状态不可观测的信息物理融合系统运行时验证

房丙午[1,2],黄志球[1],王 勇[1],李 勇[1]
(1. 南京航空航天大学计算机科学与技术学院,江苏南京 210016;2. 安徽财贸职业学院云桂信息学院,安徽合肥 230061)

摘 要: 确保信息物理融合系统(Cyber-Physical System,CPS)运行时行为正确性是至关重要的,尤其在航空航天、汽车、核电和医疗等安全攸关领域.本文针对具有随机行为且状态不可观测的 CPS,提出一种基于隐马尔科夫模型的运行时安全性验证方法.首先构造状态不可观测的 CPS 运行时安全性验证框架,该框架通过隐马尔科夫模型表示系统,使用确定性有限自动机规约系统安全属性的否定,两者的乘积自动机作为运行时监控器,从而将 CPS 运行时安全性验证问题约简到监控器上的概率推断问题.然后,提出一种增量迭代安全性验证算法以及反例生成算法.实验结果表明本文算法和粒子滤波算法相比预测错误率下降了近 20%,并且当系统违背安全属性时,本文的方法能给出反例.

关键词: 信息物理融合系统;运行时验证;隐马尔科夫模型;确定性有限自动机;安全性;反例
中图分类号: TP311 **文献标识码:** A **文章编号:** 0372-2112 (2018)12-2824-08
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2018.12.002

Runtime Verification of States Unobservable Cyber-Physical System

FANG Bing-wu[1,2], HUANG Zhi-qiu[1], WANG Yong[1], LI Yong[1]
(1. *College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu* 210016, *China*;
2. *School of YunGui Information, Anhui Finance and Trade Vocational College, Hefei, Anhui* 230601, *China*)

Abstract: Ensuring the correct behavior of cyber physical systems (CPS) at runtime is critical, and it is more so in safety-critical area, such as aerospace, automotive, nuclear power and medical. A method of runtime safety verification based on hidden Markov model (HMM) is proposed for the CPS with stochastic behavior and unobservable states. First, a runtime safety verification framework of the CPS is constructed where the system model is represented by HMM, and the negation of safety property is specified by deterministic finite automaton (DFA), and then the product automata of DFA and HMM is used as a runtime monitor. Thus, the problem of CPS runtime safety verification is reduced into a probabilistic inference problem on the monitor. Second, an incremental iterative safety verification algorithm and a counterexample generation algorithm are proposed. The experimental results show that the prediction error rate of the safety verification algorithm is nearly 20% lower than that of the particle filter algorithm. When the system violates the safety property, the proposed method can product a counterexample which the particle filtering method cannot do.

Key words: cyber-physical system; runtime verification; hidden Markov model; deterministic finite automaton; safety; counter-example

1 引言

CPS 在航空航天、汽车、核电和医疗等安全攸关的领域有着广阔的应用前景[1,2],因此,确保 CPS 运行时行为正确性是至关重要的.运行时验证通过监控系统运行时的执行轨迹来验证系统是否满足属性规约[3],是保证 CPS 运行时行为正确性的有效方法.文献[3~5]针对实时性要求较高的 CPS,验证与时间相关的功能正确性.文献[6]针对 CPS 运行时失效行为进行监控.在 CPS 运行时安全性验证方面,文献[7]给出在不可靠的网络通信下,对 CPS 的安全属性不变式进行运行时验证,不变式是一类基于状态的使用谓词逻辑描述的

收稿日期:2018-05-03;修回日期:2018-07-15;责任编辑:覃怀银
基金项目:国家重点研发计划(No. 2016YFB1000802);国家自然科学基金资助项目(No. 61772270);安徽省高校自然科学基金重点项目(No. KJ2017A859);安徽省高校学科(专业)优秀拔尖人才学术资助计划(No. gxbjZD32)

安全属性,该方法不能验证无限状态时序序列的安全属性. 文献[8]使用混成自动机建模自适应巡航控制系统,使用 CHARON 语言规约安全属性,验证了自适应巡航控制系统运行时安全性. 虽然研究人员提出了多种 CPS 运行时验证的方法,但目前的研究存在以下不足:(1)现有的研究都假设系统运行时的状态可以直接观测的,实际上大多数复杂的 CPS 在运行时其状态是不可观测的[9,10]. (2)在运行时安全性验证中,当系统违背安全属性时,而没有给出相应的反例[11].

针对上述不足,本文提出了状态不可观测的具有随机行为的 CPS 运行时安全性验证方法. 由于系统状态不可观测,因此,采用隐马尔科夫模型(Hidden Markov Model,HMM)来建模系统. 使用确定性有限自动机(Deterministic Finite Automaton,DFA)规约安全属性的否定,安全属性监控器由 HMM 和 DFA 的乘积自动机表示,从而将状态不可观测的 CPS 运行时安全性验证问题约简到监控器上的概率推断问题.

2　运行时验证框架

本文的验证问题可描述为:给定系统状态转换模型 TS,系统安全属性 φ_{safe} 以及系统运行时的观测序列 $o_1,\cdots,o_t$,目标是(1)计算在 t 时刻系统满足安全属性的概率 $\Pr_t(\text{TS}\models\varphi_{\text{safe}}\mid o_1,\cdots,o_t;\text{TS})$,(2)当系统违背安全属性时,求解系统最大可能的执行路径 $\underset{s_1,s_2,\cdots,s_t}{\operatorname{argmax}}\Pr(s_1,s_2,\cdots,s_t\mid o_1,o_2,\cdots,o_t;\text{TS})$ 作为系统违背安全属性的反例. 针对该问题,图 1 给出了本文验证方法的框架.

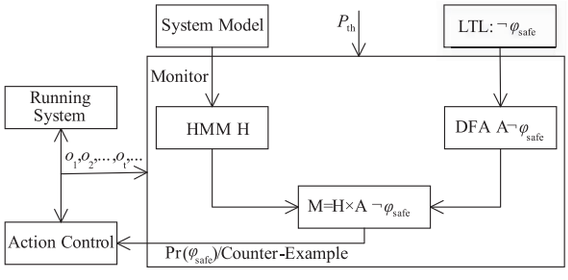


图1　CPS运行时安全性验证框架

图 1 中,虚线框中是运行时验证的监控器单元,监控器由系统模型和属性自动机的乘积来表示,其中,系统模型由 HMM 表示,属性自动机由 DFA 表示. 监控器的输入是运行时系统的观测序列和系统安全性的阈值 P_{th},监控器输出是当前系统满足安全性的概率,当系统安全性概率低于阈值时,输出反例. 系统模型的构造可以来自于系统设计模型[9],或通过领域专家指定系统状态以及状态的观测值,然后模拟系统运行获取系统观测数据并学习一个系统模型[10]. 系统的安全属性采用线性时序逻辑描述,使用 DFA 来识别安全属性的否定也就是识别系统运行中违反安全性的有限前缀.

3　相关定义与监控器模型

3.1　相关定义

本文采用 HMM 建模系统,采用 DFA 描述系统安全属性,下面给出相关的定义.

定义 1　一个 DFA 是如下形式的五元组 $A=(Q,\Sigma,\delta,q_0,F)$. 其中,

(1)Q 是有限的,非空的状态集合,

(2)Σ 是输入字母表,$\Sigma=2^{\text{AP}}$,AP 为原子命题集合,

(3)$\delta:Q\times\Sigma\to Q$ 是一个状态转换函数,

(4)$q_0\in Q$ 是一个初始状态,

(5)$F\subseteq Q$ 是接受状态的集合.

如果在 A 中 $\forall q\in Q,\forall B\in 2^{\text{AP}}$ 都有 $|\delta(q,B)|=1$,那么 A 是一个完全的 DFA. 一个完全的 DFA 对于每个输入字 $\sigma\in(2^{\text{AP}})^{\omega}$ 都存在一个唯一的运行与之对应.

定义 2　一个离散时间马尔科夫链(Discrete-Time Markov Chain,DTMC)是一个四元组 $C=(S,\boldsymbol{P},\iota_{\text{init}},L)$. 其中:

(1)S 表示可数非空的状态集合,

(2)$\boldsymbol{P}:S\times S\to[0,1]$ 是状态转换概率分布函数,对每一个状态 $s\in S$,$\sum_{s'\in S}\boldsymbol{P}(s,s')=1$,

(3)$\iota_{\text{init}}:S\to[0,1]$ 是初始状态概率分布函数,$\sum_{s\in S}\iota_{\text{init}}(s)=1$,

(4)$L:S\to 2^{\text{AP}}$ 是标签函数.

定义 3　一个 HMM 是一个三元组 $H=(C,O,\mu)$. 其中:

(1)C 是一个 DTMC,

(2)O 是系统可观测符号的有限集合,即系统模型输出的有限集合,

(3)$\mu:S\times 2^{O}\to[0,1]$ 是一个观测符号概率分布函数,对于 $\forall s\in S$,$\sum_{o\in 2^{O}}\mu(s,o)=1$.

$\mu(s,o)$ 或 $\mu_s(o)$ 表示系统在状态 s 时产生观测 o 的概率,相同的观测可能由几个不同的系统状态产生,因此,在已知目前观测的情况下系统的状态是不确定的.

3.2　监控器模型

本文将安全属性监控器定义为一个 A 和 H 的乘积自动机,系统安全属性是基于可观测符号描述,因此,A 的输入字母表 $\Sigma\subseteq 2^{O}$.

定义 4　安全属性监控器定义为一个 A 和 H 的乘积自动机 $M=A\otimes H=(S',\boldsymbol{P}',\iota'_{\text{init}},F',O,\mu')$. 其中,

(1)$S'=S\times Q$,监控器状态集合;

(2)$\boldsymbol{P}'(\langle s,q\rangle,\langle s',q'\rangle)=$

$$\begin{cases}\boldsymbol{P}(s,s')\sum_{t\in T}\mu_s(t),\text{if } T\neq\varnothing\\ 0,\text{otherwise}\end{cases}$$

表示状态转换概率分布,

其中,$T=O(s')\cap\delta^{-1}(q,q')$,$O(s')\subseteq 2^{O}$ 表示在状态 s' 观测字母的集合,$\delta^{-1}(q,q')\subseteq\Sigma$ 表示 A 中从状态 q 到 q' 的输入字母的集合;

(3)$F'=S\times F$,监控器接受状态集合;

(4)$\iota'_{\text{init}}(\langle s,q\rangle)=\begin{cases}\iota_{\text{init}}(s)\sum_{t\in T}\mu_s(t), & \text{if } T\neq\varnothing\\ 0, & \text{otherwise}\end{cases}$,监控器初始状态概率分布,其中,$T=O(s)\cap\delta^{-1}(q_0,q)$;

(5)$\mu'(\langle s,q\rangle,o)=\mu_s(o)$,$o\in T$,表示状态$\langle s,q\rangle$观测符号概率分布.

根据定义可知,监控器 M 本质上是一个含有接受状态集的 HMM,并且 M 的状态集 $S'=S\times Q$ 中存在一些从初始状态不可达的状态以及与监控属性无关的状态,因此,要从 M 中移除这些状态. 采用宽度优先算法来构造 M,首先构造 M 的所有初始状态. 然后,从初始状态出发,采用宽度优先搜索算法来生成 M 的状态. 根据定义 4,监控器构造时间复杂度为:$O(|O|\times(|S|\times|O|)^2)$,其中,$|O|$表示系统可观测符号集合的势.

4 安全性验证和反例生成算法

监控器接受当前的系统观测,更新信念状态并计算在已知整个有穷观测序列下的系统安全性概率. 基于监控器模型,本节提出一种增量迭代安全性验证算法(Incremental Iterative Verification,IIV)算法以及系统违背安全性需求时的反例生成算法.

4.1 安全性验证算法

定义 5 信念状态(belief state)是指在时刻 t,已知系统的观测序列 $o_1,o_2,\cdots,o_t$,监控器 M 处在状态 s 的概率. 形式化表示为:$b_t(s)=\Pr(s|o_1,o_2,\cdots,o_t;M)$,$\forall s\in S'$,$\sum_{s\in S'}b_t(s)=1$.

由于观测值是从系统的不同状态发出的,因此无法知道系统目前所处的准确状态也就是 M 所处的状态,但是根据系统的历史观测,能够计算出系统的信念状态. 由定义 5 可知,系统的信念状态也可由 HMM 的前向或后向概率表示[12],但前(后)向算法需要处理从初始观测开始到当前时刻的观测序列,当一个新的观测值到达时,为了更新系统信念状态,前(后)向算法每次都要处理从当前观测之前的整个观测序列. 随着系统观测序列的增长,前(后)向算法消耗的内存和计算时间也将增长. 为了能够快速地反应系统信念状态的变化,本节提出 IIV 算法,一旦收到一个新的观测值,算法立即更新系统信念状态,该算法以在线递推方式计算系统信念状态,在 $t=1$ 时,系统信念状态 $b_1(s)$ 取决于初始状态分布和初始观测值 o_1 根据 HMM 的齐次马尔科夫假设可知,系统在 t 时刻的信念状态 $b_t(s)$ 只依赖于 $b_{t-1}(s)$时刻的状态和当前的观测 o_t. 引理 1 给出了 $b_t(s)$和 $b_{t-1}(s)$之间的关系.

引理 1 给定系统的历史观测序列 $o_1,o_2,\cdots,o_t,M$ 的信念状态更新函数 $b_t(s)$可表示为如下递推公式:

$$b_t(s)=\begin{cases}\dfrac{\iota'_{\text{init}}(s)\mu'_s(o_1)}{z_1}, & t=1 \quad (1)\\ \dfrac{\sum_{s'\in S'}b_{t-1}(s')\boldsymbol{P}'(s',s)\mu'_s(o_t)}{z_t}, & t>1 \quad (2)\end{cases}$$

其中,$z_1=\sum_{s\in S'}\iota'_{\text{init}}(s)\mu'_s(o_1)$,$z_t=\sum_{s\in S'}\sum_{s'\in S'}b_{t-1}(s')\boldsymbol{P}'(s',s)\mu'_s(o_t)$.

证明 根据信念状态的定义和贝叶斯定理可得 $t=1$ 时信念状态:

$$b_1(s)=\Pr(s|o_1)=\frac{\Pr(s)\Pr(o_1|s)}{\sum_{s\in S'}\Pr(s,o_1)}=\frac{\iota'_{\text{init}}(s)\mu'_s(o_1)}{\sum_{s\in S'}\iota'_{\text{init}}(s)\mu'_s(o_1)}$$

假设在 $t>1$ 时,已知 $b_{t-1}(s)$和当前的观测 o_t,则系统在 t 时刻的信念状态:

$$\begin{aligned}b_t(s)&=\Pr(s|o_1,\cdots,o_t)\\&=\Pr(s,o_t|o_1,\cdots,o_{t-1})\frac{\Pr(o_1,\cdots,o_{t-1})}{\Pr(o_1,\cdots,o_t)}\end{aligned}\quad(3)$$

其中,

$$\begin{aligned}\Pr(s,o_t|o_1,\cdots,o_{t-1})&=\sum_{s'\in S'}\Pr(s,s',o_t|o_1,\cdots,o_{t-1})\\&=\sum_{s'\in S'}\Pr(s'|o_1,\cdots,o_{t-1})\\&\qquad\Pr(s,o_t|s',o_1,\cdots,o_{t-1})\\&=\sum_{s'\in S'}b_{t-1}(s')\boldsymbol{P}'(s',s)\mu'_s(o_t)\end{aligned}\quad(4)$$

$$\frac{\Pr(o_1,\cdots,o_{t-1})}{\Pr(o_1,\cdots,o_t)}=\frac{1}{\sum_{s\in S'}\sum_{s'\in S'}\Pr(s,s',o_t|o_1,\cdots,o_{t-1})}=\frac{1}{\sum_{s\in S'}\sum_{s'\in S'}b_{t-1}(s')\boldsymbol{P}'(s',s)\mu'_s(o_t)}=\frac{1}{z_t}\quad(5)$$

将式(4)和式(5)的结果代入式(3)即可得出式(2).

定理 1 令 φ_{safe}是系统的安全属性,A 是一个接受 φ_{safe}所有坏前缀集合的 DFA,H 是表示系统模型的 HMM,监控器 $M=A\otimes H$,$o_1,o_2,\cdots,o_t$,是系统运行时的观测序列. 那么在 t 时刻系统安全性的概率:

$$\begin{aligned}\Pr_t(H\models\varphi_{\text{safe}}|o_1,\cdots,o_t)&=\Pr_t(M\models\varphi_{\text{safe}}|o_1,\cdots,o_t)\\&=1-\sum_{s\in F'}b_t(s).\end{aligned}$$

证明 (1) 由于 A 是完全确定性自动机,所以 A 不

会影响监控器 M 中的概率计算. 另外,如果 H 中某一状态不包含在 M 的组合状态中,则该状态与监控属性无关,也不会影响监控器 M 中的概率计算. 所以,

$$\Pr_t(H\models\varphi_{\text{safe}}\mid o_1,\cdots,o_t)=\Pr_t(M\models\varphi_{\text{safe}}\mid o_1,\cdots,o_t).$$

(2)首先,自动机 A 接受安全属性的否定,也就是接受的是坏前缀,所以 M 中的接收状态集 F' 是系统的不安全状态. 其次,在 M 构造过程中删除了从初始状态不可达状态以及与监控属性无关的状态,而这些状态相对于 φ_{safe} 来说是安全状态,因此 t 时刻系统安全性的概率 $\Pr_t(M\models\varphi_{\text{safe}}\mid o_1,\cdots,o_t)=1-\sum_{s\in F'}b_t(s)$.

根据引理 1 和定理 1,算法 1 给出根据系统的当前观测 o_t 计算系统安全性概率的算法.

算法 1　安全性验证算法

输入:M,当前的观测 o_t
输出:$\Pr_t(M\models\varphi_{\text{safe}}\mid o_1,\cdots,o_t)$

```
1:  if (t = =1) then
2:    for each s∈S′do
3:      b1′(s) =ι′init(s)μ′s(o1);
4:      z1 + =b1′(s);
5:    end for
6:    for each s∈S′do
7:      b1(s) =b1′(s)/z1;
8:    end for
9:  else
10:   for each s∈S′do
11:     for each s′∈S′do
12:       b′t(s) + =bt-1(s′)P′(s′,s)μ′s(ot);
13:     end for
14:     z1 + =b′t(s);
15:   end for
16:   for each s∈S′do
17:     bt(s) =b′t(s)/zt;
18:   end for
19: end if
20: return 1 - Σ_{s∈F′} bt(s);
```

在算法 1 中,语句 1 ~ 9 是根据式(1)进行信念状态初始化,仅在算法启动时执行一次. 语句 10 ~ 15 是计算式(2)的分子部分和分母 z_t,语句 16 ~ 18 是计算 t 时刻的信念状态,语句 20 返回系统安全性概率.

从算法 1 中可以看出语句 12 执行次数最多,可以通过分析语句 12 最大执行次数来分析算法的最坏时间复杂度. 由语句 10 ~ 15 可知,语句 12 的执行次数为 $|S'|^2$,因此,算法 1 的时间复杂度为 $O(|S'|^2)$.

4.2　反例生成算法

在随机模型检测中,当系统属性被违背时,对反例的产生方法进行了大量的研究[11]. 在运行验证中,系统当前执行路径只有一条,当系统违背安全属性时,该路径就是当前运行的一个反例. 在状态不可观测的情况下,通过观测序列推断系统最大可能的执行路径,因此,当系统违背安全属性时,本文将当前观测序列下的最大可能执行路径作为系统运行时的一个反例. 定义 6 给出了最大可能执行路径的定义.

定义 6　已知系统观测序列 $o_1,o_2,\cdots,o_T$,当系统违背安全属性时,监控器 M 的最大可能执行路径 $(s_1^*,s_2^*,\cdots,s_T^*)=\underset{s_1,s_2,\cdots,s_T}{\operatorname{argmax}}\Pr(s_1,s_2,\cdots,s_T,o_1,o_2,\cdots,o_T)$,$s_i\in S'$,$i=1,2,\cdots,T$ 其中,$s_i\in S'$,$i=1,2,\cdots,T$.

由定义 6 可知,求解 M 中长度为 T 的最大可能执行路径的时间复杂度为 $O(|S'|^T)$,显然不能满足运行时验证需求. 对于该问题可以采样随机优化算法近似求解[13],本文采用动态规划算法来求解,下面的引理给出了系统反例的生成方法.

引理 2　已知监控器 M 和系统运行时观测序列 $o_1,o_2,\cdots,o_T$,(1)求解最大可能执行路径 $\underset{s_1,s_2,\cdots,s_T}{\operatorname{argmax}}\Pr(s_1,s_2,\cdots,s_T\mid o_1,o_2,\cdots,o_T)$ 等价于求解 $\underset{s_1,s_2,\cdots,s_T}{\operatorname{argmax}}\Pr(s_1,s_2,\cdots,s_T,o_1,o_2,\cdots,o_T)$. (2)定义在时刻 $t(t<T)$ 状态为 s 的所有单个路径中概率最大值 $\gamma_t(s)=\max_{s_1,s_2,\cdots,s_{t-1}}\Pr(s_1,s_2,\cdots,s_{t-1},s,o_1,o_2,\cdots,o_t)$,则在时刻 $t+1$ 状态为 s 的所有单个路径中概率最大值 $\gamma_{t+1}(s)=\max_{s_t\in S'}[\gamma_t(s_t)\boldsymbol{P}'(s_t,s)]\mu'_s(o_{t+1})$.

证明　(1)由贝叶斯定理可知,$\Pr(s_1,s_2,\cdots,s_T\mid o_1,o_2,\cdots,o_T)=\dfrac{\Pr(s_1,s_2,\cdots,s_T,o_1,o_2,\cdots,o_T)}{\Pr(o_1,o_2,\cdots,o_T)}$,由于 $\dfrac{1}{\Pr(o_1,o_2,\cdots,o_T)}$ 是规范化因子(常数),所以,$\Pr(s_1,s_2,\cdots,s_T\mid o_1,o_2,\cdots,o_T)\propto\Pr(s_1,s_2,\cdots,s_T,o_1,o_2,\cdots,o_T)$ 由此可得引理 2 中(1)成立.

(2)根据定义,$\gamma_{t+1}(s)=\max_{s_1,s_2,\cdots,s_t}\Pr(s_1,s_2,\cdots,s_t,s,o_1,o_2,\cdots,o_{t+1})$,根据贝叶斯定理,$\Pr(s_1,s_2,\cdots,s_t,s,o_1,o_2,\cdots,o_{t+1})=\Pr(s_1,s_2,\cdots,s_t,o_1,o_2,\cdots,o_t)\Pr(s\mid s_t)\Pr(o_{t+1}\mid s)$.

对上式求最大值并根据动态规划原理可得:

$$\begin{aligned}&\max_{s_1,s_2,\cdots,s_t}\Pr(s_1,s_2,\cdots,s_t,o_1,o_2,\cdots,o_t)\Pr(s\mid s_t)\Pr(o_{t+1}\mid s)\\&=\max_{s_t\in S'}\Big[\max_{s_1,s_2,\cdots,s_{t-1}}\Pr(s_1,s_2,\cdots,s_t,o_1,o_2,\cdots,o_t)\Pr(s\mid s_t)\Big]\cdot\Pr(o_{t+1}\mid s)\\&=\max_{s_t\in S'}[\gamma_t(s_t)\boldsymbol{P}'(s_t,s)]\mu'_s(o_{t+1}).\end{aligned}$$

根据引理 2,算法 2 给出了系统反例生成算法,算法 2 中的 $\xi_t(s)$ 表示在时刻 t 状态为 s 的所有单个路径中概率最大的路径的第 $t-1$ 个状态,$\text{Project}(s_1^*,s_2^*,\cdots,s_t^*,H)$ 将 M 的最大可能执行路径投影到系统模型 H 上从而获得系统的最大可能执行路径.

算法 2 反例生成算法

输入:$M,o_1,o_2,\cdots,o_T$
输出:系统反例
1:根据算法 1 计算系统在 $t-T+1$ 时刻信念状态:$b_{t-T+1}(s)$, $\forall s \in S'$;
2: for each $s \in S'$do
3: $\gamma_1(s)=b_{t-T+1}(s)\mu'_s(o_1)$;
4: $\xi_1(s)=\varnothing$;
5: end for
6: for $t=2$ To T do
7: for each $s \in S'$do
8: $\gamma_t(s)=\max\limits_{s_{t-1}\in S'}[\gamma_{t-1}(s_{t-1})\boldsymbol{P}'(s_{t-1},s)]\mu'_s(o_t)$;
9: $\xi_t(s)=\arg\max\limits_{s_{t-1}\in S'}[\gamma_{t-1}(s_{t-1})\boldsymbol{P}'(s_{t-1},s)]$;
10: end for
11: end for
12: $s_T^*=\arg\max\limits_{s\in S'}\gamma_T(s)$;
13: for $t=T-1$ to 1 do
14: $s_t^*=\xi_{t+1}(s_{t+1}^*)$;
15: end for
16: return Project$(s_1^*,s_2^*,\cdots,s_t^*,H)$;

语句 2 ~ 5 根据当前的信念状态初始化 $\gamma_1(s)$ 和 $\xi_1(s)$,语句 6 ~ 11 递推计算在 t 时刻到达状态 s 最大可能路径的概率以及概率最大路径的前驱,语句 12 获取最大可能路径的终点,语句 13 ~ 15 回溯获取最大可能执行路径,语句 16 返回系统最大可能执行路径. 从算法 2 中可以看出语句 8 执行次数最多,最大执行次数是:$T\times|S'|^2$,因此,算法 2 的时间复杂度为 $O(T\times|S'|^2)$. 算法 2 生成的反例长度取决于输入观测序列长度 T.

5 案例分析与实验

本节通过汽车安全车道保持系统[14](SAFELANE)来验证本文方法的有效性和计算效率. 汽车控制系统是一个安全攸关的 CPS,其中 SAFELANE 是商用和家用汽车主动安全系统的重要组成部分,SAFELANE 是防止驾驶员疏忽或疲劳而造成的无意识的车道偏离. 当系统工作时,一旦探测到意外的车道偏离时,系统向驾驶员发出警告或主动校正车道使其恢复到正常状态,为了能够执行车道校正,系统必须获得驾驶车辆的权限. 系统能够被驾驶员的操作行为覆盖(如驾驶员转向操作)或通过人机交互接口关闭. SAFELANE 由视觉传感、决策控制、执行、转向控制、速度控制等子系统组成,其中,决策控制子系统采用冗余计算方式,主系统和热备份系统并行计算,如果两者的计算结果不一致,则系统进入故障状态.

5.1 实验设置

针对 SAFELANE 主动控制模式进行安全属性验证,图 2 给出了该模式对应的 HMM,其中,s_1 表示系统处在空闲状态,s_2 表示系统处在决策状态,s_3 表示系统处在车道校正状态,s_4 表示主系统和热备份系统计算结果不一致状态,s_5 表示系统处在未知故障状态. 系统可观测符号,IL(In Lane)表示汽车在车道上,ULD(Unintentional Lane Departure)表示无意识的车道偏离,RC(Requesting Control)表示请求控制,CRG (Control Requesting Granted)表示控制请求已授权,AF (Actuator Fail)表示执行子系统失效,表 1 为模型参数. 待验证的系统安全属性为:"如果执行子系统失效,SAFELANE 将不能获取车辆的控制权"其 LTL 公式为:□(AF→(□¬ CRG)). 将该 LTL 公式转换为成 DFA,如图 3 所示,图中的双圆圈表示 DFA 的接受状态,DFA 输入字母表 $\Sigma=\{\{\},\{AF\},\{CRG\},\{AF,CRG\}\}$. 实验运行在 intel core i5 双核 2.4GHz CPU,8GB 内存,64 位 Win10 操作系统上. 通过 HMM MATLAB 工具箱模拟系统运行. 为了评估本文 IIV 算法的计算性能,本节将 IIV 算法和粒子滤波算法(Particle Filter,PF)[15] 进行比较. PF 算法的粒子数为 500.

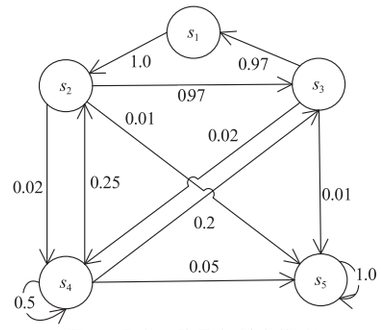


图2 系统的隐马尔科夫模型

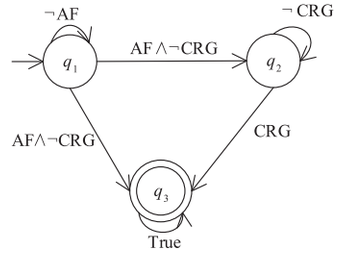


图3 安全属性自动机

表 1 系统可观测变量及参数值设置

系统状态	初始概率分布	系统观测变量及概率分布				
		IL	ULD	RC	CRG	AF
s_1	1.0	0.7	0.3	0.0	0.0	0.0
s_2	0.0	0.0	0.0	0.8	0.2	0.0
s_3	0.0	0.0	0.1	0.0	0.9	0.0
s_4	0.0	0.0	0.0	0.0	0.1	0.9
s_5	0.0	0.0	0.0	0.0	0.0	1.0

5.2 安全属性验证

模拟系统运行 100 次,每次产生观测序列的长度为 50,系统安全性概率阈值为 0.95,如果当前系统安全性概率低于阈值,监控器根据当前的监控情况产生长度为 10 的反例并终止本次运行. 在 100 次运行中,共计探测到 4 次系统安全性概率低于给定的阈值,表 2 列出了当系统安全性概率低于阈值时的观测序列及产生的反例.

表 2　系统安全性概率及生成的反例

序号	输入的观测序列	安全性概率	最大可能执行路径	是否反例
1	IL,RC,CRG,IL,RC,CRG,IL,RC,CRG,AF	0.9009	$s_1,s_2,s_3,s_1,s_2,s_3,s_1,s_2,s_3,s_4$	否
2	IL,RC,CRG,IL,CRG,CRG,IL,RC,AF,CRG	0.2447	$s_1,s_2,s_3,s_1,s_2,s_3,s_1,s_2,s_4,s_3$	是
3	ULD,RC,CRG,IL,RC,CRG,ULD,RC,AF,AF	0.9101	$s_1,s_2,s_3,s_1,s_2,s_3,s_1,s_2,s_4,s_4$	否
4	RC,CRG,IL,RC,CRG,IL,RC,AF,ULD,CRG	0.1870	$s_2,s_3,s_1,s_2,s_3,s_1,s_2,s_4,s_4,s_2$	是

从表 2 中可以看出,第一行和第三行是请求控制被授权后,系统执行车道校正时,执行机构失效了,系统安全性概率降低了,但系统没有违背安全属性,此时产生的系统最大可能执行路径不是系统反例. 第二行和第四行的系统安全性概率非常低,根据观测序列可知,系统的执行机构失效后,请求控制获得授权,系统违背了安全属性,即此时系统的最大可能执行路径是系统反例. 通过对表 2 的分析可知,系统安全性预测结果和以最大可能执行路径作为反例的预测都可能产生错误,5.3 节通过实验评估本文方法的安全性和反例预测结果的错误率.

5.3 算法预测精度和性能分析

首先定义安全性预测的两类错误(1)误报,即系统是安全的,监控器判定系统不安全,(2)漏报,即系统进入了不安全状态,监控器判定系统是安全的. 引起上述两类错误不仅和本文方法预测精度有关也和系统安全性概率阈值的选择相关. 例如,在给定阈值下,如果系统是安全的,但系统安全性概率低于阈值,引起监控器误报. 相反,如果系统是不安全的,但系统安全性概率高于阈值,会引起监控器漏报. 反例预测错误是指系统违背安全属性时,产生的最大可能执行路径不是反例. 在此定义下,反例预测的错误和阈值无关,能真实反映本文方法反例预测的错误率.

根据 5.1 节的模型和系统参数设置,不考虑安全性概率阈值,模拟运行系统 400 次,每次运行到系统违背安全属性或观测序列的长度大于 50 为止. 在 400 次运行中系统违背安全属性 64 次. 通过分析系统每次运行的观测序列、最大可能执行路径和系统实际执行路径,表 3 给出 IIV 算法和 PF 算法安全性预测错误率和本文提出的反例预测算法的错误率. 当系统违背安全属性时,我们的方法能够给出反例而 PF 算法不能给出系统反例,在安全攸关系统中,漏报的危害比误报要大,而 IIV 算法漏报次数只有 PF 算法的一半,预测错误率下降了近 20%.

表 3　IIV 和 PF 算法预测错误率

方法	安全性预测			反例预测		
	误报次数	漏报次数	预测错误率	违背安全属性次数	反例预测错误数	预测错误率
IIV	18	3	0.0525	64	7	0.1094
PF	20	6	0.0650		/	

安全性概率阈值的选择也影响安全性预测的错误率. 从图 4 可以看出,随着阈值增大,误报率上升,漏报率下降. 相反,随着阈值减小,误报率下降,漏报率上升. 当阈值在 0.8 到 0.9 之间时,系统预测错误率达到最低,因此合理选择安全性概率阈值能降低系统预测错误率,一般来说系统的漏报比误报危害要大,适当提高阈值可以减少漏报率,就本例来说安全性概率阈值设置为 0.9 较为合理.

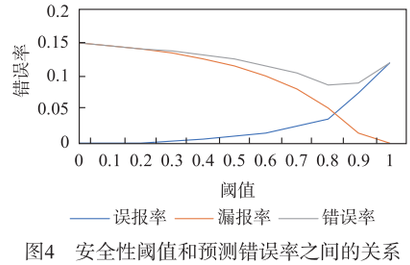
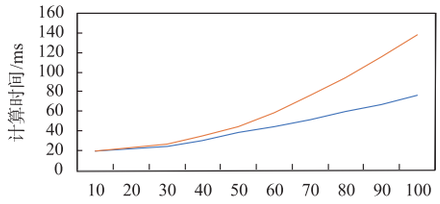


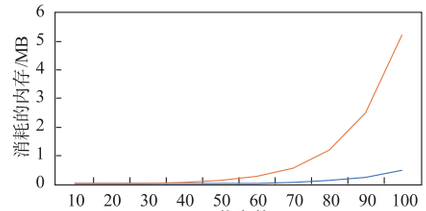
图4　安全性阈值和预测错误率之间的关系

下面比较两种算法的时间和空间复杂性. 在保证两种方法具有相同计算精度的情况下,通过不断增大系统模型状态的数量,统计算法的执行时间和消耗的内存空间. 图 5 给出了两种安全性验证算法时间和空间随系统状态数增大的变化趋势.

从图 5 可以看出,在时间性能方面,随系统状态数增大,IIV 算法的执行时间变化比较平稳,PF 算法执行时间变化较快. 在空间性能方面,IIV 算法消耗的内存空间随系统状态数增大变化不明显,而 PF 算法在状态数达到 80 以上时内存空间增长较快. 出现这种实验结果的主要原因是:(1) IIV 算法约简了系统模型和属性



(*a*) IIV和PF算法的空间性能



(*b*) IIV和PF算法的空间性能

图5 IIV和PF算法的性能比较

自动机的组合状态减少了系统执行时间和消耗的空间,而 PF 算法是直接基于系统模型和属性自动机的组合状态进行联合概率分布的计算,(2) IIV 算法每次仅处理一个观测,而 PF 算法需要保存当前观测之前的序列,(3)随着系统状态数增长,PF 算法要保持和 IIV 算法相同的精度,必须增加粒子的数量从而引起计算时间和空间的增长变快.

6 总结

本文针对状态不可观测的且行为具有随机性特征的 CPS,提出了运行时安全性验证方法,首先,构造了运行时验证框架,通过 HMM 来建模系统,使用 DFA 规约系统安全属性的否定,两者的乘积自动机作为运行时监控器,将 CPS 运行时安全性验证问题约简到属性监控器上的概率推断问题. 然后,提出了一种增量迭代安全性验证算法以及反例生成算法. 实例分析和实验结果表明本文提出的方法能正确地和快速地探测系统违背安全属性的行为,并能够产生有效的反例. 本文没有考虑 CPS 的混成系统特征,下一步工作是扩展 HMM 使其能够建模随机混成系统并对该类 CPS 进行运行时安全性验证.

参考文献

[1] 黄志球,徐丙凤,阚双龙,胡军,陈哲. 嵌入式机载软件安全性分析标准、方法及工具研究综述[J]. 软件学报,2014,25(2):200 – 218.
Huang ZQ,Xu BF,Kan SL,Hu J,Chen Z. Survey on embedded software safety analysis standards, methods and tools for airborne system[J]. Journal of Software,2014,25(2):200 – 218. (in Chinese)

[2] Chen F,Ye H,Yang J,et al. A standardized design methodology for complex digital logic components of cyber-physical systems[J]. Microprocessors & Microsystems,2015,39(8):1245 – 1254.

[3] Zheng X,Julien C,Chen H,et al. Real-time simulation support for runtime verification of cyber-physical systems[J]. ACM Transactions on Embedded Computing Systems,2017,16(4):106.

[4] Jiang Y,Song H,Wang R,et al. Data-centered runtime verification of wireless medical cyber-physical system[J]. IEEE Transactions on Industrial Informatics,2017,13(4):1900 – 1909.

[5] Medhat R,Bonakdarpour B,Kumar D,et al. Runtime monitoring of cyber-physical systems under timing and memory constraints[J]. ACM Transactions on Embedded Computing Systems,2015,14(4):79 – 108.

[6] Yu K,Chen Z,Dong W. Apredictive runtime verification framework for cyber-physical systems[A]. IEEE Eighth International Conference on Software Security and Reliability-Companion[C]. San Francisco,California,USA:IEEE,2014. 223 – 227.

[7] Bak S,Huang Z,Abad F A T,et al. Safety andprogress for distributed cyber-physical systems with unreliable communication[J]. ACM Transactions on Embedded Computing Systems,2015,14(4):1 – 22.

[8] Mao J,Chen L. Runtimemonitoring for cyber-physical systems:a case study of cooperative adaptive cruise control[A]. Second International Conference on Intelligent System Design and Engineering Application[C]. Sanya,Hainan,China:IEEE,2012. 509 – 515.

[9] Sistla A P,Žefran M,Feng Y,et al. Timely monitoring of partially observable stochastic systems[A]. International Conference on Hybrid Systems:Computation and Control[C]. Chicago,Illinois,USA:ACM,2014. 61 – 70.

[10] Stoller SD,Bartocci E,Seyster J,et al. Runtime verification with state estimation[A]. International Conference on Runtime Verification[C]. Berlin Heidelberg:Springer-Verlag,2011. 193 – 207.

[11] Han T,Katoen JP. Counter examples in probabilistic model checking[A]. International Conference on Tools and Algorithms for the Construction and Analysis of Systems[C]. Berlin Heidelberg:Springer-Verlag,2007. 72 – 86.

[12] Rabiner L. A tutorial onhidden Markov models and selected applications in speech recognition[J]. Proceedings of the IEEE,1989,77(2):257 – 286.

[13] Zhao C,Wu C,Chai J,et al. Decomposition-based multi-objective firefly algorithm for RFID network planning with uncertainty[J]. Applied Soft Computing,2017,55(6):549 – 564.

[14] Amditis A, Bimpas M, Thomaidis G, et al. A situation-adaptive lane-keeping support system: overview of the SAFELANE approach[J]. IEEE Transactions on Intelligent Transportation Systems, 2010, 11(3): 617 – 629.

[15] Kalajdzic K, Bartocci E, Smolka SA, et al. Runtimeverification with particle filtering[A]. International Conference on Runtime Verification[C]. Berlin Heidelberg: Springer-Verlag, 2013. 149 – 166.

作者简介

房丙午　男,1974 年生于安徽枞阳. 现为南京航空航天大学计算机科学与技术学院博士研究生,副教授. 主要研究方向软件工程、软件系统安全性分析.
E-mail: bingwufang@163.com

黄志球(通信作者)　男,1965 年生于江苏南京. 现为南京航空航天大学教授,博士生导师,CCF 杰出会员. 主要研究方向为软件工程、形式化方法、软件分析与验证.
E-mail: zqhuang@nuaa.edu.cn