文章编号:1674 - 2869(2017)02 - 0186 - 07

一种动态环境下多Agent的协作方法

吴 坤[1,2],刘 玮[1,2*],李 爽[1,2],王 晶[1,2]
1. 武汉工程大学计算机科学与工程学院,湖北 武汉 430205;
2. 智能机器人湖北省重点实验室(武汉工程大学),湖北 武汉 430205

摘 要:对动态环境下多Agent系统中Agent的协作问题进行了研究,提出了一种基于连续规划的动态环境下多Agent协作方法. 首先,在MA-PDDL的语法上进行了改进,通过一个自定义的函数使其能够描述连续规划方法;然后,提出了一种基于扩展的MA-PDDL的连续规划算法,能够处理动态环境在系统中所造成的不确定性;最后,实现了扩展的MA-PDDL的解析方法,解析的结果能够用于模拟Agent执行任务的过程. 选取了医疗垃圾无人运输场景进行实验,成功模拟了整个实验场景的运行过程,验证了方法的可行性.

关键词:动态环境;MAS;多Agent协作;连续规划;MA-PDDL
中图分类号:TP242 **文献标识码**:A **doi**:10. 3969/j. issn. 1674-2869. 2017. 02. 015

A Multi-Agent Cooperation Method in Dynamic Environment

WU Kun[1,2], *LIU Wei*[1,2*], *LI Shuang*[1,2], *WANG Jing*[1,2]
1.School of Computer Science and Engineering, Wuhan Institute of Technology, Wuhan 430205, China;
2. Hubei Key Laboratory of Intelligent Robot (Wuhan Institute of Technology), Wuhan 430205, China

Abstract: A multi-agent cooperation method was proposed based on multi-agent system continual planning in dynamic environment. First, the Multi-Agent Planning Domain Definition Language (MA-PDDL) was extended for describing the continual planning method by a user-defined function. Then, a continual planning algorithm based on extended MA-PDDL was proposed to deal with the uncertainty. Finally, a parsing method of the extended MA-PDDL was implemented, which simulated how agents execute tasks. The method successfully simulates the execution process of automated cart transportation in hospital, which validates its feasibility.

Keywords: dynamic environment; multi-agent system; multi-agent cooperation; continual planning; MA-PDDL

多Agent系统(multi-agent system, MAS)是分布式人工智能领域研究的热点之一,已经成为分布式人工智能研究的一个重要分支,关于Agent的协作问题一直是MAS研究的核心问题.

传统领域关于多Agent协作问题的研究是建立在MAS的环境不变或者可预测的基础之上,例如经典合同网协议(contract net protocol, CNP)[1]、组织结构协作[2]、黑板模型[3]等. 但是MAS的环境从封闭到开放,从可预知到不确定的变化,要求多Agent的协作方法必须能够适用于动态环境下的

收稿日期:2016-12-14
基金项目:国家自然科学基金(61502355);国家测绘局测绘地理信息公益性行业科研专项(201412014);湖北省自然科学基金(2013CF125);武汉工程大学科学研究基金(K201475)
作者简介:吴 坤,硕士研究生. E-mail:kris_wu@foxmaill.com
***通讯作者**:刘 玮,博士,副教授. E-mail:liuwei@wit.edu.cn
引文格式:吴坤,刘玮,李爽,等. 一种动态环境下多Agent的协作方法[J]. 武汉工程大学学报,2017,39(2):186-192.
WU K, LIU W, LI S, et al. A multi-agent cooperation method in dynamic environment[J]. Journal of Wuhan Institute of Technology, 2017, 39(2): 186-192.

MAS，上述方法均无法解决实际问题，因此关于动态环境下的多Agent协作方法研究就显得非常的重要.

近几年研究人员已经开始尝试解决动态环境下MAS中的Agent协作问题. 其中一类研究是在传统方法的基础上进行改进，例如林琳和刘锋[4]在合同网的投标过程中引入了信任度、感知系数和活跃度等智能参数，在熟人库(acquaintance)中进行招标，通过公共消息黑板发布标书让Agent主动接受任务，引入了一个评估函数定期对Agent进行评估，更新其能力以及管理Agent对其的信任度. 该方法虽然具有一定的灵活性并且可以提高效率，但是由于动态环境的实时变化，导致系统更新过于频繁以至于增大系统压力，而更新太慢又可能导致信息没有同步而发生错误. 另一类方法是和人工智能相结合，例如唐贤伦等[5]提出了基于改进蚁群算法的多Agent协作策略解决了在未知环境下的多Agent协作问题，首先由Agent在未知环境下对任务进行搜寻，然后将收集到的信息通过黑板模型进行信息共享，最终完成所有的任务. 该方法在可预测的未知环境下是可行的，但是如果环境会动态变化且无法预知则不再适用了.

智能规划(AI planning)是用人工智能理论与技术自动或半自动地生成一组动作序列，用以实现期望的目标[6]. 近年来，有关智能规划的研究在问题描述和问题求解两方面得到了新的突破，使得智能规划已成为一个热门的人工智能研究领域. 由于智能规划的研究对象和研究方法的转变，极大地扩展了智能规划的应用领域，在太空飞船[7]、卫星[8]、智能机器人[9-11]等众多领域都有着十分广泛的应用.

使用智能规划来解决Agent协作问题时具有专门的规划语言. 最早的规划语言是STRIPS[13]，由Richard Fikes和Nils J. Nilsson在1971年提出，他们在基于逻辑的规划问题表示上引入了过程化表示和语义，随后发展成为经典规划中最主要的描述语言，几乎所有的规划系统都采用STRIPS语言或其一个变种来描述规划问题. 1998年，McDermott提出了规划领域定义语言(planning domain definition language，PDDL)[14]，并逐渐成为国际智能规划比赛IPC公认的标准.

随着MAS的广泛应用与研究，关于多Agent的规划语言也得到了快速的发展，出现了大量的多Agent规划语言，多Agent规划领域定义语言(multi-agent PDDL，MA-PDDL)就是其中的一种. MA-PDDL是由Kovacs[15]在2012年提出来的，主要针对PDDL3.1的巴科斯范式(Backus-Naur Form，BNF)作了如下扩展：在requirements中加入了multi-agent的依赖，使PDDL3.1能够支持多Agent规划；在动作的定义中引入了Agent参数，当一个动作需要另一个Agent动作的效果时，可以直接将协作动作以参数的形式引用到当前的动作中，这样就可以灵活的描述多个Agent之间的协作以及动作之间的相互影响；在问题的定义中引入了goal参数，多个Agent可以有相同的goal，要实现这些goal就必须要这些Agent一起协作完成，这样就可以描述多Agent之间的协作问题了. 但是由于PDDL3.1中假设环境是静态的而且是完全可观察的，所以MA-PDDL也存在这种假设，因此MA-PDDL虽然能够描述多Agent的协作问题，但是不能直接应用于动态环境下的MAS中.

总的来说，目前缺乏一种有效可行的方法来解决动态环境下的多Agent协作问题，针对这一现状，本文对动态环境下的MAS中Agent协作问题进行了研究，提出了一种基于连续规划的多Agent协作方法. 对MA-PDDL进行了改进，在语言中加入了连续规划元素，使其能够通过连续规划算法解决动态环境下的多Agent协作问题，避免了传统方法下任务执行过程中环境发生改变而导致任务失败的情况.

1　研究基础

1.1　智能规划

智能规划问题可表示为一个五元组$<S, S_0, G, A, \Gamma>$，其中S是所有可能的状态集合，$S_0 \subset S$表示世界的初始状态，$G \subset S$表示规划系统希望实现的目标状态，A表示进行状态之间转换所需的动作集合，状态转移关系$\Gamma \subseteq S \times A \times S$定义了每个动作执行时的先决条件和效果[12]. 采用智能规划方法来解决问题的过程就是在A集合中找到一组动作序列，使得当前系统的状态由S_0变为G. 假设完成一个任务需要两个动作，用智能规划来解决问题的示意图如图1所示.

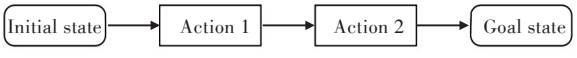

图1　智能规划示意图

Fig. 1　Schematic diagram of intelligent planning

1.2　规划语言

使用规划语言对智能规划问题进行描述时通常包括领域描述和问题描述，其中领域描述主要

是定义整个领域当中的参数,以及领域中的对象可以执行的动作,用以反映整个领域的行为. 而问题描述则包括规划的对象、系统初始状态和目标状态,是对实际规划的问题进行描述.

定义1 多Agent规划的领域描述是一个四元组$D=<O, V, F, A>$,其中O表示系统中所有的对象集合,可以是Agent,也可以是其他的普通对象;V表示系统中涉及到的所有状态变量的集合,即用来描述整个领域状态的变量,一般用逻辑谓词来描述;F表示函数的集合;A表示动作的集合.

定义2 动作a是一个三元组$<A_g, P, E>$,$a \in A$,$P \in V$,$E \in V$. 其中A_g为执行动作的Agent,P是动作执行的前提条件,E是动作执行之后所产生的效果. 一个动作$a<a_g, p, e>$表示一个Agent(a_g)在系统状态为p的情况下,执行动作a之后系统状态变为e.

定义3 问题描述为$Q=<I, G>$,I为系统的初始状态,G为需要达到的目标状态. 若经过一系列的规划动作之后,系统状态变为G,则表示该问题已经被解决了.

1.3 连续规划

在传统的规划方法中,规划者必须考虑整个系统中可能出现的所有情况,规划出Agent相应的动作,同时设定Agent从开始到结束都对整个系统有着完整的正确的知识,且动作一定会产生预期的效果. 这种方法对于环境相对稳定,Agent独立完成任务且各Agent之间没有相互影响的系统来说是可行的,但是对于动态环境中Agent的动作会对其他Agent产生影响的系统来说是行不通的.

为了解决多Agent在动态环境下的规划问题,Brenner和Nebel[16]提出了连续规划算法. 与传统的规划方法不同,连续规划并不需要事先考虑所有的可能事件,而是由Agent自己决定如何去实现他们的目标,换句话说就是Agent在完成目标任务的过程中可以自己规划自己的行动.

连续规划方法的核心思想就是当Agent对周围环境未知或者掌握的知识不足以完成规划的动作时,Agent可以延迟规划动作的执行,然后通过自身的感知能力或者通过与其他Agent通信,获取更加准确的信息,不断地更新知识直到Agent有足够的知识来完成接下来的规划任务,如图2所示. 与图1相比,图2中加入了一个重新规划步骤,需要注意的是这里并不是简单的循环执行动作Action1,而是通过系统环境重新规划出一条新的动作序列来完成后面的工作.

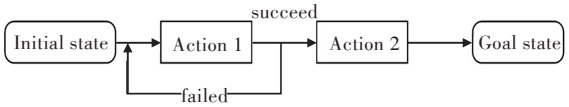

图2 连续规划示意图

Fig. 2 Schematic diagram of continual planning

例如让一个自动导引车(automated guided vehicle, AGV)将一件货物M从A点运送到B点,通过规划方法来解决这个问题可能需要执行以下几个动作:①AGV移动到A点;②装上货物M;③AGV带着货物M从A点移动到B点;④卸下货物M. 在传统规划方法中,以上几个动作均被认为是瞬间完成,但是在现实中这些动作都是需要一段时间才能完成. 假如在AGV带着M从A点移动到B点的过程中,原本规划的行动路线由于突发情况无法通过,该规划则会陷入死锁,最终导致任务无法完成. 通过连续规划方法能够避免这种情况的发生:当AGV在移动过程中发现道路不通即动作③无法完成时,则会停下来感知周围的环境,并与其他的AGV进行通信来了解最新的环境知识,然后重新规划一条移动路径继续完成动作③和动作④.

2 基于连续规划的MA-PDDL

2.1 规划过程描述

使用基于连续规划的MA-PDDL来解决MAS中的多Agent协作问题一般包括如下5个步骤:

步骤1 通过多Agent领域描述语言描述多Agent系统的当前环境;

步骤2 接收用户输入的待解决问题,并通过多Agent领域描述语言进行描述;

步骤3 根据描述后的多Agent系统的当前环境和待解决问题,规划出一条最优解决方案,该最优方案包括需要执行的若干个动作,将若干个动作分别分配给多Agent系统中能够执行相应动作的Agent;

步骤4 多Agent系统中具有执行任务的各Agent根据该最优方案依次完成分配给自己的所有动作,当确定当前动作正常完成时,各Agent继续下一个动作,直到完成所有动作,问题得到解决;

步骤5 当确定当前动作无法完成时,停止各Agent当前动作,通过Agent的感知功能对多Agent系统的当前环境进行重新感知,并通过多Agent领域描述语言进行重新描述,随后转至步骤3.

2.2　语法描述

为了在MA-PDDL中使用连续规划算法，对MA-PDDL的动作进行了如下定义.

<action> ::= (:action <action-symbol>
[:agent <agent-def>]$^{multi-agent}$
[:parameters (<typed list (variable)>)]
[:replan <predicates list>]
[action-def body])

为了实现整个多Agent系统的自适应性，保证Agent在动态环境下也能正常完成任务，我们在原来的动作定义中加入了replan这个子动作. 其工作原理如下：事先定义一个触发replan的条件，Agent在每次执行动作之前判断当前环境是否满足该条件，一旦满足条件就会暂停当前动作的执行，然后通过自身的感知能力或者与其他Agent通信来更新对系统环境的知识，最后重新规划出一条解决问题的执行方案. 如果不满足该条件则正常执行动作，若因为系统环境发生了改变而导致动作无法执行或者动作执行失败时，在系统中设置该触发条件.

使用MA-PDDL对上文中的动作③进行描述，其程序如下.

(:action SelectPath
:agent ?manager - decision_agent
:parameters (?agv - AGV ?from ?to - location)
:precondition (and (ready ?from)(ready ?to))
:effect (pathReady ?agv))
(:action move
:agent ?agv - AGV
:parameters (?from ?to - location)
:precondition (and (at ?agv ?from) (SelectPath manager))
:effect (at ?agv ?to))

其中manager是一个决策Agent，可以根据当前系统的环境规划出一条最优路径. 当agv要从A点移动到B点时，必须先选出一条最优路径，此时需要与manager协作才能完成，采用MA-PDDL可以很方便的描述这种协作关系.

如果manager规划完路径，agv在执行动作的过程中环境发生了变化，例如路径被堵塞了，那么该动作就无法完成. 使用加入连续规划的MA-PDDL对该动作进行描述，其程序如下.

(:action move
:agent ?agv - AGV
:parameters (?from ?to - location)
:replan (isDone ?false)
:precondition (and (at ?agv ?from) (SelectPath manager))
:effect (at ?agv ?to))

以上描述在原有的基础上加入了：replan，该方法的触发条件是isDone ? false. 默认情况下该条件是isDone ? true，当动作执行失败时将该条件设置为isDone ? false，满足该条件表示move动作无法完成，系统无法达到预期效果，此时Agent会暂停后续动作的执行，然后重新获取当前系统环境进行重新规划.

2.3　方法的实现

方法的具体实现过程如图3所示，主要包括以下几个步骤：

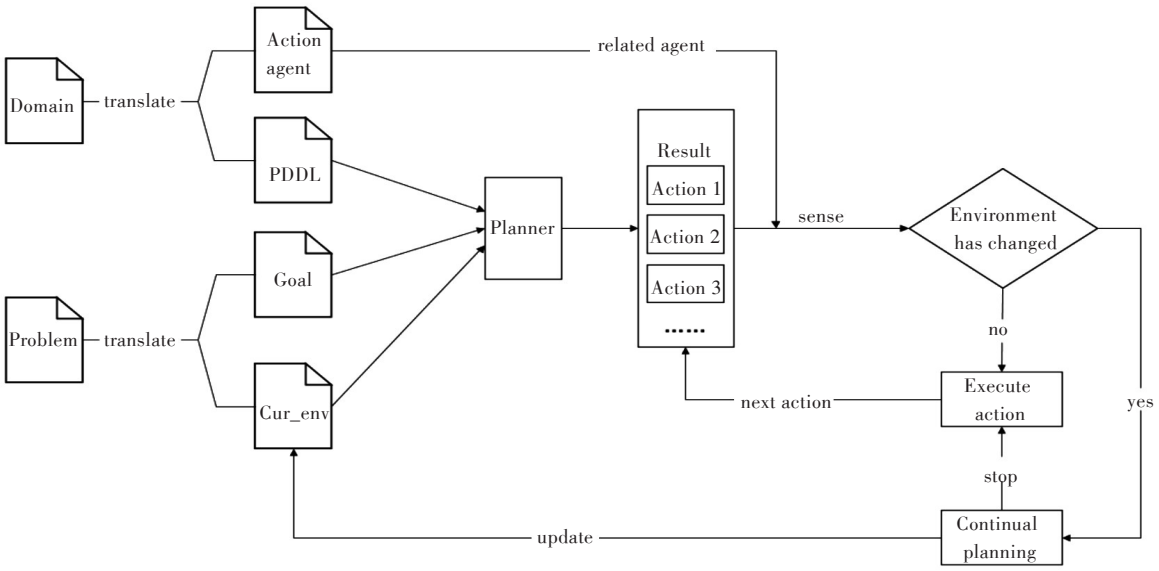

图3　方法的实现过程

Fig. 3　Realization process of method

步骤1 使用改进后的语言将MAS的领域知识和需要解决的问题描述出来作为输入；

步骤2 将领域描述Domain转换为传统的PDDL和一个描述Action与Agent关系的文件，将问题描述Problem转换为目标状态文件Goal和当前环境文件cur_env；

步骤3 根据当前环境、目标状态和MAS的领域描述规划出一条最优解决方案，即一系列需要执行的动作；

步骤4 根据动作在Action-Agent关系文件中找出一个或多个Agent来执行动作，每次执行动作之前都检查一次当前系统的环境是否发生了改变：系统环境没有变化，满足动作的前提条件，正常执行该动作，完成之后继续执行下一个动作；系统环境发生了变化，不再满足动作的前提条件，终止该动作的执行，更新描述当前系统环境的文件，跳转到步骤3.

3 场景建模

医院中的医疗垃圾往往具有传染性，对人体有很大的危害，使用人工进行运输存在着极大的安全风险，使用AGV来运输则可以避免这种风险，而且还可以免去很多人工劳动. Savant Automation公司在2015年提出了一个医院自动运输车系统(automated hospital cart transportation system)[17]，在该系统中AGV被用来运输食物、医疗器械以及医疗垃圾等，并且已经投入生产和使用. 本文的实验就是以上述系统为基础，使用了一个特定的场景进行建模，最后通过模拟运行进行实验.

在模型中使用Cart存放医疗垃圾，所有的Cart均放置于相应的Pickup上，每个Pickup上都有一个射频识别标签(Radio Frequency Indentification，RFID)用来读取Cart的重量、位置等信息，通过AGV将Cart从Pickup处运送到指定地点. 整个场景可以看作是一个多Agent系统，里面包括三种不同的Agent，分别是Cart-sensor、AGV以及decision agent. 其中Cart-sensor固定在Pickup处，通过读取Pickup上面的RFID获取Cart的重量和位置等信息，发布任务；AGV是专门用来运输Cart的小车，具有不同载重量；decision agent可以根据当前环境来选择最适合的AGV来执行，同时可以为AGV规划出一条最优的移动路线.

整个实验场景如图4所示，Pickup-Location表示Pickup的位置，也是Cart的存放地点；Charge-Location为AGV充电的地方；Destination为Cart被运输的目的地；Road为AGV运行的轨迹，表示AGV可达的路径；Wall为AGV不可达的地点，可视为障碍物.

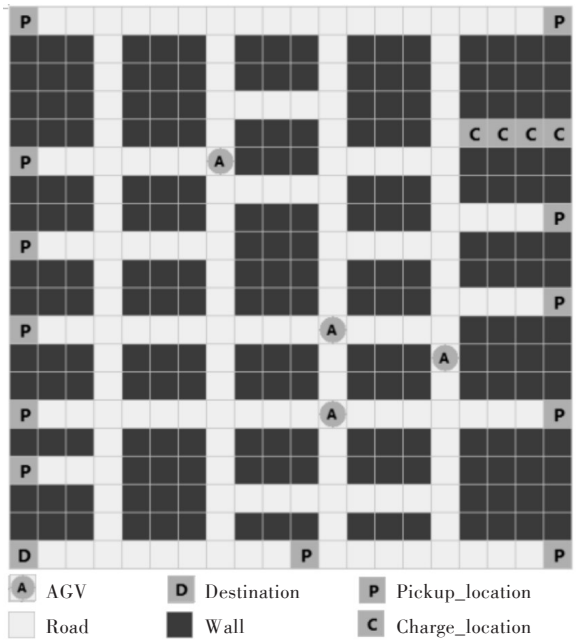

图4 实验仿真场景

Fig. 4 Experiment scenario

使用定义1对领域进行描述如下.

S = <O, V, F, A>,其中

O={decision_agent,AGV,Cart_sensor,cart, pickup,location},

V={samePosition,on,in,ready, available, is Done, pathReady},

F={distance,loadage},

A={DetectCart,ReadTag,DeliverCall,SelectAGV, SelectPath,MovetoPickup,TakeinCart,MovetoDestination,TakeoffCart}

*O*是系统中的对象集合，有decision_agent, AGV, Cart_sensor, cart, pickup和location等6个对象；*V*是系统中的状态变量集合，samePosition用来描述2个对象在同一个位置，on用来描述cart在pickup中，in用来描述cart在AGV小车上，ready用来描述pickup、AGV以及目的地的位置是否确定，available用来描述是否发布任务，isDone用来描述动作是否正常完成，pathReady用来描述行动路径是否规划完成；需要用到的函数包括distance和loadage，前者是求两点之间的距离，后者是求AGV的载重量，两个函数都是用来选择最佳执行AGV的；*A*是系统中所有Agent能够完成的动作的集合，包括DetectCart、ReadTag等9个动作.

使用定义2对领域中的动作进行描述如下.

DetectCart=<Cart_sensor, <on(cart, pickup)>, <ready(pickup)>>

ReadTag=<Cart_sensor,<ready(pickup)>,<ready(destination)>>

DeliverCall=<Cart_sensor, <ready(pickup) ready(destination)>,<available(agv_call)>>

SelectAGV=<decision_agent, <available(agv_call)>,<ready(agv)>>

SelectPath=<decision_agent, <ready(from) ready(to)>,<pathReady(agv)>>

MovetoPickup=<AGV, <ready(pickup) ready(agv)>,<samePosition(pickup, agv)>>

TakeinCart=<AGV, <samePosition(pickup, agv)>,<in(cart, agv)>>

MovetoDestination=<AGV, <ready(agv) ready(destination) in(cart, agv)>, <samePosition(destination, agv)>>

TakeoffCart=<AGV, <samePosition(destination, agv)>,< in(cart, agv) samePosition(cart, destination)>>

每个动作包含3个参数:第1个参数表示执行该动作的Agent,第2个参数是执行该动作的前提条件,是一个状态变量的集合,第3个参数是执行完该动作产生的影响,也是一个状态变量的集合.例如 *DetectCart* = <*Cart_sensor*, <*on*(*cart*, *pickup*)>, <*ready*(*pickup*)>>,表示当cart在pickup中时,Cart_sensor会执行动作DetectCart,成功执行完该动作之后,pickup的位置信息就确定了.在整个系统中,Cart_sensor可以执行3个动作,decision_agent可以执行2个动作,AGV可以执行4个动作.

整个系统的初始状态是cart被放置到pickup中,需要完成的任务是将cart运送到专门的垃圾处理点,使用定义3对问题进行描述如下所示.

P = <<on(cart, pickup)>,<samePosition(cart, destination)>>

实验场景中一共有12个pickup分别位于不同的地方,有两个载重量为20 kg的AGV和两个载重量为50 kg的AGV,50 kg载重量的AGV能耗更大,同时能耗也跟运输距离成正比,需要选择一种能耗最小的方案来完成任务.

使用java语言对MA-PDDL和改进的MA-PDDL进行解析,分别在静态环境以及动态环境这两种情况下进行实验,通过语句描述来代替具体的Agent执行动作,最后通过控制台打印出Agent的动作序列,其运行结果摘录如图5和图6所示.

Cart_sensor doing DetectCart	Cart_sensor doing DetectCart
Cart_sensor doing ReadTag	Cart_sensor doing ReadTag
Cart_sensor doing DeliverCall	Cart_sensor doing DeliverCall
decision_agent doing SelectAGV	decision_agent doing SelectAGV
decision_agent doing SelectPath	decision_agent doing SelectPath
AGV doing MovetoPickup	AGV doing MovetoPickup
AGV doing TakeinCart	AGV doing TakeinCart
decision_agent doing SelectPath	decision_agent doing SelectPath
AGV doing MovetoDestination	AGV doing MovetoDestination
AGV doing TakeoffCart	AGV doing TakeoffCart

图5　静态环境下运行的结果

Fig. 5　Results in static environment

Cart_sensor doing DetectCart	Cart_sensor doing DetectCart
Cart_sensor doing ReadTag	Cart_sensor doing ReadTag
Cart_sensor doing DeliverCall	Cart_sensor doing DeliverCall
decision_agent doing SelectAGV	decision_agent doing SelectAGV
decision_agent doing SelectPath	decision_agent doing SelectPath
AGV doing MovetoPickup	AGV doing MovetoPickup
	decision_agent doing SelectPath
	AGV doing MovetoPickup
	AGV doing TakeinCart
	decision_agent doing SelectPath
	AGV doing MovetoDestination
	AGV doing TakeoffCart

图6　动态环境下运行的结果

Fig. 6　Results in dynamic environment

图5和图6中左边是MA-PDDL的运行结果,右边是改进的MA-PDDL的运行结果.可以看到,在静态环境下,两种方法都可以解决问题.但是如果在动态环境下,MA-PDDL会停止运行,最终导致任务无法完成,而基于连续规划的MA-PDDL会通过重新规划找到一条新的解决方案,直到问题得到解决.

原系统中,当AGV在执行动作的过程中环境发生改变时,系统采取的策略是暂停动作的执行,AGV在原地等待直到系统环境恢复预期情况.如果系统环境的改变是长时间的或者是永久的,那么AGV完成任务的效率就会变低,甚至需要人工的参与才能完成任务.实验结果显示,本文提出的方法是可以解决问题的,而且可以避免上述情况的发生,这也证明了在动态环境下,本文方法与传统方法相比更有优势.

4　结　语

MA-PDDL在处理动态环境下的多Agent协作问题时,可能会出现环境突然发生改变而导致任务无法完成的情况,针对这一现状,提出了一种基于MA-PDDL的连续规划方法.对MA-PDDL进行了改进,在语言中加入了连续规划元素,使其能够通过连续规划算法解决动态环境下的多Agent协作问题,避免了上述情况的发生.最后提出了一种改进语言的解析方法,实验成功模拟了医疗垃圾无人运输场景的运行过程,实现了动态环境下的

多Agent协作.

改进之后方法虽然能够解决问题,但是在时间上却比MA-PDDL方法消耗更多,因为改进方法在每个动作执行前都会同步一次系统环境,这个是需要花费时间的,在之后的研究中将会考虑将算法进行优化,争取在时间上的消耗达到最小.

参考文献:

[1] SMITH R G. The contract net protocol: high-level communication and control in a distributed problem solver[J]. IEEE Transactions on Computers, 1980, 29(12):1104-1113.

[2] JENNINGS N R. Commitments and conventions: the foundation of coordination in multi-agent systems[J]. Knowledge Engineering Review,1993,8(3):223-250.

[3] NII P. The blackboard model of problem solving[J]. AI Magazine, 1986, 7(2):38-53.

[4] 林琳, 刘锋. 基于改进合同网协议的多Agent协作模型[J]. 计算机技术与发展, 2010, 20(3):71-75.
LIN L, LIU F. A multi-agent cooperation model based on improved contract net protocol [J]. Computer Technology and Development, 2010, 20(3):71-75.

[5] 唐贤伦, 李亚楠, 樊峥. 未知环境中多Agent自主协作规划策略[J]. 系统工程与电子技术, 2013, 35(2):345-349.
TANG X L, LI Y N, FAN Z. Multi-agent autonomous cooperation planning strategy in unknown environment [J]. System Engineering and Electronics, 2013, 35(2):345-349.

[6] 宋泾舸, 查建中, 陆一平. 智能规划研究综述——一个面向应用的视角[J]. 智能系统学报, 2007, 2(2):18-25.
SONG J G, CHA J Z, LU Y P. Survey on AI planning research——an application-oriented perspective [J]. CAAI Transactions on Intelligent Systems, 2007, 2(2):18-25.

[7] CHIEN S, KNIGHT R, STECHERT A, et al. Integrated planning and execution for autonomous spacecraft[J]. IEEE Aerospace and Electronic Systems Magazine, 2009, 24(1):23-30.

[8] IZZO D, PETTAZZI L. Autonomous and distributed motion planning for satellite swarm [J]. Journal of Guidance Control and Dynamics, 2015, 30(2):449-459.

[9] 张彦铎, 李哲靖, 鲁统伟. 机器人世界杯足球锦标赛中多机器人对目标协同定位算法的改进[J]. 武汉工程大学学报, 2013, 35(2):69-73.
ZHANG Y D, LI Z J, LU T W. Improvements of collaborative localization algorithm of multi-robot on target in ROBOCUP[J]. Journal of Wuhan Institute of Technology, 2013, 35(2):69-73.

[10] 鲁统伟, 林芹, 李熹,等. 记忆运动方向的机器人避障算法[J]. 武汉工程大学学报, 2013, 35(4):66-71.
LU T W, LIN Q, LI X, et al. Obstacle avoidance algorithm of robot based on recording move direction [J]. Journal of Wuhan Institute of Technology, 2013, 35(4):66-71.

[11] 张彦铎, 葛林凤. 一种新的基于MMAS的机器人路径规划方法[J]. 武汉工程大学学报, 2009, 31(5):76-79.
ZHANG Y D, GE L F. A novel method for robot's path planning based on and max-min ant system [J]. Journal of Wuhan Institute of Technology, 2009, 31(5):76-79.

[12] 林川. 基于PDDL的Web服务自动组合的描述[J]. 计算机应用与软件, 2008, 25(1):138-139.
LIN C. Describition of automatic web services composition based on PDDL [J]. Computer Applications and Software, 2008, 25(1):138-139.

[13] FIKES R E, NILSSON N J. Strips: a new approach to the application of theorem proving to problem solving [J]. Artificial intelligence, 1971, 2:608-620.

[14] MALIK G, HOWE A, KNOBLOCK C, et al. PDDL-the planning domain definition language, Technical Report CVC TR-98-003/DCS TR-1165[R]. Connecticut:Yale Center for Computational Vision and Control, 1998.

[15] KOVACS D L. A multi-agent extension of PDDL3.1 [C]// The Association for the Advancement of Artificial Intelligence (AAAI). Workshop on the International Planning Competition, ICAPS-2012. Atibaia:AAAI Press, 2012:19-27.

[16] BRENNER M, NEBEL B. Continual planning and acting in dynamic multiagent environments [J]. Autonomous Agents and Multi-agent Systems, 2009, 19(3):297-331.

[17] SAVANT AUTOMATION. Automated hospital cart transportation system [EB/OL]. (2015-06-15) [2017-04-10].http://www.agvsystems.com/hospital-carts.

本文编辑:陈小平