一种基于时标状态的启发式航天器任务规划算法

李朝玉[1,2]，徐瑞[1,2]
（1. 北京理工大学 深空探测技术研究所，北京 100081；
2. 飞行器动力学与控制教育部重点实验室，北京 100081）

摘　要：深空探测领域对实时性要求较高，在较短时间内找到规划解是深空探测自主任务规划中的一个要求，运用启发式规划算法是达到该要求的方法之一。而深空探测自主任务规划的另外一个特点是需要处理持续动作和数值信息。针对深空探测任务特点，采用规划领域定义语言 PDDL，建立深空探测领域中知识模型，描述操作中遇到的时间与资源约束；随后应用以条件数为代价的启发式搜索方法对深空探测规划问题进行求解，并将其与 TFD 规划器中以动作时间为代价的上下文增强累加启发式搜索方法得到的结果进行对比，得出以条件数为代价的启发式搜索方法在搜索速度方面效果更佳，满足深空探测自主规划任务实时性要求。

关键词：深空探测；启发式；持续动作；数值
中图分类号：V44　　**文献标识码**：A　　**文章编号**：2095-7777(2015)01-0020-07
DOI：10.15982/j.issn.2095-7777.2015.01.003

0 引　言

深空探测领域对实时性要求较高，在较短时间内找到规划解是深空探测自主任务规划中的一个要求。近些年，规划领域蓬勃发展，规划效率不断提高，有很多具有发展前景的规划方法出现，而启发式搜索方法就是其中之一。这种方法的中心思想就是在搜索过程中运用启发式信息引导搜索向希望的节点进行，直至找到目标节点为止。在深空探测自主任务规划中运用启发式信息进行搜索会减少搜索时间，大大提高产生规划解的效率[1]。

深空探测中约束复杂，任务执行必须精准，运用启发式过程中不宜过多地进行放松；与此同时，探测器运行速度快，有时遇到观测的现象时需要能够较快产生规划解，因此探测器自主任务规划系统要求具有较高的搜索速度，寻找一种快速高效的启发式显得尤为重要[2]。起初设计启发式时，对时间和数值方面没有考虑，深空探测领域中时间信息和数值量都是无法避免的，如电量、存储量等[3]。因此，当引进持续动作和数值量后，必须对启发式进行改进，才能用于深空探测中的时间和数值规划[4-5]。

本文针对深空探测任务要求规划解质量高、搜索速度快的特点，在 TFD 规划器中采用的以动作时间为代价的上下文增强累加启发式[6]基础上，对代价计算方法进行改进，设计了一种基于时标状态的启发式规划算法，该算法以条件数为代价，以达到对时间和数值信息的处理和对搜索过程的快速引导。应用到深空探测领域，一定程度上能够提高实时处理问题的能力。

1 深空探测器自主任务规划模型

用传统的离散动作表示深空任务中的动作已经无法解决实际问题，因此需要采用持续动作，即用一定的值表示动作执行时间。此外，还需考虑动作与动作间的并行性，以此可减少执行任务的完成时间，提高效率。

对于深空探测任务来说，探测器进入接近段后，速度非常快，姿态轨道等的调整很频繁，因此对实时性要求非常高[7]。因此本文以此阶段为例，采用规划领域定义语言 PDDL[8]对其进行描述建模，并对其进行求解。不仅能表示任务的持续时间和约束，还利于后面规划算法对时间进行处理。

1.1 领域文件的建立

深空探测器天体探测接近段中，涉及到的动作有转向、校准相机、打开相机、拍照、关闭相机、下传数据等活动。下面将对接近段任务进行领域建模。

收稿日期：2014-12-20　修回日期：2015-02-10
基金项目：国家重点基础研究发展计划(973 计划)(2012CB720000)；民用航天项目

建模过程中,包括类型、谓词、函数以及持续动作的说明[9]。其中,谓词指深空探测器中各个对象的属性,具有一定的取值范围;函数则表示动作之间的转换;持续动作的说明中则包含了动作涉及的参数,持续时间,需要满足的开始条件、持续条件、结束条件以及动作产生的效果[10]。在描述过程中,会有数值量的出现,这些数值量则表示各个活动所涉及到的数值变量和资源情况,如存储器的存储空间、燃料使用情况、电量储存量等[11]。

表 1 为该任务建模中所涉及的活动及其含义描述。领域文件中对相机系统、姿态系统、导航系统、推进系统和通信系统进行建模,共 15 个活动,且文件中对每个活动的持续时间、参数、开始条件和结束效果都作了定义。

表 1　深空探测器小天体接近段任务模型的活动及其含义描述

Table 1　Description of activities in mission model of the deep space explorer for the closing phase to a small body

活动名称	含义
Cam_Ready	准备好相机
Cam_Open	打开相机
Cam_Takephoto	相机拍照
Cam_Close	关闭相机
ATT_Maneuver	探测器转向
Nav_prepare	准备导航
Nav_process	导航过程
Prop_warmingup	推进器预热
Prop_startup	推进器启动
Prop_thrust	推进器进行推进
Prop_closing	推进器关闭
Com_system_prepare	数据传递系统准备
Com_uplink	上传数据
Com_downlink	下传数据
Com_close	数据传递系统关闭

1.2　问题文件的建立

问题文件用于描述探测任务的对象、初始状态以及任务目标,也是地面工作者操作探测器的接口。本文所建立的问题模型包括相机、导航系统、推进系统、探测器指向以及探测目标。拍照时,相机需要对准探测目标进行拍照。

图 1 表示出该探测任务的初始状态。在初始状态中,相机 cam1 处于闲置关闭状态;剩余电量为 800 W,剩余数据储存量为 500 MB,推进剂剩余 500 g;推进机 prop1 预热时间为 25 s,推进时间为 10 s;数据传输准备时间需要 20 s,数据上传时间需要 35 s,下传时间需要 50 s;探测器开始指向为初始方向 Initial。此次设定的科学观测任务目标为:利用相机 cam1 对小天体 asteroid1 和恒星 star1 进行拍照。具体描述如图 2 所示。

```
(:init
    (idle cam1)
      (pointing Initial)
      (camclosed cam1)
      (idle com1)
      ;(tracked com1 gs1)
      (= (power-left) 800)
      (= (memory-left) 500)
      (= (propellant-left) 500)
      (= (warmingtime prop1) 25)
      (= (thrusttime prop1) 10)
      (= (com_prepare_time com1) 20)
      (= (uplink_time com1) 35)
      (= (downlink_time com1) 50)
)
```

图 1　深空探测器小天体接近段任务的初始状态

Fig. 1　Initial states of the deep space explorer in the closing phase to a small body

```
(:goal (and
      (camhavepic cam1 asteroid1)
      (camhavepic cam1 star1)  )
)
```

图 2　深空探测器小天体接近段的任务目标

Fig. 2　Description of goals of the deep space explorer

2　基于时标状态的启发式规划算法

早期的规划方法并不能处理具有连续动作的规划问题,因此启发式函数也只是对瞬间的动作节点进行计算。然而,深空探测任务中,动作都具有持续时间,传统的启发式处理方法已经不再适用。随着规划技术的发展,产生了可对连续动作和数值量进行处理计算的启发式,启发式搜索方法也得到了相应的改进,以适应现实问题的需要。本文设计一种以条件数为代价的上下文增强累加启发式,达到对连读动作和数值量处理的目的。

2.1　持续动作和数值处理方法

对启发式的改进涉及到两个方面:将持续动作转换为操作,使之可以用来进行启发式的计算;对规划任务中的数值部分进行处理。在这里采用的方法是引入瞬时动作(instant actions)处理持续动作部分,估计改变比较变量值的代价处理数值部分[12]。

1)瞬时动作

为了处理持续动作,需要引入瞬时动作的概念来估计时态任务。瞬时动作又可分为四部分:压缩

动作、开始动作、等待动作、由逻辑公理导出的瞬时动作。

(1)压缩动作

主要是通过对原时态规划任务中的条件与效果进行简化处理。压缩条件三元组$<c_{\vdash}, c_{\leftrightarrow}, c_{\dashv}>$，一个瞬时动作的压缩逻辑效果表示为

$$o: v=w', z \rightarrow v=w$$

其中：o 为动作；v 和 v' 为变量；w 和 w' 为变量可取值；z 为部分状态。

压缩的数值效果$<z, v \circ v'>$表示为：$o: z \rightarrow v \circ v'$（$\circ$为符号$+=, -=, *=, /=, :=$）。

每个瞬时动作都分配给一个代价 $cost(o)$，这个代价可能是真实的值或者数值变量。在不同的代价计算公式中，$cost(o)$计算的方法也是不同的。

压缩动作图例如图 3 所示。

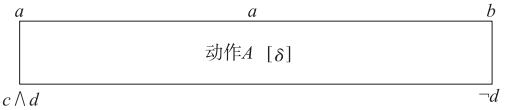

图 3 压缩动作图例

Fig. 3 Compressive actions

压缩动作转换后为：

条件：$cond=\{a, b\}$； 效果：$eff=\{c, d, \neg d\}$

(2)开始动作

将持续动作进行压缩，导致忽略了持续动作执行过程中到达的中间状态的启发式值，因此引入开始动作。它包含持续动作的开始效果，以及压缩动作所包含的所有条件，即开始、持续和结束条件。与压缩动作的唯一不同是没有结束效果。开始动作代价计算：原始持续操作的持续时间。

(3)等待动作

当启发式计算需要用到结束效果时，就用压缩动作，因为开始动作没有结束效果。而当持续动作已经在时标状态 S 下运行了，则认为动作的条件已经得到了满足，此时引入等待动作。每个调度效果$<\Delta t, c_{\leftrightarrow}, c_{\dashv}, e>$中增加一个等待动作 o，其中效果为 $effect \rightarrow e$，代价根据采用的方法计算。图 4 为等待动作图例。

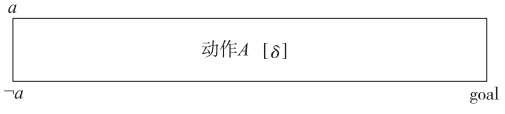

图 4 等待动作图例

Fig. 4 Waiting actions

等待动作转换为：

条件：$cond=\varnothing$； 效果：$eff=\{goal\}$

(4)由逻辑公理导出的瞬时动作

最后一组瞬时动作是从逻辑公理导出的，例如公理 $z \rightarrow v=w$ 可得到瞬时动作 o，其代价 $cost(o)=0$，效果为$\{v=w', z \rightarrow v=w \mid w' \in D_v \backslash \{w\}\}$。

2)数值信息处理

数值方面是指有些变量需要用数值表示，例如深空探测器的电量是有一定值的，如果按以前没有数值量时的情况，只能用 0 和 1 表示没电还是有电，这样就无法表示实际用电量。

在变量处理过程中，数值量没有出现在条件或者目标当中。因此通过估计改变比较变量需要的代价是一种处理数值量的有效方法。

考虑未满足条件 $v=w$，其中 $v \in V_c$ 是比较变量，$w \in \{true, false\}$，$v=v' \triangleright \triangleleft 0$ 是相关的比较公理；数值变量 v' 代表数值流集合 $F(v)$ 中的算术表达式，集合 $F(v)$ 是输入度为 0 的变量的集合，也称为 v' 的祖先。来自于瞬时动作的规则集合 $influencing(v)=\{o: z \rightarrow v_1 \circ v_2 \mid v_1 \in F(v)\}$ 作用在变量上，接下来，就从 $influencing(v)$ 中选择对原子有积极效应的规则。

为了表明比较变量的变化情况，需要定义新的符号，如式(1)所示

$$\triangleright \triangleleft^{w}=\begin{cases} < \text{if} \triangleright \triangleleft \in \{<, \leqslant, =\} \text{and } w=false \text{ or} \\ \quad \text{if} \triangleright \triangleleft \in \{>, \geqslant\} \text{and } w=true \\ > \text{if} \triangleright \triangleleft \in \{>, \geqslant\} \text{and } w=false \text{ or} \\ \quad \text{if} \triangleright \triangleleft \in \{<, \leqslant, =\} \text{and } w=true \end{cases} \tag{1}$$

在扩展状态 s 中，如果式(1)成立，$influencing(v)$中的规则 $o: z \rightarrow v_1 \circ v_2$ 是可以改变变量 v 的值 w（$v=v' \triangleright \triangleleft 0$ 是与之相关的比较公理）。

$$\begin{aligned} & s[v_1 \circ v_2](v') \triangleright \triangleleft^{w} s(v') \\ & \qquad \text{if} \triangleright \triangleleft \in \{<, \leqslant, \geqslant, >\} \\ & \mid s[v_1 \circ v_2](v') \mid \triangleright \triangleleft^{w} \mid s(v') \mid \\ & \qquad else \end{aligned} \tag{2}$$

其中，$s[v_1 \circ v_2]$是与 s 相似的状态，但是其 $v_1 \circ v_2$ 使得变量值更新和一个子序列的公理更新。

2.2 以条件数为代价的上下文增强累加启发式

由于 TFD 规划器中以动作时间为代价的上下文增强累加启发式计算较复杂，因此对代价计算方法进行简单化，尽量能够减小搜索时间。由代价值得到启发式值 $h(n)$，$h(n)$值的好坏直接影响到下个

节点的选取,因此找到一种计算快速、易于理解的启发式函数算法对整个搜索过程有着极为重要的作用[2]。

在对启发式函数进行介绍前,需要对一些符号做出说明[12]。s 表示状态,$v=w$ 为原子(以下用 x 表示),$s[v=w]$表示一种状态,在这种状态下,除了变量 v 的值以外,其他与 s 都相同;相似地,$s[s']$也为一种状态,此状态下,某些变量值与状态 s' 下相同,其他变量的值则与状态 s 下相同。var(x)指原子 x 中的变量。

一个操作是效果的集合,形式为 $v=w'$,$z\rightarrow v=w$,其中 z 是一个部分状态,v 是一个变量,w' 和 w 属于变量值域 D_v,对应于 v 不同状态下的值。这一效果意味着在与部分状态 z 相符合的当前状态 s 下,变量 v 的值为 w',作用操作 o 后,生成继承状态 s',此状态下变量 v 的值变为 w。为了将此意义表示清楚,通常写作以下形式:$o:v=w'$,$z\rightarrow v=w$。

启发式 $h(S)$定义如下

$$h(S)\xlongequal{\text{def}}\sum_{x\in s_\star}h(x\mid x_s)\tag{3}$$

其中:x_s 是状态 s 下的变量 var(x)对应的原子;$h(x\mid x_s)$是指变量 var(x)由状态 s 下的值改变到状态 $s_\star$ 下的值所用代价。

对于上面介绍的规则 $o:v=w'$,$z\rightarrow v=w$,条件 $v=w'$需要先实现,然后条件 z 在得到的状态 s 下进行评估。拓展式(3)得到

$$h(x\mid x')\xlongequal{\text{def}}\begin{cases}0 & x=x'\\ \min\limits_{o;x'',z\rightarrow x}(1+h(x''\mid x')+\sum\limits_{x_i}h(x_i\mid x_i'')) & x\neq x'\end{cases}\tag{4}$$

上述式子中,第一个被加数 1 是应用最小操作的代价,即假设代价模型的单位为 1;第二项为将原子变为 x''的代价;第三项为使规则中 z 中所有条件为真的代价。第三项中,原子 x_i'' 是变量 var(x_i)在实现 x'变为 x''的状态下的值。

以条件数为代价的启发式计算仍以瞬时动作为基础。对于压缩动作和开始动作,代价 $cost=|conditions|$,即代价为压缩动作后的前提条件数,例如对于图 3 来说,$cost=2$。对于等待动作,$cost=0$,因为动作已经在执行,说明该动作的条件数已经得到满足;从逻辑公理导出的瞬时动作,例如公理 $z\rightarrow v=w$ 可得到瞬时动作 o,其代价 $cost(o)=0$,效果为 $\{v=w',z\rightarrow v=w\mid w'\in D_v\backslash\{w\}\}$。

以条件数为代价,式(4)进行拓展,如式(5)所示。

$$h(x\mid x')\xlongequal{\text{def}}\begin{cases}0 & x=x'\\ \min\limits_{o;x'',z\rightarrow x}(\sum\limits_{i=0,\cdots,m-1}|conditions_i|+h(x''\mid x')+\sum\limits_{x_i\in z}h(x_i\mid x_i'')) & x\neq x',\text{var}(x)\notin V_c\\ \min\limits_{o;z\rightarrow v_\star v'\in prom(x',s')}(\sum\limits_{i=0,\cdots,m-1}|conditions_i|+\sum\limits_{x_i\in z}h(x_i\mid x_i')) & x\neq x',\text{var}(x)\in V_c\end{cases}\tag{5}$$

2.3　规划算法

在整个搜索过程中,采用的主要搜索算法为 A^* 算法。A^* 算法是一种静态网络中求解最短路径的最有效方法。启发式则是用来引导搜索算法的,并在父节点的确定过程中,对各个动作间进行时间约束的判断,即判断动作间是否满足先后顺序条件,不满足时间约束的节点即被移除掉,选择下一个节点,使其搜索效率更高。

在本文的搜索算法中,使用的启发式为以条件数为代价的上下文增强累加启发式。搜索过程如图 5 所示。

此算法将创建两个表,名称分别为 OpenList 和 CloseList。OpenList 存放已经生成而未被考察的节点,CloseList 则存放记录已经访问过的节点。首先将初始节点放入 OpenList 中,因为在第一步中,没有别的节点可供选择,所以将初始节点直接从 OpenList 移除,放入到 CloseList 中,并将初始节点的所有子节点放入到 OpenList 中。然后根据启发式信息对初始节点的子节点进行排序,选择优先级最高且满足时间约束的节点放入到 CloseList。再次将此节点的子节点(OpenList 中没有并且也不是 CloseList 中的节点)放入到 OpenList 中,并对 OpenList 中节点重新计算启发式,然后再选择优先级最高的节点重复处理[3]。

在搜索过程中,有两个特殊量值得注意。首先,引入一个时间小量 ,用来满足“无移动目标”规则,

当一个动作加入时标状态后就会引入此小量。其次，引入一个特殊动作“let_time_pass”，此动作可应用在任何一个状态下，此动作仅让时间流逝，而不会产生任何效果。

```
计算起点估计值；
将起点放入 OpenList 中；
While(OpenList！ ＝NULL)
{  从 OpenList 中取优先级最高的节点 n；
If(n 节点＝＝目标节点){
break；
}
for(当前节点 n 的每个子节点 x)
{  计算 x 的启发式估计值；
    If(x is in OpenList)
      {  if(x 的估计值小于原来的估计值)
        {  把 n 设置为 x 的父节点；
          将现有的估计值赋值为 x 当前的估计值；//取最小路径的估计值
        }
      }
    If(x is in CloseList)
      {  continue；}
    If(x is not in both &. 时间约束得到满足)//此处对时间约束进行判断
      {  将 n 设置为 x 的父节点；
        求 x 的估计值；
        将 x 加入到 OpenList 中；//此时并未排序
      }
}
将 N 节点插入到 CloseList 中；
按照估计值将 OpenList 中的节点进行排序；
}
保存路径，即从终点开始，每个节点沿着父节点移动至起点。
```

图 5 基于时间约束的启发式规划算法

Fig. 5 Heuristic algorithm based on time constraints

3 仿真分析

在规划系统中，搜索算法的选取对结果的产生有着重要的作用，当然在利用启发式的搜索中，启发式信息的好坏对得到的规划质量和搜索速度也至关重要。其中一个发展方向则是针对某个领域设计出具有针对性的启发式，在此领域能够取得理想的效果即可。因为本文是针对深空探测器设计的启发式，因此就需要分析这种启发式在深空探测问题中所取得的结果和效果。

采用第 1 节所建立的深空探测器小天体接近段的任务模型，利用基于时标状态的规划器进行规划的搜索求解，获得了任务的规划结果。如图 6 所示。

从得到的规划方案可看出，时间以 0.000 1 s 的单位向前推进，以满足“无移动目标规则”(no moving target rule)。对于第 1 节建立的探测器任务，规划器只需要 40.000 3 s 即可实现目标。实现目标所需要的步数根据初始状态和目标状态决定的。因为采用了时标状态及时间推进小量∈，准备相机和调整探测器指向可同时进行，所以节省整个规划任务解的时间，提高效率。

Time：	Action：	Action Duration：
0.00000000：	(cam_ready cam1)	[10.00000000]
0.00000000：	(att_maneuver att1 initial asteroid1)	[15.00000000]
10.00000000：	(cam_open cam1)	[1.00000000]
15.00000000：	(cam_takephoto cam1 asteroid1)	[5.00000000]
20.00000000：	(att_maneuver att1 asteroid1 star1)	[15.00000000]
35.00000000：	(cam_takephoto cam1 star1)	[5.00000000]
Time：	Action：	Action Duration：
0.00000000：	(cam_ready cam1)	[10.00000000]
0.00000000：	(att_maneuver att1 initial asteroid1)	[15.00000000]
10.00010000：	(cam_open cam1)	[1.00000000]
15.00010000：	(cam_takephoto cam1 asteroid1)	[5.00000000]
20.00020000：	(att_maneuver att1 asteroid1 star1)	[15.00000000]
35.00030000：	(cam_takephoto cam1 star1)	[5.00000000]

Solution with original makespan 40 found (ignoring no-moving-targets-rule).
Solution was epsilonized and rescheduled to a makespan of 40.0003.
Plan length：6 step(s).
Makespan：40.0003

图 6 深空探测器小天体接近段任务的规划方案

Fig. 6 Plans for missions in the closing phase to a small body

按照第 1 节的建模方法建立多个深空探测领域问题模型。然后分别运用以动作时间为代价的和以条件数为代价的启发式搜索方法求解，记录搜索时间和规划解的完成时间，将两种结果对比，进而分析两种启发式信息的效果。用曲线图 7 表示，形象地比较两种启发式在搜索时间方面的优劣。

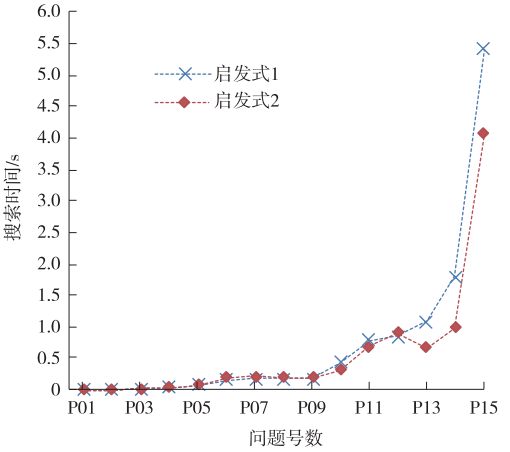

图 7 启发式搜索时间曲线比较图

Fig. 7 Comparison of heuristic 1 and heuristic 2

在图 7 中，启发式 1 为使用以动作时间为代价的上下文增强累加启发式(TFD 中)得到的搜索结

果,启发式 2 为使用以条件数为代价的启发式搜索方法得到的结果。横坐标是问题号数,问题号数是以问题复杂程度由简单到复杂排列,而问题复杂程度的判定是以领域描述复杂度及问题难易度为标准。以问题搜索时间为纵坐标,单位为秒(s),单位长度为 0.5 s,因此同一个横坐标下,曲线在下方的启发式方法搜索时间短,效果好。由图可知,随着问题复杂度的增加,问题搜索时间也增加;在两种启发式结果比较中,起初区别并不明显,而随着问题复杂度的增加,差别逐渐拉开,以条件数为代价的启发式搜索方法显示出优势。

从上述实验结果可知,对于深空探测领域问题,应用以条件数为代价的启发式搜索算法仍能得到规划结果,并且搜索时间仍然比较理想。当两种启发式得到的规划解质量相同时,在搜索时间方面,以条件数为代价的启发式搜索方法略胜一筹,适合实时性要求高的深空探测器。

4　结　论

本文针对深空探测任务特点,采用规划领域定义语言 PDDL,建立深空探测领域中多个问题模型,并对模型语句及定义方法举例说明,展示操作中遇到的时间与资源约束。随后应用以条件数为代价的启发式搜索方法对深空探测模型求解,并将其与以动作时间为代价的上下文增强累加启发式搜索方法得到的结果对比,得出以条件数为代价的启发式搜索方法在搜索速度方面略胜一筹,满足深空探测自主规划任务实时性要求。

参　考　文　献

[1] Bonet B, Geffner H. Planning as heuristic search[J]. Artificial Intelligence, 2001,129:5-33.

[2] Edelkamp S, Schrödl S. Heuristic search: theory and application [M]. United States of America: Margan Kaufmann, 2011:1-427.

[3] Cushing W, Kambhampati S, Mausam, et al. When is temporal planning really temporal? [C]// The International Joint Conference on Artificial Intelligence. Hyderabad, India:[s.n.], 2007.

[4] Haslum P, Geffner H. Heuristic planning with time and resources[C]// Proceedings of the Sixth European Conference on Planning. Toledo, Spain:[s.n.], 2001.

[5] Minh B D, Kambhampati S. Sapa: a multi-objective metric temporal planner [J]. Journal of Artificial Intelligence Research, 2003,20:155-194.

[6] Helmert M, Geffner H. Unifying the causal graph and additive heuristics[C]// Proceeding of the 18th International Conference on Automated Planning and Scheduling(ICAPS). Sydney, Australia: [s.n.], 2008.

[7] 彭婷.基于修复策略的深空探测器自主任务规划方法研究[D].北京:北京理工大学,2012.[Peng T. Autonomous mission planning method of deep space explorer based on repair planning strategy[D]. Beijing: Beijing Institute of Technology, 2012.]

[8] Fox M, Long D. PDDL2.1: an extension to PDDL for expressing temporal planning domains [J]. Journal of Artificial intelligence research, 2003,20:61-124.

[9] Smith B, Sherwood R, Govindjee A, et al. Representing spacecraft mission planning knowledge in ASPEN [C]// Artificial Intelligence Planning Systems Workshop on Knowledge Acquisition. Pittsburgh: [s.n.], 1998.

[10] 饶东宁,蒋志华,姜云飞.规划领域定义语言的演进综述[J].计算机工程与应用,2010,46(22):23-25.[Rao D N, Jiang Z H, Jiang Y F. Review on evolution of planning domains definition language [J]. Computer Engineering and Applications, 2010,46(22):23-25.]

[11] Helmert M. The fast downward planning system[J]. Journal if Artificial Intelligence Research, 2006(26):191-246.

[12] Eyerich P, Mattmüller R, Röger C. Using the context-enhanced additive heuristic for temporal and numeric planning [C] // Proceeding of the 19th International Conference on Automated Planning and Scheduling (ICAPS). Thessaloniki, Greece: [s.n.], 2009.

作者简介:

李朝玉(1990—),女,博士,主要研究方向:深空探测器自主任务规划。

通讯地址:北京市海淀区中关村南大街 5 号院北京理工大学宇航学院(100081)

电话:(010)68918910

E-mail:handylzy@163.com

Time Stamped States based Heuristic Algorithm for Spacecraft Mission Planning

LI Zhaoyu[1,2], XU Rui [1,2]

(1. Institute of Deep Space Exploration Technology, Beijing Institute of Technology, Beijing 100081, China;
2. Key Laboratory of Dynamics and Control of Flight Vehicle, Ministry of Education, Beijing 100081, China)

Abstract: For real time in deep space exploration, it is a requirement of autonomous mission planning for the explorer to find a plan as soon as possible. A kind of method is to use heuristic algorithm. At the same time, durative actions and numeric information have to be processed. According to these characteristics, this paper adapts planning domain definition language (PDDL) to establish knowledge models and describe time and resource constraints. Then the heuristic algorithm based on condition number is proposed to solve planning problems of deep space exploration. Finally, we compare this heuristic with context-enhanced additive heuristic based on action time in TFD (Temporal Fast Downward) planner. The result of the experiment shows that the heuristic algorithm we proposed is better to solve the planning problems in deep space from the point of view of real time.

Key words: deep space exploration; heuristic; durative actions; numeric

[责任编辑:高莎]