基于抽象和组合方法的网络协议验证

陈道喜[1] 张广泉[2] 徐成凯[2] 陈国彬[3]
(苏州技师学院 苏州 215009)[1] (苏州大学计算机科学与技术学院 苏州 215006)[2]
(重庆工商大学融智学院 重庆 400033)[3]

摘 要 由于模型检测存在状态爆炸问题,多主体的网络协议组合模型检测往往难以进行。为了缓解该问题,分析了通信主体数量增加对状态数量的影响,提出了组合式的抽象验证方法。首先根据所需验证的LTL性质,建立各个通信主体的Kripke结构,再对该Kripke结构进行抽象;然后组合抽象模型;最后运用Spin对组合抽象模型进行检验。为验证该方法的有效性,对NSPK协议进行了检测,结果表明,该方法所需的状态空间向量长度、搜索深度、存贮和遍历的状态数都有明显减少,有利于缓解状态爆炸问题。

关键词 Kripke结构,状态爆炸,组合抽象模型,LTL模型检测
中图法分类号 TP311 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2015.7.025

Verification of Network Protocols Based on Abstraction and Composition

CHEN Dao-xi[1] ZHANG Guang-quan[2] XU Cheng-kai[2] CHEN Guo-bin[3]
(Suzhou Senior Technician Institute, Suzhou 215009, China)[1]
(School of Computer Science & Technology, Soochow University, Suzhou 215006, China)[2]
(Rongzhi College, Chongqing Technology and Business University, Chongqing 400033, China)[3]

Abstract Due to the state explosion problem in model checking, it is always impossible to verify the composition model of a multi-agent protocol. To relieve this problem, we analyzed the impact of the increase in the number of agents on that of states and then proposed an approach based on abstraction and composition. Firstly, Kripke structures of individual agents are established according to the LTL properties to be verified, and these structures are abstracted. Then, these abstraction models are composed. Finally, the tool Spin is used to verified the composed model. To validate the efficiency of this approach, we verified the protocol NSPK. The results show that there are significant decreases in the length of state-vector, depth searched and the number of states stored and transitions, which will help relieve the state explosion problem.

Keywords Kripke structure, State explosion, Composition abstraction model, LTL model checking

模型检测是一种验证有限状态并发系统的自动化技术[1]。首先要建立模型,再用形式化方法描述系统应拥有的性质,建立性质的逻辑表达式,通过遍历模型上的所有可达状态,判断系统是否拥有所期望的性质。模型检测已被成功地应用于分析与验证计算机硬件、通信协议、网络协议、控制系统和安全认证等。

模型检测基于对系统状态空间的穷举搜索。对于并发系统,其状态的数目往往随并发分量的增加呈指数增长[2]。模型检测面临的挑战是状态空间爆炸(state explosion)问题,抽象技术[3,4]和组合模型检验[5,6]是应对状态爆炸问题的两种方法。抽象技术的基本思想是:构造抽象模型,缩减状态空间,在抽象模型上对系统性质进行检验。组合模型检验的基本思想是:利用系统的组合结构对问题进行分解,首先检验子系统的性质,然后综合推理导出整个系统的性质,这种分而治之的方法是降低模型检验复杂度的有效手段。文献[7]提出把构件组合模型检验转化为各成分构件的局部的抽象精化,通过单个构件的等价关系的精化实现构件组合的抽象模型精化。

上述组合验证方法突出了构件间的交互行为,所验证性质也存在于交互序列中;但是在网络协议的LTL模型检测中[8],某些LTL性质却在交互序列之外。因此,需对上述组合方法加以扩展,以保证在组合后保留了所需验证的状态。本文模拟了网络通信主体的增加而引发的状态爆炸问题,通过对协议中各个通信主体建立抽象Kripke模型[9],来检验组合抽象模型。运用Spin[10,11]检测抽象前后的Promela模型[12],通过实验数据说明本文方法可以缓解状态空间爆炸。

1 协议验证中的状态爆炸

在基于Spin网络协议的形式化验证中,通信主体的增加会带来两个方面的问题。一方面随着Promela模型中并发分

到稿日期:2014-06-30 返修日期:2014-10-04 本文受江苏省自然科学基金(BK2011281),苏州市应用基础研究计划(SYG201241),江苏省普通高校研究生科研创新计划(CXLX13_820),重庆市教委科学技术研究项目(KJ133103)资助。
陈道喜(1974—),男,硕士,高级讲师,主要研究方向为软件工程与形式化方法;张广泉(1965—),男,博士,教授,CCF高级会员,主要研究方向为软件工程与形式化方法;徐成凯(1989—),男,硕士生,主要研究方向为软件工程与形式化方法;陈国彬(1982—),硕士,讲师,主要研究方向为软件工程。

量的增加，状态数会有明显的增加。当通信双方到一定数量时，就会面临状态爆炸问题。以 Lowe 改进后的 Needham-Schroeder 协议为例，当 Alice 的数量为 1，而 Bob 数量在增加时，存贮状态数、转移状态数的变化如图 1 所示。其中 A 表示 Alice 主体的数量，B 表示 Bob 主体的数量。

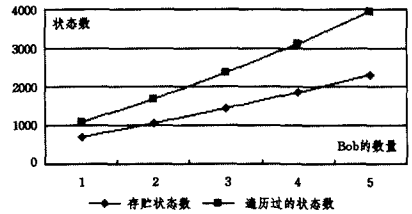

图 1　$A=1, B=n$ 时检测结果

从图 1 中可以看出，随着通信主体的增加，验证性质时所遍历的状态数也有明显增加。另一方面如果两个主体数量同时增加，那么性质的验证也比较复杂，原有的通信序列被打破，同一类主体的发送与接收往往不是同一个主体。

图 2 是在没有 Intruder 的情况下，Alice 与 Bob 数量均为 2 时协议模拟的运行图。从图 2 可以看出，当 Alice0 发送消息 msg1 给 Bob3 时，Bob3 却将回复的 msg2 发给了 Alice1。当两者的数量继续增加时，这种现象更是普遍存在。因此，基于多 agent 的 Promela 模型的检测更为复杂。为解决上述问题，提出对初始模型加以抽象，并组合抽象模型的方法。

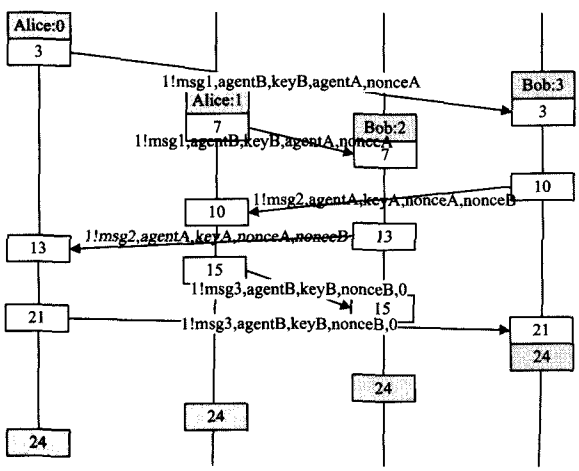

图 2　Alice 和 Bob 的数量均为 2 时模拟运行图

2　抽象与组合验证

抽象的目的是使模型的状态空间减少但保留要验证的性质[7]。在建立协议的形式化模型后，运用抽象和组合方法进行协议的验证。

2.1　协议的形式化模型

协议的模型是协议分析和设计的核心技术之一。协议的主要形式化模型技术有：有限状态机(FSM)、Petri 网和通信进程演算(CCS)等。本文采用 Kripke 结构描述协议通信主体(本文简称 agent)的行为。Kripke 结构[1,9]是一个五元组 (S, S_0, AP, R, L)，其中 S 是一个有限状态集；S_0 是初始状态的集合，并且 $S_0 \subseteq S$；AP 为原子命题的集合；$R \subseteq S \times S$ 为状态转换关系，对于每个 $s \in S$ 都有一个状态 $s' \in S$ 满足 $(s, s') \in R$；$L: S \to 2^{AP}$ 是状态标记函数，将每个状态标记为在该状态下为真的原子命题的集合。

定义 1(agent 的 Kripke 结构)　一个 agent 的行为模型是 Kripke 结构 $M=(S, S_0, AP, R, L)$，其中 $AP=\text{Input} \cup \text{Output} \cup \text{Interior}$，简写为 $AP=\text{In} \cup \text{Out} \cup \text{Ir}$，其中 In，Out，Ir 分别表示向通道中输入、输出和内部动作集合。

例如：以 NSPK 协议为例，为了简化验证，在不妨碍所需验证性质的前提下，假设信息在发送和传输过程中无延迟和丢失，不存在重发现象。根据定义 1，建立通信主体 Alice 的 Kripke 结构如图 3(a)所示，$M_A=(S_A, S_{0A}, AP_A, R_A, L_A)$，其中 $AP_A = In_A \cup Out_A \cup Ir_A$。$S_A=\{a_1, a_2, \cdots, a_{10}\}$，$S_{0A}=\{a_1\}$，$AP_A=\{A\_idle, A\_generate\_msg1, msg1, A\_transmission\_msg1, A\_wait\_2, msg2, A\_generate\_msg3, msg3, A\_transmission\_msg3, A\_ok\}$，其中 A_idle 表示 *Alice* 空闲，A 代表 *Alice*，*generate* 表示生成消息，*transmission* 表示发送消息，*wait* 表示等待消息，A_ok 表示 A 的状态 ok，已经完成协议。$In_A=\{A\_idle, msg1, msg3\}$，$Out_A=\{msg2, A\_ok\}$，$Ir_A=\{A\_generate\_msg1, A\_transmission\_msg1, A\_wait\_msg2, A\_generate\_msg3\}$。$R_A=\{(a_1,a_1),(a_1,a_2),\cdots,(a_9,a_{10}),(a_{10},a_1)\}$。$L(a_1)=\{A\_idle\}$，$L(a_2)=\{A\_generate\_msg1\}$，$L(a_3)=\{msg1\}$，$L(a_4)=\{A\_transmission\_msg1\}$，$L(a_5)=\{A\_wait\_2\}$，$L(a_6)=\{msg2\}$，$L(a_7)=\{A\_generate\_msg3\}$，$L(a_8)=\{msg3\}$，$L(a_9)=\{A\_transmission\_msg3\}$，$L(a_{10})=\{A\_ok\}$。

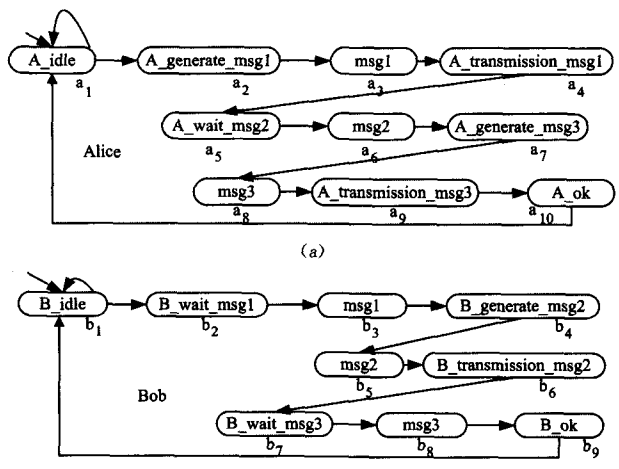

图 3　Alice 和 Bob 的 Kripke 结构图

类似地，可以建立通信主体 Bob 的 Kripke 结构图，如图 3(b)所示。其中 $In_B=\{B\_idle, msg2\}$，$Out_B=\{msg1, msg3, B\_ok\}$，$Ir_B=\{B\_wait\_msg1, B\_generate\_msg2, B\_transmission\_msg2, B\_wait\_msg3\}$。

2.2　agent 和性质的抽象

为减少组合的状态数量，并不是直接组合这两个 Kripke 结构，而是先对它们进行抽象。

定义 2(agent 的抽象)　设 $M^a=(S^a, S_0, {}^aAP^a, R^a, L^a)$ 是 M 的抽象，记为 $M \leq Ma$，当且仅当 $AP^a \subseteq AP$，$\forall s, s' \in S$，$s^a \in S^a$，$R(s,s') \wedge \alpha(s,s^a) \Rightarrow \exists s^{a\prime} \in S^a$，$R^a(s^a, s^{a\prime}) \wedge \alpha(s', s^{a\prime})$，$\alpha$ 是抽象映射。

对 agent 抽象的方法是建立具体状态与抽象状态之间的映射关系，将一些与验证性质无关的状态合并，相应的迁移关系也要进行抽象，相应的性质有时也要抽象。

定义 3(性质的抽象)　对于时序逻辑公式 φ；性质抽象公式 φ^a，当且仅当状态子公式 $p \subseteq \varphi$，$\varphi(p) \Rightarrow \varphi^a(\alpha(p))$，$AP_\varphi$ 为性质 φ 中的原子命题。

例如：在 NSPK 协议中，分别对上述的 Alice 和 Bob 建立抽象模型，假设验证性质：$\varphi=[](($*A_ok*&&*B_ok*$)\rightarrow(($*partnerA*==*agentB*$)$ &&$($*partnerB*==*agentA*$))$，其中$($*partnerA*==*agentB*$)$与$($*partnerB*==*agentA*$)$分别在 *msg*1 中，以$($*flag*==*ok*$)$表示$($*partnerA*==*agentB*$)$&&$($*partnerB*==*agentA*$)$，得到 $\varphi^a=[](($*A_ok*&&*B_ok*$)\rightarrow($*flag*==*ok*$))$。根据定义 2 和定义 3 建立 Alice 与 Bob 的抽象模型，分别如图 4(a)和图 4(b)所示。

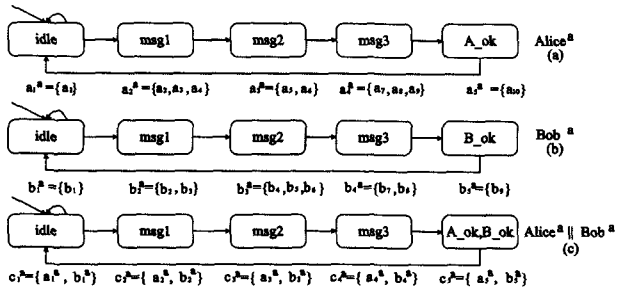

图 4　Alice 与 Bob 抽象组合图

为便于自动化验证，对比抽象前后的状态数的变化，建立 Lowe 改正后的多主体的 NSPK 协议的 Promela 模型和抽象模型。

2.3　抽象组合验证

定义 4(多个 agent 的组合)　设 $M_1,M_2,\cdots,M_i,M_j,\cdots,M_n$ 是可组合的 agent，则它们的同步组合 $M=M_1\parallel M_2\parallel\cdots\parallel M_i\parallel M_j\parallel\cdots\parallel M_n=(S,S_0,AP,R,L)$，$i,j,n\in N^+$，满足 $S=(s_i,s_j)\mid L_i(s_i)\cap AP_j=L_j(s_j)\cap AP_i\}\cup\{AP_\varphi\}$；$S_0=(S_{0i}\times S_{0j})\cap S$；$AP=AP_1\cup AP_2\cup\cdots\cup AP_i\cup AP_j\cup\cdots\cup AP_n$；$((s_i,s_j),(s_i',s_j'))\in R$ 当且仅当$(s_i,s_i')\in R_i,(s_j,s_j')\in R_j$；$L((s_i,s_j))=L_i(s_i)\cup L_j(s_j)$。

本文将文献[7]中的组合方法 $S=\{(s_i,s_j)\mid L_i(s_i)\cap AP_j=L_j(s_j)\cap AP_i\}$扩展成 $S=\{(s_i,s_j)\mid L_i(s_i)\cap AP_j=L_j(s_j)\cap AP_i\}\cup\{AP_\varphi\}$，以保证在组合后保留了所需验证的状态。这是因为在基于 LTL 性质验证中，不仅涉及到 agent 之间的交互消息，而且也涉及到每个 agent 的输出消息。多个 agent 的组合可以构成复杂的协议系统。设 $M_i=(S_i,S_{0i},AP_i,R_i,L_i)$表示第 i 个 agent，其中 $AP_i=In_i\cup Out_i\cup Ir_i$。具体应用到网络协议的分析时，往往涉及到两个以上的 agent 组合。

定理 1(抽象组合)　设 M_1 和 M_2 是两个 Kripke 结构，$M_1\leq M_1^a,M_2\leq M_2^a,\varphi$ 为由 AP 构成的 LTL 公式，$\varphi\leq\varphi^a$。如果 $M_1^a\parallel M_2^a\vdash\varphi^a$，那么 $M_1\parallel M_2\vdash\varphi^a$。

证明：由于 $M_1\leq M_1^a$，那么 $M_1\parallel M_2\leq M_1^a\parallel M_2\leq M_1^a\parallel M_2^a$，即 $M_1^a\parallel M_2^a$ 是 $M_1\parallel M_2$ 的抽象。由定义 2 可知，$\forall s,s'\in S,s^a\in S^a,R(s,s')\wedge\alpha(s,s^a)\Rightarrow\exists s^{a\prime}\in S^a,R^a(s^a,s^{a\prime})\wedge\alpha(s',s^{a\prime})$，$\alpha$ 是抽象映射。已知 $M_1^a\parallel M_2^a\vdash\varphi^a$，则 $M_1^a\parallel M_2^a$ 中所有的路径 $\pi^a\vdash\varphi^a$，那么在 $M_1\parallel M_2$ 中就一定存在对应的路径 $\pi\vdash\varphi^a$。又因为 $\varphi^a\leq\varphi$，那么 $\pi\vdash\varphi$，从而 $M_1\parallel M_2\vdash\varphi$。

由定理 1 可知，由于多 agent 的抽象组合模型是它们的组合模型的抽象，而且检测抽象组合模型又具有较小的探索状态空间，因此通常可以使用多 agent 的抽象组合模型检测。由定义 4 可得到 Alice 和 Bob 两者的抽象组合模型如图 4(c)所示，那些没有实际意义的状态组合被忽略。

3　实例分析

以上述 NSPK 协议为例，参与通信的有合法的主体 Alice 和 Bob 以及入侵者 Intruder，其中 Intruder 的 Kripke 结构类似定义 1。

3.1　协议性质检测结果分析

在没有 Intruder 时，Alice 与 Bob 抽象组合如图 4(c)所示，可以验证性质 φ 是正确的。表 1 列出了在检验性质 φ 时的每个 agent 抽象和组合抽象模型的状态数。其中可能的状态数为 agent 状态数的乘积，有效状态数小于可能状态数。

表 1　实例模型抽象组合状态数

	agent 状态数		agent 组合状态数	
	Alice	Bob	可能状态数	有效状态数
直接组合	10	9	90	<90
抽象组合	5	5	25	5

以 Lowe 改正后的 NSPK 协议为例，抽象前后的模型在 Spin Version 5.1.4 下检验，当 Alice 和 Bob 的数量均为 2 时，实验结果对比如表 2 所列。从表 2 中可以看出状态数量的减少，当 Alice 与 Bob 的数量都增加时，抽象前后的状态数之差会进一步加大，例如：当两者的数量均为 4 时，抽象前存贮状态数为 186017，抽象后存贮状态数为 42957；抽象前遍历的状态数为 508445，而抽象后遍历的状态数为 168049。其余项目的对比均有类似的结果。

表 2　实例模型 Spin 检测状态数

	状态空间向量(byte)	搜索深度	存贮状态数	遍历的状态数
抽象前	64	51	279	491
抽象后	36	47	141	257

真实的网络环境下有入侵者，由于 Intruder 可以根据安全协议分析的假设而设定不同的知识结构，也就是说，Intruder 的能力各不相同，因此，一方面如果对 Intruder 的能力进行限制，例如不能够破译密钥，也不能进行消息存储转发等，则可以验证性质 φ 是正确的。但是另一方面如果Intruder能够任意获取网上传递的消息，并可对消息进行再生成，则可以验证性质 φ 是不正确的，这就是著名 Needham-Sch-roeder协议的 Lowe 攻击。这两种情况下 Alice 和 Bob 以及 Intruder3 者的抽象组合模型分别如图 5(a)和图 5(b)所示，其中一些与验证性质无关的状态合并在 other1 和 other2 中。

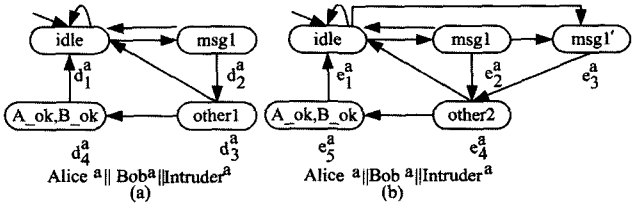

图 5　Alice 和 Bob 以及 Intruder 3 者的抽象组合模型

图 5(a)中 Intruder 不可以对消息进行再生成，只能运用自己的身份标识和自己的密钥，不可以重发自己已经截取的密钥，从 d_1^a 开始到 d_2^a 发送 *msg*1，然后返回到 d_1^a，结束 Intruder 的一次运行。在图 5(b)中 Intruder 可以对消息进行再生成，生成不同于 *msg*1 的新消息 *msg*1′。从而可以看出存在一条路径$(e_1^a,e_3^a),(e_3^a,e_4^a),(e_4^a,e_5^a)$，该路径不满足性质 φ，由基于反例的抽象求精可以判断，在实际状态中存在一条路径与之对应，在这里使用基于反例的抽象求精的目的是去掉伪反例。

3.2　基于多 agent 的安全性和活性的验证

在抽象组合模型基础上，对基于多 agent 的安全性和活

性属性进行验证时，验证结果往往与通信双方每方只有一个agent时的结果并不一致。

基于Promela模型Spin检测安全性和活性时，还可以采取其它一些方面应对状态爆炸问题。如：利用工具Spin选择safety state properties对Promela模型进行安全性检测，由于可采用穷举算法对状态空间进行可达性分析，因此对于大规模的系统，可可采用Bit State Hashing[13]等方法有选择地搜索部分状态空间。一般地，如果安全性质不满足，则状态数不多；而安全性质满足时，状态数量多，状态爆炸的问题显现出来。因此，在检测时，使用Promela中的atomic和d_step，尽量使用同步channel、Spin里的partial order reduction和slicing algorithm等。最重要的是建立的模型既要能反映原协议的时序逻辑，又要足够的抽象，没有冗余计算和冗余数据。利用工具Spin对Promela模型进行活性检测时，可以直接选择liveness对含有标号为progress或accept的模型进行活性验证。

在模拟多个agent运行时，对图2中的时序进行修正，使得两个主体间的通信能够完成协议所规定的要求，则在Promela建模时，要加入对通道的限制如xr和xs，此时运行的模拟如图6所示。从图6中可以看出，一组(Alice0，Bob3)和另一组(Alice1，Bob2)之间3个消息的时序是正确的。

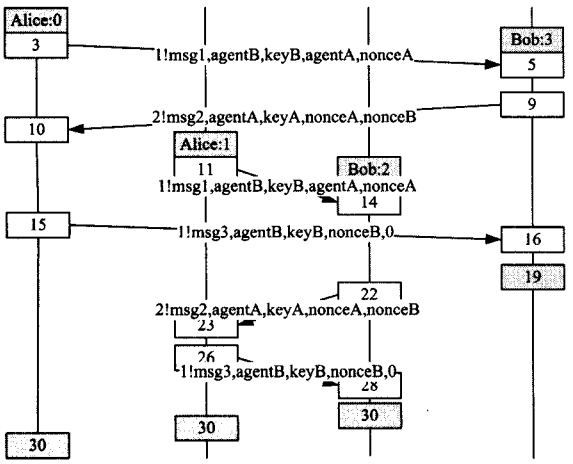

图6　修正后的模拟运行图

结束语　本文分析了通信主体数量增加而引发的状态爆炸问题，提出了组合式的抽象验证方法。根据待验证LTL性质，通过对各个通信主体建立抽象模型，再检验组合抽象模型，对比了组合抽象前后的状态数量的变化。最后通过实例说明，组合抽象模型检测可以使状态数量明显减少，有利于缓解状态爆炸问题。如果既考虑到模型又考虑到了验证性质，那将比单一应用某种技术能更好地节省内存空间，这些都是进一步研究的方向。

参考文献

[1] Edmund M C, Oran G, Doron A P. Model Checking [M]. Cambridge: MIT Press, 1999

[2] 林惠民，张文辉. 模型检测：理论、方法与应用[J]. 电子学报，2002，30(12A)：1907-1912
Lin Hui-min, Zhang Wen-hui. Model Checking: Theories, Techniques and Applications[J]. Acta Electronica Sinica, 2002, 30(12A): 1907-1912

[3] Edmund M C, Oran G, David E L. Model checking and abstraction [J]. ACM Transactions on Programming Languages and Systems, 1994, 16(5): 1512-1542

[4] 刘志锋，孙博，周从华. 概率实时时态认知逻辑模型检测中抽象技术的研究[J]. 电子学报，2013，41(7)：1343-1351
Liu Zhi-feng, Sun Bo, Zhou Cong-hua. Abstraction in Model Checking Probabilistic Real-Time Temporal Logic of Knowledge[J]. Acat Electronica Sinica, 2013, 41(7): 1343-1351

[5] Grumberg O, Vardi M Y, Sifakis J, et al. 2010 CAV award announcement[J]. Formal Methods in System Design, 2012, 40(2): 117-120

[6] Mendoza L E, Capel M I, Perez M A. Conceptual framework for business processes compositional verification [J]. Information and software technology, 2012, 54(2): 149-161

[7] 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
Zeng Hong-Wei, Miao Huai-Kou. Verification of Component Composition Based on Abstraction Refinement[J]. Journal of Software, 2008, 19(5): 1149-1159

[8] Carbone R. LTL model-checking for security protocols[J]. AI communications, 2011, 24(3): 385-396

[9] 高建华，蒋颖. 基于余归纳的最小Kripke结构的求解[J]. 软件学报，2014，25(1)：16-26
Gao Jian-hua, Jiang Ying. Coinduction-Based Solution for Minimization of Kripke Structures[J]. Journal of Software, 2014, 25(1): 16-26

[10] Gerard J H. The SPIN Model Checker: Primer and Reference Manual [M]. Boston: Addison-Wesley, 2004

[11] 吕威，黄志球，陈哲，等. ESpin：基于SPIN的Eclipse模型检测环境[J]. 计算机工程与应用，2013，49(7)：45-51
Lv Wei, Huang Zhi-qiu, Chen Zhe, et al. ESpin: SPIN-based Eclipse model checking environment[J]. Computer Engineering and Applications, 2013, 49(7): 45-51

[12] Patig S, Stolz M. A pattern-based approach for the verification of business process descriptions [J]. Information and Software Technology, 2013, 55(1): 58-87

[13] Ikeda, Satoshi, Jibiki, et al. Coverage Estimation in Model Checking with Bitstate Hashing[J]. IEEE Transactions on Software Engineering, 2013, 39(4): 477-486

(上接第98页)

[11] Zhang Lu, Tang Xue-yan. Client assignment for improving interactivity in distributed interactive applications[C]// Proceedings of IEEE international Conference on INFOCOM. 2001: 3227-3235

[12] Zhang Lu, Tang Xue-yan. The client assignment problem for continuous distributed interactive applications[C]// Proceedings of IEEE Conference on ICDCS. 2001: 203-214

[13] Zhen Han-ying, Tang Xue-yan. An Enhanced Genetic Algorithm for Server Placement in Distributed Interactive Applications[C]// Proceedings of IEEE Conference on Parallel and Distributed Systems. 2012: 596-603

[14] Gelfand S B, Mitter S K. Analysis of simulated anneling for optimization[J]. Decision and Control, 1985, 24(1)

[15] Rolland E, Schilling D A, Current J R. An efficient tabu search procedure for the p-Median problem[J]. European Journal of Operational Research, 1997, 96(2): 329-342