文章编号: 1001- 2486(2010) 01- 0095- 06

参数化系统二维抽象框架*

屈婉霞, 庞征斌, 郭 阳, 李 暾, 杨晓东
(国防科技大学 计算机学院, 湖南 长沙 410073)

摘 要: 针对参数化系统状态空间爆炸问题提出了一个通用的参数化系统二维抽象框架TDA。对所有进程单独进行抽象, 利用参数化系统的设计思想, 隐藏系统参数构建全系统的抽象模型, 最大限度地剔除了原始系统中的冗余信息。建立的具有真并发语义的参数化系统的形式化模型, 更适合描述一般意义上的并发系统, 较好地解决了验证大规模同构和异构系统的空间激增问题。理论推导和实例均证实了 TDA 的正确性和合理性。

关键词: 参数化系统; 模型检验; 抽象; 多处理机系统; Cache 一致性协议
中图分类号:TP391. 72 文献标识码: A

A Generic Framework of Two-dimension Abstraction for Parameterized Systems

QU Wan-xia, PANG Zheng-bin, GUO Yang, LI Tun, YANG Xiao-dong
(College of Computer, National Univ. of Defense Technology, Changsha 410073, China)

Abstract: To address the state explosion problem in model checking parameterized systems, this paper proposes a generic framework of two-dimension abstraction (TDA), in which the size of the state transition graph for individual process is reduced independently at first, and the whole system composed of reduced processes is then abstracted based on the design method of parameterized systems, thus avoiding the construction of the unreduced model that might be too big to fit into memory. Formal model with true concurrency semantics for parameterized systems, more suitable for describing general concurrency systems, is introduced, which effectively solves the problem of verifying both homogeneous and heterogeneous systems. Results from the theoretical reasoning and the example given demonstrate that TDA is correct and feasible.

Key words: parameterized system; model checking; abstraction; multiprocessor; Cache coherence protocol

参数化系统由一组结构相同、协同运行的进程组成(进程个数称为系统参数, 因为系统参数可变, 参数化系统本质上是一种无界系统) , 这类具有重复结构的系统在现实中非常普遍, 如通信协议、Cache 一致性协议和互斥协议等。如何高效验证参数化系统是当前验证领域一项非常具有挑战性的工作, 一方面, 参数化系统中每个进程的行为不仅依赖于自身的状态, 同时还依赖于外界环境的行为, 参数化系统是由这些相互交互的进程并发组合而成的一个复杂有机体; 另一方面, 参数化系统验证的一个最根本问题是要检验给定的性质是否在包含任意多个进程的系统中也成立。系统参数的任意性和进程之间交互的复杂性导致了参数化系统的状态空间激增。

针对上述问题, 研究人员提出了多种解决方案, 谓词抽象[1-2] 是应用最成功的技术之一。目前, 参数化系统的抽象研究采用的方法主要分两类: 一类是设法将无界系统转换为有界系统[3-4], 但是此方法不能处理进程行为非常复杂的系统; 另一类方法采用分而治之的策略, 先对单个进程进行抽象, 然后考虑进程之间的交互, 曾红卫[5] 和 Igor[6] 在这方面进行了积极尝试, 曾红卫的方法仅限于处理同步并发系统, 而 Igor 则假定进程的状态是有限的。

本文提出了一个通用的参数化系统二维抽象框架 TDA(Two-dimension Abstraction) , 对所有进程单独

* 收稿日期:2009- 05- 18
基金项目:国家自然科学基金资助项目(60573173,60773025) ; 新世纪优秀人才支持计划资助项目
作者简介:屈婉霞(1972—) , 女, 助理研究员, 博士。

进行抽象,然后利用参数化系统的设计思想隐藏系统参数,构建全系统的抽象模型。TDA由相互独立的$Y-$抽象和$X-$抽象两个子过程组成,通过分解—抽象—组合—再抽象,最大限度地剔除了原始系统中的冗余信息,降低了问题的复杂度。

1 具有真并发语义的参数化系统模型

根据进程之间交互方式的不同,参数化系统通常分为同步并发和异步并发两类。异步并发系统的行为表现为进程动作的交织运行,但是,大多数实际的异步并发系统允许进程同时动作,它们之间没有先后顺序关系,保留了并发的真正含义,本文称之为具有真并发语义的异步并发,研究这类系统更具理论和实际意义。

本文用Kripke结构$M=(AP,S,I,R,L)$描述进程,进程状态是有限状态变量集合V上的一个解释,其中,V^i为内部变量集合,V^e为外部变量集合,并且满足$(V^i\cup V^e=V)\wedge(V^i\cap V^e=\cong)$,具有真并发语义的异步并发参数化系统模型可通过合成运算得到,两个Kripke结构必须满足一定条件才能进行组合。

定义1 相容结构(Compatible Structure):给定两个Kripke结构$M_1=(AP_1,S_1,I_1,R_1,L_1)$和$M_2=(AP_2,S_2,I_2,R_2,L_2)$,如果$V_1^i\cap V_2^i=\cong$并且$V_1^e=V_2^e$,则$M_1$和$M_2$是相容的Kripke结构。

定义2 相容状态(Compatible State):$M_1=(AP_1,S_1,I_1,R_1,L_1)$和$M_2=(AP_2,S_2,I_2,R_2,L_2)$是两个相容的Kripke结构,如果$L_1(s_1)\cap AP_2=L_2(s_2)\cap AP_1$,则状态$s_1\in S_1$和状态$s_2\in S_2$是相容的。

定义3 真并发异步合成(Asynchronous Composition with True Concurrency Semantics):设$M_k=(AP_k,S_k,I_k,R_k,L_k)$是$n$个相容的Kripke结构中的第$k(k\in[1..n])$个,这$n$个结构的真并发异步合成$M=\mathop{\|}\limits_{k=1}^{n}{}_a M_k$是如下定义的一个Kripke结构$(AP,S,I,R,L)$:

(1) $AP=\bigcup\limits_{k=1}^{n}AP_k$;

(2) $S=\{<s_1,s_2,\cdots,s_n>\mid s_k\in S_k(k\in[1..n])$是相容的状态$\}\subseteq\prod\limits_{k=1}^{n}S_k$;

(3) $I=\{<s_1,s_2,\cdots,s_n>\mid\bigwedge\limits_{k=1}^{n}s_k\in I_k\}\subseteq S$;

(4) $R=\{(<s_{1,i},s_{2,i},\cdots,s_{n,i}>,<s_{1,i+1},s_{2,i+1},\cdots,s_{n,i+1}>)\mid\exists j,1\leqslant j\leqslant n,(s_{j,i},s_{j,i+1})\in R_j\}$;

(5) $L(<s_1,s_2,\cdots,s_n>)=\bigcup\limits_{k=1}^{n}L_k(s_k)$。

后文所述异步并发均指具有真并发语义的异步并发。

2 二维抽象框架

设参数化系统包含n个进程,假定所有进程结构完全相同,每个进程的状态数为m,图1分析了影响异步并发参数化系统状态空间规模的因素,m个正方形叠加而成的长方形表示进程的状态空间,灰色正方形表示进程的当前状态,长方形的数目表示进程数,连接n个长方形的连续折线表示一个全局状态。由于允许多个进程同时发生状态迁移,系统的全局状态数最多可达m^n个。本文提出采用对m和n构成的二维空间进行抽象处理的方法:一方面对单个进程的状态空间进行抽象,即沿Y方向对矩形的高度进行压缩,另一方面对系统的参数n进行抽象,设法用有限状态描述无限状态,即沿X方向进行抽象。

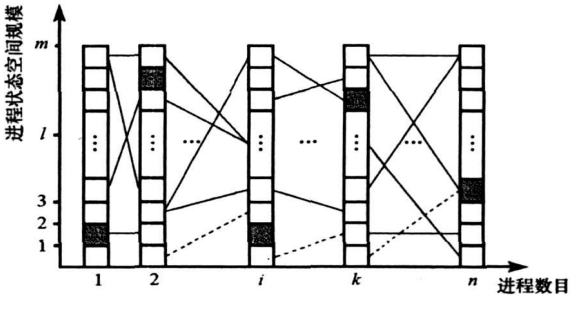

图1 异步并发参数化系统状态空间规模的影响因素

Fig. 1 Factors resulting in state space explosion in asynchronous parallel parameterized systems

定义 4　二维抽象框架TDA 对于异步并发参数化系统而言, 首先对所有进程的状态空间进行独立压缩, 然后用有限状态描述方法“隐藏”系统参数, 这种分别从 Y 和 X 两个方向对系统进行抽象的方法, 称为二维抽象框架TDA。前者称为 $Y-$ 抽象, 得到的模型称作 $Y-$ 抽象模型, 后者称为 $X-$ 抽象, 得到的模型称作TDA 模型。

为保证验证的正确性, 需要在TDA 模型和原始模型之间建立一种等价关系, 模拟关系[7] 通过限制逻辑以及放松结构必须满足恰好相同公式集的要求, 有可能获得较大的状态缩减。如果存在一个模拟关系 H, 使得对 M_1 中的每一个初始状态 $s_{10} \in I_1$, 在 M_2 中都存在一个初始状态 $s_{20} \in I_2$ 满足 $H(s_{10}, s_{20})$, 则称 M_2 模拟 M_1, 记作 $M_1 \preceq M_2$。

后文用 $PS^c(n)$ 表示由 n 个进程组成的异步并发参数化系统的原始模型, $PS^y(n)$ 表示对 $PS^c(n)$ 进行 $Y-$ 抽象后得到的模型, $PS^i(n)$ 表示对 $PS^y(n)$ 进行 $X-$ 抽象后得到的模型。

2.1　$Y-$ 抽象

从系统性质和协议描述中提取谓词集合 $\Phi = \{\varphi_1, \varphi_2, \ldots, \varphi_r\}$, 并用标准的谓词抽象方法进行抽象。谓词在进程 $M_k^c = (AP_k^c, S_k^c, I_k^c, R_k^c, L_k^c)$ $(k \in [1..n])$ 的状态集合 S_k^c 上定义一个等价关系, 这个等价关系对 S_k^c 进行划分, 一个等价类用一个抽象状态表示。原始状态用该状态满足的谓词公式 φ 标记, 标记函数 L_k^c 是从原始状态到谓词集合的映射。相应的 $Y-$ 抽象模型 $M_k^y = (AP_k^y, S_k^y, I_k^y, R_k^y, L_k^y)$ 的状态集合 S_k^y 是变量 $b_1, b_2, \ldots, b_r$ (b_j $(j \in [1..r])$ 与谓词 φ_j 对应) 上的一组正交布尔表达式, 抽象状态是 r 个布尔变量的某个赋值, 标记函数 L_k^y 是从抽象状态到布尔表达式的映射。原始状态和抽象状态之间的关系由抽象算子 H_k^{φ} 确定。

从抽象状态的构造过程、标记函数 L_k^c 和 L_k^y 的定义可知, $H_k^{cy} \subseteq S_k^c \times S_k^y$ 是 M_k^c 与 M_k^y 之间的一个模拟关系, 即有下述结论成立:

定理 1　$M_k^c \preceq M_k^y$ $(1 \leqslant k \leqslant n)$。

定理 2　异步合成运算 $\|_a$ 关于 $\preceq$ 单调, 即: $M_k^c \preceq M_k^y$ $(k \in [1..n]) \Rightarrow PS^c(n) \preceq PS^y(n)$。

$\|_a$ 关于 $\preceq$ 的单调性保证了 $Y-$ 抽象模型关于 ACTL* 公式弱保持, 因此有下列定理成立:

定理 3　对于每一个 ACTL* 公式 f $(AP_f \subseteq AP^y)$, $PS^y(n) \models f \Rightarrow PS^c(n) \models f$。

验证大规模参数化系统时, 定理 3 非常有用。如果一个 ACTL* 公式 f 在 $PS^y(n)$ 中成立, 那么 f 在 $PS^c(n)$ 中也成立, 而 $PS^y(n)$ 具有较小的状态空间, 因此有助于加快验证速度。

2.2　$X-$ 抽象

设计人员在设计参数化系统时, 通常以一个进程(记为 *hub*) 为主, 考虑与其他进程(记为 *rim*) 的交互, 以此简化设计过程。设 $PS^y(n)$ 包含 $n-1$ $(n>1)$ 个 *rim* 和 1 个 *hub*, 所有 *rim* 构成 *hub* 的外界环境, 第 k $(k \in [1..n-1])$ 个 *rim* 的 Kripke 结构为

$$M_k^y = (AP_k^y, S_k^y, I_k^y, R_k^y, L_k^y)$$

hub 的 Kripke 结构为

$$M_h^y = (AP_h^y, S_h^y, I_h^y, R_h^y, L_h^y)$$

则 $PS^y(n)$ 是 M_k^y 和 M_h^y 的异步合成, 即

$$PS^y(n) = (AP^y, S^y, I^y, R^y, L^y) = (\|_{k=1}^{n-1}{}_a M_k^y) \|_a M_h^y$$

将

$$L^y(< s_1^y, \ldots, s_k^y, \ldots, s_{n-1}^y, s_h^y >) = \bigcup_{k=1}^{n-1} L_k^y(s_k^y) \cup L_h^y(s_h^y)$$

展开得到:

$$L^y(< s_1^y, \ldots, s_k^y, \ldots, s_{n-1}^y, s_h^y >) = L_1^y(s_1^y) \cup L_2^y(s_2^y) \cup \ldots \cup L_k^y(s_k^y) \cup \ldots \cup L_{n-1}^y(s_{n-1}^y) \cup L_h^y(s_h^y)$$

可以发现, 等号右侧的每一项是相应进程的状态标记, 即该进程的当前状态满足的所有原子命题,

这些原子命题反映了进程具备的性质。

定义5 进程性质(Process Property):公式 $l(k), k\in[1..n-1,h]$ 表示进程 k 满足性质 l。k 满足的所有性质用 $L(k)=\{l(k)\}$ 表示。进程性质也称标记(label)。

借助进程性质的概念,$Y-$ 抽象模型的全局状态标记可以简化:

$L^y(<s^y_1,\ldots,s^y_k,\ldots,s^y_{n-1},s^y_h>)=L(1)\cup\ldots L(k)\cup\ldots L(n-1)\cup L(h)=\{l(d),s^y\vDash l(d),d\in[1..n-1,h]\}$

即 $Y-$ 抽象状态 s^y 的标记是它满足的所有的进程性质。

定义6 描述(Description):描述 $\delta(k)\triangleq l(k)\wedge(\bigwedge_{j\neq k}l(j))$ 是一个涉及多个进程的复杂公式,不仅刻画了进程 k 的性质,而且刻画了 k 的环境具备的性质。所有描述用 $D=\{\delta(k)\}$ 表示。

$\delta(k)$ 给出了 $PS^y(n)$ 上的一个等价类划分所满足的条件:如果 s^y 中存在一个进程 d 满足性质 $\delta(k)$,即 $s^y\vDash\delta(d)$,则 $\delta(k)$ 是 s^y 的一个抽象状态。所有满足上述条件的 s^y 是一个等价类。

为了能够构造TDA模型,标记集合 L 和描述集合 D 必须满足覆盖性和一致性。覆盖性是指每一个 $Y-$ 抽象状态均能由某些抽象状态 $\delta(k)$ 反映出来,一致性是指 $\delta(k)$ 包含详细的信息,足以判断一个进程性质是否在一个描述中成立。如果 $\delta(k)\Rightarrow l(k)$,则抽象状态 $\delta(k)$ 具有标记 $l(k)$。

给定满足覆盖性和一致性条件的描述集合 D 和标记集合 L,$PS^y(n)$ 经过 $X-$ 抽象后得到的TDA模型是一个Kripke结构 $PS^t=<AP^t,S^t,I^t,R^t,L^t>$,其中,

(1) $AP^t=AP^y$ 是构成进程性质的原子命题集合;

(2) $S^t=D$ 是TDA状态空间:抽象算子 $\alpha_h(s^y)=\{\delta(k)\in D\mid s^y\vDash\delta(h)\}$,将所有 *hub* 进程满足 $\delta(k)$ 的 $Y-$ 抽象状态 s^y 映射到TDA抽象状态 $\delta(k)$。覆盖性条件保证了TDA抽象状态集合 $\alpha_h(s^y)$ 非空;

(3) I^t 是TDA初始抽象状态集合:如果存在一个 $PS^y(n)$、一个 $Y-$ 抽象状态 $s^y\in I^y$ 满足 $\delta(k)\in\alpha_h(s^y)$,则 $\delta(k)\in I^t$;

(4) L^t 是标记函数:对于 $\forall\delta(k)\in S^t$,$L^t(\delta(k))=\{l(k):\delta(k)\Rightarrow l(h)\}$,即用 *hub* 进程满足的性质 $l(k)$ 标记TDA抽象状态 $\delta(k)$;

(5) R^t 是TDA抽象状态迁移关系集合:对于 $\forall\delta_1(k),\delta_2(k)\in S^t$,如果存在一个 $PS^y(n)$、两个 $Y-$ 抽象状态 $s^y\in S^y,t^y\in S^y$ 满足 $\delta_1(k)\in\alpha_h(s^y)\wedge\delta_2(k)\in\alpha_h(t^y)\wedge(s^y,t^y)\in R^y$,则 $(\delta_1(k),\delta_2(k))\in R^t$。

定理4 对于单索引ACTL* 公式 $\forall x.\varphi(x)$,$\varphi(x)$ 的原子公式是 L^t 中的标记,那么 $PS^t\vDash\varphi(x)\Rightarrow\forall n.PS^y(n)\vDash\forall x.\varphi(x)$。

2.3 TDA的正确性与合理性

TDA的正确性是指TDA模型相对原始模型关于单索引ACTL* 公式弱保持。定理2~4共同保证了TDA的正确性。同时,定理4指出了TDA是合理的,即在TDA模型中成立的单索引ACTL* 公式在任意规模的原始模型中都成立。

3 二维抽象实例分析

考虑由3个处理器构成的分布共享多处理机系统 $PS^c(3)$,通过MESI协议实现共享数据访问的一致性。每个Cacheline至少包括存储器地址(*addr*)、Cache状态(*achestate*)和Cache数据(*cachedata*)三个域。若限定 *cachedata* 值域为0和1,则单个处理器关于一个存储器块的MESI状态机虽然只有7个有效状态,但状态迁移关系已多达30个,如图2左图所示。如果允许 *cachedata* 和 *addr* 在其有效的值域内取值,不可能画出对应的状态机。

假定要验证系统性质"处理器共享一个存储器块时,允许某个其他处理器中没有此存储器块的备份",即 $\forall p,\exists q.(p.cachestate=S\wedge p\neq q\Rightarrow q.cachestate=I)$。根据TDA的定义,首先对单处理器的MESI协议进行 $Y-$ 抽象。系统性质只涉及存储器块的状态,并不关心存储器块的值,因此存储器块的值是与待验证性质无关的冗余信息,采用谓词抽象删除冗余信息后的MESI协议的Kripke结构如图2右

图所示。根据定义 3, 具有真并发语义的 $Y-$ 抽象模型 $PS^{y}(3)$ 如图 3 左上图所示, 合法状态多达 14 个, 每一个状态是所有处理器中当前存储器块满足的谓词构成的向量。

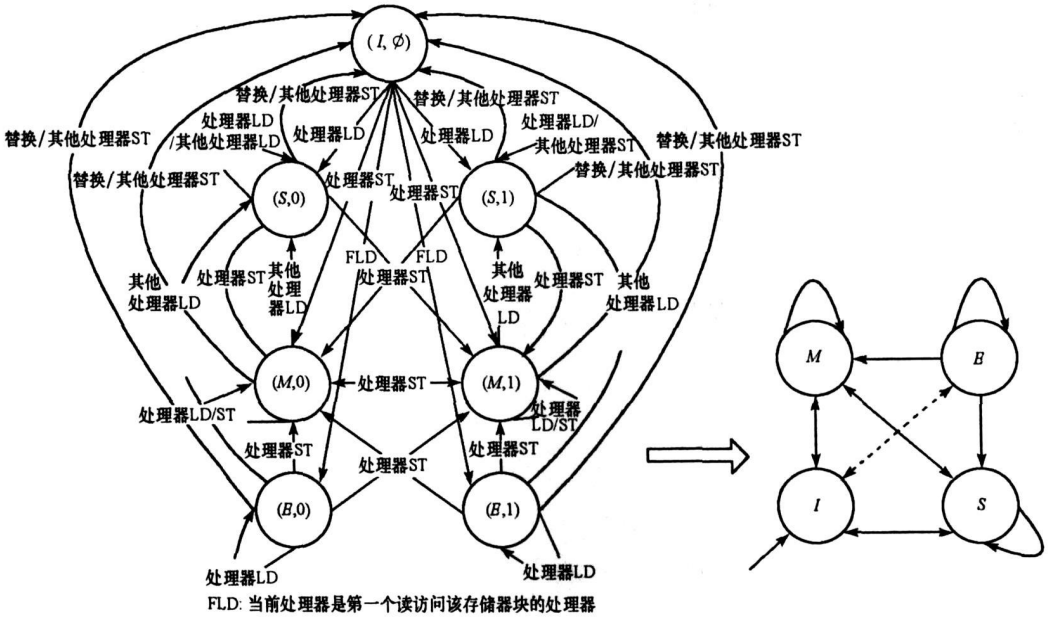

图 2 MESI 状态机和经过 $Y-$ 抽象的 Kripke 结构

Fig. 2 MESI state machine and its Kripke structure reduced by $Y-$abstraction

然后再对 $PS^{y}(3)$ 进行 $X-$ 抽象, 用

$$\delta(k) = (cachestate[k] = S) \wedge (\bigvee_{j \neq k} cachestate[j] = I)$$

表示处理器 k 的 $cachestate$ 为 S 并且存在一个其他处理器的 $cachestate$ 为 I。令

$$l_1(k) \triangleq (cachestate[k] = S),\ l_2(k) \triangleq (I \in env(k))$$

则

$$\delta(1) = l_1(1) \wedge l_2(1),\quad \delta(2) = l_1(2) \wedge l_2(2),\quad \delta(3) = l_1(3) \wedge l_2(3)$$

如果用 $\delta(1)$ 对 $PS^{y}(3)$ 的状态进行划分, 状态分类结果及相应的标记如表 1 所示, 对应的 TDA 模型如图 3 左下图所示, 该模型仅包含 4 个状态, 状态< 11> 标示的等价类包含 $Y-$ 抽象状态 SIS、SSI 和 SII, 表示处理器 1 共享一个存储器块并且处理器 2 或 1 和 3 中没有该存储器块的共享备份, 即系统性质可满足, 证明了 TDA 的正确性。由于对系统参数 n 进行了存在量化, 一个 TDA 模型可以表示满足相同条件的一组不同规模的参数化系统。图 3 右上图是与 $PS^{y}(3)$ 具有相同 TDA 模型的 $Y-$ 抽象模型 $PS^{y}(4)$, 它们具有不同的系统规模, 但满足相同的系统性质, 表明 TDA 是合理的。

表 1 $PS^{y}(3)$ 状态空间按照 $\delta(1)$ 划分的结果

Tab. 1 $PS^{y}(3)$ state space partition according to $\delta(1)$

等价类	等价类标记	标记符号
{SIS, SSI, SII}	$l_1(1) \wedge l_2(1) = \delta(1)$	< 11>
{ISI, III, IEI, IMI, EII, MII, IIE, IIS, IIM}	$\neg l_1(1) \wedge l_2(1) = \neg\delta(1)$	< 01>
{SSS}	$l_1(1) \wedge \neg l_2(1) = \neg\delta(1)$	< 10>
{ISS}	$\neg l_1(1) \wedge \neg l_2(1) = \neg\delta(1)$	< 00>

4 结 论

TDA 从两个方面最大限度地剔除了原始系统中的冗余信息, 对进程状态空间是否有限、进程间的交

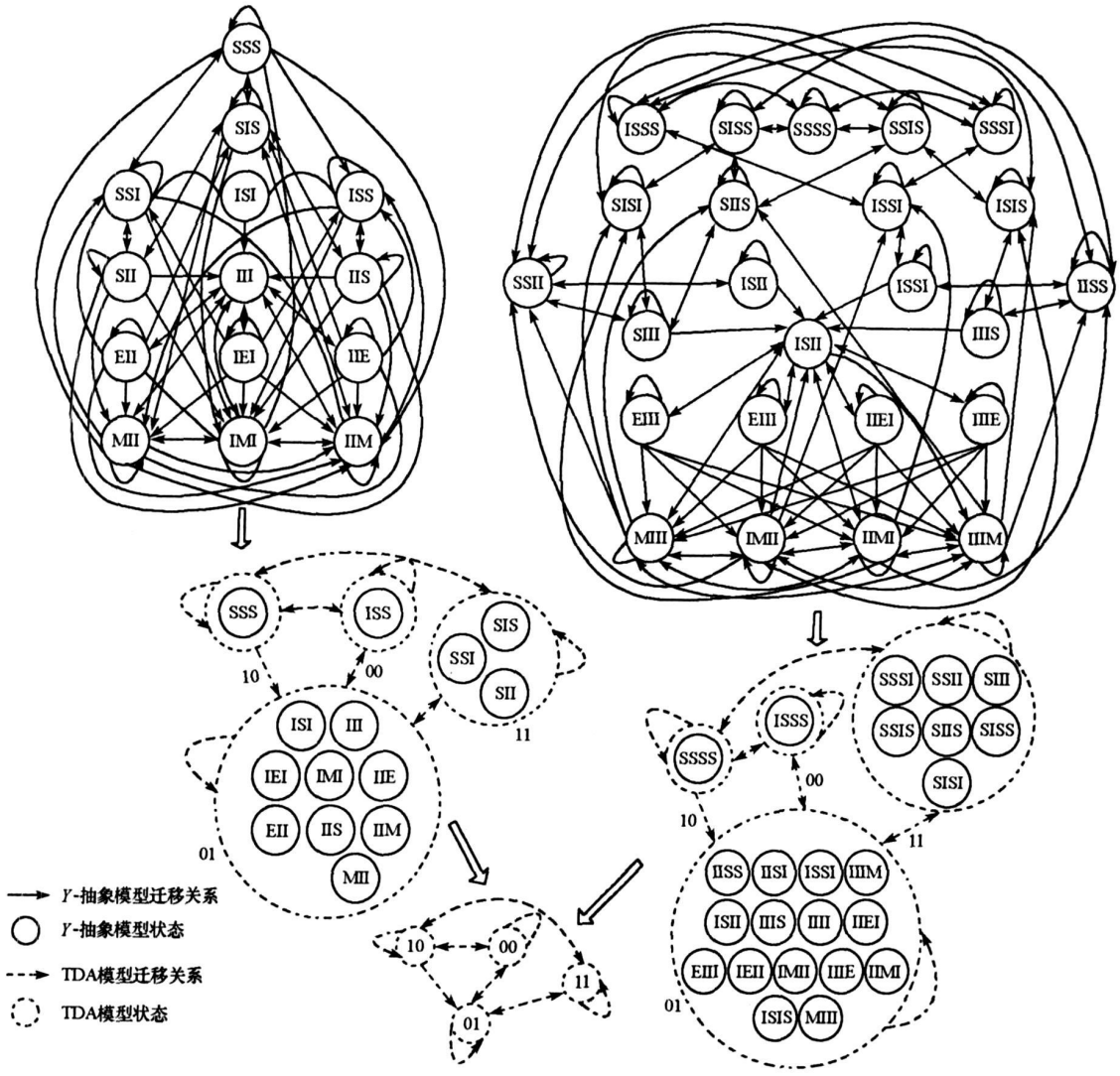

图3 基于MESI协议的 $Y-$ 抽象模型 $PS^{y}(3)$ 和 $PS^{y}(4)$ 及其TDA模型

Fig. 3 Y-model $PS^{y}(3)$, $PS^{y}(4)$ and their TDA-model

互方式均无严格限制,为简化参数化系统的验证提供了理论依据。下一步的工作是实现TDA的自动化,结合现有的模型检验工具对自主设计的参数化Cache协议进行验证。

参考文献:

[1] 屈婉霞, 李暾, 郭阳, 等. 谓词抽象技术研究[J]. 软件学报, 2008, 19(1): 27-38.

[2] Shuvendu K L, Randal E B. Indexed Predicate Discovery for Unbounded System Verification[C]//Proc. of the 16th International Conference on Computer Aided Verification (CAV'04), Boston, USA, 2004.

[3] Amir P, Jessica Xu, Lenore Z. Liveness with (0; 1; ∞) Counter Abstraction[C]//Proc. of the 14th International Conference on Computer Aided Verification 2002 (CAV'02), Copenhagen, Denmark, 2002.

[4] Muralidhar T. Abstraction Techniques for Parameterized Verification[D]. Pittsburgh: Carnegie Mellon University, 2006.

[5] 曾红卫, 缪淮扣. 构件组合的抽象精化验证[J]. 软件学报, 2008, 19(5): 1149-1159.

[6] Igor V K, Vladimir A Z. An Invariant-based Approach to the Verification of Asynchronous Parameterized Networks[C]//Workshop on Invariant Generation (WING 2007), Hagenberg, Austria, 2007.

[7] Milner R. An Algebraic Definition of Simulation Between Program[C]//Proc. of the 2nd International Joint Conference on Artificial Intelligence, William Kaufmann, London, 1971.