软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2017,28(5):1118−1127 [doi: 10.13328/j.cnki.jos.005207]

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

# 多机器人路径规划的安全性验证*

刘 涛, 王淑灵, 詹乃军

(计算机科学国家重点实验室(中国科学院 软件研究所),北京 100190)

通讯作者: 詹乃军, E-mail: znj@ios.ac.cn

**摘 要**: 近年来,伴随着人工智能领域的浪潮,机器人越来越多地出现在我们的日常生活中,例如足球机器人、无人机、无人车等.如何保证这些自治机器人尤其是多个机器人在移动过程中的安全,成为人们一直很关心的问题.混成通信顺序进程(hybrid communicating sequential process,简称 HCSP)是一种针对混成系统的形式化建模语言,在通信顺序进程(communicating sequential process,简称 CSP)的基础上引入了微分方程以描述混成系统中的连续行为和控制逻辑,可以方便而高效地对大型控制系统,尤其是在有通信事件发生时的情形进行形式化建模.用 HCSP 建模多机器人的路径控制算法,并用定理证明工具 HProver 进行形式化验证,结果表明,在满足一定的初始条件下,机器人团队在整个运行途中不会发生碰撞.

**关键词**: 人工智能;机器人;混成通信顺序进程;混成系统;形式化验证;定理证明

**中图法分类号**: TP311

中文引用格式: 刘涛,王淑灵,詹乃军.多机器人路径规划的安全性验证.软件学报,2017,28(5):1118−1127. http://www.jos.org.cn/1000-9825/5207.htm

英文引用格式: Liu T, Wang SL, Zhan NJ. Safety verification of trajectory planning for multiple robots. Ruan Jian Xue Bao/ Journal of Software, 2017,28(5):1118−1127 (in Chinese). http://www.jos.org.cn/1000-9825/5207.htm

## Safety Verification of Trajectory Planning for Multiple Robots

LIU Tao, WANG Shu-Ling, ZHAN Nai-Jun

(State Key Laboratory of Computer Science (Institute of Software, The Chinese Academy of Sciences), Beijing 100190, China)

**Abstract**: In recent years, with the rapid development of artificial intelligence, people encounter more and more robots, such as soccer robots, unmanned aerial vehicles, and unmanned automobiles in their daily life. How to guarantee the safety with those autonomous robots, especially team of robots on the move, has drawn people's attention. Hybrid CSP (HCSP) is a modelling language for hybrid systems. As an extension of CSP, HCSP introduces into CSP differential equations to describe the continuous behavior and control logic in hybrid systems. It can be used to model large-scale control systems, especially those with occurrences of communication events. In this paper, the trajectory planning algorithm is first modeled for multi robots using HCSP, and some properties are then formally proved with the theorem prover HProver. The paper finally proves that collision will never happen during the whole process under some initial conditions.

**Key words**: artificial intelligence; robot; hybrid communicating sequential process; hybrid system; formal verification; theorem proving

近年来,伴随着人工智能领域的浪潮,机器人越来越多地出现在我们的日常生活中.谷歌、百度、特斯拉等都相继要推出无人驾驶汽车,亚马逊、京东等大型电子商务网站也在部署自己的无人机送货系统,类似于大疆创新科技的无人机创新企业也在慢慢进入人们的生活,人工智能、机器人已成为当今的世界潮流.在机器人和

* 基金项目: 国家自然科学基金(91118007, 6110006, 61304185)
Foundation item: National Natural Science Foundation of China (91118007, 6110006, 61304185)
收稿时间: 2016-06-15; 修改时间: 2016-09-25; 采用时间: 2016-12-07; jos 在线出版时间: 2017-01-20
CNKI 网络优先出版: 2017-01-20 16:06:30, http://www.cnki.net/kcms/detail/11.2560.TP.20170120.1606.002.html

人工智能飞速发展的同时,我们也面临一个十分严峻但又不可避免的安全问题.安全问题成为制约无人车、无人机走向大众的关键问题,系统的任何错误都可能导致灾难性后果.例如,2016 年 2 月 14 日,谷歌一辆无人车在加州街头与一辆公交车相撞,虽然损失不大,但对人们的生命财产造成直接的威胁.

在所有机器人团队的应用场景中,都要考虑每个机器人的路径问题,包括为每个机器人分配一个目标位置和为每个机器人产生一条安全的运动路径.在目标分配问题中,目前采用最多的是最优化方法,即让所有机器人运行的总的路径和最小.考虑到实用性和实时性,这里我们主要关注线性分配算法,非线性算法复杂度很高,验证起来过于麻烦.目前,线性分配算法中应用最广泛的是 Hungarian Algorithm,其复杂度是 $O(N^3)$.目标位置分配完以后,然后就要为每个机器人制定一条安全的路径.对于单个机器人,经典的算法如图上搜索算法 $A^*$和 $D^*$是解决这个问题最有效的方法,但是复杂度仍然很高,尤其是同时为多个机器人搜索安全路径,其搜索空间随着机器人个数的增长呈指数增长.CAPT 和 D-CAPT 算法是一种解决此类问题的比较好的方法,兼顾复杂度和实时性,在实际使用中有很大价值.

尽管人们十分关注机器人的安全性问题,其验证工作仍然是一项十分有挑战性的话题.现实中的机器人要同时考虑很多因素,比如环境的不确定性、测量元件和执行元件的误差、算法的实时性等.同时,现有机器人融入了很多统计和概率的算法,比如机器学习算法、神经网络算法以及目前十分火热的深度学习算法.本来,像深度学习算法在学术上就是一种黑箱算法,学术界对其运行原理不甚了解,这类算法的引入,更是给机器人控制算法的安全性验证工作带来了困难.

本文中,我们主要考虑机器人团队在独立运行期间的碰撞性问题,而忽略环境的变化,比如突然而来的障碍物所带来的影响.我们使用著名的 D-CAPT 算法并验证其安全性问题.和现有的工作,比如文献[1]不同,我们使用 HCSP 和 HProver 来形式化建模并验证机器人的行为,兼顾了其离散的控制策略和连续的物理行为.本文的贡献主要如下:

(1) 提出了一个描述机器人,尤其是有通信发生时的多机器人的行为模型;

(2) 形式化地证明了多机器人的路径控制算法 D-CAPT 的安全性,即运行中不会发生碰撞.

本文第 1 节首先介绍国际上关于机器人建模和安全性验证的相关工作.第 2 节简介背景知识,包括 HCSP(我们用来建模机器人运动行为的建模语言)和混成霍尔逻辑(hybrid Hoare logic,简称 HHL).第 3 节介绍多机器人情况下的路径规划模型,给出安全条件下的形式化描述,并验证其满足安全性条件.第 4 节总结本文并展望未来的工作.

## 1 相关工作

随着机器人、无人机、无人车越来越多地出现在人们生活中,其安全性越来越受到工业界和学术界的重视.之前的验证工作主要集中在避碰算法的正确性上,避碰算法主要是在机器人的碰撞规避调度器模块上,独立于机器人的路径选择.而且之前的工作很少考虑机器人团队之间的碰撞问题,尤其是机器人之间有通信发生的情形.

Platzer 等人[2]提出了被动安全(passive safety)和被动友好安全(passive friendly safety)的概念.被动安全即机器人在移动过程中不会和其他物体发生碰撞.被动友好安全即在被动安全的基础上,当机器人停止运动后,仍旧与障碍物保持安全距离,保证不会发生碰撞.他们用混成系统模型和定理证明技术验证机器人的离散控制决策和连续的物理行为,并以著名的避碰算法动态窗口算法为例,证明了其满足被动安全和被动友好安全条件.该工作是用微分动力逻辑来建模混成系统模型,由于该逻辑的限制,这里并没有进一步考虑在有通信发生时的情形.

Hu 等人[3]提出用数值模拟算法来近似描述飞行器行为的离散马尔科尔链模型,主要近似从一个初始集合出发的有限时间的带概率的可达集.与之不同的是,我们这里考虑的是既有离散控制又有连续物理行为的混成系统,而不是任意的概率连续行为.

Hwang 等人[4]提出了一种直线飞行的飞行器避碰调度算法.该算法涉及到复杂三角运算上的优化问题.他

们用随机数值模拟的方法对该算法进行验证.由于该方法没有采用严格的形式化方法进行验证,故并不能完全保证正确性.

Platzer 等人[5]用微分动力逻辑验证了多个飞行器在绕圆盘旋飞行时的安全性问题.将整个过程按进入圆、绕圆飞行和飞离圆分解为 7 个子模块分别进行验证.该方法直接分析混成系统,兼顾了碰撞检测和碰撞规避.他们的思想和方法与我们的较为相近,都是直接建模混成系统,然后用定理证明工具进行形式化证明.由于微分动力逻辑不能建模有通信发生的情形,因此,这里虽然考虑多个飞行器,但是将它们独立考虑,而在实际中,飞行器之间会有通信发生.

Zhan 等人[6,7]用 HCSP 建模并验证了中国高速铁路列控系统中的一些关键场景.由于高速铁路列控软件和硬件规模过于复杂,主要采用 Simulink\Stateflow 建模工具对其行车许可、等级升级及部分模式转换场景进行建模并仿真,HCSP 的形式验证只是起到辅助性的作用,且只是取其一个不正常的场景进行形式验证.

LQG-MP[8]是一种十分复杂的路径规划算法.该算法适用于在不确定信息的环境下,综合考虑传感器、控制器和运动轨迹计算出随机产生的若干条路径发生碰撞的概率,并选择其中概率最小的路径.该算法适应性较强,但是不能保证完全的安全性,即只是寻找具有较小碰撞概率的路径.

总而言之,我们的方法和之前的方法有所不同.

- 我们不是验证避碰算法的正确性,而是验证路径规划的正确性,即保证机器人按照给定的算法产生的路径飞行不会发生碰撞.
- 我们考虑的是多机器人,即机器人团队共同运行的情形,而且机器人之前有通信发生,彼此之间的运动轨迹会相互影响.这种情形更贴近现实,但是验证起来也更加复杂.
- 我们考虑的是机器人团队之间不会有碰撞发生,没有考虑环境中的影响,即:我们考虑机器人与机器人之间的碰撞问题,而忽略机器人与环境中障碍物的碰撞问题.
- 我们使用符号验证而不是数值计算,因此避免了数值计算中的浮点数计算带来的潜在错误的可能性.
- 我们不是最小化概率意义上的碰撞,而是用定理证明了碰撞在机器人之间不可能发生.

## 2 背景知识

在自然状态下,多自治机器人的共同运动构成一个很复杂的混成系统,涉及到单机器人的运动控制和机器人之间的通信问题.为了建模和验证这一类问题,我们使用了 HCSP 和 HHL.

### 2.1 混成CSP

HCSP(hybrid communicating sequential process)[9−12]是一种针对混成系统的形式化建模语言.它在 CSP (communicating sequential process)的基础上引入了微分方程,以描述在混成系统中的连续动态行为;并引入中断机制(条件中断和通信中断)来中断一个连续的微分行为,进而使连续微分行为和离散控制行为交错运行.HCSP 能够同时刻画系统的连续行为和控制逻辑,并具有高可组合性的特性,是一种很好的用来描述大型控制系统的形式化建模语言.在 HCSP 中,各个进程之间以消息通信的方式进行数据交换,HCSP 禁止进程间以共享变量的形式共享数据.

HCSP 的语法如下所示:

$$\begin{aligned} P ::= & \ \textbf{\textit{skip}} \mid x := e \mid \textbf{\textit{wait}}\ d \mid ch?x \mid ch!e \mid P;Q \mid B \to P \mid P \sqcup Q \mid P^* \\ & \mid \langle F(\dot{s},s) = 0 \,\&\, B \rangle \mid \langle F(\dot{s},s) = 0 \,\&\, B \rangle \trianglerighteq_{i \in I} (io_i \to Q_i) \\ & \mid \langle F(\dot{s},s) = 0 \,\&\, B \rangle \trianglerighteq_d Q, \\ S ::= & \ P \mid S \parallel S. \end{aligned}$$

其中,$P,Q,Q_i,S$ 均为 HCSP 进程,$x$ 和 $s$ 为进程变量,$ch$ 为通道名,$io_i$ 为通信事件(输入事件 $ch?x$ 或输出事件 $ch!e$),$B$ 和 $e$ 为布尔表达式和算术表达式.以上语法结构的非形式化语义表示如下:***skip*** 不执行任何操作,立即终止;$x:=e$ 将表达式 $e$ 的值赋给 $x$,该过程不消耗时间,程序立刻终止;$ch!e$ 将 $e$ 的值发送到通道 $ch$,$ch?x$ 从通道 $ch$ 中得到一个值,并将其赋给 $x$,通信双方需等待对方进入准备状态后进行通信;$P;Q$ 先执行进程 $P$,当进程 $P$ 终止时执行进

程 $Q$;$B\rightarrow P$ 在 $B$ 为真时执行 $P$,否则立刻终止;$P$;$\sqcup Q$ 由系统随机选择是执行进程 $P$ 还是 $Q$;$P^*$表示有限次重复执行进程 $P$;$\langle F(\dot{s},s)=0\ \&\ B\rangle$ 代表微分动态过程,该微分过程相关的变量取值必须使得布尔表达式 $B$ 始终取真值,否则该语句立即终止;$\langle F(\dot{s},s)=0\ \&\ B\rangle\,|\,\langle F(\dot{s},s)=0\ \&\ B\rangle \trianglerighteq_{i\in I}(io_i\rightarrow Q_i)$ 与 $\langle F(\dot{s},s)=0\ \&\ B\rangle$ 执行过程基本一致,但是当通信事件 $io_i$ 发生时会立刻执行对应进程 $Q_i$,否则它会一直执行到微分过程终止为止;$\langle F(\dot{s},s)=0\ \&\ B\rangle \trianglerighteq_d Q$ 与 $\langle F(\dot{s},s)=0\ \&\ B\rangle$ 执行过程也基本一致,当执行时间超过 $d$ 时会立刻执行进程 $Q$;$S_1\|S_2$ 中,$S_1$ 和 $S_2$ 为并发进程,在不需要通信时它们各自独立运行,当需要通信时,通信双方需要同步执行该通信过程,并发进程之间不能存在共享变量或是共享输入或者输出通道.

在混成系统中,系统时间消耗分为物理环境连续迁移和控制计算两种.现代计算单元的计算速度非常快,计算时间非常短,在分析混成系统的行为时,常常会将控制计算时间忽略不计,这种分析问题的方法被称为超稠密计算.HCSP 采用了超稠密计算,因此离散的计算(如 ***skip*** 和 $B\rightarrow P$)不消耗时间.通信双方可能会等待对方准备通信而消耗时间,同步通信过程不消耗时间.

### 2.2 混成霍尔逻辑

混成霍尔逻辑作为霍尔逻辑的混成补充,是形式化建模语言 HCSP 的逻辑证明系统.混成霍尔逻辑的断言包括两个部分:一阶逻辑公式和历史公式[13].一阶逻辑公式用于描述离散性质,历史公式用于描述系统的连续特性.在混成霍尔逻辑中,混成系统使用 HCSP 进程表示.因此,逻辑系统混成霍尔逻辑包含以下 3 个部分:一阶逻辑的公理和推导规则、历史公式的公理和推导规则及 HCSP 的公理和推导规则.

#### 2.2.1 历史公式

混成霍尔逻辑使用历史公式(HF)记录 HCSP 进程的连续性质,即进程的运行历史.历史公式使用时段演算的以下子集定义:

$$HF::=\ell<T|\ell=T|\ell>T|\lceil S\rceil^0|\lceil S\rceil|\neg HF|HF_1\wedge HF|HF_1\vee HF,$$

其中,$\ell$表示公式作用的时间区段的时间长度,$T\in R^+$是一个常数.状态公式 $S$ 是变量构成的一阶逻辑公式.$\ell<T$($\ell=T$ 或 $\ell>T$)表示相应的时间区段小于(等于或大于)$T$;$\lceil S\rceil^0$ 表示状态公式 $S$ 在一个 $\ell=0$ 的时间点成立;$\lceil S\rceil$表示状态公式 $S$ 在相应的时间区段上除右端点以外的其他时间点成立;$HF_1\wedge HF_2$ 表示对应的时间区段可以分成两个子区段,并且在第 1 个区段上 $HF_1$ 成立,第 2 个区段上 $HF_2$ 成立;逻辑连接符$\neg$和$\vee$与一阶逻辑中的理解相同.

#### 2.2.2 混成霍尔断言

混成霍尔断言包括 4 个部分:前置断言、HCSP 进程、后置断言和历史公式,记作:

$$\{Pre\}P\{Post;HF\},$$

其中,$Pre$ 表示进程 $P$ 执行前变量满足的性质;$Post$ 表示进程运行终止时系统满足的性质;历史公式 $HF$ 是 DC 公式的子集,用于描述进程 $P$ 的执行历史.

对于并发进程,霍尔断言变成:

$$\{pre_1,\ldots,pre_n\}P_1\|\ldots\|P_n\{post_1,\ldots,post_n;HF_1,\ldots,HF_n\}.$$

历史公式 HF 用于描述 HCSP 进程的连续性质,一阶逻辑公式 $Pre$ 和 $Post$ 用于描述系统的离散行为.因此,历史公式 HF 和一阶逻辑公式 $Pre$ 和 $Post$ 共同描述了系统的连续离散行为.

在文献[12]中,我们通过在经典的霍尔逻辑中引入历史公式构建了 HCSP 的证明系统混成霍尔逻辑.文献[6,10]将该逻辑在 Isabelle/HOL 中实现,即工具 HProver,并利用该工具验证了中国铁路控制系统中的一个实际例.HProver 能够机械地证明 HCSP 的模型满足的性质,并具备一定的自动性.

## 3 D-CAPT 算法及其验证

### 3.1 D-CAPT算法描述

D-CAPT 算法是一种针对多机器人和多目标路径规划的算法,是 CAPT 算法的一种推广.假设有 $N$ 个机器人和 $N$ 个目标位置,D-CAPT 问题要解决下面两个问题:

1) 为每个机器人分配一个目标位置,同时保证不能有两个机器人被分配到同一个目标位置;

2) 为每个机器人规划一个运行轨迹,保证任意两个机器人在运行期间不能发生碰撞.

在 $n$ 维欧式空间中,考虑 $N$ 个半径为 $R$ 的机器人从不同初始位置出发去 $N$ 个不同的目标位置.我们定义从 1 到正整数 $Z$ 之间的集合表示为 $I_Z\equiv\{1,2,\ldots,Z\}$.因此,假设有 3 个目标位置,$N$=3,可以有 $I_N\equiv\{1,2,3\}$.用 $\boldsymbol{x}_i\in R^n$ 记作机器人 $i$ 的位置,$\boldsymbol{x}_i(t)$表示机器人 $i$ 在 $t$ 时刻的位置,$\boldsymbol{g}_i$ 表示机器人 $i$ 的目标位置.

该算法要求初始和目标位置分别满足下面的条件:

$$\|\boldsymbol{x}_i(t_0)-\boldsymbol{x}_j(t_0)\|>2\sqrt{2}R,\forall i\neq j\in I_N \tag{1}$$

$$\|\boldsymbol{g}_i-\boldsymbol{g}_j\|>2\sqrt{2}R,\forall i\neq j\in I_N \tag{2}$$

定义 $Nn$ 维状态向量 $\boldsymbol{X}\in R^{Nn}$ 表示机器人团队在 $t$ 时刻的位置:

$$\boldsymbol{X}(t)=[\boldsymbol{x}_1(t)^T,\boldsymbol{x}_2(t)^T,\ldots,\boldsymbol{x}_N(t)^T]^T.$$

类似地,定义目标状态向量 $\boldsymbol{G}\in R^{Nn}$:

$$\boldsymbol{G}=[\boldsymbol{g}_1^T,\boldsymbol{g}_1^T,\ldots,\boldsymbol{g}_N^T]^T.$$

定义分配矩阵 $\phi\in R^{N\times N}$ 如下:

$$\phi_{i,j}=\begin{cases}1, & \text{if robot } i \text{ is assigned to goal } j\\ 0, & \text{otherwise}\end{cases} \tag{3}$$

D-CAPT 的目的就是为了找到函数:

$$\gamma(t):[t_0,t_f]\rightarrow\boldsymbol{X}(t).$$

$t_0$ 和 $t_f$ 分别是初始时间和结束时间.机器人的初始位置 $x_i(t_0)$,其中,$\forall i\in I_N$,满足:

$$\gamma(t_0)=\boldsymbol{X}(t_0) \tag{4}$$

D-CAPT 的思想是求解下面的公式:

$$\underset{\varphi,\gamma(t)}{\text{minimize}}\ \sum_{i=1}^{N}\int_{t_0}^{t_f}\dot{\boldsymbol{x}}_i(t)^T\dot{\boldsymbol{x}}_i(t)\mathrm{d}t \tag{5}$$

等价于求解:

$$\underset{\varphi,\gamma(t)}{\text{minimize}}\ \int_{t_0}^{t_f}\dot{\boldsymbol{X}}(t)^T\dot{\boldsymbol{X}}(t)\mathrm{d}t \tag{6}$$

该公式的求解即 CAPT 算法的思想.其结果是,首先按照公式(6)求得一个解,为每个机器人分配一个目标,然后求出每个机器人各自到达目标位置的最小时间,从求得的最小时间中选出一个需要的最大时间作为每个机器人到达各自目标位置需要的时间,每个机器人在运行途中需要保持匀速,因此不同机器人的运行速度可能不一样,可以证明该方法可以保证每个机器人在运行途中不会发生碰撞.但是当 $N$ 很大时,求解上述公式的代价变得很大.D-CAPT 如算法 1 给出一种实时的、可行的得到局部最优解的方法.该算法以 CAPT 的结论作为基础,没有要求每个机器人到达目标位置所需时间一致.在该算法中,机器人目标函数 $\phi$ 是不知道的,因此定义 $f_i$ 作为初始分配给机器人 $i$ 的目标.$C_i(t)$定义为 $t$ 时刻机器人 $i$ 的通信范围内的邻近机器人集合,即

$$C_i(t)=\{j\,|\,\|\boldsymbol{x}_j(t)-\boldsymbol{x}_i(t)\|\leqslant h,i\neq j\}\subset I_N.$$

$U_i(t)\subset C_i(t)$是机器人 $i$ 准备发送信息的目标机器人集合.$t_c$ 表示当前时间满足:

$$t_0\leqslant t_c\leqslant t_f.\boldsymbol{r}_{i,j}=\boldsymbol{x}_j(t_f)-\boldsymbol{x}_i(t_c),\boldsymbol{n}_{i,j}=\boldsymbol{x}_j(t_c)-\boldsymbol{x}_i(t_c),\boldsymbol{w}_{i,j}=\boldsymbol{x}_j(t_f)-\boldsymbol{x}_i(t_f).$$

机器人 $i$ 在接下来的时间段 $t\in[t_c,t_f]$的运行轨迹可以表示为

$$\boldsymbol{x}_i(t)=\left(1-\frac{t-t_c}{t_f-t_c}\right)\boldsymbol{x}_i(t_c)+\left(\frac{t-t_c}{t_f-t_c}\right)\boldsymbol{f}_i \tag{7}$$

**算法 1**. $\boldsymbol{x}_i$=*D-CAPT*($\boldsymbol{x}_i(t_0)$,$\boldsymbol{f}_i$).

1: compute trajectory using CAPT

2: $U_i=C_i(t_0)$

3: $t_{prev}\leftarrow t_0$

4: **while** $t < t_f$ **do**
5: $t_c \leftarrow t$
6: **for** $j \in U_i$ **do**
7: **if** $j \notin C_i(t_{prev})$ **then**
8: $U_i = U_i \cup j$
9: $U_i = U_i \cap C_i(t_c)$
10:
11: **for** $j \in U_i$ **do**
12: request $x_j(t_c)$ and $f_j$ from agent $j$ (communication)
13: **if** $\boldsymbol{u}_{ij}^T \boldsymbol{w}_{ij} < 0$ **then**
14: send to robot $j$ to change goal location to $f_i$ (communition)
15: $f_i \leftarrow f_j$
16: $U_i = U_i \setminus j$
17:
18: **if** robot $j$ sends $f_j$ as the new robot $i$ goal **then**
19: $f_i \leftarrow f_j$
20: $U_i = C_i(t_c)$
21: recompute trajection using CART
22: $U_i = U_i \setminus j$
23: $t_{prec} \leftarrow t_c$

图 1(a)中,*robot* 0 和 *robot* 1 某一时刻距离小于 $h$,即发生通信;之后,如图 1(b)所示,两个机器人互相交换目标位置,即 $\boldsymbol{x}_0$ 目标位置变成 $\boldsymbol{g}_1$,$\boldsymbol{x}_1$ 的目标位置变成 $\boldsymbol{g}_0$.

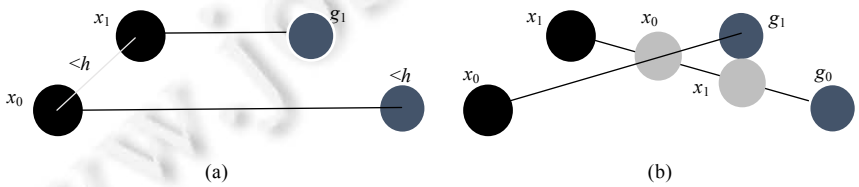



Fig.1
图 1

### 3.2 HCSP建模与验证

D-CAPT 算法是一种实时的、计算复杂度可行的多机器人路径规划算法,尤其在运行大量机器人的情形下,避免了 CAPT 算法中的高计算复杂度问题.为了验证此算法在应用中不会发生机器人碰撞事件的发生,不失一般性,这里任意选取两个机器人,建模其运动轨迹,并验证其安全性.由于 HCSP 采用超稠密计算,在分析混成系统的行为时,常常会将控制计算时间忽略不计,即离散的计算不消耗时间,因此,多个机器人的情况不会导致时间上的差别.但是在用 HCSP 建模时,机器人的个数 $N$ 必须是固定的,即 $P_1||P_2...||P_N$.当 $n$ 变大时,模型的验证工作量会大为增加,但过程并没有本质区别.因此,这里我们选取任意两个机器人,并假设其余的机器人的控制逻辑与此相同,其结果可以推广到多个机器人.任意两个机器人在运动过程中的情况可以分为两种:

i) 两个机器人在飞行过程中某一时刻的距离小于 $h$;

ii) 两个机器人在整个飞行途中的距离始终大于 $h$.

对于第一种情况,模型 1 描述了两个机器人在整个过程中的运行情形.初始位置与目标位置分别为 $\boldsymbol{s}_0$,$\boldsymbol{s}_1$,$\boldsymbol{gg}_0$,$\boldsymbol{gg}_1$,目标位置随机分配,彼此的状态通过通信获知.不失一般性,我们假设机器人之间相隔 $T$ 时间通信一次,也就是说,实际中,两个机器人相隔 $T$ 时间更新对方的状态.由于 HCSP 采用超稠密计算,通信过程不消耗时间,对于多个机器人的情况,同样是每隔 $T$ 时间更新对方状态.

**模型 1**. D-CAPT1.

$P_0||P_1$

$$P_0 \equiv \boldsymbol{x}_0=\boldsymbol{s}_0;\boldsymbol{g}_0=\boldsymbol{g}\boldsymbol{g}_0;ch_{x_0 2x_1}!\boldsymbol{x}_0;ch_{x_1 2x_0}?\boldsymbol{x}_1';ch_{x_0 2x_1 v}!\boldsymbol{v}_0;ch_{x_1 2x_0 v}?\boldsymbol{v}_1';$$
$$\left(\|\boldsymbol{x}_0-\boldsymbol{x}_1\|>h+T(v_0+v_1)\rightarrow\left\langle \dot{\boldsymbol{x}}_0=\boldsymbol{v}_0\frac{\boldsymbol{g}_0-\boldsymbol{x}_0}{\|\boldsymbol{g}_0-\boldsymbol{x}_0\|}\ \&\ \text{true}\right\rangle \trianglerighteq_T(ch_{x_0 2x_1}!\boldsymbol{x}_0;ch_{x_1 2x_0}?\boldsymbol{x}_1')\right)^*;$$
$$ch_{x_0 2x_1 g}!\boldsymbol{g}_0;ch_{x_1 2x_0 g}?\boldsymbol{g}_1';(\boldsymbol{g}_1'-\boldsymbol{g}_0)(\boldsymbol{x}_1'-\boldsymbol{x}_0)<0\rightarrow\boldsymbol{g}_0=\boldsymbol{g}_1';t_0=\frac{\|g_0-x_0\|}{v_{0\max}};$$
$$ch_{x_0 2x_1 t}!t_0;ch_{x_1 2x_0 t}?t_1';t_0<t_1'\rightarrow t_0=t_1';$$
$$v_0=\frac{\|\boldsymbol{g}_0-\boldsymbol{x}_0\|}{t_0};ch_{x_0 2x_1 v}!\boldsymbol{v}_0;ch_{x_1 2x_0}?\boldsymbol{v}_1';ch_{x_0 2x_1 g}!\boldsymbol{g}_0;ch_{x_1 2x_0 g}?\boldsymbol{g}_1';$$
$$\left\langle \dot{\boldsymbol{x}}_0=v_0\frac{\boldsymbol{g}_0-\boldsymbol{x}_0}{\|\boldsymbol{g}_0-\boldsymbol{x}_0\|};\dot{\boldsymbol{x}}_1'=v_1'\frac{\boldsymbol{g}_1'-\boldsymbol{x}_1'}{\|\boldsymbol{g}_1'-\boldsymbol{x}_1'\|}\ \&\ (\boldsymbol{x}_0\neq\boldsymbol{g}_0\wedge\boldsymbol{x}_1'\neq\boldsymbol{g}_1')\right\rangle$$
$$P_1 \equiv \boldsymbol{x}_1=\boldsymbol{s}_1;\boldsymbol{g}_1=\boldsymbol{g}\boldsymbol{g}_1;ch_{x_0 2x_1}?\boldsymbol{x}_0';ch_{x_1 2x_0}!\boldsymbol{x}_1;ch_{x_0 2x_1 v}?\boldsymbol{v}_0';ch_{x_1 2x_0 v}!\boldsymbol{v}_1;$$
$$\left(\|\boldsymbol{x}_1-\boldsymbol{x}_0\|>h+T(v_0+v_1)\rightarrow\left\langle \dot{\boldsymbol{x}}_1=v_1\frac{\boldsymbol{g}_1-\boldsymbol{x}_1}{\|\boldsymbol{g}_1-\boldsymbol{x}_1\|}\ \&\ \text{true}\right\rangle \trianglerighteq_T(ch_{x_0 2x_1}?\boldsymbol{x}_0';ch_{x_1 2x_0}!\boldsymbol{x}_1)\right)^*;$$
$$ch_{x_0 2x_1 g}?\boldsymbol{g}_0';ch_{x_1 2x_0 g}!\boldsymbol{g}_1;(\boldsymbol{g}_0'-\boldsymbol{g}_1)(\boldsymbol{x}_0'-\boldsymbol{x}_1)<0\rightarrow\boldsymbol{g}_1=\boldsymbol{g}_0';t_1=\frac{\|\boldsymbol{g}_1-\boldsymbol{x}_1\|}{v_{1\max}};$$
$$ch_{x_0 2x_1 t}?t_0';ch_{x_1 2x_0 t}!t_1;t_1<t_0'\rightarrow t_1=t_0';$$
$$v_1=\frac{\|\boldsymbol{g}_1-\boldsymbol{x}_1\|}{t_1};ch_{x_0 2x_1 v}?\boldsymbol{v}_0';ch_{x_1 2x_0 v}!\boldsymbol{v}_1;ch_{x_0 2x_1 g}?\boldsymbol{g}_0';ch_{x_1 2x_0 g}!\boldsymbol{g}_1;$$
$$\left\langle \dot{\boldsymbol{x}}_1=v_1\frac{\boldsymbol{g}_1-\boldsymbol{x}_1}{\|\boldsymbol{g}_1-\boldsymbol{x}_1\|};\dot{\boldsymbol{x}}_0'=v_0'\frac{\boldsymbol{g}_0'-\boldsymbol{x}_0'}{\|\boldsymbol{g}_0'-\boldsymbol{x}_0'\|}\ \&\ (\boldsymbol{x}_1\neq\boldsymbol{g}_1\wedge\boldsymbol{x}_0'\neq\boldsymbol{g}_0')\right\rangle$$

我们要验证的算法的安全性是指在整个运行过程,即机器人从初始位置出发到目标位置的过程中不会发生碰撞事件,用公式表示如下:

$$Post=\{\boldsymbol{x}_0=\boldsymbol{g}_0,\boldsymbol{x}_1=\boldsymbol{g}_1;\lceil\|\boldsymbol{x}_0-\boldsymbol{x}_1\|>2R\rceil\}.$$

根据 D-CAPT,pre-condition 可用公式表示如下:

$$Pre=R>0\wedge\|\boldsymbol{s}_0-\boldsymbol{s}_1\|>2\sqrt{2}R\wedge\|\boldsymbol{g}\boldsymbol{g}_0-\boldsymbol{g}\boldsymbol{g}_1\|>2\sqrt{2}R\wedge h>2\sqrt{2}R\wedge 0<v_0<v_{0\max}\wedge 0<v_1<v_{1\max}\wedge T>0,$$

其中,$R>0$ 意思是机器人有实际大小,这里用圆盘近似.$0<v_0<v_{0\max}\wedge 0<v_1<v_{1\max}$ 保证机器人在运行过程中速度不会大于其最大速度.$T>0$ 表明机器人之间每隔 $T$ 时间通信一次,且通信间隔大于 0.实际中,$T$ 是一个很小的数值,即有一个上限.这里,我们没有对其限制,而是将其作为一个参数,即可以变化.$\|\boldsymbol{s}_0-\boldsymbol{s}_1\|>2\sqrt{2}R\wedge\|\boldsymbol{g}\boldsymbol{g}_0-\boldsymbol{g}\boldsymbol{g}_1\|>2\sqrt{2}R$ 是 D-CAPT 算法的条件,机器人初始位置之间间隔和目标位置之间间隔要大于 $2\sqrt{2}R$. $h>2\sqrt{2}R$ 意思是通信距离或理解成实际中传感器(雷达、红外等)的有效探测距离也必须大于 $2\sqrt{2}R$.

所以,证明目标可以表示为

$$\{Pre;Pre\}P_0\|P_1\{\boldsymbol{x}_0=\boldsymbol{g}_0,\boldsymbol{x}_1=\boldsymbol{g}_1;\lceil\|\boldsymbol{p}_0-\boldsymbol{p}_1\|>2R\rceil\}.$$

我们用 HProver,一个关于 HCSP 的定理证明工具证明了上述目标,即当初始满足 *Pre* 条件时,算法整个运行过程中不会有碰撞发生.

对于第 2 种情况,模型 2 描述了整个运行期间的过程.由于不会发生 $h<2\sqrt{2}R$ 的时刻,所以整个过程相对比较简单.很容易证明,在这种情况下不会发生碰撞.用公式表示,即

$$\{Pre;Pre\}P_0\|P_1\{\boldsymbol{x}_0=\boldsymbol{g}_0,\boldsymbol{x}_1=\boldsymbol{g}_1;\lceil\|\boldsymbol{p}_0-\boldsymbol{p}_1\|>2R\rceil\}.$$

**模型 2**. D-CAPT2.

$$P_0\|P_1$$
$$P_0 \equiv \boldsymbol{x}_0=\boldsymbol{s}_0;\boldsymbol{g}_0=\boldsymbol{g}\boldsymbol{g}_0;ch_{x_0 2x_1}!\boldsymbol{x}_0;ch_{x_1 2x_0}?\boldsymbol{x}_1';ch_{x_0 2x_1 v}!\boldsymbol{v}_0;ch_{x_1 2x_0 v}?\boldsymbol{v}_1';$$
$$\left(\|\boldsymbol{x}_0-\boldsymbol{x}_1\|>h+T(v_0+v_1)\rightarrow\left\langle \dot{\boldsymbol{x}}_0=v_0\frac{\boldsymbol{g}_0-\boldsymbol{x}_0}{\|\boldsymbol{g}_0-\boldsymbol{x}_0\|}\ \&\ \boldsymbol{x}_0\neq\boldsymbol{g}_0\right\rangle \trianglerighteq_T(ch_{x_0 2x_1}!\boldsymbol{x}_0;ch_{x_1 2x_0}?\boldsymbol{x}_1')\right)^*;$$
$$P_1 \equiv \boldsymbol{x}_1=\boldsymbol{s}_1;\boldsymbol{g}_1=\boldsymbol{g}\boldsymbol{g}_1;ch_{x_0 2x_1}?\boldsymbol{x}_0';ch_{x_1 2x_0}!\boldsymbol{x}_1;ch_{x_0 2x_1 v}?\boldsymbol{v}_0';ch_{x_1 2x_0 v}!\boldsymbol{v}_1;$$
$$\left(\|\boldsymbol{x}_0-\boldsymbol{x}_1\|>h+T(v_0+v_1)\rightarrow\left\langle \dot{\boldsymbol{x}}_1=v_1\frac{\boldsymbol{g}_1-\boldsymbol{x}_1}{\|\boldsymbol{g}_1-\boldsymbol{x}_1\|}\ \&\ \boldsymbol{x}_1\neq\boldsymbol{g}_1\right\rangle \trianglerighteq_T(ch_{x_0 2x_1}?\boldsymbol{x}_0';ch_{x_1 2x_0}!\boldsymbol{x}_1)\right)^*;$$

### 3.3 形式验证

类似于文献[6],我们利用 HHL 定理证明工具 HProver 来完成上述验证工作,主要包括 4 个文件,分别对应于模型的变量定义、进程定义、断言定义和最终的证明目标,它们分别是 varDef.thy,assertionDef.thy,processDef.thy 和 goal.thy.

assertionDef.thy 主要结构如下,主要定义了 HCSP 进程语句间的断言、前置条件以及中间的不变式等.

```
theory assertionDef.thy
    imports "varDef.thy"
begin
(*Define consts for processes definition.*)
Definition Inv::fform where
"Inv==(pa [−] pb) [>] (Real 2) [*] R"
Definition Pre::fform where
"Pre==(R [>] (Real 0) [&] (pa [−] pb [>] (Real 2) [*] (Sqrt 2) [*] R) [&] (ga [−] gb [>] (Real 2) [*] (Sqrt 2) [*] R))"
…
end
```

processDef.thy 主要结构如下,主要定义了 HCSP 模型的整体框架.

```
Theory processDef
    imports "assertionDef"
begin
definition va_init::proc where
  "va_init==va:=((ga [−] pa)[**] t)"
definition vb_init::proc where
  "vb_init==vb:=((gb [−] pb) [**] t)"
definition robot_Diff::proc where
  "robot_Diff==(⟨Inv3⟩&&([~](pa[=]ga)[&][~](pb[=]gb))>:WTrue"
Definition pc1::proc where
  "pc1==(va_init;assert1;vb_init); assert2; robot_Diff"
…
end
```

在 HHL Prover 中,我们证明的结论如下:

```
(*Goal for the whole process.*)
lemma goal: "{Pre}Pro{Post; high ((pa [−] pb [>] (Real 2 [*] R))}"
```

证明过程主要是将证明目标根据 HHL 的顺序组合规则、赋值语句规则、连续语句规则等一步步分解成原子语句来证明.

限于篇幅,这里给出整个证明框架的核心部分代码,全部代码见 http://pan.baidu.com/s/1hsIxSMg.

```
apply(cut_tac px=Pre and qx="WTure" and Hx="l[=](Real 0)[∧]high((pa[−]pb)[>](Real 2[*]R))" in ConsequenceS, auto)
prefer 2
apply (rule conjR, rule impR, rule basic)
apply (rule conjR, rule impR, rule baisc)
apply (rule impR, rule LL3a, rule basic)
```

apply (simp add: Pro_def)

apply (simp add: assert2_def)

apply (rule Sequence)

apply (simp add: assert1_def)

apply (*cut_tac px=Pre* and *qx*="*pa*[−]*pb*[>](*Real* 2) [*] *varDef.R* [&] *ga*[−]*gb*[>](*Real* 2)[*]*varDef.R*[&]*va*[=](*ga*[−]*pa*)[**]*t*[&]*vb*[=](*gb*[−]*pb*)[**]*t*" and *Hx*="*l*[=](*Real* 0) [∧]*l*[=](*Real* 0)" in ConsequenceS, auto)

prefer 2

apply (rule conjR, rule impR, rule basic)

apply (rule conjR, rule impR, rule basic)

apply (rule impR, rule LL3a, rule basic)

## 4 总结与未来的工作

在本文中,我们提出了用 HCSP 来建模验证机器人路径规划中的安全性问题,尤其是多机器人在互相有通信发生时的情形,并用 HHL 定理证明器 HPover 对其进行形式验证.我们针对 D-CAPT 算法进行了建模,证明了在满足一定初始条件的前提下,机器人在整个运行途中不会发生碰撞.虽然模型只考虑了两个机器人的情形,但是很容易推广到任意多机器人,且结论也适用于任意多机器人.由于现实中机器人,尤其是机器人团队之间的通信过程和路径分配远远比该算法要复杂,本文是做了一个尝试,同时有很多地方可以进一步改进.比如,本算法中没有考虑机器人的加速问题,也没有考虑机器人在转向时的弧线轨迹,这在未来可以继续探索.

另外,本例考虑的是确定性的算法和轨迹,在现实中往往会出现一些随机扰动,在建模时考虑这些随机元素具有重要意义.詹乃军等人[14]对 HCSP 进行了概率和随机的扩充,这对我们未来的工作是一个方向上的指导.

**References**:

[1] Turpin M, Michael N, Kumar V. Capt: Concurrent assignment and planning of trajectories for multiple robots. The Int'l Journal of Robotics Research, 2014,33(1):98−112. [doi: 10.1177/0278364913515307]

[2] Mitsch S, Ghorbal K, Platzer A. On provably safe obstacle avoidance for autonomous robotic ground vehicles. 2013. [doi: 10.15607/RSS.2013.IX.014]

[3] Hu JH, Prandini M, Sastry S. Probabilistic safety analysis in three dimensional aircraft flight. In: Proc. of the 42nd IEEE Conf. on Decision and Control. IEEE, 2003. 5335−5340. [doi: 10.1109/CDC.2003.1272485]

[4] Hwang I, Kim J, Tomlin C. Protocol-Based conflict resolution for air traffic control. Air Traffic Control Quarterly, 2007,15(1):1−34.

[5] Platzer A, Clarke EM. Formal Verification of Curved Flight Collision Avoidance Maneuvers: A Case Study. Berlin, Heidelberg: Springer-Verlag, 2009. [doi: 10.1007/978-3-642-05089-3_35]

[6] Guo DQ, Lü JD, Wang SL, Tang T, Zhan NJ, Zhou DT, Zou L. Formal anlysis and verification of Chinese train control system. Science China Information Sciences, 2015,45(3):417−438 (in Chinese).

[7] Zou L, Lü JD, Wang SL, Zhan NJ, Tang T, Yuan L, Liu Y. Verifying Chinese train control system under a combined scenario by theorem proving. In: Proc. of the Verified Software: Theories, Tools, Experiments (VSTTE 2013). LNCS 8164, Berlin, Heidelberg: Springer-Verlag, 2013. 262−280.

[8] Van Den Berg J, Abbeel P, Goldberg K. LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. The Int'l Journal of Robotics Research, 2011,30(7):895−913. [doi: 10.1177/0278364911406562]

[9] Zhan NJ, Wang SL, Zhao HJ. Formal modelling, analysis and verification of hybrid systems. In: Proc. of the Unifying Theories of Programming and Formal Engineering Methods. Berlin, Heidelberg: Springer-Verlag, 2013. 207−281. [doi: 10.1007/978-3-642-39721-9_5]

[10] Wang SL, Zhan NJ, Zou L. An improved HHL prover: An interactive theorem prover for hybrid systems. In: Proc. of the Formal Methods and Software Engineering. Springer Int'l Publishing, 2015. 382−399. [doi: 10.1007/978-3-319-25423-4_25]

[11] Zou L, Zhan NJ, Wang SL, Fränzle M, Qin SC. Verifying simulink diagrams via a hybrid hoare logic prover. In: Proc. of the 11th ACM Int'l Conf. on Embedded Software (EMSOFT 2013). IEEE Press, 2013. 1−10. [doi: 10.1109/EMSOFT.2013.6658587]

[12] Liu J, Lü JD, Quan Z, Zhan NJ, Zhao HJ, Zhou CC, Zou L. A calculus for hybrid CSP. In: A keynotes of APLAS 2010. LNCS 6461, Berlin, Heidelberg: Springer-Verlag, 2010. 1−15. [doi: 10.1007/978-3-642-17164-2_1]

[13] Zhou CC, Hoare CAR, Ravn AP. A calculus of durations. Information Processing Letters, 1991,40(5):269−276. [doi: 10.1016/0020-0190(91)90122-X]

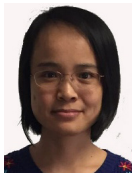
[14] Peng Y, Wang SL, Zhan NJ, Zhang LJ. Extending hybrid CSP with probability and stochasticity. In: Proc. of the Dependable Software Engineering: Theories, Tools, and Applications (SETTA 2015). LNCS 9409, Springer Int'l Publishing, 2015. 87−102. [doi: 10.1007/978-3-319-25942-0_6]

**附中文参考文献**:

[6] 郭丹青,吕继东,王淑灵,唐涛,詹乃军,周达天,邹亮.中国高速铁路列控系统的形式化分析与验证.中国科学:信息科学,2015,45(3):417−438.

**刘涛**(1993－),男,河南信阳人,硕士生,主要研究领域为嵌入式系统形式化验证.

**詹乃军**(1971－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为形式化方法.



**王淑灵**(1981－),女,博士,助理研究员,CCF 专业会员,主要研究领域为形式化方法,混成系统的建模与验证.