < \lambda$ 时, 可以认为当前对环境状态的估计与得到的观察有较大的差距. 定义 $W(B, o) < \lambda$ 时的信念分布调整函数为 $Mod(o, B)$. 一般情况下, 它的时间复杂度可以控制在 $O(|S|)$ 范围内. $Mod(o, B)$ 的定义与系统实际特性相关.

系统的计划选择基于主体的当前信念和计划执行的回报. 系统获得的观察 o 或目标 g 称为事件, 表示为 $notice(o)$ 或者 $g = achieve(l)$. 当事件 e 为观察事件 $notice(o)$ 时, 若计划 p 的触发子为 $notice(t)$, 且存在 $q \subseteq o$ 和 $mgu \sigma$ 使得 $t\sigma = q\sigma$, 则称 p 为事件 e 的相关计划. 当事件 e 为目标事件 $achieve(l)$ 时, 若计划 p 的触发子为 $achieve(t)$, 且存在 $q \subseteq l$ 和 $mgu \sigma$ 使得 $t\sigma = q\sigma$, 则称 p 为事件 e 的相关计划. 设 B 为系统获得事件 e 后的信念分布, p 为 e 的相关计划, c_p 为 p 的前提条件. 若

$$\sum_{s \in S} b(s) [s \rightarrow c_p] \geq \epsilon \quad (1)$$

则称 p 为事件 e 的可执行相关计划; 其中

$$[s \rightarrow c_p] = \begin{cases} 1, & \exists qq \subseteq s, \exists mgu \delta q \delta = c_p \delta \\ 0, & \text{其它} \end{cases}$$

可以响应某一事件的可执行相关计划可能不止一个; 在这种情况下, 系统根据它们的期望回报值加以选择. 计划 p 在信念分布 B 下的评估函数 $V(B, p)$ 递归地定义为

$$V(B, p) = R(B, a_p) + \sum_{i=1}^m \rho_i V(\tau_a(B, a_p), p - (c_{\rho_i})),$$

$$R(B, a_p) = \sum_{s \in S} R(s, a_p) b(s) \quad (2)$$

其中 ρ_i 为执行结点 a_p 后选择其后继的第 i 个分支的概率. 式(2)可简化为

$$V(B, p) = \sum_{s \in S} V(s, p) b(s) \quad (3)$$

$V(s, p)$ 为在状态 s 下执行计划 p 的回报值, 它先验地存在于计划的评估值表中.

由分析文献[13]可知, POMDPRS 的系统循环时间复杂度最高的为信念更新过程, 为 $O(|S|^2)$. 这虽然是一个多项式级别的时间, 但是在状态空间很大的时候, 对于实时系统的决策来说仍然是不能接受的. 以 RoboCup 四腿机器人足球[14]为例. 根据机

器人自身定位的需求, 场地被划分为 1134 块, 机器人身体方向划分为 8 个方向, 机器人与足球的相对关系为 24 种. 这样对机器人来说, 其信念的状态空间就总共有 $1134 \times 8 \times 24 = 217728$ 个状态. 这对于 RoboCup 使用的 AIBO 机器人是一个很大的数字. 其 CPU 主频为 576MHz, 完成 217728^2 次计算理论上就至少需要 82.3s, 这是完全不满足实时系统的要求的.

FHMM^[15] 是 HMM 的一个变种. FHMM 将 HMM 中的状态拆分为一些层, 模型的状态由这些层中的“元状态”组成: $s_i = s_i^{(1)}, \dots, s_i^{(m)}$. 每个层形成了小的层状态集, 层与层之间是独立的, 没有依赖关系. 信念更新由此变成几个比较小的层状态集上的更新, 可以有效地降低 HMM 中的状态转换函数的复杂度, 减少信念更新的计算复杂度. 但是并不是所有的环境都是可以将状态这样分层的. 或者有的时候即便分层, 单个层的状态集仍然很大, 仍然无法满足实时系统的需求.

3 Monte Carlo 滤波对 POMDPRS 信念更新的改进

3.1 Monte Carlo 滤波

Monte Carlo 滤波^[16] 将可能性分布用这个分布上的一些实际值来进行近似的表示. 这些实际值称为样本(sample)或者粒子(particle). 所有用于表示这个分布的样本组成一个样本集. Monte Carlo 滤波通过对这个样本集的更新和维护来维持一个对相应可能性分布的近似估计. 对样本集的基本维护方法称为 SIR 方法(Sampling/Importance Re-sampling). 其基本思路是在系统得到观察以后, 计算所有样本相对于这个观察的符合程度, 得到一个概率权重. 然后根据这个权重, 从原样本集中进行重采样, 形成新的样本集. 权重越高的(越“重要”)的样本被重采样的可能性越大. 这样, 经过重采样后的新样本集合就更加符合实际分布的情况. Monte Carlo 滤波因其适于计算机编程实现, 在机器人定位^[17] 等领域得到了比较广泛的应用.

3.2 Monte Carlo 滤波对 POMDPRS 的改进

为提高系统决策效率, 将 Monte Carlo 滤波应用于 POMDPRS 形成 MC-POMDPRS. 其主要思想为: 对系统信念, 将原先使用的状态集上的可能性分布用一个状态子集上的可能性分布来代替, 即 Monte Carlo 滤波中的样本集方法. 主体的信念更新过程随之转变为通过 SIR 方法对这个样本集的

更新. 将 Monte Carlo 滤波引入 POMDP 是通过将信念及其更新使用近似的方法表示, 牺牲一部分信念表示的准确度, 来达到提高整个系统决策效率的目的.

为将 Monte Carlo 滤波引入 POMDP, 需对其信念的表示和信念更新过程重新定义: 定义样本集 Z 为状态集 S 的真子集: $Z \subseteq S$. 信念重新定义为样本集 Z 上的信念分布:

$$\langle \text{Belief} \rangle := \{b(z_j)\}, \quad z_j \in Z, \quad \sum_{j=1}^{|Z|} b(z_j) = 1.$$

根据动作进行的信念更新过程定义为

$$\begin{aligned} & \tau_o(B, a) \\ & B: \langle \text{Belief} \rangle \\ & a: \langle \text{Action} \rangle \\ & B': \langle \text{Belief} \rangle \end{aligned}$$

$$p(s) = \sum_{z \in Z} T(z, a, s) b(z), \quad b(z) \in B \quad \wedge \quad (a)$$

$$w(s) = \frac{p(s)}{\sum_{s' \in S} p(s')} \quad \wedge \quad (b)$$

$$Z' = \{z \mid \text{pick from } S \text{ according } w(s) \text{ for } |Z| \text{ times}\} \quad \wedge \quad (c)$$

$$B' = \left\{ b'(z') = \frac{p(z')}{\sum_{z \in Z'} p(z)} \mid z' \in Z' \right\} \quad (d)$$

return B'

对于根据观察 o 的修正, 首先定义观察 o 与当前信念分布 B 的符合度

$$W(B, o) = \sum_{z \in Z} O(o, z) b(z), \quad b(z) \in B \quad (4)$$

假定 $O(o, s)$ 是可靠的. 则当 $W(B, o) \geq \lambda$ 时, 可以认为当前对环境状态的估计与得到的观察在一定程度上是一致的. 定义观察修正过程 $\tau_o(B, o)$

$$\begin{aligned} & \tau_o(B, o) \\ & B: \langle \text{Belief} \rangle \\ & o: \langle \text{Observation} \rangle \\ & B': \langle \text{Belief} \rangle \end{aligned}$$

$$p(z) = O(o, z) b(z), \quad b(z) \in B \quad \wedge \quad (e)$$

$$w(z) = \frac{p(z)}{\sum_{z' \in Z} p(z')} \quad \wedge \quad (f)$$

$$Z' = \{z \mid \text{pick from } Z \text{ according } w(z) \text{ for } |Z| \text{ times}\} \quad \wedge \quad (g)$$

$$B' = \left\{ b'(z') = \frac{p(z')}{\sum_{z \in Z'} p(z)} \mid z' \in Z' \right\} \quad (h)$$

return B'

当 $W(B, o) < \lambda$ 时, 可以认为当前对环境状态的估计与得到的观察有较大的差距. 这种差距的来源可能有两个. 一是主体对环境的估计出现严重的偏差, 这可能是环境中一些主体未知的因素造成的. 二是主

体对环境的观察出现严重不准, 甚至是得到了错误的观察. 这种情况下主体应该根据可能的错误来源来确定信念分布的更新方法. 如果主体对传感器的准确度有比较高的信任度, 则倾向于认为是自己当前对环境的估计有误, 可以按照以上的方法更新. 如果主体对传感器的准确程度信心不足, 认为其经常出错, 则可以倾向于保持对环境的原有估计, 维持信念分布不变. 或者将信念分布作较小的修正, 如将分布平均化, 降低峰值等等. 定义 $W(B, o) < \lambda$ 时的信念分布调整函数为 $\text{Mod}(o, B)$. 一般情况下, 它的时间复杂度可以控制在 $O(|S|)$ 范围内.

计划的选择过程与 POMDP 基本一致, 只是在确定事件相关计划和计算计划的回报评估值时, 只使用样本集 Z 上的信念分布. 即式(1), (3)分别变为

$$\sum_{s \in Z} b(s) [s \rightarrow c_p] \geq \epsilon \quad (1')$$

$$V(B, p) = \sum_{s \in Z} V(s, p) b(s) \quad (3')$$

由以上定义可以看出, Monte Carlo 滤波的使用并不与状态空间的因子化冲突. MC-POMDP 可以单独在整个状态空间上使用, 也可以在因子化后的层状态集上单独使用. 这使得 POMDP 可以先通过因子化来降低状态空间的大小, 然后对仍然比较大的层状态集使用 Monte Carlo 滤波进一步缩减信念更新的时间复杂度.

3.3 系统性能分析

(1) MC-POMDP 时间性能分析

由上节定义可知, 完成 MC-POMDP 的一次系统决策循环需要执行以下几个过程: $\tau_o(B, a)$, 式(4), $\tau_o(B, a)$ 或 $\text{Mod}(o, B)$, 式(1'), 式(3').

对于 $\tau_o(B, a)$, 首先要对所有状态计算式(a), 这需要 $|Z| \times |S|$ 次运算. 然后根据式(b)计算每个状态的权值, 这是一个需要 $2 \times |S|$ 次计算的过程. 式(c)是重采样过程, 共需 $|Z|$ 次运算, 最后的式(d)为概率归一化过程, 共需 $2 \times |Z|$ 次计算. 因此, 执行 $\tau_o(B, a)$ 共进行了 $(2 + |Z|) \times |S| + 2 \times |Z|$ 次运算. 计算式(4), 易知其共需 $|Z|$ 次运算. 对于 $\tau_o(B, o)$ 的计算, 基本过程与 $\tau_o(B, a)$ 相同, 只是计算的范围只限于 $|Z|$, 易知其总计算次数为 $6 \times |Z|$. 对于式(1')和式(3'), 易知它们均各需 $|Z|$ 次运算.

归纳以上分析, 一次决策循环所需计算次数为 $(2 + |Z|) \times |S| + 11 \times |Z|$ 次. 因此可知其时间复杂度为 $O(|Z| \times |S|)$ 量级. 由于通常情况下 $\text{Mod}(o, B)$ 的时间复杂度为 $O(|Z|)$, 因此当使用它来代替 $\tau_o(B, o)$ 进行根据观察的更新时, 也不会对总决策流

程的时间复杂度造成影响. 一般来说, $|Z| \ll |S|$, 因此 MC-POMDPRS 相较于 POMDPRS 的 $O(|S|^2)$ 决策时间复杂度有了较大的改善.

(2) MC-POMDPRS 的局限

Monte Carlo 滤波的 SIR 重采样使得样本集的元素都集中在概率比较大的状态上, 因此可以比较好地反映实际信念分布情况. 但这毕竟是以部分状态上的信念分布近似代表整个状态空间上的分布. 这对计划的选择可能造成不良影响. 如一个计划在某状态的回报值较高, 而这个状态又不在样本集里, 则这个回报值就无法体现在计划的总回报评估值中. 这可能造成最终选择的计划不是最优的. 例如有计划 p_1 和 p_2 可以满足当前目标, 其评估值表分别为

	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9	s_{10}
$V(s, p_1)$	100	100	200	100	50	0	100	100	50	50
$V(s, p_2)$	100	100	100	100	1000	0	100	100	100	100

POMDPRS 的状态集 $|S|$ 中有 10 个状态, 当前信念分布为

s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9	s_{10}
0.1	0.15	0.25	0.1	0.05	0.05	0.1	0.13	0.05	0.02

如果采用 MC-POMDPRS, 设样本集 $|Z|$ 大小为 5. 根据 SIR 方法, 样本集可能为 s_2, s_3, s_4, s_7, s_8 . 据此我们可以计算出在 POMDPRS 和 MCPOMDPRS 下这两个计划各自的回报评估值:

	$V(p_1)$	$V(p_2)$
POMDPRS	114	140
MCPOMDPRS	134.2	100

可见, 在不同的系统下, 这两个计划的回报评估值的大小关系是不一致的. 这说明在 MCPOMDPRS 中可能会选择实际不是最优的, 而是选择了一个较优的计划来响应事件.

之所以会出现以上的现象, 一方面因为使用一个样本集来近似代替整个状态空间. 另一方面是因为计划 p_2 在某些状态下的回报评估值大大高于其他状态下的评估值. 这就是说, 在某些概率很小情况下具有很高回报的计划会由于样本集没有覆盖到这个状态而被舍弃. 不过对于一般实时性要求比较高的系统, 只要能在合理的时间内选择一个比较优的计划就是可以满足要求的了.

对于以上问题的一种解决问题的方法是通过插值来补全不在样本集中的状态的概率. 这种方法在一定程度上可以解决这个问题, 但是需要额外的计算来支持. 而且与采用的插值方法有密切的关系. 另一种方法是尽可能扩大样本集的大小, 使得样本集尽可能体现实际分布的情况. 在实际应用中, 系统设

计者可以通过调整样本集的大小来寻找速度与最优解之间的平衡点.

4 应用实例

仍以 RoboCup 机器人足球为例. 根据前述说明, 在 POMDPRS 下, 状态空间总大小为 217728 个状态. 如果对其引入因子化, 可以将机器人的自身定位和机器人与球的相对关系之间划分为两个层状态空间. 则两个层状态空间的大小分别为 9072 和 24. 可以看到, 因子化后, 机器人自身定位的层状态空间仍然比较大, 因此可以对其引入 Monte Carlo 滤波.

图 1 显示了在 Intel P4 2.4GHz CPU 的 PC 上进行仿真实验的结果. 图中时间为一次决策的时间, 每种情况测试 100 次, 取平均值. ①为 POMDPRS, ②为因子化后的 POMDPRS, ③为在机器人自身定位的层状态空间中引入 Monte Carlo 滤波的结果.

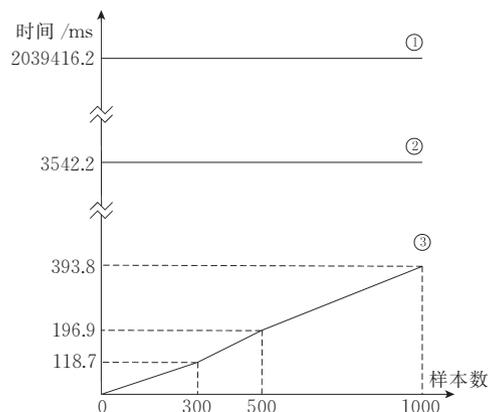


图 1 机器人信念更新进行仿真实验结果

由图 1 结果可以明显看出, 原始的 POMDPRS 一次决策需要半个小时以上, 完全不可使用. 因子化后的情况要好很多, 但是一次决策仍然要花费 3s 以上的时间. 对于实时系统来讲, 一般是不能满足需要的. 进一步引入 Monte Carlo 滤波后, 决策时间才降到了 1s 以下, 可以基本满足实时系统的需要.

5 结 论

本文为解决 POMDPRS 系统中主体信念分布更新时间消耗过高的问题, 提出了引入 Monte Carlo 滤波的方法来对此加以改善. 先通过状态空间因子化将整个状态空间进行分层, 形成多个较小的子状态空间进行处理, 降低信念更新复杂度. 再通过分层后仍然很大的子状态空间中引入 Monte Carlo 滤波将信念更新复杂度进一步降低. 从而满

足实时系统的要求.

参 考 文 献

- [1] Procedural reasoning system; User's guide. Artificial Intelligence Center, SRI International; Technical Report, 2001
- [2] d'Inverno M, Kinny D, Luck M, Wooldridge M. A formal specification of dMARS//Singh et al, eds. Proceedings of the 4th International Workshop on Agent Theories, Architectures, and Languages (ATAL'97). LNAI, Springer, 1998, 1365: 155-176
- [3] Lee Jaeho, Huber Marcus, Durfee Edmund, Kenny Patrick. UM-PRS: An implementation of the procedural reasoning system for multirobot applications//Proceedings of the AIAA/NASA Conference on Intelligent Robots in Field, Factory, Service, and Space (CIRFFSS '94). Houston, Texas, 1994: 842-849
- [4] Busetta Paolo, Ronnquist Ralph, Hodgson Andrew, Lucas Andrew. JACK intelligent agents - components for intelligent agents in Java. Agent Oriented Software Pty. Ltd, Melbourne, Australia; Technical Report AOS TR9901, 1998
- [5] Huber Marcus. JAM; A BDI-theoretic mobile agent architecture//Proceedings of the 3rd International Conference on Autonomous Agents (Agents'99). Seattle, 1999: 236-243
- [6] Kaelbling L P, Littman M L, Cassandra A R. Planning and acting in partially observable stochastic domains. Artificial Intelligence, 1998, 101(1-2): 99-134
- [7] Murphy K. A survey of POMDP solution techniques. Berkeley U. C. : Technical Report, 2000
- [8] Cassandra A R. A survey of POMDP applications//Michael Littmann ed. Working Notes; AAAI Fall Symposium on Planning with Partially Observable Markov Decision Processes, AAAI. Orlando, Florida, 1998: 17-24
- [9] Cassandra A, Littman M, Zhang N. Incremental pruning: A simple, fast, exact method for partially observable Markov decision processes//Proceedings of the 13th Annual Conference on Uncertainty in Artificial Intelligence (UAI-97). San Francisco, CA, 1997: 54-61
- [10] Zhang N L, Zhang W. Speeding up the convergence of value iteration in partially observable Markov decision processes. Journal of Artificial Intelligence Research, 2001(14): 29-51
- [11] Pineau J, Gordon G, Thrun S. Point-based value iteration: An anytime algorithm for POMDPs//Proceedings of the 18th International Joint Conference on Artificial Intelligence. Acapulco, Mexico, 2003: 1025-1032
- [12] Pineau J, Thrun S. High-level robot behavior control using POMDPs//Proceedings of the AAAI Workshop Notes, 2002
- [13] Li Xiang, Chen Xiao-Ping. A real-time planning system in dynamic nondeterministic environments. Chinese Journal of Computers, 2005, 28(7): 1163-1170(in Chinese)
(李 响,陈小平. 一种动态不确定性环境中的持续规划系统. 计算机学报, 2005, 28(7): 1163-1170)
- [14] Chen Xiao-Ping. Advancements and achievements of international RoboCup research. Technology and Application of Robotics, 2001, (1): 25-28(in Chinese)
(陈小平. 国际机器人足球(RoboCup)最新进展. 机器人技术与应用, 2001, (1): 25-28)
- [15] Ghahramani Z, Jordan M I. Factorial hidden Markov models. Machine Learning, 1997(29): 245-273
- [16] Kitagawa G. Monte Carlo filter and smoother for non-Gaussian state space models, Journal of Computational and Graphical Statistics, 1996, 5(1): 1-25
- [17] Fox Dieter, Burgardy Wolfram, Dellaert Frank, Thrun Sebastian. Monte Carlo localization: Efficient position estimation for mobile robots//Proceedings of the 16th National Conference on Artificial Intelligence (AAAI-99). Orlando, Florida, 1999: 343-349



LI Xiang, born in 1976, Ph. D., senior engineer. His main research interests include multi-agent system, continual planning and 3G communication.

Background

The work described in this paper was finished in Multi-Agent Lab of Computer Science Department, USTC.

Continual planning is one of important fields of artificial intelligence. PRS is based on continual planning and BDI model. It is a mature system that has been applied in same real world engineering applications. But PRS is lack of ability to planning under nondeterministic environments. POMDP is one of the main branches of artificial intelligence research in the world these years. It offers a mathematical model for generating best action strategy. However, the efficient bottleneck of POMDP is the main obstacle in the road to real world engineering applications.

In the author's foregoing paper, the author presented a novel continual planning system, POMDPRS, which combines POMDP and PRS to gain stronger abilities of adapting to dynamic nondeterministic environments. On the one hand, this paper offers a probabilistic extension to PRS based on POMDP. On the other hand, it introduces continual planning mechanism into POMDP, so that gains up its efficiency. There is not any other works that is similar to POMDPRS in China.

This paper is based on above paper. It introduces Monte Carlo filter into POMDPRS to gain up efficiency farther more. This makes a firm foundation to apply POMDPRS into engineering applications in more complex environments.