基于 STeC-Stateflow 转换系统的实时系统仿真与验证方法*

纪 政, 李慧勇, 陈仪香†
(华东师范大学 软件学院 教育部软硬件协同设计技术与应用工程研究中心, 上海 200062)

摘 要: 物联网以及信息物理融合系统对形式化建模提出了新的挑战,引入了实时系统规范语言 STeC,为刻画实时系统的时空一致性提供了规范语言。针对 STeC 语言建立 STeC 至 Stateflow 自动转换系统,提出一种基于 STeC 至 Stateflow 转换的仿真及验证方法,该方法使用 STeC 语言对实时系统进行形式化建模,再建立实时监控的 Simulink 仿真模型,并使用 Checkmate 对系统进行安全性验证。通过对京沪高铁运行的实例研究,表明该方法对高铁运行系统实时仿真的有效性,并能够验证高铁运行系统的安全性。

关键词: 实时系统; 实时系统规范语言; 时空一致性; 系统仿真与验证; Stateflow; Checkmate
中图分类号: TP301.2 **文献标志码**: A **文章编号**: 1001-3695(2014)02-0448-06
doi:10.3969/j.issn.1001-3695.2014.02.030

Real-time system simulation and verification approach based on STeC to Stateflow transformation system

JI Zheng, LI Hui-yong, CHEN Yi-xiang†
(*MoE Engineering Center of Software/Hardware Co-design Technology & Applications, Software Engineering Institute, East China Normal University, Shanghai* 200062, *China*)

Abstract: Internet of Things or cyber-physical systems provide a new challenge for formal modeling methods related to the aspect of physical elements such as location and time. Recently, this paper introduced a specification language called STeC to stress the spatio-temporal consistency for real-time systems. The operational and denotational semantics of and tool set related to this language have been given. The aim of this paper was to establish a STeC to Stateflow automatic transformation system and to propose a simulation and verification approach based on this transformation system. It firstly gave a formal model for an object system in STeC language, and then set up a real-time monitoring simulation model using Simulink. After that, it presented a verification approach for the system safety property based on Checkmate. Finally, it gave a case about Jinghu Gaotie (high speed train) running timetable to show that the proposed approach is effect and usable.

Key words: real-time system; specification language for real-time system; spatial-temporal consistence; system simulation and verification; Stateflow; Checkmate

0 引言

物联网以及信息物理融合系统 (cyber physical systems, CPS)是综合计算、网络、物理环境的多维复杂系统,具有重要的研究价值和应用意义[1,2]。

物联网将数以亿计的物理世界对象连接起来,通过信息技术检测、分析和控制,潜力巨大[2]。而 CPS 的核心是 3C 技术,即计算(computation)、通信(communication)和控制(control)之间的有机融合与深度协作[1]。该系统既包括物理世界的连续变化,也包括计算机系统的离散状态迁移。

为了对物联网以及 CPS 系统建模,笔者引入了实时系统规范描述语言 STeC[3]。STeC 是一种描述物理事件触发的实时系统描述语言,重点强调实时系统智能体在网络环境下具有时空一致性的行为。时空一致性强调在指定时间到达指定位置,同时在指定的时间完成规定的任务。这里的位置概念是物理的空间元素,时间包括了时刻和时段。

笔者已经建立了 STeC 语言的语义模型[4],和基于 Maude 的 STeC 语言形式推理工具[5]。物联网以及 CPS 系统中的许多领域都具有时空一致性的要求,如高速列车、城市交通、航空航天等,STeC 语言可以用来描述和分析这种实时系统的时空一致性[3~5]。

本文的贡献如下:

a)本文针对实时系统规范语言 STeC,建立了从 STeC 至 Stateflow 的自动转换系统,从而可对 STeC 语言的进程进行模型化的表达和分析,并将 STeC 语言扩展到应用层,既拥有了形式化规范性,又保证了模型的可用性。

b)建立一种基于 STeC 至 Stateflow 自动转换系统的仿真

收稿日期: 2013-06-19; **修回日期**: 2013-08-11 **基金项目**: 国家"973"计划基金资助项目(2011CB302802);国家自然科学基金资助项目(61202104);上海高校知识创新工程(085)建设项目

作者简介:纪政(1989-),男,山东曹县人,硕士研究生,主要研究方向为软件工程、软硬件协同设计;李慧勇(1980-),男,山西太原人,博士研究生,主要研究方向为车联网、CPS 系统;陈仪香(1961-),男(通信作者),江苏徐州人,教授,博导,博士,主要研究方向为物联网、实时协同规范语言设计、程序语义模型、软件可信度量与评估(yxchen@sei.ecnu.edu.cn).

及验证方法,将 STeC 语法对应到有限混成自动机 Stateflow,并生成仿真模型和验证模型。形式化验证保证了系统的安全性,而仿真模型是对系统实时监控的有效手段。

c)通过京沪高铁运行模型实例,说明了基于 STeC 至 Stateflow 自动转换系统的仿真及验证方法的有效性。

本文中使用了 Simulink/Stateflow 和 Checkmate 工具。Stateflow 是基于有限混成自动机的 MATLAB 图形化实现。在当前的软件开发中,对复杂控制逻辑的动态系统,通过使用 MATLAB Simulink/Stateflow 建模方法,可以建立相应的系统仿真模型[6]。结合 Simulink Coder 和 Stateflow Coder 自动代码生成技术,可以将软件应用至实际大型复杂控制系统。但单纯的 Simulink/Stateflow 仿真模型需要人工干预,缺乏理论推理,不能保证系统的安全性。

Checkmate 是一种基于 MATLAB 的混成自动机的形式化验证工具,也适合于物联网以及 CPS 系统的形式化验证[7]。Checkmate 工具箱中含有切换连续系统模块 SCSB、多面体阈值模块 PTHB、混成状态机模块 FSMB 等。该验证工具支持连续状态下的线性和非线性状态方程,而混成状态自动机处理离散状态的变迁,输出连续状态变量,从而可对混成系统建模[8]。

Checkmate 自动验证时,将图形表示的混成自动机模型转换成多面体不变集混成自动机(PIHA)。之后对连续变量的状态空间进行划分,并通过流管道近似技术,得到系统的可达集。最后使用 CTL(计算树逻辑)逻辑语言对系统进行形式化的模型验证。但 Checkmate 和 Simulink/Stateflow 都没有考虑实时系统的时空一致性。本文在上述工具的基础上设计基于 STeC 至 Stateflow 自动转换系统的仿真及验证方法,较好地解决了上述问题。

本文简单介绍 STeC 语言,建立了 STeC 至 Stateflow 的自动转换系统和仿真验证方法,并以京沪高铁 G147、G39、G17 高速列车运行为例,说明本文所建立的方法的有效性和实用性。

1　STeC 语言

1.1　语法

STeC 语言看起来像是形式化描述语言 CSP 和 CCS 的扩展,它的语法描述了进程(process),其中包括了动作 α、状态 β 和通信机制。

$$A::=\mathrm{Send}^{G\to G'}_{(l,t)}(m,\delta)\mid\mathrm{Get}^{G\leftarrow G'}_{(l,t)}(m,\delta)$$

$$B::=\alpha^{G}_{(l,t)}(l',\delta)\mid\beta^{G}_{(l,t)}(\delta)$$

$$P::=\mathrm{Stop}_{(l,t)}\mid\mathrm{Skip}_{(l,t)}\mid A\mid B\mid P;P\mid$$
$$[\,]_{(i\in I)}(B_i\to P_i)\mid P\triangleright_{\delta}P\mid$$
$$P\trianglerighteq(\mathrm{Get}^{G\leftarrow G'}_{(l,t)}(m)\to Q)\mid P\parallel P$$

智能体之间的通信通过两个原子通信进程 Send 和 Get 实现。原子进程 $\mathrm{Send}^{G\to G'}_{(l,t)}(m,\delta)$ 定义了智能体 G 在位置 l 和时间 t 发送消息 m 给智能体 G',该动作消耗 δ 单位时间。与之类似,$\mathrm{Get}^{G\leftarrow G'}_{(l,t)}(m,\delta)$ 原子进程定义了在位置 l 和时间 t,智能体 G 收到消息 m,并花费 δ 单位时间。

对于每个智能体,本文定义了其动作 α 和状态 β。

动作指的是智能体执行任务。$\alpha^{G}_{(l,t)}(l',\delta)$ 表示智能体 G 在位置 l 和时间 t 执行动作 α,消耗 δ 单位时间,执行动作之后智能体的新位置为 l'。因此,执行动作将改变智能体的时间和位置。$\beta^{G}_{(l,t)}(\delta)$ 表示智能体 G 在位置 l 和时间 t 开始处于状态 β,并保持状态 δ 单位时间。

进程 $\mathrm{Stop}_{(l,t)}$ 表示如果不被打断将一直执行,而 $\mathrm{Skip}_{(l,t)}$ 进程不消耗时间。Send 和 Get 原子操作的时间由智能体情况决定。

B_i 在卫士选择进程 $[\,]_{i\in I}(B_i\to P_i)$ 中表示卫士条件。STeC 语言要求只有一个卫士条件为真,选择进程选择该条件并执行相应的 P_i。

延时进程 $P\triangleright_{\delta}Q$ 表示进程 P 执行 δ 单位时间,之后跳转到 Q 进程。交互进程 $P\trianglerighteq(\mathrm{Get}^{G\leftarrow G'}_{(l,t)}(m)\to Q)$ 表示智能体首先执行进程 P,当 $\mathrm{Get}^{G\leftarrow G'}_{(l,t)}(m,\delta)$ 为真时,打断 P,执行 Q 进程。而并行进程 $P\parallel Q$ 表示进程 P 和 Q 独立执行。

本文使用的 STeC 至 Stateflow 自动转换也是基于上述 STeC 的语法进行解析。

1.2　执行时间

根据 STeC 语法,定义 STeC 进程的执行时间如下所示。其中进程 P 的执行时间使用 $\tau(P)$ 表示。

$$\tau(\mathrm{Stop}_{(l,t)})=\infty\quad\tau(\mathrm{Skip}_{(l,t)})=0$$
$$\tau(\mathrm{Send}^{G\to G'}_{(l,t)}(m))=\tau(\mathrm{Send})$$
$$\tau(\mathrm{Get}^{G\leftarrow G'}_{(l,t)}(m))=\tau(\mathrm{Get})$$
$$\tau(\alpha^{G}_{(l,t)}(l',\delta))=\delta\qquad\tau(\beta^{G}_{(l,t)}(l',\delta))=\delta$$
$$\tau(P\triangleright_{\delta}Q)=\delta+\tau(Q)\quad\tau(P;Q)=\tau(P)+\tau(Q)$$
$$\tau(P\parallel Q)=\max\{\tau(P),\tau(Q)\}$$
$$\tau([\,]_{(i\in I)}(B_i\to P_i))=\max_{i\in I}\{\tau(B_i)+\tau(P_i)\}$$
$$\tau(P\trianglerighteq(\mathrm{Get}^{G\leftarrow G'}_{(l,t)}(m)\to Q))=\tau(P)+\tau(\mathrm{Get})+\tau(Q)$$

其中,并行进程当所有子进程都结束时才结束。STeC 语言的动作均消耗时间,这点和有限混成自动机不同。本文根据 STeC 的执行时间,规定混成自动机的状态持续时间。

2　STeC 至 Stateflow 转换系统

2.1　自动转换系统

使用 STeC 语言的形式化描述规范而清晰,并可以满足时空一致性的要求。而使用混成状态自动机的模型可以直观地反映系统的状态迁移变化。Stateflow 是一种在业界广泛使用并认可的有限混成自动机工具,利用 MATLAB 的配套仿真器,十分便捷地实现实时仿真和 C 代码的生成。

本文将 STeC 语言的语法和有限混成自动机建立对应规则,并用 Stateflow 工具自动实现转换。首先给出从 STeC 语法公式生成 Stateflow 状态的规则。

规则 1　STeC 进程均由基本进程构成,每个基本进程和一个状态图对应,基本进程之间可以转移、触发和嵌套。

规则 2　有限状态机初始时,自动调用 Start 迁移至初始状态。有限状态机的进程执行完毕后进入正常终止状态,表示执行顺利完成。对应于 STeC 的 E 进程。

规则 3　有限状态机的状态迁移使用条件迁移和事件触发两种,条件迁移的谓词都使用时间和空间条件与 STeC 对应,事件触发对应 STeC 的 Send 和 Get 原子进程。

规则 4　有限状态机的同一状态迁移不能有多个处于使能状态,状态迁移全部为立即迁移,使用状态表示 STeC 的延迟迁移动作。

规则 5　表示 STeC 延迟迁移动作的有限自动机状态使用 STeC 的执行时间(2.2 节)表示状态持续时间。

基本公式转换表如表 1 所示，每个 STeC 语法中的基本公式进程(process)对应一个子状态图。

表 1 STeC 及有限混成自动机基本公式转换

STeC 语法	有限混成自动机
$\mathrm{Send}^{G\to G'}_{(l,t)}(m,\delta)$	Before Send_G —[time=t & distance=l]→ Sending exit:time= time+ δ; —Send(m,G')→ After Send_G
$\mathrm{Get}^{G\leftarrow G'}_{(l,t)}(m,\delta)$	Before Get_G —[time=t & distance=l]→ Getting exit:time= time+ δ; —m→ After Get_G
$\alpha^{G}_{(l,t)}(l',\delta)$	Before α_G —[time=t & distance=l]→ α_G —[time=t+ δ & distance=l']→ After α_G
$\beta^{G}_{(l,t)}(\delta)$	Before β_G —[time=t & distance=l]→ β_G —[time=t+ δ]→ After β_G
$\mathrm{Stop}_{(l,t)}$	Before Stop —[time=t & distance=l]→ Stop
$\mathrm{Skip}_{(l,t)}$	Before Skip —[time=t & distance=l]→ Skip → After Skip
$P;P$	P → P
$[\,]_{(i\in 1)}(B_i\to P_i)$	Before B → B_1→P_1; B_2→P_2; …; B_i→P_i
$P\triangleright_{\delta}P$	P entry: time=t; —time=t+ δ→ P
$P\trianglerighteq\mathrm{Get}^{G\leftarrow G'}_{(l,t)}(m)\to Q)$	P —[time=t & distance=l]→ Getting exit:time= time+ δ; —m→ G
$P\parallel P$	P P

其中，原子动作 Send 和 Get 表示通过迁移决策的逻辑判断时间 t 和位置 l，并通过时间 δ 完成该动作。这里使用了三个状态代替这两个原子动作，并使用了事件 m 立即触发。

卫士选择进程表示 $B_1, B_2, \cdots, B_i$ 中有且只有一个为真，执行 B_i 也需花费相应的时间，当 B_i 成立时执行 P_i。因此在状态自动机中设置 B_i 也为状态，并可以设置相应的时间。

并发进程表示两个进程独立执行，进程之间不考虑通信关系。延迟进程显式地指定进程 P 推迟 δ 时间，而规定推迟时间在原进程 P 中完成。

除该表所示的对应规则之外，还加入了默认迁移，即设定条件为假或事件未发生的状态都有一个通向状态的默认迁移。转换过程中还生成了相应的数据对象、事件对象、反馈控制对象、时间空间对象等。

进行语言转换时，对 STeC 语言的每条公式进行词法和语法解析，得到 STeC 语法树，根据语法树结构将公式转换为 STeC 进程(process)中对应的元素，对这些元素利用转换规则，生成相应的状态图，并可以根据 STeC 语言的语法关系合成相应的子状态图。生成状态图时使用了 Stateflow API 进行代码自动化。

通过 STeC 至 Stateflow 的自动转换系统，输入 STeC 形式化语言，自动输出 Stateflow 模型，从而可以对 STeC 语言的进程进行模型化的分析，并将 STeC 语言扩展到应用层，既拥有了形式化规范性，又保证了模型的可用性。

2.2 STeC 至 Stateflow 仿真及验证方法

本节从形式化体系结构角度出发，考虑到物联网以及 CPS 系统的实时性和时空一致性，在使用时空一致的 STeC 语言形式化描述的基础上，将其语法对应到有限混成自动机 Stateflow，表达物联网以及 CPS 系统的状态及其迁移，并生成仿真模型和验证模型。其中 Simulink 物理仿真模型由 IP 核和 STeC 语言提供的特征生成。而 Stateflow 模型可以使用 Checkmate 验证器对系统的安全性等进行形式化验证。基本构架如图 1 所示，其中，IP 核针对不同的物理环境提供不同的微分方程。

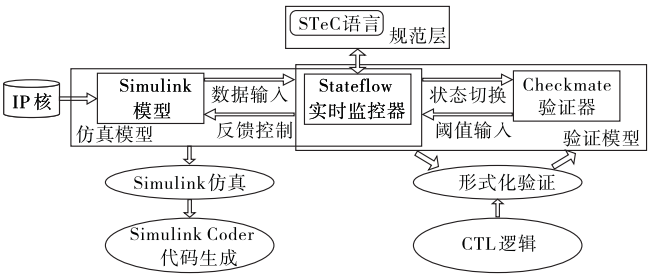

图 1 STeC 至 Stateflow 仿真验证方法

Stateflow 混成自动机是本方法中的实时监控器，其接受 Simulink 的辅助变量，加以反馈控制，并可以对 Stateflow 模型进行时空一致性质验证。

由 Simulink 和 Stateflow 共同构成的系统模型可以在 MATLAB 环境下实时仿真，而最终可以通过 Simulink/Stateflow Coder 生成 C 语言代码。

利用 STeC 至 Stateflow 自动转换系统，将 Checkmate 中混成状态机模块 FSMB，使用 STeC 规范描述得到的 Stateflow 状态机表示，从而可以保证 CPS 系统的时空一致性，且更便捷地对 CPS 系统形式化建模，并使用 Checkmate 进行模型验证，具有较强的应用价值，其中形式化验证使用 CTL 逻辑描述安全性性质。

3 案例分析

列车运行的安全性和准时性一直是研究的重点。随着列车运行速度的提高，300 km/h 高速列车的运行特点与普通列车有很大不同[9~11]。本章将选用高速列车 CRH-3 型进行分析，使用时空一致语言 STeC 进行形式化建模，并利用基于 STeC 至 Stateflow 转换的方法，进行仿真和验证。

3.1 STeC 语言形式化描述

G147、G39、G17 三列高速列车北京至上海段运行时刻图如图 2 所示。首先使用 STeC 语言进行形式化描述。高速列车 G147 运行过程如下：

$\mathrm{Running}_{(\mathrm{Beijing},14:36)}(\mathrm{Tianjin},\ 34\mathrm{m})$; $\mathrm{TStop}_{(\mathrm{Tianjin},15:10)}(2\mathrm{m})$;
$\mathrm{Running}_{(\mathrm{Tianjin},15:12)}(\mathrm{Jinan},\ 63\mathrm{m})$; $\mathrm{TStop}_{(\mathrm{Jinan},16:15)}(2\mathrm{m})$;
$\mathrm{Running}_{(\mathrm{Jinnan},16:17)}(\mathrm{Zaozhuang},\ 50\mathrm{m})$; $\mathrm{TStop}_{(\mathrm{Zaozhuang},17:07)}(1\mathrm{m})$);
$\mathrm{Running}_{(\mathrm{Zaozhuang},17:08)}(\mathrm{Xuzhou},\ 18\mathrm{m})$; $\mathrm{TStop}_{(\mathrm{Xuzhou},17:26)}(2\mathrm{m})$);
$\mathrm{Running}_{(\mathrm{Xuzhou},17:28)}(\mathrm{Nanjing},\ 74\mathrm{m})$; $\mathrm{TStop}_{(\mathrm{Nanjing},18:42)}(4\mathrm{m})$;
$\mathrm{Running}_{(\mathrm{Nanjing},18:46)}(\mathrm{Wuxi},\ 44\mathrm{m})$; $\mathrm{TStop}_{(\mathrm{Wuxi},19:30)}(6\mathrm{m})$;
$\mathrm{Running}_{(\mathrm{Wuxi},19:36)}(\mathrm{Shanghai},\ 28\mathrm{m})$; $\mathrm{Arrive}_{(\mathrm{Shanghai},20:04)}(5\mathrm{m})$.

其中:Running 为 STeC 语言的动作,TStop 为状态。

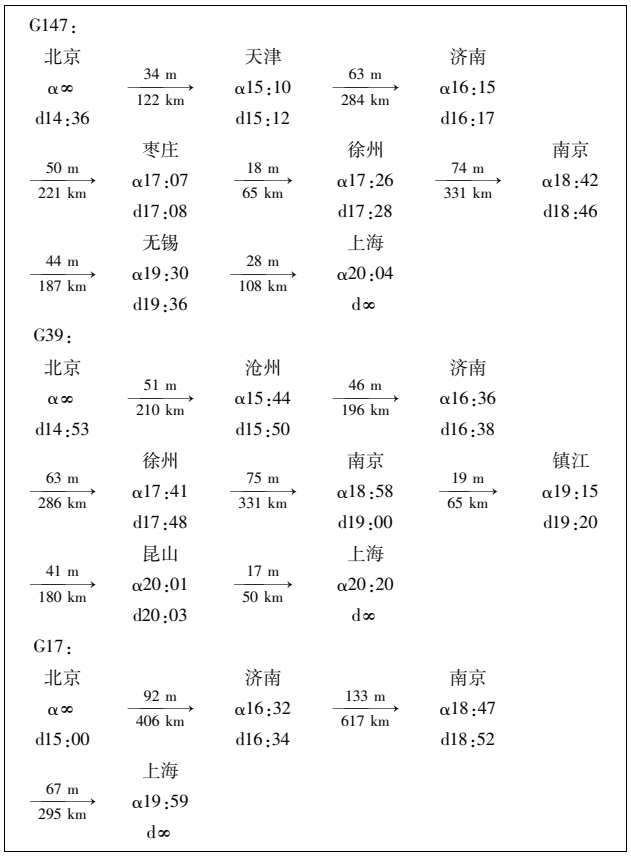

图 2　三列高速列车运行时刻图

其他两列高速列车的 STeC 语言形式化描述与之类似。

高速列车 G39:

$\text{Running}_{(\text{Beijing},14:53)}$(Cangzhou, 51m); $\text{TStop}_{(\text{Cangzhou},15:44)}$(6m);
$\text{Running}_{(\text{Cangzhou},15:50)}$(Jinan, 46m); $\text{TStop}_{(\text{Jinan},16:36)}$(2m);
$\text{Running}_{(\text{Jinnan},16:38)}$(Xuzhou, 63m); $\text{TStop}_{(\text{Xuzhou},17:41)}$(2m);
$\text{Running}_{(\text{Xuzhou},17:43)}$(Nanjing, 75m); $\text{TStop}_{(\text{Nanjing},18:58)}$(2m);
$\text{Running}_{(\text{Nanjing},19:00)}$(Zhenjiang, 19m); $\text{TStop}_{(\text{Zhenjiang},19:19)}$(1m);
$\text{Running}_{(\text{Zhenjiang},19:20)}$(Kunshan, 41m); $\text{TStop}_{(\text{Kunshan},20:01)}$(2m);
$\text{Running}_{(\text{Kunshan},20:03)}$(Shanghai, 17m); $\text{Arrive}_{(\text{Shanghai},20:20)}$(5m)。

高速列车 G17:

$\text{Running}_{(\text{Beijing},15:00)}$(Jinan, 92m); $\text{TStop}_{(\text{Jinan},16:32)}$(2m);
$\text{Running}_{(\text{Jinan},16:34)}$(Nanjing, 133m); $\text{TStop}_{(\text{Nanjing},18:47)}$(5m);
$\text{Running}_{(\text{Nanjing},18:52)}$(Shanghai, 67m); $\text{Arrive}_{(\text{Shanghai},19:59)}$(5m)。

3.2　高速列车运行模型

3.2.1　牵引过程

牵引力由列车产生,是列车前进的动力。牵引力的大小主要由运行速度决定。本文使用文献[10]的牵引力曲线。牵引力 F_R(N)与列车速度 v(km/h)的关系为

$$\begin{cases} F_R = -0.4222 \times v + 514.5314 & 146.6 > v \geqslant 0 \\ F_R = \dfrac{9250 \times 3.8}{v} & 400 \geqslant v \geqslant 146.6 \end{cases} \tag{1}$$

3.2.2　制动过程

制动力是人为地控制列车速度的阻力。高速列车制动力一般包括再生制动力和空气制动力。列车制动优先采用再生制动,当列车速度较低时,切换为空气制动。再生制动的制动力 F_D(N)与列车速度 v(km/h)的关系为

$$\begin{cases} F_D = \dfrac{9250 \times 3.6}{v} & 400 \geqslant v \geqslant 160 \\ F_D = 185.25 & 160 > v \geqslant 0 \end{cases} \tag{2}$$

空气制动的制动力 F_A(N)与列车速度 v(km/h)的关系为

$$F_A = M(1+r)\beta \tag{3}$$

其中:M 为列车的质量(t);β 为列车的减速度(m/s^2);r 为列车的回转系数,一般取 0.06。

3.1 节的 $\text{Running}_{(\text{Beijing},14:36)}$(Tianjin, 34m)动作表示了 G147 高速列车从北京到天津段的运行过程。可以根据上述列车运行模型,将这段过程细化为五个阶段,并使用 STeC 描述。其中,$T_1 + T_2 + T_3 + T_4 + T_5 = 34$m。

$\text{Running}_{(\text{Beijing},14:36)}$(Tianjin, 34m) 转换为
$\text{RunningRise1}_{(\text{Beijing},14:36)}$($D_1$, T_1);
$\text{RunningRise2}_{(D_1,\ 14:36+T_1)}$($D_2$, T_2);
$\text{RunningConstant}_{(D_2,\ 14:36+T_1+T_2)}$($D_3$, T_3);
$\text{RunningDown1}_{(D_3,\ 14:36+T_1+T_2+T_3)}$($D_4$, T_4);
$\text{RunningDown2}_{(D_4,\ 14:36+T_1+T_2+T_3+T_4)}$(Tianjin, T_5)。

从 RunningRise1 到 RunningDown2 的每个动作代表了一段列车运行方程。同理,其他的每个 Running 动作均可分为五个阶段。

本文选用 CRH-3 型的高速列车进行具体分析。其中,列车为八节编组,四动四拖。最大牵引力功率为 8250 W,最大再生制动力功率为 8250 W。列车的质量为 686 t,设计时速为 400 km/h,实际运行限速为 300 km/h。

本文针对 STeC 描述的列车运行模型,对应京沪高铁 G147、G39、G17 三列列车的北京至上海段,使用 STeC 至 Stateflow 转换方法进行分析。

3.3　仿真验证

下面针对 G147 高速列车使用 STeC 至 Stateflow 转换方法对 STeC 语言描述进行处理。另两辆列车的转换过程类似。

3.3.1　STeC 至 Stateflow 自动转换

首先利用自动转换,使用 Stateflow 工具表示 STeC 描述的 G147 高速列车运行模型如图 3 所示。

使用 G147 运行时刻图中各站之间的里程和时间,数字化 STeC 语言的描述。该 Stateflow 模型的时间 time 单位为 s,距离 distance 单位为 km。

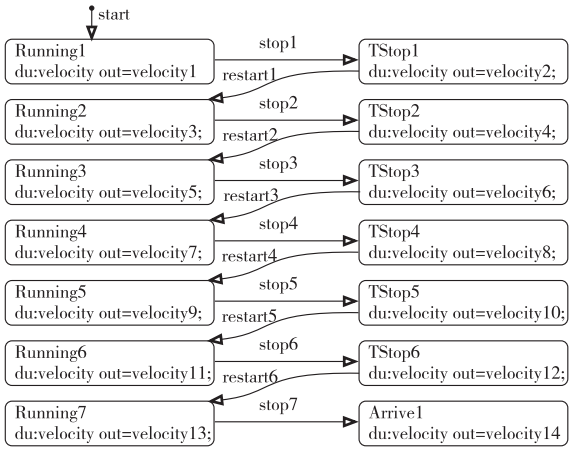

图 3　G147 运行过程模型 Stateflow 表示图

根据 STeC 至 Stateflow 转换规则,将 Running 和 TStop 转换为对应的状态,并使用状态迁移对应 Running、TStop、Arrive 之间的顺序算子。状态之间的迁移使用事件 event 触发的形式表示,而事件的触发满足谓词表述如表 2 所示,体现了 STeC 语言时空一致性的特点,即当在指定时间到达指定位置时,才能进行状态的切换,否则不切换。

表 2 G147 列车运行 STeC 至 Stateflow 事件谓词对应

事件	谓词表述
start	[time == 0]
stop1	[(time == 2040) & (distance1 ≥ 122 − e) & (distance1 ≤ 122 + e)]
restart1	[time ≥ 2160]
stop2	[(time == 5940) & (distance3 ≥ 284 − e) & (distance3 ≤ 284 + e)]
restart2	[time ≥ 6060]
stop3	[(time == 9060) & (distance5 ≥ 221 − e) & (distance5 ≤ 221 + e)]
restart3	[time ≥ 9120]
stop4	[(time == 10200) & (distance7 ≥ 65 − e) & (distance7 ≤ 65 + e)]
restart4	[time ≥ 10320]
stop5	[(time == 14760) & (distance9 ≥ 331 − e) & (distance9 ≤ 331 + e)]
restart5	[time ≥ 15000]
stop6	[(time == 17640) & (distance11 ≥ 187 − e) & (distance11 ≤ 187 + e)]
restart6	[time ≥ 18000]
stop7	[(time == 19680) & (distance13 ≥ 108 − e) & (distance13 ≤ 108 + e)]

如表 2 所示，由 Running 到 TStop 和 Arrive 的状态迁移：stop1 ~ stop7 必须满足运行时间 time 和运行距离 distance 的双重要求。这里允许模型的运行距离有误差，用 e 表示，这里选取 $e = 1$。为了计算方便，这里以第一列高速列车 G147 的发车时间为起始时间，以秒为单位，将时刻时间转换为时段时间。

由 TStop 到 Running 的状态迁移：restart1 ~ restart6 必须满足运行时间 time 的要求，由于 TStop 是 STeC 的状态，在 STeC 中状态不涉及位置变化，所以对应这里没有运行距离的要求。

这里再对 G147 北京天津段的 STeC 细化表示使用 STeC 至 Stateflow 自动转换。结果如图 4 所示。

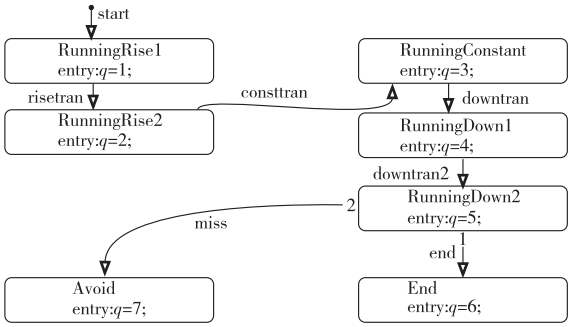

图 4 G147 北京天津段的细化 Stateflow 表示图

这里加入 Avoid 状态，当列车不能按时准确到达车站时，触发 miss 事件，进入该状态。正常到达时触发 end 事件，进入 End 状态。此图的状态变迁形式与上例相似，不再赘述。

3.3.2 仿真模型

根据上述 STeC 至 Stateflow 转换，采用 G147 运行过程的 Stateflow 模型（图 3）作为仿真模型的实时监控器，并建立 Stateflow-Simulink 模型。其中，每个 Running、TStop 或 Arrive 对应一个 Simulink 模块，模块由输入的 M 文件计算该段的即时速度和路程。

M 文件中包含了由运行模型的式(1) ~ (3)构成的微分方程和起止时间、位置等信息。

Simulink 模块将输入数据传递给 Stateflow 实时监控器。由 Stateflow 监控器控制仿真模型的输出结果。

为了方便图形显示，仿真验证的时间单位均为 10 s，速度单位为 m/s，距离单位为 km。

仿真结果如图 5 和 6 所示。

图 5 和 6 的仿真结果与列车运行模型一致。图 6 的曲线反映了三列高速列车交互的情况，其中高速列车 G17 对另两列分别停靠沧州站和无锡站的列车进行超越，保证了运行的安全性。该仿真结果符合了列车运行的客观规律，并且较好地满足了时空一致性的要求，可以较好地保证列车准点到达。其中仿真运行拥有较高的精度，时间误差小于 1 s，路程的误差小于 1 km，能够满足工程实际应用的要求。

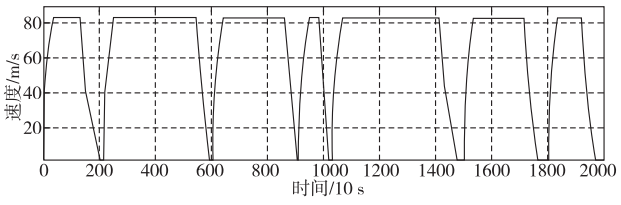

图 5 G147 列车仿真速度—时间曲线

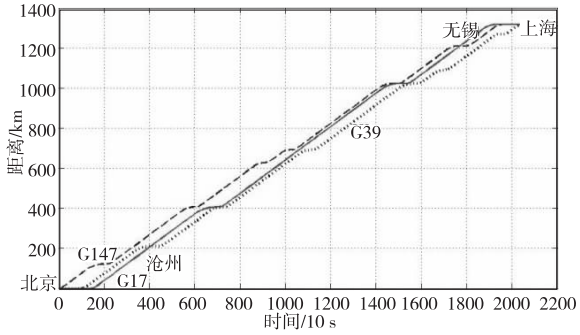

图 6 三列列车仿真路程—时间曲线

3.3.3 验证模型

基于 G147 运行过程的 Stateflow 模型的上述仿真模型较好地反映了列车运行过程，但对于形式化验证，该模型仍需精化。

下面将高速列车 G147 北京至上海段按停靠站分为：a）北京至天津段；b）天津至济南段；c）济南至枣庄段；d）枣庄至徐州段；e）徐州至南京段；f）南京至无锡段；g）无锡至上海段。对每段分别使用细化 STeC 语言描述的 Stateflow 模型和 Checkmate 工具验证模型的安全性，从而说明整体的安全性。

下面针对北京至天津段进行分析，建立 Checkmate 模型如图 7 所示。其他段的过程类似。

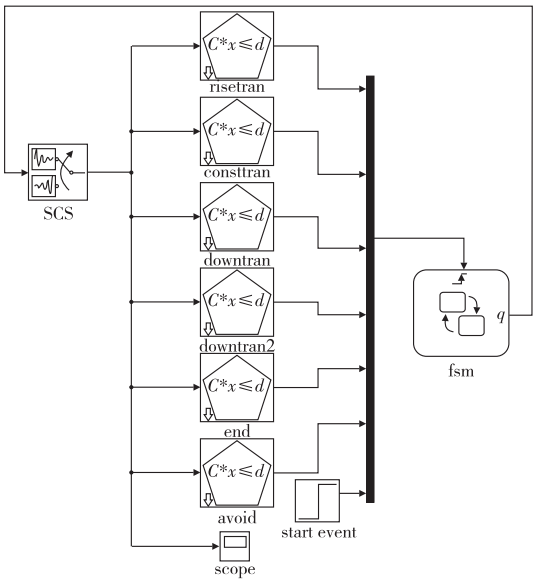

图 7 Checkmate 验证模型

其中：SCS 模块反映模型的连续状态变量情况，并通过 fsm 模块的输入切换连续状态；PTHB 模块 risetran、consttran 等分别为系统建立切换面，并使用线性约束方程表示不变集的状态空间；fsm 模块即图 4 的细化 Stateflow 模型，可以根据上升或下降

沿两种触发方式反映事件 event 的信号。

Checkmate 可以通过 verify 对该验证模型进行形式验证。

本文定义该列车运行模型的安全性为列车运行准确到达指定地点,不发生过站或不能到达的情况,并使用 CTL 公式

$$(AG \sim fsm == avoid) \& (AG \sim out\_of\_bound)$$

对系统的安全性进行验证。

该公式表示该系统永远不会到达 avoid 的离散状态,并且状态变量永远不会偏离规定集。这里设定规定集为距离和时间均大于等于 0。

从图 8 的验证结果可知:系统验证在经过初始划分后得到的迁移系统可达集,满足 CTL 逻辑描述的规范,即从初始区域出发,系统经过 Stateflow 控制器之间的有限次切换,最终到达的状态总是状态 End,并且不会超过规定集。高铁运行北京至天津段模型满足了安全性要求。而同理可以验证,在 G147 的各行驶段的运行模型均满足该安全性。所以说明了整体的安全性。

```
Parsing specification 1:(AG ~out_of_bound)&(AG ~fsm == avoid)
Compiling list of atomic propositions:(AG ~out_of_bound)&(AG ~fsm == avoid)
 * out_of_bound
 * fsm_in_avoid
Making refinement decision.
System never enters the state "avoid"
total verification time is 943.34 seconds.
```

图 8　Checkmate 验证结果

4　结束语

形式化描述语言 STeC,描述物理元素触发的实时系统。STeC 语言重点强调实时系统智能体在网络环境下具有时空一致性的行为。

本文针对 STeC 形式化语言的语法,使用有限混成自动机的图形化工具 Stateflow 对 STeC 进行逻辑解释,并基于该转换建立实时监控器仿真模型和形式化验证模型。通过京沪高铁的例子说明该方法的有效性和合理性,并对 STeC 描述进行安全性验证。

下一步工作包括动态调控仿真模型和提供更多的属性进行形式验证。

参考文献:

[1] 何积丰. 信息物理融合系统[J]. 中国计算机学会通讯, 2010, 6(1): 25-29.

[2] 武建佳,赵伟. WInternet:从物网到物联网[J]. 计算机研究与发展, 2013,50(6):1127-1134.

[3] CHEN Yi-xiang. STeC: a location-triggered specification language for real-time systems[C]//Proc of the 15th IEEE International Symposium on Object/Component/Service- Oriented Real- Time Distributed Computing. 2012:1-6.

[4] WU Heng-yang, CHEN Yi-xiang, ZHANG Min. On denotational semantics of spatial-temporal consistence language STeC[C]//Proc of the 7th International Symposium on Theoretical Aspects of Software Engineering. 2013:113-120.

[5] 栾天骄,陈仪香,王江涛. 实时系统规范语言 STeC 的 Maude 重写系统[J]. 计算机工程, 2013,39(10):57-62,67.

[6] 陶继平,徐文艳,杨根科,等. 基于 Stateflow 的 Petri 网仿真方法[J]. 计算机仿真, 2006, 23(12):96-99.

[7] CLARKE E M, FEHNKER A, HAN Zhi, *et al*. Verification of hybrid systems based on counter example-guided abstraction refinement [C]//Proc of the 9th International Conference on Tools and Algorithms for the Construction and Analysis of Systems. 2003:192-207.

[8] 方敏,张雅顺,李辉. 混成系统的形式验证方法[J]. 系统仿真学报, 2006,18(10): 2921-2928.

[9] 唐金金,周磊山,佟路,等. 单列高速列车运行仿真模型与算法[J]. 中国铁道科学, 2012,33(3): 109-115.

[10] 唐金金,周磊山,佟路,等. 基于 Bezier 函数的列车特性曲线数据处理方法研究[J]. 交通运输系统工程与信息, 2011,11(3): 131-137.

[11] 上官伟,蔡伯根,王晶晶,等. 时速 250 km 以上高速列车制动模式曲线算法[J]. 交通运输工程学报, 2011,11(3): 41-46.