【电子与信息科学／Electronics and Information Science】 ·人工智能与数字经济专题·

基于单工架构的信息物理系统运行时安全性保障方法

王强[1]，陈强[1]，曹伟朋[2]，何文锋[3]

1）军事科学院系统工程研究院，北京 100070；2）人工智能与数字经济广东省实验室(深圳)，广东深圳 518107；3）深圳大学计算机与软件学院，广东深圳 518060

摘　要：运行时安全性保障方法旨在系统运行过程中确保安全性属性得到满足．基于单工(simplex)架构的运行时安全性保障方法能够在使用复杂且安全性未经验证的控制器的情况下，为系统提供安全性保证．其基本思想是将复杂且安全性未经验证的复杂控制器与经过严格形式化验证的安全控制器相结合，前者工作在无安全风险的情况下，若检测发现系统存在潜在的安全风险，则通过决策模块完成复杂控制器与安全控制器之间的切换，从而确保系统的安全性．本文重点关注基于simplex架构的信息物理系统运行时安全性保障方法，从simplex架构决策模块的设计、改进与拓展，以及simplex架构的应用3个方面展开研究，对相关工作进行了梳理和总结，指出当前面临的技术挑战，并展望未来的发展方向，认为基于simplex架构的运行时安全性保障方法将会是解决智能信息物理系统安全性保障问题的有效途径之一．

关键词：人工智能；形式化方法；运行时安全性保障；运行时验证；形式化建模与验证；信息物理系统；系统建模与验证；嵌入式系统

中图分类号：TP311　　**文献标志码：**A　　**DOI：**10.3724/SP.J.1249.2024.03253

Runtime safety assurance methods for cyber physical systems based on simplex architecture

WANG Qiang[1], CHEN Qiang[1], CAO Weipeng[2], and HE Wenfeng[3]

1) Institute of System Engineering, Academy of Military Sciences, Beijing 100070, P. R. China

2) Guangdong Laboratory of Artificial Intelligence and Digital Economy (Shenzhen), Shenzhen 518107, Guangdong Province, P. R. China

3) College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, Guangdong Province, P. R. China

Abstract: Runtime safety assurance is a method that can ensure security attributes being satisfied during system operation. The simplex architecture-based runtime safety assurance methods are able to provide the safety guarantees even in the presence of unverified complex controllers. The basic idea is to combine the complex controller with a rigorously formalized safe controller, where the former operates in the absence of safety risk. Whenever a potential safety risk is detected, a decision module then completes the switching between the complex controller and the safe controller in order to guarantee the safety. This paper investigates the runtime safety assurance methods for cyber physical systems and summarizes the most relevant work from three aspects: the design of simplex decision-making module, the extension and improvement of simplex architecture, and the application of simplex architecture. This paper also provides an analysis of technical challenges and future directions of simplex architecture-based runtime safety assurance methods, indicating their potential and effectiveness in solving the runtime safety assurance issues in intelligent cyber-physical systems.

Received: 2023-07-31; **Accepted:** 2023-11-02; **Online** (CNKI)**:** 2024-04-19
Foundation: National Natural Science Foundation of China (62106150)
Corresponding author: Associate research fellow CAO Weipeng (caoweipeng@gml. ac. cn)
Citation: WANG Qiang, CHEN Qiang, CAO Weipeng, et al. Runtime safety assurance methods for cyber physical systems based on simplex architecture [J]. Journal of Shenzhen University Science and Engineering, 2024, 41(3): 253-263. (in Chinese)

http://journal.szu.edu.cn

Key words: artrficial intelligence; formal methods; runtime safety assurance; runtime verification; formal modeling and verification; cyber physical system; system modeling and verification; embedded system

目前，信息物理系统(cyber physical systems，CPS)已被广泛地应用于交通、医疗、能源和国防等与安全攸关的领域，因此，其安全性保障变得尤为重要．近年来，随着深度学习(deep learning，DL)算法的快速发展，CPS越来越多地采用DL算法来实现感知、决策和控制等功能．但是，DL算法的广泛使用，在提升CPS自主性的同时，也给系统的安全性保障带来了新的挑战．一方面，DL算法自身的安全性保障仍是一项极具挑战的难题．DL算法往往通过数据驱动的方式训练神经网络模型，模型的行为机理缺乏可解释性，且模型的准确性严重依赖于训练样本的质量．另一方面，CPS与外界环境存在着大量的动态交互，且系统的感知数据具有较高的不确定性．为应对复杂不确定的环境输入或场景变化，CPS通常会集成多个异构的控制器[1]．其中不仅包括了既有的控制器，也包括了针对特定环境输入或场景而设计的新的控制器．尽管这种复杂的异构特性能够提升CPS在多种不同场景下的控制效率，但也使得系统的功能及架构变得更加复杂，导致传统的基于形式化验证的安全性保障方法很难应用于CPS．

运行时安全性保障(runtime safety assurance)是一种在系统运行过程中为系统提供安全性保障的方法[2]．该方法通过对系统的运行状态进行监控，以期检测出可能出现的违背安全性质的行为，并在必要时采取纠正措施以避免出现违背安全性质的行为．单工(simplex)架构[3]作为一种典型的运行时安全性保障方法，基于容错思想，通过使用冗余的具有安全性保障的组件，为复杂且未经验证的组件提供安全性保障．其中，未经安全性验证的组件工作在无安全风险的情况时，若检测发现系统可能存在安全风险，则通过决策模块将工作模式到切换到具有安全性保障的模块．

近年来，基于simplex架构的运行时安全性保障方法在学术界和工业界都受到了越来越多的关注，并已被广泛应用于诸多类型的CPS．本研究从simplex架构的基本原理、决策模块设计方法、改进拓展，以及其针对不同领域系统的应用4方面对运行时安全性保障方法进行详细剖析，并对基于simplex架构的运行时安全性保障方法目前面临的挑战进行了梳理，进而展望未来研究的方向．

1 Simplex架构介绍

Simplex架构[3]旨在提高控制系统的可靠性，并降低控制软件的开发成本．图1给出了基于simplex架构的运行时安全性保障方法示意．其中，复杂控制器(complex controller)通常指安全性难以进行严格形式化验证或证明的控制器，如基于深度学习的智能控制器(intelligent controller)或者先进控制器(advanced controller)等；安全控制器(safety controller)通常指安全性能够得到严格形式化验证或证明的控制器；决策模块(decision module)实现对系统运行状态的实时监控，并据此判断受控系统(plant)在复杂控制器的作用下(当前或未来)的运行状态是否会违背安全性约束，若存在违背安全性约束的情况，则将系统切换至安全控制器．

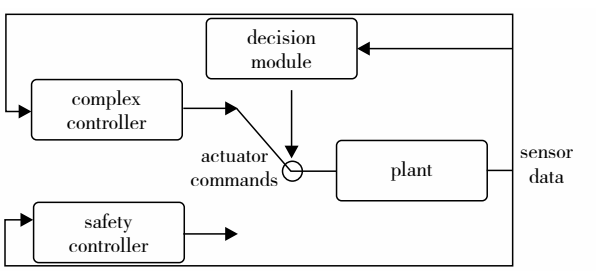


图1 基于simplex架构的运行时安全性保障方法[3]

Fig. 1 A runtime safety assurance method based on simplex architecture[3].

图2从状态空间的角度展示了simplex架构的基本原理．在系统运行期间，复杂控制器和安全控制器均可获取系统的状态信息以及外界环境的感知数据，并输出对系统的控制命令．当系统处于可恢复状态(状态1和状态2)时，复杂控制器接管对系统的控制，但是在安全关键状态下(状态3)，若系统继续采用复杂控制器的输出，则可能导致系统进入不安全状态(状态4)．因此，simplex架构的决策模块需要基于系统的当前状态和外界环境的感知数据等信息，预测复杂控制器作用下系统在下一时刻的状态．如果系统在下一时刻内不进入违背安全性约束的状态，那么决策模块将采用复杂控制器的输出；否则，就将控制从复杂控制器切换到安全控制器．例如，若系统处于状态3，在复杂控制器作用下，系统将到达不安全的状态4，则需将系统切换

http://journal.szu.edu.cn

至安全控制器．此外，为确保系统从安全控制器返回至复杂控制器后不会进入不安全状态，需确保系统处于可恢复状态．例如，状态 2 为可恢复状态，若此时系统从安全控制器切换至复杂控制器，则系统仍将保持在安全状态空间．

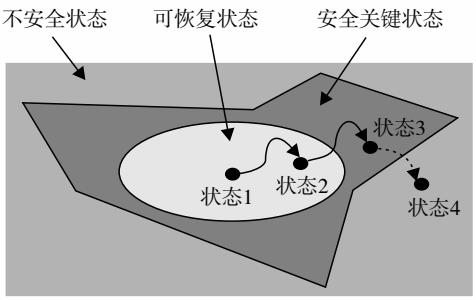


图2　Simplex架构基本原理

Fig. 2　Basic principles of simplex architecture.

可见，基于simplex架构的运行时安全性保障方法的关键在于实现系统状态预测与控制器切换逻辑的决策模块，该模块的关键又在于计算系统的可恢复状态空间．此外，决策模块需要有正确性保证，即当系统进入安全关键状态，并可能出现违背安全性约束的情况时，决策模块能够监控系统状态的变化，并及时从复杂控制器切换至安全控制器．需指出的是，系统所要保障的安全性约束与决策模块所监控的安全性约束性质是不同的．因此，不能直接监控系统所要保障的安全性约束，否则，决策模块将在运行时保障的安全性约束被违背时才会做出决策，导致无法避免地违背了安全性约束．因此，决策模块所监控的安全性约束需要是可预测的：该安全性约束的违背必须发生在系统所保障的安全性约束之前．理论上，决策模块还需具有最优性．一个简单的决策逻辑是始终采用安全控制器，显然这不是最优的．最优是指系统只有在必要的情况下才切换至安全控制器，即系统在复杂控制器作用下必将进入不安全状态的情况（图2状态3），而在非必要条件下，系统能够持续采用复杂控制器（图2状态1和状态2的情况）．

2　决策模块设计方法

Simplex架构中决策模块的设计方法主要有基于Lyapunov函数的方法[3]、基于可达集计算的方法[4-6]和基于屏障证书（barrier certificates）的方法[7-9]．其中，基于Lyapunov函数的方法主要用于计算线性系统的控制切换条件；基于可达集计算的方法将CPS抽象描述为混成自动机（hybrid automaton），并基于混成自动机的前向（或后向）可达性设计决策模块的切换条件；基于屏障证书的方法也将系统建模为混成自动机，通过计算混成自动机的屏障证书来合成决策模块的切换条件．JOHNSON等[4]将Lyapunov函数方法与可达集计算方法相结合，给出了决策模块的通用设计框架．

基于Lyapunov函数和可达状态空间方法设计的决策模块也称为切换式决策模块，即基于使用当前状态，选择使用其中一个控制器的输出；另外一种切换模块的设计方式是混合两个控制器的输出以创建最终输出，也称为过滤式决策模块，通常是通过使用控制屏障函数（control barrier function，CBF）混合来自安全控制器和复杂控制器的输出，从而使系统保持在安全状态．

此外，NIGAM等[10]讨论了基于可满足性模理论（satisfiability modulo theory，SMT）求解的simplex架构决策模块切换逻辑的自动化证明．

2.1　基于Lyapunov函数的方法

Simplex架构最初采用基于Lyapunov函数的方法来计算决策模块中的切换条件．该方法基于自动控制理论，设$\boldsymbol{x}$为CPS的状态向量，$\boldsymbol{x} \in \mathbf{R}^n$，其中，$n$为系统状态的维度，则CPS状态变化描述为

$$\dot{\boldsymbol{x}} = (\boldsymbol{A} + \boldsymbol{BK})\boldsymbol{x} \tag{1}$$

其中，$\boldsymbol{A}$和$\boldsymbol{B}$皆为系统的状态矩阵；$\boldsymbol{K}$为安全控制器的控制率．设$(\boldsymbol{A} + \boldsymbol{BK}) = \boldsymbol{P}$，则系统的安全性约束为

$$\boldsymbol{P}^{\mathrm{T}}\boldsymbol{Q} + \boldsymbol{QP} < 0 \tag{2}$$

其中，$\boldsymbol{Q}$为系统Lyapunov函数．所有满足$\boldsymbol{x}^{\mathrm{T}}\boldsymbol{Qx} = 1$条件的状态构成了安全状态空间的边界，而所有满足$\boldsymbol{x}^{\mathrm{T}}\boldsymbol{Qx} \leqslant 1$条件的状态构成了系统状态空间的一个子空间，后者通常被称为安全控制器作用下的安全包络或系统的可恢复空间．当系统的初始状态处于上述子空间内时，系统在安全控制器的作用下将一直处于该子空间内．

基于上述结论，simplex架构可采用$\boldsymbol{x}^{\mathrm{T}}\boldsymbol{Qx} \leqslant 1$作为决策模块的切换条件：当系统状态满足该不等式时，即系统状态位于可恢复空间，系统可切换至复杂控制器，此时，系统的可恢复空间就是复杂控制器的作用空间；当系统状态不满足上述不等式时，系统将切换至安全控制器．

该方法的优势在于决策模块的切换条件可

http://journal.szu.edu.cn

通过离线计算的方式得到．通常，线性系统的Lyapunov函数$\boldsymbol{Q}$可采用基于线性矩阵不等式(linear matrix inequalities，LMIs)的方法来计算．在系统运行过程中，决策模块只需基于系统的当前状态进行简单的条件判断，即$\boldsymbol{x}^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{x} \leqslant 1$．考虑到实际情况下，由于控制器存在控制周期和延迟，系统的切换控制无法立即生效．为确保切换过程的安全稳定性，复杂控制器的作用空间与可恢复空间还应当存在一定的余量d．因此，实际情况下决策模块的切换条件通常为$\boldsymbol{x}^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{x} \leqslant 1-d$．然而，离线计算只能得到可恢复空间的一个下近似，这就意味着切换条件可能会“保守”，即在不存在安全风险的状态下切换至安全控制器，无法充分发挥复杂控制器的效率．

2.2 基于可达集计算的方法

该方法通过混成自动机对CPS建模．混成自动机是一种能够描述CPS离散行为和连续行为的形式化模型，定义为$\boldsymbol{H}=(X, L, X_0, L_0, I, F, T)$．其中，$X$为系统的连续状态集合；$L$为系统的离散模态集合；$X_0$为系统的初始连续状态集合，$X_0 \subset X$；$L_0$为系统的初始模态，$L_0 \in L$；$I$：$L \to 2^X$为系统的不变式，对于每个离散模态$l$，$I(l)$为系统在该模态上所允许的连续状态集合，$I(l) \subseteq X$；$F$：$X \to 2^X$为系统的连续状态变化函数；$T$为系统的离散模态迁移关系，$T \subseteq X \times X$，×表示求两个集合的笛卡尔乘积，迁移$\big((l,\boldsymbol{x}),(l',\boldsymbol{x}')\big) \in T$表示系统从模态$l$转换到模态$l'$，并且将连续状态$\boldsymbol{x}$更新为$\boldsymbol{x}'$．

通常，模态迁移包括了1个激活条件和1个重置操作．给定初始状态，混成自动机的连续状态将依据函数F中所定义的状态变化关系而变化．若当前状态满足离散迁移的激活条件，则执行该离散迁移，并进入新模态，然后执行重置操作对并更新连续状态变量；若当前状态下同时存在多个激活的离散迁移，则随机选取并执行1个离散迁移．通过重复上述执行过程，将得到混成自动机的一条执行迹．

记系统的状态空间为集合S，$S=L\times X$，初始状态空间为集合S_0，$S_0=L_0\times X_0$，若给定混成自动机$\boldsymbol{H}$，则当且仅当$\boldsymbol{H}$存在一条从初始状态$s_0 \in S_0$到该状态的执行迹(包括连续状态变化和离散模态迁移)时，称状态$s \in S$是可达的．将复杂控制器作用下CPS的混成自动机模型记为$\boldsymbol{H}_{\mathrm{CC}}$，安全控制器作用下CPS的混成自动机模型记为$\boldsymbol{H}_{\mathrm{SC}}$，从$s \in S$开始，将所有可达状态集记为$R(\boldsymbol{H}, s, \infty)$，所有在$t$时间内的可达状态记为$R(\boldsymbol{H}, s, \leqslant t)$，所有在$t$时刻能够达到的状态记为$R(\boldsymbol{H}, s, =t)$．给定不安全状态集合$S_{\mathrm{u}}$，当且仅当$\boldsymbol{H}$的可达状态集与$S_{\mathrm{u}}$不相交，即$R(\boldsymbol{H}, s, \infty) \cap S_{\mathrm{u}} = \varnothing$时，称$\boldsymbol{H}$是安全的．

基于上述定义，基于可达集方法的simplex架构决策模块的切换逻辑为：给定系统的状态s、控制周期Δt和不安全状态集S_{u}，当且仅当条件(3)和(4)同时成立时，系统能够从安全控制器切换到复杂控制器．

$$R(\boldsymbol{H}_{\mathrm{CC}}, s, \leqslant \Delta t) \cap S_{\mathrm{u}} = \varnothing \tag{3}$$

$$R\big(\boldsymbol{H}_{\mathrm{SC}}, R(\boldsymbol{H}_{\mathrm{CC}}, s, =\Delta t), \infty\big) \cap S_{\mathrm{u}} = \varnothing \tag{4}$$

上述切换逻辑的基本思路是：首先，判断系统在复杂控制器作用下，是否可能在1个控制周期Δt内进入不安全状态(3)．其次，判断系统在复杂控制器工作1个控制周期后，并且在安全控制器作用下，是否可能进入不安全状态(4)．若在这两个条件下，系统都不会从当前状态s进入到不安全状态，则可采用复杂控制器；反之，系统需在下一个控制周期内切换到安全控制器．

上述可达集通常被称为基于前向可达性的状态集合，一个等价的切换逻辑是基于后向可达性．给定系统的当前状态s、控制周期Δt和不安全状态集S_{u}，则复杂控制器作用的条件为

$$s \notin R_{\mathrm{b}}\big(\boldsymbol{H}_{\mathrm{CC}}, R_{\mathrm{b}}(\boldsymbol{H}_{\mathrm{SC}}, S_{\mathrm{u}}, \infty), \leqslant \Delta t\big) \tag{5}$$

其中，$R_{\mathrm{b}}(\boldsymbol{H}_{\mathrm{SC}}, S_{\mathrm{u}}, \infty)$为系统在安全控制器作用下，所有可能达到不安全状态$S_{\mathrm{u}}$的状态集合(不限制控制周期)，也称为不可恢复状态集合，表示系统从这类状态开始，将不可避免地进入不安全状态；$R_{\mathrm{b}}(\boldsymbol{H}_{\mathrm{CC}}, R_{\mathrm{b}}(\boldsymbol{H}_{\mathrm{SC}}, S_{\mathrm{u}}, \infty), \leqslant \Delta t)$为系统在复杂控制器作用下，所有在1个控制周期内可能到达不可恢复状态的状态集合．换言之，若系统的当前状态不属于上述集合，则系统不会进入不可恢复状态，此时系统可继续运行复杂控制器；否则，系统必须切换至安全控制器．因此可证明，若系统初始状态是安全的，则在上述决策逻辑下，即便复杂控制器缺乏安全性保障，系统仍将一直处于安全状态空间内．

上述切换逻辑具有通用性，不仅适用于线性系统，也应用于非线性系统．但此切逻辑在实际应用中仍然存在两个难点：一是计算可达状态集合需要构建复杂控制器的混成自动机模型，且该模型的精

http://journal.szu.edu.cn

度决定了可达状态集合的准确度．然而在实践中，复杂控制器模型(如基于深度学习的控制器)很难被构建．二是上述切换逻辑的判断条件针对的是系统的当前状态s，这使得决策模块需在系统运行过程中对可达状态集合进行实时的在线计算．但是，混成自动机的可达集计算具有较高的复杂度．若要得到离线计算的判断条件，需遍历所有的(无穷)状态空间，显然，这种计算是不现实的．

2.3　基于屏障证书的方法

屏障证书是一种证明混成自动机安全性的方法，其基本思路是找到一个能够分割混成自动机可达状态集和不安全状态集的屏障，从而使系统从安全状态开始，经过任意的离散模态转换或连续状态变化后，后继状态不会“穿越屏障”，并始终处于安全状态空间内．

给定混成自动机$\boldsymbol{H}=(X, L, X_0, L_0, I, F, T)$，将其不安全状态集合记为$S_{\mathrm{u}}\subseteq S$，模态$l\in L$下的不安全状态记为$\mathrm{Unsafe}(l)=\{\boldsymbol{x}\in X\mid(l,\boldsymbol{x})\in S_{\mathrm{u}}\}$，模态$l\in L$下的初始状态记为$\mathrm{Init}(l)=\{\boldsymbol{x}\in X\mid(l,\boldsymbol{x})\in S_0\}$，并给定非负常量$\sigma_{(l,l')}$，其中$(l,l')\in L\times L$，则模态$l\in L$下的屏障证书为满足条件(5)至(8)的函数$B_l(\boldsymbol{x})$．

$$\forall x\in\mathrm{Unsafe}(l)\,B_l(\boldsymbol{x})>0 \tag{6}$$

$$\forall x\in\mathrm{Init}(l)\,B_l(\boldsymbol{x})\leqslant 0 \tag{7}$$

$$\forall(l,\boldsymbol{x})\in I\times X\ \frac{\partial B_l}{\partial\boldsymbol{x}}(x)\,f_l(\boldsymbol{x})\leqslant 0 \tag{8}$$

$$\forall(\boldsymbol{x},\boldsymbol{x}')\in X^2,\ \boldsymbol{x}'\in T(l',l)(\boldsymbol{x}'),\quad B_l(\boldsymbol{x})-\sigma(l',l)B_{l'}(\boldsymbol{x}')\leqslant 0 \tag{9}$$

因此，条件$B_l(x)=0$定义了模态$l\in L$下不安全状态$\mathrm{Unsafe}(l)$和安全状态之间的屏障．给定模态$l\in L$，则可将可恢复集合记为$\mathrm{Recov}(l)=\{\boldsymbol{x}\mid\boldsymbol{x}\in I(l)\wedge B_l(\boldsymbol{x})=0\}$.

基于屏障证书的决策模块切换条件为：给定系统的当前状态s，若同时满足条件(10)和条件(11)，则决策模块选择复杂控制器；否则，决策模块选择安全控制器．

$$R(\boldsymbol{H}_{\mathrm{CC}},s,\leqslant\Delta t)\cap S_{\mathrm{u}}=\varnothing; \tag{10}$$

$$R(\boldsymbol{H}_{\mathrm{CC}},s,=\Delta t)\subseteq\{\mathrm{Recov}(l)\mid l\in L\} \tag{11}$$

与基于可达集计算的方法相比，基于屏障证书的方法不使用显式计算CPS在复杂控制器或安全控制器作用下的所有可达状态集合$R(\boldsymbol{H}_{\mathrm{SC}},U,\infty)$或$R(\boldsymbol{H}_{\mathrm{CC}},U,\infty)$，而是基于屏障证书合成系统的可恢复状态$\{\mathrm{Recov}(l)\mid l\in L\}$．同样可以证明，如果系统初始状态是安全的，那么在上述决策逻辑下，系统仍将处于安全状态空间内．

3　Simplex架构的改进与拓展

Simplex架构自被提出以来已经历了不同维度的拓展．PHAN等[11]针对组件化的CPS运行时安全性保障，提出基于组件的simplex架构(component-based simplex architecture，CBSA)．该架构能够对多个组件进行独立的安全保障，从而避免了将多个组件视为一个单一的整体，进行安全保障所带来的复杂性；通过应用假设-保证推理(assume-guarantee reasoning，AGR)技术对多组件决策模块进行设计，实现了系统安全性保障的组合推理；能够实现协同切换(coordinated switching)，即在多个组件存在协作的情况下，若其中一个组件进行了控制器切换，其他协作组件也可能需要进行相应的切换．

随后，PHAN等[12]又针对基于神经网络的复杂控制器，对simplex架构进行了拓展，提出neural simplex架构(图3)．该架构在神经网络控制器、安全控制器和决策模块的基础上，增加了一个自适应模块，用于获取对神经网络控制器的训练数据，使其能够在提升复杂控制器安全性的同时还能优化神经网络的性能．但是，该方法需获取神经网络控制器的内部结构等信息，且只能对单一控制器组件进行防护．但是，文献[12]并未给出决策单元中切换条件的系统性构造方法，且案例中的决策单元采用的是简单的、确定性的切换条件．

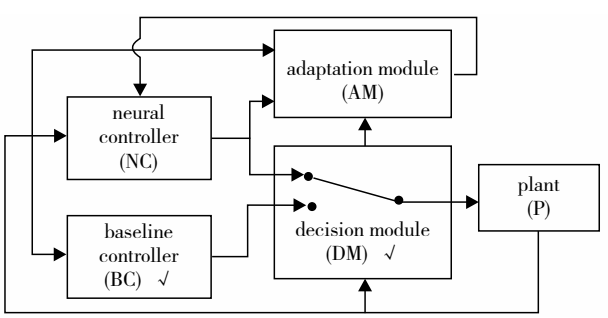


图3　Neural simplex架构[12]

Fig. 3　Neural simplex architecture[12].

在经典的simplex架构中决策模型输出的通常是简单的二元值，分别代表复杂控制器或安全控制器．这种控制器之间的直接切换，容易带来切换控制的抖动问题．RAMAKRISHNA等[13]讨论了简单

http://journal.szu.edu.cn

加权策略和上下文有关加权策略两种加权策略，通过强化学习方法对权值进行动态计算．为提高切换控制的平滑性，同时避免在多个控制器之间频繁切换，RAMAKRISHNA等[14]对经典simplex架构进行了加权扩展，提出了动态加权的simplex架构．该架构的决策模块将计算所有控制器输出的加权集合，从而在不影响系统性能的前提下，使切换控制过程达到最优平衡．

经典simplex架构侧重在单智能体(agent)系统或集中控制系统的运行时安全保障，MEHMOOD等[15-16]提出了一种面向多智能体系统(multi-agent system，MAS)运行时安全性保障的分布式simplex架构(distributed simplex architecture，DSA)，如图4．其中，$S_{i,m}$为智能体i感知得到的智能体m的状态．该simplex架构由多个智能体构成，每个智能体均由1个独立的simplex架构实例进行安全保障，所有simplex架构实例集成共同实现MAS的安全保障．该研究考虑的MAS中，每个智能体是同构且对称的．其中，同构指每个智能体的物理模型相同；对称指每个智能体所采用的simplex架构均具有相同的决策模块和安全控制器．对于多智能体相关的安全性约束，首先需将该安全性约束分解为单智能体的约束．例如，设两智能体agent_i和agent_j之间的安全性约束为

$$\begin{bmatrix}\boldsymbol{P}_{ij} & \boldsymbol{Q}_{ij}\end{bmatrix}\begin{bmatrix}\boldsymbol{u}_i\\ \boldsymbol{u}_j\end{bmatrix}\leqslant b_{ij} \tag{12}$$

其中，$\boldsymbol{P}_{ij}$和$\boldsymbol{Q}_{ij}$分别为智能体agent_i和agent_j的状态矩阵；$\boldsymbol{u}_i$和$\boldsymbol{u}_j$分别为智能体agent_i和agent_j的控制输入；b_{ij}为智能体agent_i和agent_j相关的安全性边界条件．上述安全性约束将被分解成了两个独立的约束，即

$$\begin{cases}\boldsymbol{P}_{ij}\boldsymbol{u}_i\leqslant b_{ij}/2\\ \boldsymbol{Q}_{ij}\boldsymbol{u}_j\leqslant b_{ij}/2\end{cases} \tag{13}$$

则两智能体可分别单独进行决策模块和安全控制器的设计．

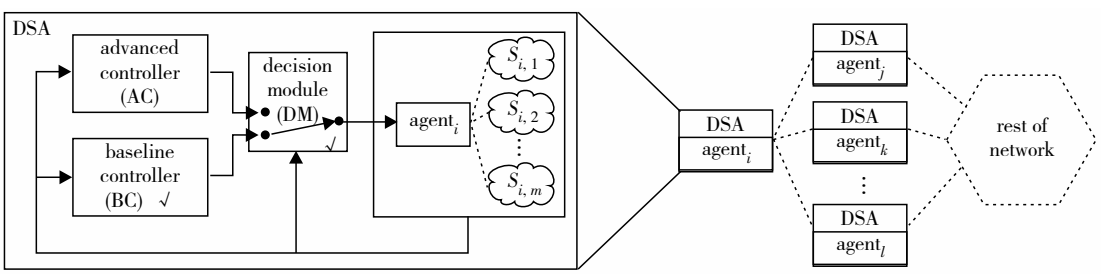


图4 面向多智能体系统的分布式simplex架构[15]

Fig. 4 Distributed simplex architecture of multi-agent system[15].

经典simplex架构要求安全控制器和决策模块必须能够被完全验证或证明．决策模块不仅监控系统的状态，还能够在复杂控制器可能导致安全违背的情况下，将系统的控制从复杂控制器切换到安全控制器．安全控制器的正确性和安全性应当能够被形式化证明，但这对实际的信息物理系统来说，目前仍是一项具有挑战的任务．许多信息物理系统的控制器，如基于模型预测控制技术的控制器，在实践中可能奏效，但其正确性和安全性仍难以被形式化证明，使它无法作为simplex架构的安全控制器．MEHMOOD等[17-18]提出的黑盒simplex架构(图5)，能够在不要求对安全控制器进行严格形式化验证或证明的情况下，依然保证全系统的安全性．黑盒simplex架构中的复杂控制器采用先进控制器(advanced controller)，而安全控制器为前向基线控制器(lookahead baseline controller)．该架构与经典simplex架构的主要区别在于，安全控制器输出的不再是控制指令，而是指令序列．此外，未经安全性证明的先进控制器需将其自身的指令提供给安全控制器，后者从复杂控制器当前时刻的指令开始，计算一个指令序列，该序列能够使受控系统(plant)及其底层控制器(low-level controller)处于可证明的安全状态．随后，经过安全性验证的决策模块会在运行时验证给定的指令是否满足安全性约束．如果满足安全性约束，决策模块将存储该指令，并在下一个时间步骤中作为备份命令使用，从而防止安全控制器无法生成相应的安全命令序列．

大部分simplex架构中决策模块的切换条件都是离线计算的，且只支持从复杂控制器向安全控制器的单向切换．LUO等[19]提出动态simplex架构，实现了多控制器之间的双向切换，且为避免“短视”的频繁的双向切换策略，基于半马尔可夫决策

http://journal.szu.edu.cn

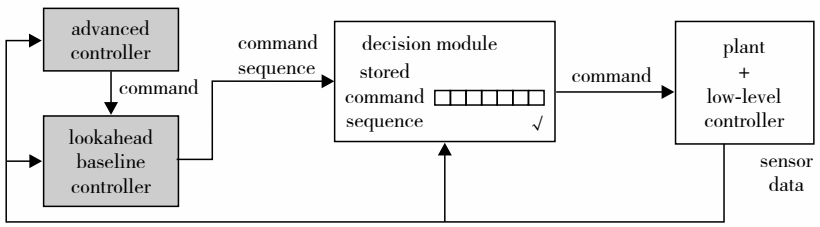


图5　黑盒simplex架构[17]

Fig. 5　Black-box simplex architecture[17].

过程实现了多控制器之间的双向切换，以动态的在线方式计算切换条件．该架构包括以下组件：①“短视”动作选择器；② 基于近似启发式搜索算法的“非短视”规划器；③ 用于监测环境参数的变化和传感器故障的运行时监控器(runtime monitors)．控制逻辑首先要找出正在执行控制的控制器．若复杂控制器正在执行控制，则切换器基于训练的代理模型激活“短视”动作选择器，以估算两个控制器的性能和安全奖励．如果复杂控制器奖励更高，则不执行切换．否则，将执行正向切换，即从复杂控制器切换至安全控制器；如果安全控制器正在执行控制，切换器就会激活规划器，决策模块使用一组生成模型来模拟系统未来可能的轨迹，并选择一个动作，使得期望的累积奖励最大化．

WHEATMAN等[20]提出一种面向分布式智能控制系统的simplex架构(图6)．该架构增加了黑盒监控器(black-box monitor)和白盒监控器(white-box monitor)．其中，黑盒监控器接收系统的状态信息，用于判断系统是否进入了错误的路径或状态；白盒监控器接收复杂控制器(文献主要考虑基于人工智能的控制器)的状态信息，用于判断复杂控制器输出的可信度；决策模块在判断切换条件时，将同时考虑上述黑盒监控器和白盒监控器的输出．通过结合两种监控器，该架构能够进一步发挥复杂控制器的效能，大幅提升系统的控制性能．

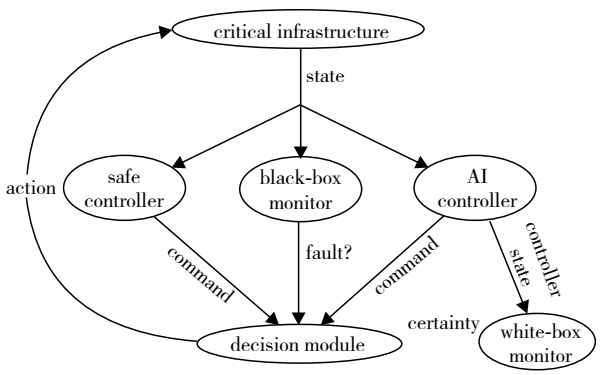


图6　面向分布式智能控制系统的simplex架构[20]

Fig. 6　Simplex architecture for distributed intelligent control systems[20].

经典simplex架构主要用于应对软件失效或故障等问题．为应对硬件故障问题，WANG等[21]提出了L1Simplex架构，在经典simplex架构基础上增加了安全监控器，用于实时监控系统的状态并预测硬件故障．随后，WANG等[22]又提出一种鲁棒的simplex架构(RSimplex)，其决策模块用来检测复杂控制器的故障，而新增的监控器用来检测硬件故障．该架构基于安全包络的切换条件来拓展基于监控器的切换条件，即当不确定性监控器计算的物理故障估计超过预设阈值，或系统状态位于安全包络的边界上且即将离开此包络时，系统将切换到安全控制器，如图7．

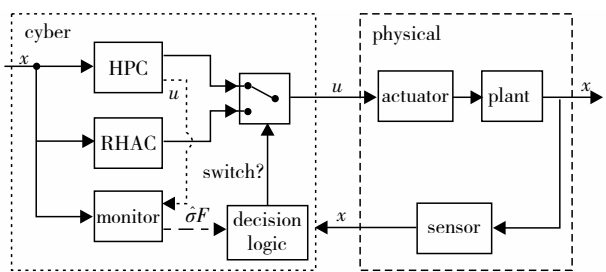


图7　鲁棒的simplex架构[22]

Fig. 7　Robust simplex architecture[22].

上述simplex架构主要是针对应用软件故障的运行时安全性保障，无法应对硬件平台、操作系统或中间件的故障．BAK等[23]提出了一种系统级的simplex架构，不仅能够应对应用软件中可能出现的逻辑故障，也能够应对与软硬件相关的资源共享故障．首先，基于体系架构分析与设计语言(architecture analysis and design language，AADL)构建simplex架构的逻辑模型，通过模型检测验证该模型的安全性，再使用自动化模型转换得到simplex架构的系统模型．该模型考虑了逻辑模型与硬件平台的映射关系，还可用于安全关键代码的自动生成．经典simplex架构主要考虑控制器的输出对

http://journal.szu.edu.cn

系统安全性的影响，并未考虑系统中不可靠组件之间的故障传播，例如，如何减少传感器感知数据失真对系统控制输出的影响等．CRENSHAW等[24]给出了一般化的simplex参考架构模型，阐明了如何在系统存在不可靠组件的情况下限制故障传播，以确保系统的安全性．YAO等[25]针对网络化控制系统提出NetSimplex架构，并在存在网络通信延迟和参数变化等情况下，讨论了网络化控制系统安全控制器和决策模块的设计方法．

ZHONG等[26]基于simplex架构的思想提出一种应用于随机CPS的运行时安全性保障方法，称为“沙盒(sandbox)”方法．为确保未经验证的复杂控制器(unverified controller)的安全性，该方法增加了监视器(supervisor)和安全顾问(safety advisor)，如图8．其中，安全控制器是依据系统所需满足的安全属性而设计的；监视器用来检测未经验证控制器可能存在的危险行为，并判断是否采用安全控制器的输出作为系统的控制，作用等价于simplex架构中的决策模块．该文献将随机CPS抽象为离散时间马尔可夫决策过程，并通过计算最优安全策略来设计安全控制器，其安全目标是在有限时间维度内使得系统进入不安全状态空间的概率最小．

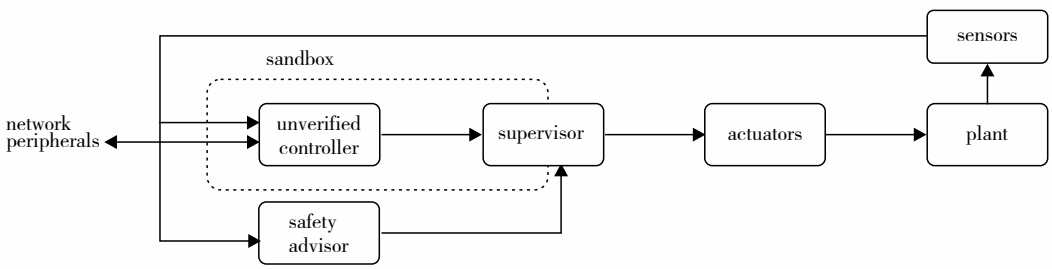


图8 面向随机CPS的“沙盒”simplex架构[26]

Fig. 8 Sandbox simplex architecture for random CPS systems[26].

4 Simplex架构的应用

早期的simplex架构主要应用于嵌入式控制系统的在线更新[27]，可以在不中断控制系统运行的情况下，实现嵌入式控制系统的安全在线更新．近年来，CPS大量采用深度学习等人工智能算法，使得学习能力成为系统的重要构成要素，这类系统通常被称作学习使能系统(learning enabled system)．其中，具备学习能力的组件被称作学习型组件(learning enabled component)．针对学习使能系统，特别是基于神经网络的飞机器滑行控制系统，COFER等[28]讨论了基于simplex架构的运行时保障方法，以实现系统的安全运行．

DESAI等[29]基于机器人操作系统(robot operating system，ROS)，提出一套称作SOTER的面向分布式机器人系统的运行时安全保障框架，用以支持采用运行时保证方法来设计安全的机器人软件系统．SOTER将机器人系统的各个功能组件定义为运行时保障模块，每个模块作为simplex架构的一个实现，包含了2个控制器(分别为高效控制器和安全控制器)和1个决策单元(实现两个控制器之间的切换逻辑)．

BIONDI等[30]指出，将深度神经网络(deep neural network，DNN)应用于安全攸关系统仍面临着诸多挑战：一方面，DNN存在安全可信漏洞，且易遭受安全攻击；另一方面，DNN的开发过程与安全攸关软件的标准开发规范不一致，使得研究人员难以对DNN模型开展安全认证．为此，他们提出一种基于simplex架构的安全攸关软件架构，用以确保DNN的安全可靠和可预测性．

VIVEKANANDAN等[31]提出一种基于simplex架构的无人机系统容错架构，以实现智能且可靠的系统控制．该架构采用两种不同的硬件平台．其中，高安全保障(high assurance，HA)平台具有较高的安全可靠性，用来运行simplex架构中的安全控制器和决策逻辑；高性能(high performance，HP)计算平台更侧重在运行性能优化，用来执行simplex架构中的复杂控制器．但是，高性能计算平台可能出现软硬件故障，如操作系统的故障或者计算单元的瞬态故障等．

运行时安全保障方法也应用在车辆辅助驾驶系统或无人驾驶系统领域[32-36]，其基本思路是在系统架构层面，通过增加监控组件，对不安全组件进行实时监控，当这类组件失效或出现异常时，监控组件能检测出异常并接管系统的运行控制，以确保系

http://journal.szu.edu.cn

统的安全性．其中，KWON 等[32]为提升端到端的自动驾驶运动规划的可靠性，提出了基于 simplex 架构的高安全可靠运动规划方法．MUSAU 等[33]将 simplex 架构应用于自动驾驶系统，并讨论了基于可达性分析的决策模型设计．MAO 等[36]针对自动驾驶系统跟车和换道等场景，设计了基于 simplex 架构的安全控制器．LAL 等[37]通过研究一种用于飞行器的安全混合控制器，实现了在基于线性二次调节器(linear quadratic regulator，LQR)的经典控制器和基于人工神经网络(artificial neural network，ANN)的智能控制器之间的安全切换．

目前，已有多种方法可用于设计 simplex 架构的决策模块．MILLER 等[38]提出一种用于 simplex 架构性能评估的框架 RTAEval．该框架包括了一个用于定义场景的模块，一个用于执行所定义的场景的模块，以及用于收集和可视化执行数据的模块．RTAEval 框架允许在使用较少代码的情况下，接入不同的智能体模型、ERsimplex 决策模块和用户自定义的度量等．

结　语

基于 simplex 架构的运行时安全性保障是一种在系统动态运行过程中确保系统满足相应安全属性的方法．该方法目前面临的主要挑战以及未来的研究方向包括：① 安全控制器的设计与实现的挑战，需要解决复杂 CPS 的形式化建模表征与安全性分析的难题，例如，高效且精确的可达状态空间的计算难题；② 复杂控制器作用下系统状态预测的挑战，特别是在具有不确定性输出的智能控制器的作用下，需要解决预测系统的行为状态并判断安全性约束是否得到满足的难题；③ simplex 架构在复杂 CPS 中的系统级应用与验证的挑战，特别是对感知-决策-控制闭环中，如何实现可组合、模块化的运行时安全性保障架构的难题．

基于 simplex 架构的运行时安全性保障凭借其简单而有效的特点，已经在工业界得到了广泛的应用．其中，Barron Associates 和 Lockheed Martin 公司成功地将该方法应用于复杂飞行器控制软件，以确保飞行器控制的安全性[39]．该方法也逐步被诸多行业标准所采纳，如在无人飞行器控制系统的标准 ASTM F3269-17[40]中，建议采用运行时安全性保证架构，通过在运行时监控系统中潜在的非预期功能，从而对复杂控制器作用下的系统行为提供安全性限界约束．随着以深度学习和大模型为代表的人工智能算法的快速发展，CPS 越来越多地采用人工智能算法来实现感知、决策和控制等功能，在提升系统智能自主能力的同时，也给系统的安全性保障带来了新的挑战．基于 simplex 架构的运行时安全性保障方法将会是解决智能自主 CPS 安全性保障问题的有效途径之一．

基金项目：国家自然科学基金资助项目(62106150)
作者简介：王强(18513688908@163.com)，军事科学院副研究员、博士．研究方向：形式化方法、模型验证、人工智能．
引　　文：王强，陈强，曹伟朋，等．基于单工架构的信息物理系统运行时安全性保障方法[J]．深圳大学学报理工版，2024，41(3)：253-263.

参考文献 / References：

[1] HUANG Chao, XU Shichao, WANG Zhilu, et al. Opportunistic intermittent control with safety guarantees for autonomous systems [C]// The 57th ACM/IEEE Design Automation Conference. Piscataway, USA: IEEE, 2020: 1-6.

[2] CLARK M, KOUTSOUKOS X, PORTER J, et al. A study on run time assurance for complex cyber physical systems [EB/OL]. (2013-4-18) [2024-5-11]. [S. l.]: Air Force Research Laboratory Aerospace Systems Directorate. https://leepike. github. io/pubs/RTA-CPS. pdf.

[3] SHA L. Using simplicity to control complexity [J]. IEEE Software, 2001, 18(4): 20-28.

[4] JOHNSON T T, BAK S, CACCAMO M, et al. Real-time reachability for verified simplex design [J]. ACM Transactions on Embedded Computing Systems, 2016, 15(2): 26.

[5] BAK S, GREER A, MITRA S. Hybrid cyberphysical system verification with simplex using discrete abstractions [C]// The 16th IEEE Real-Time and Embedded Technology and Applications Symposium. Piscataway, USA: IEEE, 2010: 143-152.

[6] BAK S, MANAMCHERI K, MITRA S, et al. Sandboxing controllers for cyber-physical systems [C]// The IEEE/ACM 2nd International Conference on Cyber-Physical Systems. Piscataway, USA: IEEE, 2011: 3-12.

[7] YANG Junxing, ISLAM M A, MURTHY A, et al. A simplex architecture for hybrid systems using barrier certificates [C]// Computer Safety, Reliability, and Security. Cham, Switzerland: Springer International Publishing, 2017: 117-13.

http://journal.szu.edu.cn

[8] DAMARE A, ROY S, SMOLKA S A, et al. A barrier certificate-based simplex architecture with application to microgrids [C]// Runtime Verification. Cham, Switzerland: Springer International Publishing, 2022: 105-123.

[9] KONG Hui, HE Fei, SONG Xiaoyu, et al. Exponential-condition-based barrier certificate generation for safety verification of hybrid systems [C]// The 25th International Conference on Computer Aided Verification. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013: 242-257.

[10] NIGAM V, TALCOTT C. Automating recoverability proofs for cyber-physical systems with runtime assurance architectures [C]// Theoretical Aspects of Software Engineering. Cham, Switzerland: Springer Nature Switzerland, 2023: 1-19.

[11] PHAN D, YANG Junxing, CLARK M, et al. A component-based simplex architecture for high-assurance cyber-physical systems [C]// The 17th International Conference on Application of Concurrency to System Design. Piscataway, USA: IEEE, 2017: 49-58.

[12] PHAN D T, GROSU R, JANSEN N, et al. Neural simplex architecture [C]// NASA Formal Methods. Cham, Switzerland: Springer International Publishing, 2020: 97-114.

[13] RAMAKRISHNA S, HARSTELL C, BURRUSS M P, et al. Dynamic-weighted simplex strategy for learning enabled cyber physical systems [J]. Journal of Systems Architecture, 2020, 111: 101760.

[14] RAMAKRISHNA S, DUBEY A, BURRUSS M P, et al. Augmenting learning components for safety in resource constrained autonomous robots [C]// The 22nd International Symposium on Real-Time Distributed Computing. Piscataway, USA: IEEE, 2019: 108-117.

[15] MEHMOOD U, D STOLLER S, GROSU R, et al. Collision-free 3D flocking using the distributed simplex architecture [M]// BARTOCCI E, FALCONE Y, LEUCKER M. Formal Methods in Outer Space: Essays Dedicated to Klaus Havelund on the Occasion of his 65th Birthday. Cham, Switzerland: Springer International Publishing, 2021: 147-156.

[16] MEHMOOD U, ROY S, DAMARE A, et al. A distributed simplex architecture for multi-agent systems [J]. Journal of Systems Architecture, 2023, 134: 102784.

[17] MEHMOOD U, BAK S, SMOLKA S A, et al. Safe CPS from unsafe controllers [C]// Proceedings of the Workshop on Computation-Aware Algorithmic Design for Cyber-Physical Systems. New York, USA: Association for Computing Machinery, 2021: 26-28.

[18] MEHMOOD U, SHEIKHI S, BAK S, et al. The black-box simplex architecture for runtime assurance of autonomous CPS [C]// NASA Formal Methods. Cham, Switzerland: Springer International Publishing, 2022: 231-250.

[19] LUO Baiting, RAMAKRISHNA S, PETTET A, et al. Dynamic simplex: balancing safety and performance in autonomous cyber physical systems [C]// Proceedings of the ACM/IEEE 14th International Conference on Cyber-Physical Systems (with CPS-IoT Week 2023). New York, USA: Association for Computing Machinery, 2023: 177-186.

[20] WHEATMAN B, CHEN J, SOOKOOR T, et al. RADICS: runtime assurance of distributed intelligent control systems [C]// The 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W). Piscataway, USA: IEEE, 2021: 182-187.

[21] WANG Xiaofeng, HOVAKIMYAN N, SHA L. L1simplex: fault-tolerant control of cyber-physical systems [C]// The ACM/IEEE International Conference on Cyber-Physical Systems. Piscataway, USA: IEEE, 2013: 41-50.

[22] WANG Xiaofeng, HOVAKIMYAN N, SHA L. RSimplex: a robust control architecture for cyber and physical failures [J]. ACM Transactions on Cyber-Physical Systems, 2018, 2(4): 27.

[23] BAK S, CHIVUKULA D K, ADEKUNLE O, et al. The system-level simplex architecture for improved real-time embedded system safety [C]// The 15th IEEE Real-Time and Embedded Technology and Applications Symposium. Piscataway, USA: IEEE, 2009: 99-107.

[24] CRENSHAW T L, GUNTER E, ROBINSON C L, et al. The simplex reference model: limiting fault-propagation due to unreliable components in cyber-physical system architectures [C]// The 28th IEEE International Real-Time Systems Symposium. Piscataway, USA: IEEE, 2007: 400-412.

[25] YAO Jianguo, LIU Xue, ZHU Guchuan, et al. Netsimplex: controller fault tolerance architecture in networked control systems [J]. IEEE Transactions on Industrial Informatics, 2013, 9(1): 346-356.

[26] ZHONG Bingzhuo, ZAMANI M, CACCAMO M. Sandboxing controllers for stochastic cyber-physical systems [C]// International Conference on Formal Modeling and Analysis of Timed Systems. Cham,

http://journal.szu.edu.cn

Switzerland: Springer International Publishing, 2019: 247-264.

[27] SETO D, KROGH B, SHA L, et al. The simplex architecture for safe online control system upgrades [C]// Proceedings of the American Control Conference. Piscataway, USA: IEEE, 1998: 3504-3508.

[28] COFER D, AMUNDSON I, SATTIGERI R, et al. Run-time assurance for learning-enabled systems [C]// NASA Formal Methods. Cham, Switzerland: Springer International Publishing, 2020: 361-368.

[29] DESAI A, GHOSH S, SESHIA S A, et al. SOTER: a runtime assurance framework for programming safe robotics systems [C]// The 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks. Piscataway, USA: IEEE, 2019: 138-150.

[30] BIONDI A, NESTI F, CICERO G, et al. A safe, secure, and predictable software architecture for deep learning in safety-critical systems [J]. IEEE Embedded Systems Letters, 2020, 12(3): 78-82.

[31] VIVEKANANDAN P, GARCIA G, YUN H, et al. A simplex architecture for intelligent and safe unmanned aerial vehicles [C]// The 22nd International Conference on Embedded and Real-Time Computing Systems and Applications. Piscataway, USA: IEEE, 2016: 69-75.

[32] KWON S K, SEO J H, LEE J W, et al. An approach for reliable end-to-end autonomous driving based on the simplex architecture [C]// The 15th International Conference on Control, Automation, Robotics and Vision. Piscataway, USA: IEEE, 2018: 1851-1856.

[33] MUSAU P, HAMILTON N, LOPEZ D M, et al. An empirical analysis of the use of real-time reachability for the safety assurance of autonomous vehicles [EB/OL]. [2022-05-03]. https://arxiv. org/abs/2205. 01419.

[34] HEN Shengduo, SUN Yaowei, LI Dachuan, et al. Runtime safety assurance for learning-enabled control of autonomous driving vehicles [C]// The International Conference on Robotics and Automation. Piscataway, USA: IEEE, 2022: 8978-8984.

[35] WANG Qiang, KOU Guang, CHEN Longquan, et al. Runtime assurance of learning-based lane changing control for autonomous driving vehicles [J]. Journal of Circuits, Systems and Computers, 2022, 31(14): 2250249.

[36] MAO Yanbing, GU Yuliang, HOVAKIMYAN N, et al. S $\mathcal{L}_1$-simplex: safe velocity regulation of self-driving vehicles in dynamic and unforeseen environments [J]. ACM Transactions on Cyber-Physical Systems, 2023, 7 (1): 2.

[37] LAL R, MCKINNIS A, HAUPTMAN D, et al. Formally verified switching logic for recoverability of aircraft controller [C]// Computer Aided Verification. Cham, Switzerland: Springer International Publishing, 2021: 566-579.

[38] MILLER K, ZEITLER C K, SHEN W, et al. RTAEval: a framework for evaluating runtime assurance logic [C]// Automated Technology for Verification and Analysis. Cham, Switzerland: Springer Nature Switzerland, 2023: 302-313.

[39] SCHIERMAN J D, DEVORE M D, RICHARDS N D, et al. Runtime assurance framework development for highly adaptive flight control systems [R]. Barron Associates, Inc. Charlottesville, 2015.

[40] ASTM International. Standard practice for methods to safely bound flight behavior of unmanned aircraft systems containing complex functions: ASTM F3269-17 [S]. West Conshohocken, USA: ASTM International, 2017.

【中文责编：英子；英文责编：木柯】

http://journal.szu.edu.cn