并发系统中基于优先级的调度分析

朱振宇 张 仕 蒋建民 吴亚洲 杨启帆
(福建师范大学数学与计算机科学学院福建省网络安全与密码技术重点实验室 福州 350007)

摘 要 当前复杂的并发系统多采用模块化、逐步求精和信息隐藏等非形式化的原则来指导系统的开发,而这些指导原则抽象且无法保证分解系统的正确性。为此,对基于优先级控制的系统分解方法展开研究,提出一种系统分解的方法,并在理论上证明该分解方法的正确性。首先采用基于事件的行为模型对系统进行建模;接着定义调度、调度策略和调度策略正确性的概念;然后研究调度策略的分解方法,并证明了调度策略分解方法的正确性;最后根据该方法,开发出一种支持依赖模型建模和调度策略分解的原型工具,通过实例的演示,说明了使用该方法可以把系统分解成若干个子系统,从而设计出正确和有效的调度策略,以达到正确分解系统的目的。

关键词 优先级,调度策略,分解,正确性
中图法分类号 TP316 **文献标识码** A

Analyzing Scheduling Based on Priority in Concurrent Systems

ZHU Zhen-yu ZHANG Shi JIANG Jian-min WU Ya-zhou YANG Qi-fan
(Fujian Provincial Key Laboratory of Network Security and Cryptology, College of Mathematics and Computer Science, Fujian Normal University, Fuzhou 350007, China)

Abstract Due to the complexity of the concurrent system, it is difficult for engineers to develop a complex system as a whole directly. They often apply informal principles, such as modular, stepwise refinement and information hiding, to guide system development. These guidelines are abstract and can't guarantee the correct decomposition of system. In this paper, we focused on the system decomposition method based on priority, proposed a method to decompose the system, and proved its correctness. First, we modeled system with an event-based behavioral model. Next, based on such a model, we formally defined the schedule, the scheduling policy and the correctness of a scheduling policy. After that, we proposed a method for decomposing schedule policy, and proved the method's correctness. Finally, according to the decentralized method, we developed a toolkit, which supports event-based behavioral model, for modeling system and decomposition of scheduling policy. An experiment demonstrates that these results may help engineers to design correct and efficient schedule policies in a system to realize decomposition.

Keywords Priority, Scheduling policy, Decomposition, Correctness

1 引言

随着并发系统功能越来越丰富,结构越来越多样,规模也越来越庞大,开发者面对的系统越来越复杂。由于系统的复杂性,开发者很难将其作为一个整体直接开发,多采用模块化、逐步求精和信息隐藏等非形式化的原则来指导系统的开发,而这些指导原则比较抽象且无法保证开发者能够正确地分解并发系统。因此,本文将重点研究如何把基于优先级控制的并发系统分解为若干独立的子系统,并且从理论上证明分解过程的正确性。

为了在理论上证明可以对并发系统进行正确的分解,需要先使用形式化模型对系统建模。由于系统的多样性,很多形式化模型还无法对其有效地建模[1]。例如,混合系统[2]虽然能够建模动态的物理系统,却没有考虑调度方面的问题。BIP[3]虽然可以对无死锁和含有优先级调度等方面的系统进行组合验证,但仅仅考虑了组件端口方面的偏序关系。一些程序模型,例如 PTIDES[4]虽然也有调度机制,但是仅基于事件的时间戳来进行,如果多个事件同时发生,那么该调度策略采用随机选择一个事件来执行的方法。

本文采用基于事件的行为模型-依赖结构[5,6]对系统建模,主要基于以下原因:1)依赖结构是基于事件的建模模型,事件可以定义为信息事件或物理事件,事件具有类别、时间和空间等信息[7];2)在依赖结构的形式化模型中,事件以数据信息为基本要素,所以数据流和控制流并不是独立的[8];3)依赖结构能够统一地定义同步通信、异步通信和广播通信 3 种交互方式,而其它的模型为了定义这 3 种交互方式,需要在网络

本文受国家高技术研究发展计划(863)(2012AA011205),国家自然科学基金(61175123),上海知识服务平台项目(ZF1213),上海高可信计算重点实验室开放课题(07dz22304201401),福建省自然科学基金(2014J01221),福建师范大学优秀骨干教师基金(fjsdjk2012047)资助。
朱振宇(1982-),男,硕士生,主要研究领域为软件工程、形式化方法,E-mail:teazzy@163.com;张 仕(1977-),男,博士,副教授,主要研究领域为软件工程、形式化方法;蒋建民(1972-),男,博士,副教授,主要研究领域为软件工程、形式化方法;吴亚洲(1989-),男,硕士生,主要研究领域为软件工程、形式化方法;杨启帆(1990-),男,硕士生,主要研究领域为软件工程、形式化方法。

通信方面受一些条件限制，而依赖结构却避免了这种情况，所以其更加符合实际情况；4）依赖结构可以对系统中的调度建模（调度表示含有优先级的多个事件在系统运行的过程中存在先后的执行顺序）。当系统中多个事件并发执行时，还需要考虑优先级问题，而现有的一些形式化方法（例如进程代数和自动机），却没有考虑这方面的问题。同时，依赖结构还支持优先级以及多并发事件调度的建模。

为了保证分解过程的正确性，还需要设计调度机制（具有优先级控制的调度策略[9,10]）来保证执行过程的正确性。在系统中，任何一种调度都需要确定事件的执行顺序，而事件的执行顺序恰恰体现了被调度事件的优先级。传统上调度算法[11-14]的正确性是用无死锁判定，然而系统中的调度不是独立的，而是相互依赖的，因此考虑单个调度的正确性并不能代表整个系统的正确性。文中提出调度策略的概念，调度策略表示一个或者多个的调度的集合。系统中的调度策略不但需要正确地执行而且需要到达终止状态。因此，本文采用弱终止的概念作为调度策略正确性的判定标准。最后，提出了调度策略的分解方法以及从理论上证明它的正确性。

本文的主要贡献有两个方面：1）定义了调度、调度策略和调度策略正确性的概念；2）研究调度策略分解以及调度策略分解的正确性。

本文第 2 节给出一个家庭安防系统作为引例；第 3 节介绍了依赖结构的形式化模型；第 4 节定义了调度、调度策略和调度策略的正确性概念；第 5 节讨论了调度的分解，证明了调度分解的正确性；第 6 节讨论了支持依赖模型建模和调度策略分解的工具；最后，总结了本文工作，并提出将来工作方向。

2 系统建模

家庭安防系统是一个可以发现家中危险情况（例如非法闯入、一氧化碳浓度超标和漏水等），并采取应对措施（报警和通知房屋主人）的系统。该系统通过各种传感器感知异常情况，并采取相应的处理措施，如图 1 所示。当有烟雾发生时，控制中心打开喷水设备进行喷水，同时发出警报声音并通过电话通知房屋主人。家庭安防系统的控制中心控制着电话、警报和喷水设备的调度与执行。

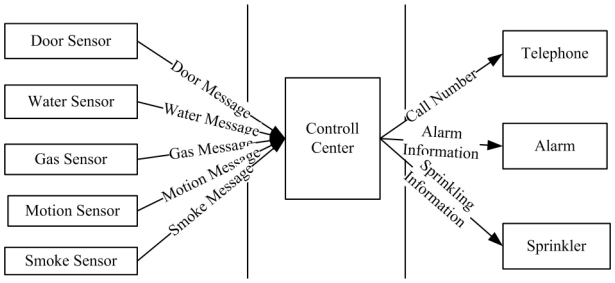

图 1 家庭安防系统

2.1 事件

基于事件的建模方法常应用于形式化建模中[7]，其中的事件不仅指活动或动作的出现，也包含了时空属性和观察者信息。

定义 1 事件 e 用三元组 $\langle A,T,L\rangle$ 表示，其中 A 代表事件的类别等相关信息，T 代表事件的时间信息，L 代表事件的空间信息。

例如：10 点时厨房产生烟雾，用 SS 来命名该事件。该事件的类别表明产生烟雾的类型（例如由液化气着火引起的烟雾）信息。时间信息 T 可以用时间间隔 $[a,b]$ 来表示，当 $a=b$ 时，表示的是一瞬间的事件。该例中 $T=[10,-]$。空间信息 L 可以是一些地名、GPS 信息等。该例中的 L 为“厨房”。

本文采用基于事件符号系统的“依赖结构”模型[6]对系统建模。

在依赖结构模型中，事件是具有三元组 $\langle A,T,L\rangle$ 信息的事件，该信息可以不是具体的值，而是一种可以对应于具体值的抽象符号[15]。如果一个事件具有具体的值，那么该事件是可用事件（或称为事件实例），否则是不可用事件（抽象事件）。

本文中的事件用名称描述，事件的具体信息 $\langle A,T,L\rangle$ 部分是不可见的。在系统的交互过程中，给定一个消息 m（组成或者对象）对应 3 个不同的事件 $!m$、$?m$ 和 $!?m$，分别表示消息 m 的发送事件、接收事件和存储事件[16]。例如，在第 2 节中烟雾消息 SM 对应的三个事件为 $!SM$、$?SM$ 和 $!?SM$。

2.2 依赖关系

依赖结构以事件集为基本元素。事件集是事件的集合。如果事件集中的所有事件是可用的，则称该事件集是可用的，否则称为不可用的。依赖结构使用 5 种依赖关系对系统进行建模，这 5 种依赖关系是：转换依赖、同步依赖、选择依赖、独立依赖和优先级依赖。

转换依赖是一种二元关系，直观上可以理解为事件集 A 转换到事件集 B，也就是在事件集 A 中所有事件可用的条件下，通过转换后事件集 B 中所有事件也是可用的，使用 $A\rightarrow B$ 表示。如图 2 所示，传感器表示从环境中感应信息并对环境中感应到的信息进行反馈。对传感器进行建模，烟雾传感器先接收到烟雾信号（SS），接着向系统发送烟雾消息（SM），可以使用转换依赖 $\{SS\}\rightarrow\{!SM\}$ 表示。执行器（例如显示器、警报器等）表示在接收到消息时可以把消息反馈给外界环境的设备，它也可用转换依赖来定义。如图 2 所示，警报器接收到警报消息 AT 后发出报警信号 Rin，使用转换依赖 $\{?AT\}\rightarrow\{Rin\}$ 表示。传感器和执行器的定义在文献[17]中有更详细的阐述。

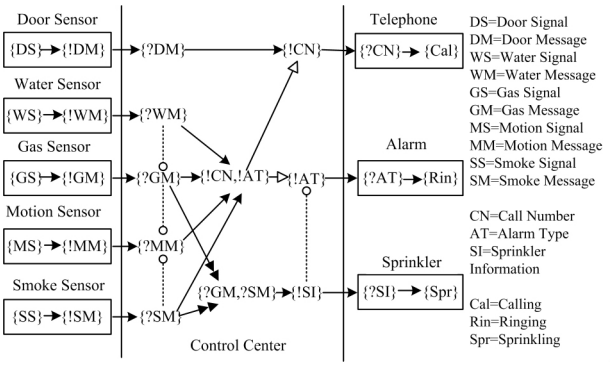

图 2 家庭安防系统的建模

同步依赖是指可用的事件需要等待其它事件以便让它们同时发生。例如，存在事件集 A 和事件集 B，且 $A\subset B$。如果事件集 A 中的事件是可用的，但是需要等待事件集 B 中不同于事件集 A 中的事件也是可用的，它们才可以进行下一步操作，那么 A 和 B 的关系就称为同步依赖，使用 $A\twoheadrightarrow B$ 来表示。如图 2 所示，当家庭安防系统的控制中心同时接收到烟雾消息（SM）和气体消息（GM）时才会触发喷水器的喷水操作，则使用同步依赖 $\{?SM\}\twoheadrightarrow\{?SM,?GM\}$ 和 $\{?GM\}\twoheadrightarrow\{?SM,?GM\}$ 表示。同步依赖可用于建模同步等待和信息融合。

选择依赖是一个二元关系，表示事件集 A 中的部分事件集 A' 发生($A'\subset A$)，同时，事件集 A 中的其它事件没有发生，则使用 $A \multimap B$ 表示(其中 $A'=B$)。如图 2 所示，当安防控制中心接收到相应信息，并且处理后，需要确定是触发警报还是电话通知用户，可以使用如下两个依赖关系表示：$\{!AT, !TNT\} \multimap \{!AT\}$ 和 $\{!AT, !TNT\} \multimap \{!TNT\}$，其中 $!AT$ 表示发送警报消息，$!TNT$ 表示电话通知屋主。选择依赖可以用来建模选择关系。

独立依赖是一种事件集间对称的关系，表示系统中的事件集可以独立发生。独立依赖用于建模真实的并发事件。依赖结构没有把独立依赖作为一种基本的元素，但是就像事件结构[18]，它实际蕴含于依赖结构中。

优先级依赖是一种事件集间的二元关系，依赖结构利用优先级关系控制集合中事件的执行顺序。在优先级关系中，系统控制任务执行的先后顺序，称为优先级依赖，使用 $\multimap\!\circ$ 表示。如图 2 所示，系统规定在烟雾消息 SM 和物体移动消息 MM 同时可用时，应该先处理烟雾消息 SM，建模该优先级依赖为 $\{?SM\} \multimap\!\circ \{?MM\}$。优先级依赖存在于需要使用优先级控制的系统中，使用调度策略对系统的执行顺序进行控制。

2.3 通信建模

依赖结构定义了同步通信、异步通信和广播通信[16]。在同步通信条件下，接收事件依赖于发送事件。如图 2 所示，当烟雾传感器发送同步消息 SM 到控制中心时，转换依赖 $\{!SM\} \rightarrow \{?SM\}$ 就是这方面的例子。在异步通信中，接收事件可以先存储事件，然后再让存储的事件发送消息。例如，$\{!SM\} \rightarrow \{!?SM\}$ 和 $\{!?SM\} \rightarrow \{?SM\}$ 表示的就是异步通信。

2.4 辅助说明

为了能够进行完整的系统建模，还需要定义初始可用事件集、终止可用事件集、输入事件集和输出事件集等概念，其中初始可用事件集表示系统开始可用的事件集，如图 2 所示，因为传感器只有在异常发生的情况下才能感应到信号，所以初始可用集为 $\emptyset$。

输入和输出事件集表示系统和环境交互时所发送和接收的消息。图 2 中输入事件集是 $\{\{DS\},\{WS\},\{GS\},\{MS\},\{SS\}\}$，该系统没有发送消息到其它系统，所以输出事件集为 $\emptyset$。

终止事件集表示系统或者子系统停止运行时的事件集合。图 2 中终止事件集是 $\{\{Cal\},\{Rin\},\{Spr\}\}$。

该系统可以看作是包含信息和物理组件的混合系统，依赖结构可以对它的组件之间以及组件与环境之间的交互进行建模。

3 形式化模型

本节将详细介绍依赖结构的形式化模型。

3.1 依赖结构的定义

为了叙述方便，首先给出一些辅助性的记法。假定 M 是事件的集合，$|M|$ 和 $P'(M)$ 分别表示集合 M 中元素的个数和集合 M 的幂集，而且 $P(M)=P'(M)\backslash\{\emptyset\}$。

定义 2 依赖结构(DS)是一个九元组 $\langle M, I, \mathcal{T}, \mathcal{S}, \mathcal{C}, \mathcal{P}, Out, In, \mathcal{F}\rangle$，其中：1) M 是非空、有限的事件集合；2) $I \subseteq P'(M)$ 是初始可用的事件集的集合；3) $\mathcal{T} \subseteq P(M)\times P(M)$ 是转换关系，且 $\forall A,B\in P(M): (A,B)\in \mathcal{T} \Rightarrow A\not\subseteq B \wedge A\not\supseteq B$；4) $\mathcal{S} \subseteq P(M)\times P(M)$ 是同步关系，且 $\forall A,B\in P(M): (A,B)\in \mathcal{S} \Rightarrow A\subset B$；5) $\mathcal{C} \subseteq P(M)\times P(M)$ 是选择关系，且 $\forall A,B\in P(M): (A,B)\in \mathcal{C} \Rightarrow A\Leftrightarrow B$；6) $\mathcal{P} \subseteq P(M)\times P(M)$ 是优先级关系，并且满足：$\forall A,B\in P(M): (A,B)\in \mathcal{P} \Rightarrow (\exists A'\in P(M): (A,A')\in \mathcal{T}\cup\mathcal{S}\cup ? \vee (A',A)\in \mathcal{T}\cup\mathcal{S}\cup\mathcal{C}) \wedge (\exists B'\in P(M): (B,B')\in \mathcal{T}\cup\mathcal{S}\cup\mathcal{C} \vee (B',B)\in \mathcal{T}\cup\mathcal{S}\cup\mathcal{C})$；7) $Out\subseteq P'(M)$ 是输出接口中事件集的集合；8) $In\subseteq P'(M)$ 是输入接口中事件集的集合，并且 $I\cap In=\emptyset$ 和 $Out\cap In=\emptyset$；9) $\mathcal{F}\subseteq P'(M)$ 最后可用的事件集的集合。

这里，$\forall A,B\subseteq M, (A,B)\in \mathcal{T}$(或 $\mathcal{S}$，$\mathcal{C}$，$\mathcal{P}$)被称为是转换依赖(或同步依赖、选择依赖、优先级依赖)，分别记作 $A\rightarrow B$(或 $A \twoheadrightarrow B$，$A \multimap B$，$A \multimap\!\circ B$)，都读作“B 依赖 A”，并且 A 和 B 分别称作依赖 (A,B) 的前置事件集和后置事件集。通常，$\forall B\subseteq P(M)$，前置事件集 $^{\cdot}B=\{X\in P(M)\mid (X,B)\in \mathcal{T}\cup\mathcal{S}\cup\mathcal{C}\}$，后置事件集 $B^{\cdot}=\{X\in P(M)\mid (B,X)\in \mathcal{T}\cup\mathcal{S}\cup\mathcal{C}\}$。为了叙述方便，事件集 $A\in In$(或 $A\in Out$)称作依赖 (A,B) 的输入事件集(或输出事件集)，A 中的任何事件称为输入事件(或输出事件)。

为区分事件集间的依赖关系，该模型将依赖关系分为以下 4 类：转换依赖、同步依赖、选择依赖和优先级依赖。该分类思想基于进程代数的组合运算、Petri 网的同步等待机制(对应于同步依赖)和事件结构中的事件依赖(顺序和互斥)关系(对应转换依赖和选择依赖)[18]。优先级依赖用于建模调度策略以及控制转换、同步和选择依赖的先后执行顺序。如果先前被提到的依赖在事件中不存在，那么这些事件之间的关系是独立的(即并发的)。因此，依赖结构可以建模并发系统。

3.2 执行语义

依赖结构的执行语义模拟了所建模系统的执行过程。如前文所述，除优先级依赖外，依赖结构中的每一个依赖均对应一个活动(操作或动作)，表示从一个事件集演化为另一个事件集，也就是系统中的一个执行步骤。一个依赖在它的前置集可用的前提下，则被称为已激活。只有已激活的依赖才能执行。为此，将执行过程中每个中间步骤的可用事件集的集合，以及当前已激活的依赖作为一个整体，称为一个“状态”。

定义 3 设 $DS=\langle M, I, \mathcal{T}, \mathcal{S}, \mathcal{C}, \mathcal{P}, Out, In, \mathcal{F}\rangle$ 是一个依赖结构，DS 的一个状态 S 是一个二元组 $\langle\Delta,\Gamma\rangle$，其中，$\Delta\subseteq P'(M)$ 是可用事件集的集合，$\Gamma\subseteq \mathcal{T}\cup\mathcal{S}\cup\mathcal{C}$ 是已激活的依赖的集合，并且满足 $\forall (A,B)\in\Gamma\Rightarrow A\in\Delta$。$DS$ 的初始状态 $S_0=\langle\Delta_0,\Gamma_0\rangle$，其中，$\Delta_0=I$ 和 $\Gamma_0=\{(A,B)\mid A\in I, (A,B)\in \mathcal{T}\cup\mathcal{S}\cup\mathcal{C}\}$。

显然，对于一个依赖结构的初始状态，由于所有的依赖都还未执行，它由所有初始情况下可用的事件集以及由这些初始可用的事件集作为前置集的依赖构成。

不同类型的依赖在执行语义上存在一些差异。当转换依赖被执行后，它们的后置集就变得可用。同步依赖被执行前，所有同步的前置事件集必须都可用，并且对应的同步依赖已激活。对于拥有同一个后置集的两个同步依赖而言，若其中的一个执行，另外一个也必须执行，否则任何一个都得不到执行，同步依赖必须一起执行。选择依赖则表示互斥，只要其中的一个选择依赖执行，拥有相同前置集的其余的选择依赖就成为不可激活的依赖。

另一方面，可以通过从外部环境中的输入事件来改变系统的状态。当系统被外部事件触发，该事件变为可用事件集，

从而使该输入事件集作为前置集的依赖被激活。所有能够得到执行的已激活依赖必须用优先级依赖来加以控制，也就是让优先级依赖来决定已激活依赖的执行顺序。为了方便，将一个可以使输入事件集 W 可用的触发事件记作“$re(W)$”，称“接收到一个事件集 W”。由此，给出了以下的定义。

定义 4 设 $DS=\langle M,I,T,S,C,P,Out,In,F\rangle$ 是一个依赖结构。

(1)对于 DS 中的状态 $S=\langle\Delta,\Gamma\rangle$，如果存在依赖 $(A,B)\in\Gamma$ 满足以下条件时：

① $(A,B)\in T\cup C$，或 $(A,B)\in S\wedge \bigcup_{W\in {}^*B} W=B$，其中 ${}^*B=\{V\in P(M)\mid (V,B)\in\Gamma\cap S\}$；

② $\exists C,D\in P(M):(C,D)\in\Gamma\wedge(D,B)\in P$；

③ $\exists E,F\in P(M):(E,F)\in\Gamma\wedge(E,A)\in P$；

则称 $S=\langle\Delta,\Gamma\rangle$ 是可演化的。

(2)如果存在一个事件集 $W\in In$，使得 $W\notin\Delta$，且存在一个依赖，该依赖的前置集等于 W，则称 $S=\langle\Delta,\Gamma\rangle$ 是可输入触发的。

当前可以执行一个或多个依赖的状态被称为可演化的状态。在输入接口不存在可用事件的状态被称为可输入触发状态。接下来给出依赖结构的执行规则。

定义 5 设 $DS=\langle M,I,T,S,C,P,Out,In,F\rangle$ 是一个依赖结构，并且 $S_1=\langle\Delta_1,\Gamma_1\rangle$ 与 $S_2=\langle\Delta_2,\Gamma_2\rangle$ 是 DS 的两个状态。有如下情形：

(1)若 $(A,B)\in\Gamma_1$，S_1 是可演化的，$\Delta_2=\Delta_1\cup\{B\}$，并且

$$\Gamma_2=\begin{cases}(\Gamma_1\backslash\{(A,B)\})\cup\Gamma^n:(A,B)\in T\\(\Gamma_1\backslash\{(X,B)\mid(X,B)\in S\})\cup\Gamma^n:(A,B)\in S\\(\Gamma_1\backslash\{(A,Y)\mid(A,Y)\in C\})\cup\Gamma^n:(A,B)\in C\end{cases}$$

则称 S_1 能通过执行依赖 (A,B) 演化成 S_2，记作 $S_1\xrightarrow{(A,B)}S_2$。其中 $\Gamma^n=\{(B,Y)\mid(B,Y)\in T\cup S\cup C\}$。

(2)若 $W\in In$，S_1 是可输入触发的，且 $\Delta_2=\Delta_1\cup\{W\}$，$\Gamma_2=\Gamma_1\cup\{(W,X)\mid(W,X)\in T\cup S\cup C\}$，则称 S_1 能通过接收事件集 W 演化成 S_2，记作 $S_1\xrightarrow{re(W)}S_2$。

该定义给出了一种计算所有可能状态的方法。

定义 6 设 $DS=\langle M,I,T,S,C,P,Out,In,F\rangle$ 是一个依赖结构，并设 S_0 是 DS 的初始状态。

(1)如果在 DS 中存在一个状态 S，且存在一个状态序列 $S_1,\cdots,S_{n-1}$，使得 $S_0\xrightarrow{d_1}S_1\cdots S_{n-1}\xrightarrow{d_n}S$，其中 $d_i\in T\cup S\cup C\cup\{re(V)\mid V\in In\}$，$i\in\{1,\cdots,n\}$，则称状态 S 在 DS 中是可达的。如果在 DS 中存在两个状态 S 和 S' 并且存在状态序列 $S_1',\cdots,S'_{n-1}$，使得 $S'\xrightarrow{d_1'}S_1'\cdots S'_{n-1}\xrightarrow{d_n'}S$，其中 $d_i'\in T\cup S\cup C\cup\{re(V)\mid V\in In\}$，$i\in\{1,\cdots,n\}$，则称状态 S' 可达至状态 S，表示为 $S'\xrightarrow{*}S$。$Sta(DS)$ 表示 DS 中所有可达状态的集合。

(2)设 $S=\langle\Delta,\Gamma\rangle\in Sta(DS)$，且 $\exists S'\in Sta(DS):S'\xrightarrow{d}S$。如果 $d=(A,B)\in T\cup S\cup C$ 或 $d=re(W)\in\{re(V)\mid V\in In\}$，则称 B 或 W 是 S 的“最近事件集”。初始状态 S_0 最近的事件集为空集 $\emptyset$。

(3)设 $S=\langle\Delta,\Gamma\rangle\in Sta(DS)$，且 X 是事件集 S 的最近事件集。如果 $\Gamma=\emptyset$，并且 S 的最近事件集 $X\in F$，则称 S 是终止状态。如果在 DS 中存在一个状态 S 不是终止的，不是可演化的，也不是输入触发的，则称 S 是不可终止的。

(4)如果存在一个依赖序列 $\sigma=d_1\cdots d_n$，其中 $d_i'\in T\cup S\cup C\cup\{re(V)\mid V\in In\}$，$i\in\{1,\cdots,n\}$，使得 $S_0\xrightarrow{d_1}S_1\cdots S_{n-1}\xrightarrow{d_n}S_n$，则称 σ 是依赖结构 DS 的一个迹。$|\sigma|=\{d_1,\cdots,d_n\}$，表示迹的所有依赖集合。$Tr(DS)$ 表示 DS 的所有迹的集合。

(5)DS 中含有死锁当且仅当 $\exists S\in Sta(DS):S$ 是不可终止的。相反，$\exists S\in Sta(DS):S$ 是不可终止的，称 DS 是无死锁的；即在 DS 中任何状态都是可终止的，则称 DS 是无死锁的。对于所有的状态 $S\in Sta(DS)$，S 是终止状态，或存在终止状态 $S_t\in Sta(DS)$ 使得 $S\xrightarrow{*}S_t$，则称 DS 是弱终止的。

作者所在研究小组已经在文献[19,20]中对系统中死锁的存在与否进行了研究，并提出了一个可达性算法用于检测死锁。

4 调度

本节首先定义调度和调度策略，然后定义调度策略的正确性。

定义 7 设 $DS=\langle M,I,T,S,C,P,Out,In,F\rangle$ 是一个依赖结构。

DS 中的执行过程 $\pi=A_1\cdots A_n$ 称为调度当且仅当 $(A_j,A_{j+1})\in P(j=1,\cdots,n-1)$，$\exists A\in P(M):(A,A_1)\in P$ 和 $\exists B\in P(M):(A_n,B)\in P$。$[\pi]$ 表示调度 π 中的优先级依赖集，$[\pi]=\{(A_1,A_2),\cdots,(A_{n-1},A_n)\}$。

$Sches(DS)$ 表示 DS 中所有调度的集合。集合 $S_P\subseteq Sches(DS)$ 是 DS 中的一个调度策略。

调度是指独立的优先级依赖的执行序列。在该序列中第一个事件集的优先级最高，最后一个事件集的优先级最低。调度策略是调度的集合，如图 2 所示，当 3 个消息 SM、MM 和 GM 同时被接收到时，优先级调度控制的目标是检测哪个信息最先被处理。优先级依赖关系是 $P=\{(\{?WM\},\{?GM\}),(\{?GM\},\{?MM\}),(\{?SM\},\{?MM\}),(\{!SI\},\{!AT\})\}$。因此，存在3 种调度 $\{?SM\}\{?MM\}$、$\{?WM\}\{?GM\}\{?MM\}$ 和 $\{!SI\}\{!AT\}$。集合 $\{\{?SM\}\{?MM\},\{?WM\}\{?GM\}\{?MM\},\{!SI\}\{!AT\}\}$ 称为一个调度策略。

命题 1 设 $DS=\langle MI,T,S,C,P,Out,In,F\rangle$ 是一个依赖结构。π 是 DS 中的一个调度。

(1)$\langle M,I,T,S,C,[\pi],Out,In,F\rangle$ 是一个依赖结构；

(2)$P=\bigcup_{\pi\in Sches(DS)}[\pi]$。

证明：由调度和优先级依赖关系的定义可知，该命题显然成立。

在设计时，即使是正确(弱终止)的，没有考虑优先级依赖时也会有错误发生。如图 2 所示，如果没有优先级依赖循环，显然它是弱终止的，但是图 3 中的家庭安防系统有优先级依赖循环 $\{(\{?GM\},\{?SM\}),(\{?SM\},\{?MM\}),(\{?MM\},\{?GM\})\}$。如果依赖 $\{!SM\}\rightarrow\{?SM\}$、$\{!MM\}\rightarrow\{?MM\}$ 和 $\{!GM\}\rightarrow\{?GM\}$ 都是已激活的，那么事件集 $\{?SM\}$、$\{?MM\}$ 和 $\{?GM\}$ 将同时可用，根据定义 4 和定义 5，这 3 个依赖无法执行。根据定义 6 可知，包含 3 个优先级依赖的状态就是不可终止的，也就是该家庭安防系统不是弱终止的。根据定义 6 含有优先级依赖循环 $\{(\{?GM\},\{?SM\}),(\{?SM\},\{?MM\}),(\{?MM\},\{?GM\})\}$ 的调度就是不正确的。同时优先级依赖 $\{GM\}\ ?\ ?SM\}$ 还控制了两个同步依赖 $\{?SM\}\twoheadrightarrow\{?SM,?GM\}$ 和

$\{?GM\}\twoheadrightarrow\{?SM,?GM\}$，这两个同步依赖需要同时执行。这个例子中的调度是不正确的。

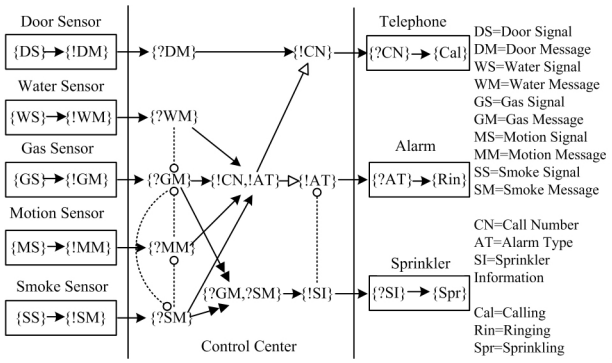

图 3　不正确的调度控制

传统的方法采用有无死锁来判断一个调度是否正确，然而系统中存在相互影响的多种调度，所以不能只考虑单个调度的正确性。因此这里提出调度策略正确性的定义。系统中的调度策略需要正确地执行到终止状态，所以采用弱终止作为调度策略正确性与否的判定标准。

定义 8　设 $DS=\langle M,I,T,S,C,[\pi],Out,In,F\rangle$ 是一个依赖结构。一个调度策略 $S_p(S_p\subseteq Sches(DS))$ 正确，当且仅当 $\langle M,I,T,S,C,\bigcup_{\pi'\in S_p}[\pi'],Out,In,F\rangle$ 弱终止。

命题 2　一个依赖结构 DS 是弱终止的，当且仅当 $Sches(DS)$ 是正确的。

证明：由命题 1 和定义 8 中调度策略的正确性可知，该命题显然成立。

该命题表明一个弱终止的依赖结构对应于一个正确的调度策略，反之亦然。

5　调度的分解

构建复杂的系统通常需先分解系统，把一个大的系统分解为若干子系统，然后分别构建分解后的子系统。这可以成为解决中心化控制的前提，因为中心化控制存在可扩展性、单点失败和网络连接等方面的问题，去中心化是解决这些问题的一种方法。去中心化的首要步骤是分解系统，即把一个大的系统分解为若干子系统。因此，本节将讨论调度策略的分解。下面先介绍一些新的符号和定义。

定义 9　设 $DS_1=\langle M_1,I,T,S,C,P,Out,In,F_1\rangle$ 和 $DS_2=\langle M_2,I,T,S,C,P,Out,In,F_2\rangle$ 是两个依赖结构。

(1)如果 $M_1\subseteq M_2$，$I_1\subseteq I_2$，$T_1\subseteq T_2$，$S_1\subseteq S_2$，$C_1\subseteq C_2$，$P_1\subseteq P_2$，$Out_1\subseteq Out_2$，$In_1\subseteq In_2$，$F_1\subseteq F_2$ 和 $I_1\cup In_1\neq\varnothing\wedge F_1\neq\varnothing$，则 DS_1 是 DS_2 的子结构，表示为 $DS_1\prec DS_2$。

(2)如果 $((T_1\cup S_1\cup C_2)\cap P_2=\varnothing)\wedge((T_2\cup S_2\cup C_2\cap P_1=\varnothing)$，当 $M'=M_1\cup M_2$，$I'=I_1\cup I_2$，$T'=T_1\cup T_2$，$S'=S_1\cup S_2$，$C'=C_1\cup C_2$，$P'=P_1\cup P_2$，$Out'=Out_1\cup Out_2$，$In'=In_1\cup In_2$ 和 $F'=F_1\cup F_2$ 时，DS_1 和 DS_2 的并集 $DS_1\uplus DS_2=Out',In',F'\rangle$。

(3)如果 $M_1\subseteq M_2$(或 $M_1\supseteq M_2$)，$I_1=I_2$，$T_1=T_2$，$S_1=S_2$，$C_1=C_2$，$P_1=P_2$，$Out_1=Out_2$，$In_1=In_2$，$F_1=F_2$，那么 DS_1 和 DS_2 是等价的，即表示为 $DS_1=DS_2$；否则，DS_1 和 DS_2 不等价，表示为 $DS_1\neq DS_2$。

两个独立依赖结构等价是指它们具有相同的依赖关系，否则就是不等价的关系。

命题 3　假设 DS_1，DS_2 和 DS_3 是依赖结构 DS 的 3 个子结构，它们满足如下性质：

(1)(**自反性**)$DS_1\prec DS_1$。

(2)(**传递性**)$DS_1\prec DS_2\wedge DS_2\prec DS_3\Rightarrow DS_1\prec DS_3$。

(3)(**幂等性**)$DS_1\uplus DS_1=DS_1$。

(4)(**交换性**)$DS_1\uplus DS_2=DS_2\uplus DS_1$。

(5)(**结合性**)$(DS_1\uplus DS_2)\uplus DS_3=DS_1\uplus(DS_2\uplus DS_3)$。

证明：根据定义 9 中子结构和并集的定义容易证明上述命题，具体证明过程不再详述。

为了实现分解调度策略的方法，首先定义子结构的控制完全性，以及调度策略的独立性。

定义 10　假设 DS 是一个依赖结构，并且是 $DS=\langle M_1,I_1,T_1,S_1,C_1,P,Out_1,In_1,F_1\rangle$ 的一个子结构。DS_1 是控制完全的，当且仅当 $DS_2=\langle M_2,I_2,T_2,S_2,C_2,P_2,Out_2,In_2,F_2\rangle\prec DS$，$I_1=I_2$，$T_1=T_2$，$S_1=S_2$，$C_1=C_2$，$P_1\subset P_2$，$Out_1=Out_2$，$In_1=In_2$，$F_1\subseteq F_2$。

子结构满足控制完全性是指具有相同的初始和终止可用集，相同输入和输出集以及相同的转换、同步和选择依赖的最大的优先级依赖。该定义保证了一个独立功能的子结构包含所有的调度策略。例如，图 4(b)中的系统就是一个控制完全的子系统。如果删除图 4(b)中的优先级依赖 $\{?SM\}\multimap\{?MM\}$，那么就不是控制完全的子系统。

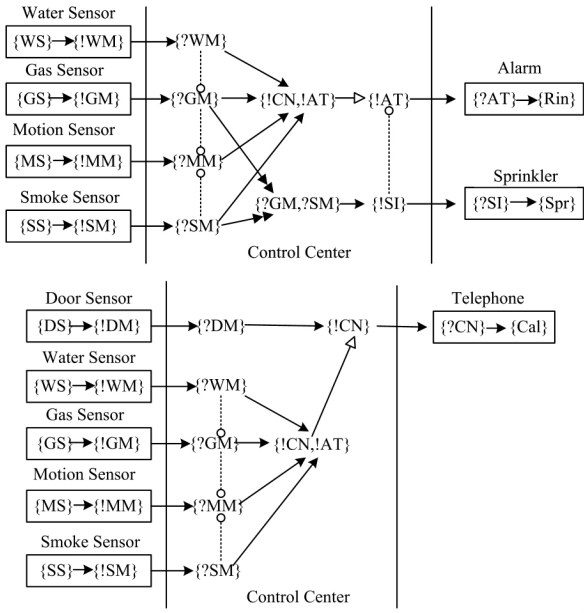

图 4　两个独立的子系统

定义 11　DS 是一个依赖结构并且 $Sches(DS)\neq\varnothing$。在 DS 中调度策略 S_p 是独立的，当且仅当 $\exists DS'\prec DS$，DS' 是控制完全的，$S_p=Sches(DS')$ 是正确的和 $\exists\sigma\in Tr(DS):\sigma\notin Tr(DS')$。

该定义提供了使用独立调度策略来分解的方法。在该定义下，一个独立调度策略和一个子系统是相匹配的。为了保证分解的正确性，每个由复杂系统分解出来的子系统都应该具有独立的功能。

定理 1　假设 DS 是一个弱终止的依赖结构。如果存在一个独立调度策略 S_p，那么就会存在两个弱终止依赖结构 DS_1，$DS_2\prec DS(DS_1\neq DS,DS_2\neq DS)$，且 $DS_1\uplus DS_2=DS$ 和 $Sches(DS)=Sches(DS_1)\bigcup Sches(DS_2)$。

证明：因为 S_p 是独立的，由定义 11 可知，存在一个独立结构 $DS' \prec DS$ 使得 DS' 是控制完全的，在 DS' 中的 S_p 是正确的和 $\exists \sigma \in Tr(DS): \sigma \notin Tr(DS')$。根据命题 2，$DS'$ 是弱终止的。由定义 6，可以得到 DS' 中所有终止迹的集合，使用 $TerTr(DS')$ 表示。显然，$\exists \sigma \in Tr(DS): \sigma \notin Tr(DS'), DS' \neq DS$。同理，因为 DS 是一个弱终止的依赖结构，可以得到 DS 中所有终止迹的集合，使用 $TerTr(DS)$ 表示。

因为 DS' 是控制完全的，调度策略 $Sches(DS')$ 包含了 DS' 中所有的优先级依赖，所以 DS' 是 DS 中的一个独立子结构。由于 $\exists \sigma \in Tr(DS): \sigma \notin Tr(DS')$，可以得到另外一个独立子结构 DS''，原因如下：1) $\sigma \in Tr(DS'')$，DS'' 含有在 $TerTr(DS)/TerTr(DS')$ 中的终止迹；2) DS'' 具有 $TerTr(DS)/TerTr(DS')$ 中可执行的依赖和参与的优先级依赖的集合；3) DS'' 含有 DS' 中没有的但是在 DS 中的其它的依赖和事件集，因此得到 $DS' \uplus DS'' = DS$。因为 DS' 是控制完全的，由定义 10 可得 DS' 中的调度是 DS 中的调度，同时 DS'' 中的调度也是 DS 中的调度，所以 $Sches(DS) = Sches(DS') \bigcup Sches(DS'')$。又因为 DS 是弱终止的，由定义 6 可知，DS 中的迹能够执行到终止状态，所以 DS'' 是弱终止的。显然，$\exists \sigma \in Tr(DS): \sigma \notin Tr(DS''), DS'' \neq DS$。

定理 1 表明如果在系统中存在一个独立的调度策略，不但存在两个独立功能的子系统以及这两个子系统的并集就是该系统，而且两个子系统中的调度和整个系统的是相同的。事实上，在系统中的每个独立调度策略对应的就是一个子系统。

以家庭安防系统为例，利用上述方法将其分解为多个具有独立调度策略的子系统。例如，图 2 表示的家庭安防系统的最大调度策略是：$Sches(safehome) = \{\{?SM\}\{?MM\}, \{?WM\}\{?GM\}\{?MM\}, \{!SI\}\{!AT\}\}$，显然该系统是弱终止的。如图 4(a)所示，它的子系统是弱终止的，该子系统的调度策略 $Sches(subsafehome_1) = \{\{?SM\}\{?MM\}, \{?WM\}\{?GM\}\{?MM\}, \{!SI\}\{!AT\}\}$ 是控制完全的，所以调度策略 $\{\{?SM\}\{?MM\}, \{?WM\}\{?GM\}\{?MM\}, \{!SI\}\{!AT\}\}$ 是独立的。图 4(b)所示的子系统也是独立的。因此，家庭安防系统能够分为图 4 所示的 2 个弱终止子系统。

按照该方法可以将系统分解成两个子系统，以此类推可以将两个子系统分解成更小的子系统，一直到该子系统无法再分解成具有独立调度策略的子系统为止。例如，图 4(b)中的子系统可以分解成图 5(a)和图 5(b)所示的两个子系统。因此图 2 中的家庭安防系统可以分解成 3 个子系统(图 4(a)、图 5(a)和图 5(b))。

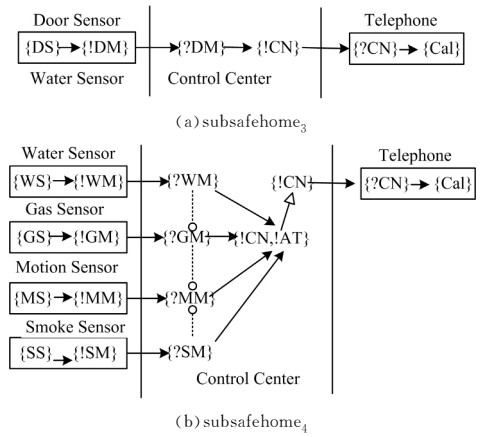

图 5　两个更小的子系统

上面的例子有助于了解定义 11。两个调度策略$\{\{?SM\}\{?MM\}, \{?WM\}\{?GM\}\{?MM\}, \{!SI\}\{!AT\}\}$和$\{\{?SM\}\{?MM\}, \{?WM\}\{?GM\}\{?MM\}\}$各自对应图 4(a)和图 5(b)的两个子系统。$\{!SI\}\{!AT\}$调度间接依靠$\{?SM\}\{?MM\}$和$\{?WM\}\{?GM\}\{?MM\}$这两个调度，这些调度策略之间是彼此关联的，因此它们之间不独立。调度策略$\{\{?SM\}\{?MM\}, \{?WM\}\{?GM\}\{?MM\}, \{!SI\}\{!AT\}\}$对应的是图 4(a)中所示的安防子系统 $subsafehome_1$。显然 $\exists \sigma \in Tr(safehome): \sigma \notin Tr(subsafehome_1)$。因此调度策略$\{\{?SM\}\{?MM\}, \{?WM\}\{?GM\}\{?MM\}, \{!SI\}\{!AT\}\}$是独立的。

尽管图 5(b)表示的子系统($subsafehome_4$)中的两个调度$\{?SM\}\{?MM\}$和$\{?WM\}\{?GM\}\{?MM\}$有相互依赖的共同的事件集$\{?MM\}$，但是它们可以独立运行，可以执行事件集$\{!CN, !AT\}, \{!CN\}$和$\{?CN\}$，最后系统会到达可用事件集$\{Cal\}$。同时 $\exists \sigma \in Tr(safehome): \sigma \notin Tr(subsafehome_4)$ 成立。因此，调度策略$\{\{?SM\}\{?MM\}, \{?WM\}\{?GM\}\{?MM\}\}$独立。显然该分割系统的方法是符合实际情况的，分割后的子系统和原系统有相同的功能，而且满足相同的需求。

6　工具支持

本研究小组已经开发了一种名为 DAT 的工具，该工具具有分解系统的功能。

根据对调度策略分解的分析，本研究小组开发了一种基于可达性[21,22]的分解算法。该算法先要查找所有的终止迹，接着按照一定的规则对终止迹进行分组。该规则如下所述：1)单个调度不能被分解；2)有终止迹的调度策略包含有一组或多组调度；3)含有相同终止迹的调度组成独立的调度策略。根据相同终止迹的组合和对应的具有独立调度策略的规则得到独立的子系统。如果存在一个非优先级依赖循环，依赖中关于循环的终止迹只能出现一次，这样可以避免出现无限的迹。该方法保证了分解算法的正确性。

算法以 XML 格式的文件为输入，以图的广度优先遍历为基础，结合调度的分解算法加以改进。通过对最终实现的算法进行分析可知，该算法的时间复杂度为指数级，空间复杂度为指数级。

DAT 工具可以对调度进行分解，图 6 所示的是家庭安防系统中调度的分解。

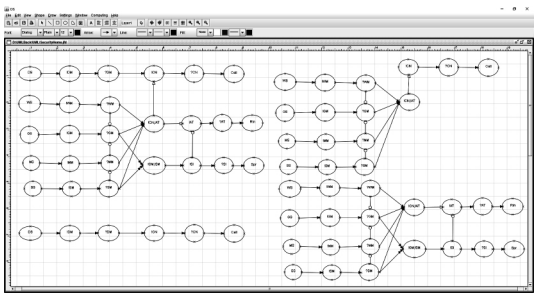

图 6　调度的分解

结束语　本文采用依赖结构模型对系统进行建模，定义了调度、调度策略和调度策略正确性的概念，研究了调度策略的分解方法，并证明了调度策略分解方法的正确性。本文的结论可以成为系统中的设计原则并且指导复杂系统的开发。

本研究小组的目标是发展一个运用于系统中的调度理论，本文的工作只是将来长期工作的一部分。在未来的工作

(下转第 535 页)

表 2　实验结果

故障原因	Bootflag	OSflag	Status	N
A 片引导程序故障	0x0B	0x01	0x01	1
B 片引导程序故障	0x0B	0xFF	0x0F	3
存储器单字单比特错误	0x0A	0x01	0x01	1
#1 单字多比特错误	0x0A	0x02	0x01	1
#1#2 单字多比特错误	0x0A	0x03	0x01	1
#1#2#3 单字多比特错误	0x0B	0xFF	0x0F	3

结束语　本文基于抗辐照龙芯设计了一种新型高可靠星载计算机结构，可靠性仿真结果满足星载计算机在轨工作 10 到 15 年的可靠性要求。为减小单粒子效应的影响，采用 EDAC 技术及通过定期刷新存储数据提高了星载计算机存储数据的可靠性。针对星载计算机因受空间环境的影响而在启动过程中可能会出现的问题，容错启动方案利用硬件冗余资源并且结合软件冗余和编码技术，保证星载计算机能够可靠地引导启动。图 2 所示切换电路使系统能够在引导程序出错或存储器故障时从备份 NOR FLASH 引导星载计算机启动。并通过对 NAND FLASH 采用硬件三模冗余及对操作系统和应用程序代码进行软件备份，确保能引导正确的操作系统及加载正确的应用程序。如何将在轨软件重构和容错启动有效地结合是进一步研究的方向。

参 考 文 献

[1] 龙芯中科技术有限公司. 龙芯 1E 处理器用户手册 [R]. 北京：龙芯中科技术有限公司，2011

[2] Bentoutou Y. A Real Time EDAC System for Applications On-board Earth Observation Small Satellites [J]. IEEE Transactions on Aerospace and Electronic Systems，2012，48(48)：648-657

[3] Saleh A M，Serrano J J，Patel J H. Reliability of Scrubbing Recovery-techniques for Memory Systems [J]. IEEE Transactions on Reliability，1990，39(1)：114-122

[4] Maestro J A，Reviriego P. Reliability of Single-error Correction Protected Memories [J]. IEEE Transactions on Reliability，2009，58(1)：193-201

[5] Li Y，Nelson B，Wirthlin M. Reliability Models for SEC/DED Memory with Scrubbing in FPGA-based Designs [J]. IEEE Transactions on Nuclear Science，2013，60(4)：2720-2727

[6] 支天，杨海刚，蔡刚，等. 嵌入式存储器容错方案可靠性评估 [J]. 微电子学，2015，45(2)：275-280

[7] 辛宁，邱乐德，张立华，等. 一种星载计算机操作系统容错引导算法研究 [J]. 宇航学报，2013，34(6)：818-823

[8] Koren I，Krishana C M. 容错系统 [M]. 杨志，唐宏，王海龙，等，译. 北京：国防工业出版社，2015：10-25

[9] 邹逢兴，张湘平，龙志强，等. 计算机应用系统的故障诊断与可靠性技术基础 [M]. 北京：中国水利水电出版社，2012：166-184

[10] Sweetman D. MIPS 体系结构透视 [M]. 李鹏，鲍峥，石洋，等译. 北京：机械工业出版社，2007：38-89

(上接第 528 页)

中，将致力于如下研究：

(1)并发系统中是否有其它的元素影响调度策略；

(2)当调度控制的条件给定后能够自动产生调度策略；

(3)在不同的层级上获取能够满足限制条件的最优化调度策略。

参 考 文 献

[1] Wing J，et al. Cyber-Physical Systems Summit summary report [R]. the April 2008 CPS Summit，St. Louis，MO，2008

[2] Alur R，Courcoubetis C，et al. The algorithmic analysis of hybrid systems[J]. Theoretical Computer Science，1995，138：3-34

[3] Bliudze S，Sifakis J. The algebra of connectors：structuring interaction in BIP[C]// EMSOFT. ACM，2007：11-20

[4] Eidson J C，Lee E A，Matic S，et al. Distributed Real-Time Software for Cyber-Physical Systems[J]. Proceedings of the IEEE，2012，100(1)：45-59

[5] Jiang J，Zhang S，et al. Configuring business process models[J]. ACM SIGSOFT Software Engineering Notes，2013，38(4)：1-10

[6] Jiang J，Zhu H，et al. Configuration of Services based on Virtualization[C]// TASE. 2014：178-184

[7] Tan Y，Vuran M C，Goddard S，et al. A concept lattice-based event model for cyber-physical systems[C]// ICCPS. 2010：50-60

[8] Talcott C. Cyber-Physical Systems and Events[M]. Software-Intensive Systems and New Computing Paradigms. Berlin，Heidelberg：Springer-Verlag，2008：101-115

[9] Bornot S，Ga G，Sifakis G J. On the construction of live timed systems[M]. Graf S，Schwartzbach M. eds. ，2000：109-126

[10] Altisen K，Gössler G，Sifakis J. Scheduler modeling based on the controller synthesis paradigm [J]. Journal of Real-Time Systems，Special Issue on Control Theoretical Approaches to Real-time Computing，2002，23(1/2)：55-84

[11] Zhang F，Szwaykowska K，Wolf W，et al. Task scheduling for control oriented requirements for cyber physical systems[C]// Real-Time Systems Symposium. 2008：47-56

[12] Li Q. Scheduling in Cyber-Physical Systems[J]. Dissertations，2012：91

[13] Li Q，Negi R. Maximal scheduling in wireless ad hoc networks with hypergraph interference models[J]. IEEE Transactions on Vehicular Technology，2012，61(1)：297-310

[14] Tang Q，Gupta S K S，Varsamopoulos G. A Unified Methodology for Scheduling in Distributed Cyber-Physical Systems. ACM Trans[J]. Embedded Comput. Syst. ，2012，11(S2)：57

[15] Hennessy M，Lin H. Symbolic Bisimulations[J]. Theor. Comput. Sci. ，1995，138(2)：353-389

[16] Jiang J，Zhang S，et al. Modeling and analyzing mixed communications in service-oriented trustworthy software[J]. Science China Information Science，2012，55(12)：2738-2756

[17] Lanese I，Bedogni L，Di Felice M. Internet of things：a process calculus approach[J]. SAC，2013：1339-1346

[18] Winskel G，Nielsen M. Models for Concurrency，Handbook of Logic in Computer Science，Semantic Modelling[J]. Oxford Science Publications，Oxford，1995，4：1-148

[19] Jiang J，Zhang S，et al. Configuring business process models[J]. ACM SIGSOFT Software Engineering Notes，2013，38(4)：1-10

[20] Jiang J，Zhang S，et al. Message Dependency-Based Adaptation of Services[J]. APSCC，2011：442-449

[21] Finkel A，Leroux J. Recent and simple algorithms for Petri nets [J]. Software & Systems Modeling，2015，14(2)：719-7251

[22] Folschette M，Paulevé L，Magnin M，et al. Sufficient Conditions for Reachability in Automata Networks with Priorities[J]. Theoretical Computer Science，2015，608：66-83