计算机科学2000Vol.27№.1

基于转换系统的广义反应系统形式模型

Formal Models of Generalized Reactive Systems Based on Transition Systems

张广泉

（重庆师范学院计算机科学系　重庆400047）（中国科学院软件所　北京100080）

Abstract This paper presents computation models of the three levels(reactive, real-time and hybrid system)for generalized reactive systems by extends traditional transition systems. For each of the three models(FTS, TTS, PTS),we analysis their properties.

Keywords Reactive systems, Formal models, FTS, TTS, PTS

1. 引言

反应系统(Reactive Systems)是与其环境保持不断交互作用的复杂软件系统，而不是在计算终止时产生一些最终结果[1]。并发和实时程序、嵌入式系统、操作系统、通信网络及飞行控制系统等都是这类系统的典型例子，如何保证反应系统的正确性和可靠性是具有挑战性的重要课题，我们认为通过建立系统的形式化模型，对其计算行为进行描述和分析，可以深入认识系统的本质特性，有助于解决上述问题。

随着对反应系统研究的不断深入，反应系统被分为三个不同的抽象层次[2]：第一、最抽象的定性(狭义的)反应式系统，这类系统的模型仅能刻画时间的时态次序，反映两个离散事件之间的先后(发生)次序；第二、定量的反应式系统——实时系统，这类系统的模型能对时间进行定量分析，反映两个事件发生间隔的物理时间；第三、最细最具体的混合系统，这是一种离散部件和连续部件融合在一起的反应系统，如数字控制器、机器人、容错飞行控制系统等。目前还没有任何一种模型能同时满足这三个不同抽象层次的反应系统，基于此，本文通过对基本转换系统进行相应的扩充，分别给出三类反应系统的计算模型，并分析它们各自不同的特点。

2. 基本转换系统模型与分析

2.1 基本转换系统模型

基本转换系统(Basic Transition System，简称BTS)最早是R. M. Keller于1976年提出的并发程序的一种抽象模型，该模型是建立在一个基本的一阶语言之上的[3]。一个基本转换系统是：

$S=(V,\Sigma,\Theta,T)$

其中：$V=\{u_1,\cdots,u_n\}$是系统变量的有穷集合，分数值变量和控制变量两类；

$\Sigma=\{s_0,s_1,\cdots\}$是状态的非空集合，每一状态$s\in\Sigma$是V的一个解释，即给每一变量$u\in V$赋予相应的值，用$s[u]$表示；

$\Theta\subseteq\Sigma$是初始条件，为描述系统开始时运行状态的断言，若一个状态$s\in\Sigma$满足Θ(记作$s\models\Theta$)，则称s为初始状态；

$T=\{\tau_0,\tau_1,\cdots\}$是转换的有穷集合，每一转换$\tau\in T$描述系统的一个状态转换动作，用函数$\tau:\Sigma\to 2^{\Sigma}$表示，将每一状态$s\in\Sigma$映射到$\Sigma$的一个$\tau$-后继状态集$\tau(s)$上。

一个转换τ在状态s下是能行的(enabled)，当且仅当$\tau(s)\neq\varnothing$，否则称τ在此状态下是不能行的；τ_I称为空转换，如果对每一状态$s\in\Sigma$，$\tau_I(s)=\{s\}$；一个状态$s\in\Sigma$称为终止状态，如果有空转换τ_I在s上是能行的。每一转换τ都由一个叫做转换关系的断言$\rho_\tau(V,V')$表示，它将状态$s\in\Sigma$与它的τ-后继状态$s'\in\tau(s)$通过无上撇号和有上撇号的状态变量联系起来，此时，转换τ的能行条件可表示为：$En(\tau):(\exists V')\rho_\tau(V,V')$。

转换系统S上的一个计算(也称执行序列或运行)是状态的一个无穷序列$\delta:s_0,s_1,\cdots$，当且仅当δ满足下列三个条件：

①初始化条件：$s_0\models\Theta$，即s_0是初始状态；

②连续性：对每一$j=0,1,\cdots$，状态s_{j+1}是状态s_j的τ-后继状态，即存在一个转换$\tau\in T$，$s_{j+1}\in s_j$，此时我们称转移τ在位置j被执行；

③勤奋性：δ中要么包含无穷多个非τ_I转换，要么包含一个终止状态。

一个计算通常用 $s_0 \xrightarrow{\tau_0} s_1 \xrightarrow{\tau_1} s_2 \rightarrow \cdots$ 来表示，一方面表示 $s_0, s_1, \cdots$ 是一个计算，同时也说明转换 $\tau_i \in T$，使系统从状态 s_i 转换到 s_{i+1}。

2.2 基本转换系统 BTS 模型分析

作为并发/反应系统的一种最常用的基本模型，BTS 具有如下几个特点：

第一、BTS 是一种状态转换模型，即给出系统（程序）的状态集合以及对应状态间变迁的转换（也称操作）的集合，通过描述系统的行为来体现系统具有的性质，是一种构造性的形式化建模方法（也称为是基于模型的方法）。

第二、BTS 本质上是一个交替并发计算模型。即系统的并发性是通过交替执行来表示的，进程间的并发执行不是真并发（这一点不同于 Petri 网），其计算模式是一个线性状态序列，两个并发进程永远不会在同一时间点上同时执行它们的语句，而是轮流执行其原子转换，因而 BTS 适用于并发程序的形式化分析和处理[1]。

第三、BTS 提供了反应系统计算模型的基础框架，为不同机制语言的统一处理提供了一种抽象机制。它可以映射到许多具有不同语法表示及通信/协调机制（如共享变量、消息传递等）的程序设计语言及其它并发模型（如 Petri 网）结构上；提供 CCS 的基本操作语义和 CSP 的方法论基础，并为时态逻辑用于反应系统规约和验证提供了良好的基础[13]。

3. BTS 的三种扩充模型

3.1 FTS 模型与分析

我们将公平性需求引入传统的转换系统 BTS 中，得到反应系统的一种定性计算模型——公平转换系统（Fair Transition System，简称 FTS）[3,12]如下：

$S=(V,\Sigma,\Theta,T,J,C)$

其中：V,Σ,Θ,T 含义同 BTS；

·$J\subseteq T$ 称为弱公平性转换集合；对转换 $\tau\in J$，不允许一个计算，其中转换 τ 从某一点开始一直能行，但只有有穷多次被执行。

·$C\subseteq T$ 称为强公平性转换集合；对转换 $\tau\in C$，不允许一个计算，其中转换 τ 无穷多次能行，但只有有穷多次被执行。

公平转换系统 S 上的一个计算是一个无穷状态序列 $\delta: s_0, s_1, \cdots$，δ 除满足 BTS 的三个条件（①初始化条件；②连续性；③勘奋性）外，还满足：

④弱公平性：对每一转换 $\tau\in J$，保证不会发生 τ 在 δ 中某一位置以后一直能行，但只有有限多次被执行。

⑤强公平性：对每一转换 $\tau\in C$，保证不会发生 τ 在 δ 中无穷多次能行，但只有有限多次被执行。

作为反应系统的一种定性计算模型，FTS 是一个附加公平性条件的转换系统。所谓公平性是指：如果一个转换（操作）是能行的，那么系统必须最终执行它[4]。FTS 从 BTS 的所有转换中挑选出特殊的子集（J 和 C），对转换实施两类不同的公平性限制：弱公平性和强公平性，从而解决了并发进程间不公平执行的问题。这里弱公平性实际上是对程序中进程的一个极小约束，使得系统不要永远忽略一个在某计算位置以后一直能行的转换的执行；强公平性实际上是对程序中进程的一个较大级别的约束，用以保证程序整体的执行。这种定性的公平性需求反映了时间的抽象概念，刻画了反应系统一个（可观察到的）计算中事件发生的先后次序（公平迹）。文[1,4]对 FTS 的原子转换及其粒度划分施加一种 LCR（限制临界引用）约束，并说明了满足 LCR 条件的 FTS 是可信的，其交替计算与实际系统的重叠执行是一致的。

3.2 TTS 模型与分析

我们将时间约束引入传统的转换系统 BTS 中，得到实时系统的一种定量计算模型——时间转换系统（Timed Transtition System，简称 TTS）如下：

$S=(V,\Sigma,\Theta,T,\iota,\mu)$

其中：V,Σ,Θ,T 含义同 BTS。

·对每一转换 $\tau\in T$，存在一个极小延迟（也称为下界）$\iota_\tau\in R^+$（非负实数），规定 $\iota_{\tau_1}=0$。

·对每一转换 $\tau\in T$，存在一个极大延迟（也称为上界）$\mu_\tau\in R^\infty=R^+\cup\{\infty\}$。

对所有转换 $\tau\in T$，有 $0\leqslant\iota_\tau\leqslant\mu_\tau$，如果 τ 在 Θ 的任何初始状态上均是能行的，则 $\mu_{\tau_1}=\infty$。说明：在 TTS 的最初模型中[5,6]，这是以非负整数 N 作为（离散）时间定义域的，本文将 N 推广至非负实数域 R^+ 上。

为了便于描述 TTS 的计算，我们引入一个特殊变量（称为时钟变量）$T\in R^+$，用于表示系统执行的任意位置的当前时间，$V_T=V\cup\{T\}$ 称为情形（situation）变量集合，每一情形 $(s,t)\in\Sigma_T$（情形集合）是 V_T 的一个解释，这里 $s\in\Sigma$ 是状态，$t\in R^+$ 是时钟 T 的解释。时间转换系统 S 上的一个计算是情形的一个无穷序列 δ：$(s_0,t_0),(s_1,t_1),\cdots$，满足：

①初始化条件：$s_0\models\Theta$ 且 $t_0=0$。

②连续性：对每一 $j=0,1,\cdots$，存在转换 $\tau\in T$，$t_j=t_j+1$ 且 $s_j+1\in\tau(s_j)$，即 τ 在位置 j 被执行或 $s_j=s_j+1$ 且 $t_j<t_j+1$（该步骤称为"滴答步"，表示时间的增长）。

③下界：对每一转换 $\tau\in T$ 和位置 $j\geqslant 0$，如果 τ 在 j 处被执行，则在一个位置 $i, i\leqslant j$，$t_i+\iota_\tau\leqslant t_j$ 且 τ 在 $s_i, s_{i+1}, \cdots, s_j$ 上是能行的；该条件要求 τ 在它被执行之前，至少在 ι_τ 时间单元内是连续能行的。

④上界：对每一转换 $\tau\in T$ 和位置 $i\geqslant 0$，如果 τ 在 i

处能行，则存在一个位置 $j\geqslant i$，$t_j\leqslant t_i+\mu_\tau$ 使得 τ 在 s_j 处不能行，换句话说，τ 在没有被执行前，不可能在多于 μ_τ 时间单元内连续执行。

⑤时间发散：随着 i 的增加，t_i 增长不受任何限制。

与 FTS 不同，TTS 并不要求每一状态至少有一个转换在其上是能行的，这是因为，即使所有转换都不能行，系统也可以通过"滴答步"保证所有计算是无穷的。另外，TTS 以定量的"下界 ι"和"上界 μ"时间约束来取代 FTS 中定性的"公平性需求"（弱公平性 J 和强公平性 C），对每一转换施加不同的极小和极大时间延迟，以保证转换发生既不太早也不太迟，TTS 的计算以情形的无穷序列（又称为时间迹）代替 FTS 的状态无穷序列，对任一情形 (s_i,t_i)，$t_i=s_i[T]$，标记在状态 s_i 上被观察（抽样）的时刻。作为实时系统的一种定量计算模型，TTS 通过时间执行序列描述系统的语义，能够刻画事件发生间隔的物理时间，可以满足各种不同的实时性要求，如超时，进程中断及进程调度等，并可对应到许多具体的实时系统如多处理机系统，多道程序设计系统等[5~8]。

3.3 PTS 模型与分析

我们将连续构件的一些概念引入 BTS 中，得到混合系统的一种形式化模型——阶段转换系统（Phase Transition System，简称 PTS）如下：

$\Phi=(V,\Sigma,\Theta,T,A,I)$

其中：Σ,Θ 含义同 BTS，而 V，T 扩充了 BTS 中相应的概念。

·$V=V_d\cup V_c$ 由离散变量 V_d 和连续变量 V_c 两部分构成，$V_c\in R$（实数，V_d 类型不受限制。

·T 是离散/连续转换的有穷集合，对每一转换 $\tau\in T$，τ 的能行条件 $En(\tau)=En_d(\tau)\wedge En_c(\tau)$，这里 $En_d(\tau)$ 是不依赖于连续变量的最大子公式，$En_c(\tau)$ 称为能行条件的连续部分。

·A 是活动的有穷集合：每一活动 $a\in A$ 对应不同条件的方程：$a_c\rightarrow e_a$，其中 a_c 是 V_d 上的布尔表达式，称为"活动条件"，e_a 是形如 $\dot{x}=e$ 的方程，$x\in V_c$，$\dot{x}$ 表示 x 的右导数，e 是 V 上表达式，一个活动 α 在状态 s 下是可操作的当且仅当 a_c 在该状态上成立。

·I 是重要事件的集合：I 至少包含 $\{En_c(\tau)\}$ 及所有出现在系统形式描述中的断言。

PTS 上的一个抽样计算是一个无穷情形序列 $(s_0,t_0),(s_1,t_1),(s_2,t_2),\cdots$，且满足下列条件：

①初始化条件：$s_0\models\Theta$ 且 $t_0=0$；（同 TTS）

②连续性：对每一 $j=0,1,\cdots$，存在转换 $\tau\in T$，$t_j=t_{j+1}$ 且 $s_{j+1}\in\tau(s)$，即 τ 在 j 上被执行；或 (s_{j+1},t_{j+1}) 是 (s_j,t_j) 的一个活动后继 $(t_j<t_{j+1})$，即一个连续阶段发生在 j 步；

③紧迫性：没有连续步使一个能行的转换不能执行，即：如果转换 τ 是能行的，但每一个有明确长度的连续步将使它变为不能行的，则 τ 必定在任何连续步之前被执行或不能行；

④时间发散：当 i 增加时，t_i 的增长不受任何边界限制。（同 TTS）

随着计算机技术的发展，混合系统的应用范围越来越广泛，在实际中愈来愈重要，对混合系统的研究已成为近几年来控制理论和计算机科学领域中十分重要的课题[7,9~11]。目前，已出现多种混合系统的模型，如：混合自动机、混合 Petri 网、扩充时段演算、TLA＋等，文[7]提出一种基于 FTS 的混合系统的状态转换模型，该模型与我们不同之处是引入一对实数 $l_\tau\leqslant\mu_\tau$ 分别表示每一转换 τ 可被连续执行的时间长度的下界和上界，通过实数对（表示时间）和下标（表示离散步）到状态的函数——混合迹 (δ,δ)，来描述系统模型的混合计算。本节给出混合系统的一种新的计算模型——阶段转换系统 PTS，将连续状态、活动、重要事件等概念引入转换系统，通过混合系统连续行为可观察的时间点的一种抽样——抽样计算来描述系统模型的语义。

参考文献

1 Manna Z, Pnueli A. The Temporal Logic of Reactive and Concurrent Systems: Specification. Springer-Verlag, 1991
2 Manna Z, Pnueli A. Models for reactivity. Acta Informatica, 1993, 30(7): 609~678
3 Manna Z, Pnueli A. How to cook a temporal proof system for your pet language. In: Proc. 10th ACM Symp. on Princ. of Prog. Lang. 1983. 141~154
4 Manna Z, Pnueli A. On the Faithfulness of Formal Models. In MFCS, LNCS520, 1991. 28~42
5 Henzinger T A, et al. Temporal proof methodologies for real-time systems. In: Proc. 18th ACM Symp. on Princ. of Prog. Lang. 1991. 353~366
6 Henzinger T A, et al. Timed Transition Systems. In Real-time: theory in practice. LNCS 600, 1992. 226~251
7 Maler O, et al. From Timed to Hybrid Systems. Same to 6, 1992. 447~484
8 Henzinger T A, et al. Temporal Proof Methodologies for Timed Transition Systems. Inf. Comput. Academic Press, 1994, 112: 273~337
9 Manna Z, Pnueli A. Verifying Hybrid Systems. In Hybrid Systems, LNCS 736, 1993. 4~35
10 Kapur A, et al. Proving Safety Properties of Hybrid Systems. In: FTRTFT'94, LNCS 863, 1994. 431~434
11 de Alfaro L, et al. Hybrid Diagrams: A Deductive-Algorithmic Approach to Hybrid System Verification. In: Proc. 14th Symp. on Theor. Comput. Sci. LNCS1200. 1997. 153~164
12 Zhang G Q（张广泉）. An Extend State Transition Model Based on Temporal Logic. '98 SSTC, 1998. 46~49
13 Zhang G Q（张广泉）. Basic Models, Specification and Verification of Generalized Reactive Systems. ICYCS'99（第5届国际青年计算机学术会议），1999