前向可修正属性算术验证的研究

吴海玲[1] 周从华[1] 鞠时光[1] 王 基[2]
(江苏大学计算机科学与通信工程学院 镇江 212013)[1] (镇江市地方税务局 镇江 212003)[2]

摘 要 目前验证前向可修正属性的"展开方法"是不完备的,即当"展开定理"的局部条件不满足时,不能判断出系统不满足前向可修正属性。为此,提出一种基于状态转换系统的前向可修正属性验证方法,该方法将前向可修正属性的验证归约为可达性问题,进而可借助可达性检测技术完成属性的验证。该方法是完备的,且当属性不成立时,可以给出使属性失效的反例,反例的给出对非法信息流的消除和控制具有直接的帮助。最后,通过磁臂隐通道的例子说明了方法的有效性和实用性。

关键词 前向可修正属性,信息流,无干扰,状态转换系统
中图法分类号 TP309 **文献标识码** A

Algorithmic Verification of Forward Correctability

WU Hai-ling[1] ZHOU Cong-hua[1] JU Shi-guang[1] WANG Ji[2]
(School of Computer Science and Telecommunication Engineering, Jiangsu University, Zhenjiang 212013, China)[1]
(Zhenjiang Municipal Local Taxation Bureau, Zhenjiang 212003, China)[2]

Abstract Due to the incompleteness of the "Unwinding Theorem", a system can't be judged to fail to satisfy the forward correctability, when some local conditions of "Unwinding Theorem" are not satisfied. This paper proposed an algorithmic verification technique to check the forward correctability based on the state transition system. The technique reduces forward correctability checking to the reachability problem and the reduction enables us to use the reachability checking technique to perform forward correctability checking. Our method is complete and it can give a counter-examples to control and eliminate the illegal information flow when a system fails to satisfy the forward correctability. Finally, Disk-arm Convert Channel illustrates the effectiveness and practicality.

Keywords Forward correctability, Information flow, Non-interference, Sate transition system

1 引言

目前,为保证计算机系统中的信息机密性,自主存取控制和强制存取控制策略在计算机多级安全系统中得到了广泛的应用。然而,实施了这两种策略的计算机系统仍存在安全隐患,如安全系统中的隐通道问题[1-3]。高安全级进程可以利用隐通道向低安全级进程泄漏信息,对系统安全构成巨大威胁。因此,对高安全等级的安全操作系统,各种安全标准都要求进行隐通道分析。自 D. E. Denning[4] 和 J. K. Millen[5] 提出采用信息流分析技术确定系统是否存在非法信息流后,信息流分析技术被用来寻找系统中潜在的各种信息泄漏问题,随后并成为隐通道分析的一个重要方法。

1982年 Goguen 和 Meseguer 首次引入了基于信息流分析的无干扰方法(noninterference)[6] 来控制确定型系统中信息的直接流动和间接流动。Goguen 提出的无干扰仅仅适用于确定型系统,随后研究人员在 Meseguer 和 Goguer 的工作基础上,相继提出适用不确定型系统的不可推断性(noninference)[7]、不可演绎属性(nondeducibility)[8,9]、广义无干扰属性(gerneralized noninterference)[10] 等。

J. K. Millen 在文献[11]中指出,一个有效的信息安全的定义至少比带输入保护的不可演绎属性强并且该属性是可复合的。但实际很多例子表明无干扰属性、不可演绎属性及广义无干扰属性都不是可复合的。为了解决上述问题,McCullough 提出"耦合安全性质"——限制属性(restrictiveness)[12,13]。它是第一个可复合的安全性质。此后不久,Jhnson, Thayer 于1988年引入一个与限制属性相似并且保持了限制属性基本要求的安全性质——前向可修正属性(forward correctability)[14]。前向可修正属性不仅比不可演绎属性强,而且是可复合的,此外它还稍弱于限制属性,比限制属性适用于更多的系统。

引入前向可修正属性的最终目的是确保多级安全系统中没有相应的非法信息流,因此如何验证系统满足前向可修正

到稿日期:2010-04-17 返修日期:2010-09-01 本文受国家自然科学基金(60773049),江苏大学高级人才科研启动基金(07JDG014),江苏省高校自然科学基金(08KJD520015),教育部博士点基金(20093227110005)资助。
吴海玲(1984—),女,硕士生,主要研究方向为信息安全,E-mail:whlbell@126.com;周从华(1978—),男,博士,副教授,硕士生导师,主要研究方向为信息流安全、模型检测和模态逻辑;鞠时光(1955—)男,教授,博士生导师,主要研究方向为隐通道、数据库管理系统;王 基(1982—),男,硕士生,主要研究方向为网络安全。

属性一直是一个重要的议题。在 Jhnson，Thayer 提出前向可修正属性后，国内外学者对前向可修正属性和它的验证方法进行了研究。Jonathan K. Millen[15] 基于事件模型研究前向可修正属性的相关性质并提出前向可修正属性的“展开方法”。周伟[16]等基于进程代数框架研究前向可修正属性，此外 Ron van[17]等在基于状态机的基础上进行研究。但后两者只是推导出前向可修正属性的一些相关性质，并未提出具体可行的验证方法。而前向可修正属性的“展开方法”首先是要构造一个“展开定理”，然后基于该定理将信息流安全属性这一全局约束归约到仅仅涉及单步状态转换的局部条件的验证。该技术建立了系统安全与单个状态转换命令之间的关系，因此称为“展开方法”。但是“展开方法”存在一个问题，它是可靠的，但并不是完备的，这里可靠性是指如果局部条件得到满足，则可推断出系统满足安全属性，不完备是指如果局部条件没有得到满足，则不能推断出系统不满足安全属性。

本文对基于状态转换系统的前向可修正属性进行分析，提出一种可靠且完备的前向可修正属性验证方法；第 2 节简要地介绍有限状态转换系统模型和前向可修正属性的定义，并分析得到前向可修正属性的一些性质。第 3 节提出前向可修正属性的验证方法；第 4 节给出实际的例子来验证算法的有效性；最后给出本文的总结和以后的工作。

2 前向可修正属性的描述

考察如图 1 所示的系统 M，它由两个子系统 M_1 和 M_2 组成。系统 M 要求，若子系统 M_1 和 M_2 是安全的，则 M 也应该是安全的。但不可演绎性、不可推断性和广义无干扰等属性不能描述图 1 所示的复合系统安全问题。

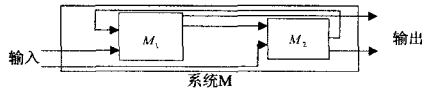

图 1 复合系统的安全问题

1988 年 Johnson. D. M. 和 F. J. Thayer 引入满足复合性的前向可修正属性，该文是在迹模型上对前向可修正属性进行定义分析。随后 Ron van 和 Chenyi Zhang 于 2006 年将前向可修正属的定义扩展到状态机模型上，本文的研究基础沿用状态机模型上的定义。

2.1 状态转换系统模型

文中我们利用有限状态转化系统来描述多级安全系统的动态行为，该安全系统称为非确定安全标记 Kripke 结构，简记为 NSLKS。与其它文献一样，在不失一般性的情况下，假设只有两个安全域，即 H 和 L。记 $Domain=\{H,L\}$。

定义 1 一个 NSLKS M 是一个六元组 $(S,s_0,next,obs,dom,A)$：

- S 是有限状态集；
- $s_0\in S$ 是初始状态；
- A 是有限的动作集；
- $next:S\times A\to P(S)\backslash\{\emptyset\}$ 是状态转换函数，$next(s,a)$ 表示在状态为 s 时执行动作 a 后所能够到达的状态集合，其中 $s\in S,a\in A$；
- $dom:A\to Domain$ 为每个动作指定安全域，其中 $Domain=\{H,L\}$；
- $obs:S\times Domain\to O$ 为观察函数，表示每个状态在具体的安全域上可观察到的值。为了描述简便，这里用 obs_u 表示当 $u\in D$ 时，$S\to O$。

系统 M 的一个运行序列用 $r=s_0a_1s_1a_2s_2\cdots a_ns_n\in S(AS)^*$ 表示，其中对于 $1\leqslant i\leqslant n$，有 $s_i\in next(s_{i-1},a_i)$。定义 $r(i)=s_i$ 表示运行序列 r 中第 i 个状态，$r^a(j)=a_j$ 表示 r 上的第 j 个动作。两个运行序列可以执行连接操作，如 $r=r_1\circ r_2$，则存在 $m,1\leqslant m\leqslant n$，使得 $r_1=s_0a_1s_1a_2s_2\cdots a_ms_m$ 且 $r_2=s_ma_{m+1}s_{m+1}a_{m+2}s_{m+2}\cdots a_ns_n$。

定义 2(定义函数) $Cond:X^*\to X^*$：对所有的 $a,b\in X$，$\alpha\in X^*$，有

$Cond(\ell)=\ell,Cond(a)=a,\ell$ 表示空序列；

$$Cond(\alpha\cdot a\cdot b)=\begin{cases}Cond(\alpha\cdot a)\cdot b, & \text{如果 } a\neq b\\ Cond(\alpha\cdot a), & \text{如果 } a=b\end{cases}$$

非形式化地讲，$Cond(\alpha\cdot a\cdot b)$ 是序列 $\alpha\cdot a\cdot b$ 通过压缩后得到的一个序列。$Cond$ 是压缩函数，它通过删除序列 $\alpha\cdot a\cdot b$ 中重复元素后得到一个尽可能短的序列。

定义 3 对于 $u\in Domain$，定义运行序列上的观察函数 $Obs_u:S(AS)^*\to O^+(AO^+)^*$：$Obs_u(s)=obs_u(s)$，

$$Obs_u(\delta\cdot a\cdot s)=\begin{cases}Obs_u(\delta)\cdot a\cdot obs_u(s), & \text{如果 } dom(a)=u\\ Obs_u(\delta)\cdot obs_u(s), & \text{如果 } dom(a)\neq u\end{cases}$$

定义 4 定义函数 $view_u:S(AS)^*\to O^+(AO^+)^*$：$view_u(r)=Cond(Obs_u(r))$

为了更好地理解定义 3 和定义 4，下面举一例子进行说明。假设 NSLKS 系统 M 有一运行序列 $r=s_0a_1s_1a_2s_2a_3s_3$，其中 $obs_L(s_0)=p_1,obs_L(s_1)=obs_L(s_2)=p_2,obs_L(s_3)=p_3,\{a_1,a_3\}\in L,a_2\in H$。根据定义 3 可得 $Obs_L(r)=p_1a_1p_2p_2a_3p_3$，再根据定义 4 得 $view_L(r)=Cond(Obs_L(r))=p_1a_1p_2a_3p_3$。

2.2 前向可修正属性

在给出具体前向可修正属性的定义前，先定义一些符号。定义运行序列上的函数 $Act:S(AS)^*\to A^*$，表示删除运行序列中的状态后得到的动作序列，如 $r=s_0a_1s_1a_2s_2a_3s_3$，则 $Act(r)=a_1a_2a_3$。定义运行序列间的关系 $\equiv$，$r_1\equiv r_2$ 当且仅当 r_1 和 r_2 有相同长度，$r_1^a(j)=r_2^a(j)$，且对于所有 $1\leqslant j\leqslant n$，有 $obs_u(r_1(i))=obs_u(r_2(i))$，其中 $0\leqslant i\leqslant n,u\in D$。

定义 5 一个 NSLKS M 满足前向可修正属性(简记 FC)当且仅当

(1) M 上任意运行序列 $r=r_1\circ r_2$，若 $Act(r_1)=\alpha,Act(r_2)=\alpha'$ 其中 $\alpha'\in A_L^*$，则对于任意高级别动作 $a\in A_H$，存在另一运行序列 $r'=r_1'\circ r_2'$ 使得 $r_1\equiv r_1',Act(r_2')=a\cdot\alpha'$ 且 $view_L(r)=view_L(r')$。

(2) M 上任意运行序列 $r=r_1\circ r_2$，若 $Act(r_1)=\alpha,Act(r_2)=a\cdot\alpha'$ 其中 $a\in A_H$ 和 $\alpha'\in A_L^*$，则存在另一运行序列 $r'=r_1'\circ r_2'$，使得 $r_1\equiv r_1',Act(r_2')=\alpha'$ 且 $view_L(r)=view_L(r')$。

本质上，由定义 5 可看出若 NSLKS M 满足 FC，即 M 上的任意运行序列 $r=s_0a_1s_1a_2s_2\cdots s_{i-1}a_is_ia_{i+1}s_{i+1}\cdots\cdots a_ns_n$(见图 2)，$r_1=s_0a_1s_1a_2s_2\cdots s_{i-1},r_2=s_{i-1}a_is_ia_{i+1}s_{i+1}\cdots a_ns_n$，子序列 r_2 的动作序列为低级别动作序列，此时在动作 a_i 前执行对运行序列 r 的扰乱，如插入一个高级别动作 H 后，存在另一运行序列 $r'=r_1'\circ r_2'$ 使得 $r_1\equiv r_1',Act(r_2')=a\cdot\alpha'$ 且 $view_L(r)=view_L(r')$。也就是说 r_1 和 r_1' 在高、低级别上的观察结果均一致，而 r_2 与 r_2' 只在低级别上的观察结果一致。即 NSLKS M 满足前向可修正属性，则对运行序列 r 的扰乱的修正是在扰乱之后的子运行序列上(即在 r 的子序列 r_2 上)。因此可

看出前向可修正属性比 McCllough 提出的广义无干扰属性强，因为广义无干扰属性要求对运行序列的扰乱的修正可以在扰乱前的子运行序列，也可以在扰乱后的子运行序列。为了更好理解这点，下面通过考察图 3 中的系统 M_1 来进行比较说明。

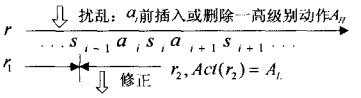

图 2　前向可修正属性

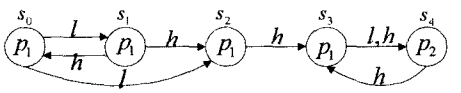

图 3　非确定安全标记 Kripke 结构 M_1

对于系统 M_1 中的运行序列 $r=s_0ls_1hs_2ls_2$，$Act(r)=lhl$，在该运行序列的高级别动作 h 之后最后一个低级别动作 l 之前插入一个高级别动作 h 执行一次扰乱，即 $Act(r')=lhhl$。而运行序列 $r'=s_0ls_1hs_0hs_0ls_1$ 是对 r 扰乱的一个修正（因为 $view_L(r)=view_L(r')$），且该修正是在扰乱之前以及之后进行，因此这种情况下，$r=s_0ls_1hs_2ls_2$ 满足广义无干扰属性但不满足前向可修正属性。

3　前向可修正属性的算术验证

根据定义 5 来直接判断一不确定型系统 M 是否满足前向可修正属性是相当困难的，因为 M 上的运行序列可以有无穷多个并且可以是任意长。因此，直接用定义 5 作为满足 FC 的判定准则行不通，下面我们提出一个与定义 5 等价的搜索方法来作为不确定型系统 M 满足前向可修正属性的判断方法。

引理 1　一个 NSLKS M，s_0 是 M 上的初始状态，s_i 是 s_0 经过任意动作序列后的可达状态，若 M 满足前向可修正属性当且仅当以 s_i 为出发状态，M 上的任一个运行序列 $r=s_ia_{i+1}s_{i+1}\cdots a_ns_n$，$Act(r)=A_L^*$，在 s_i 后插入任意一个高级别动作 b，存在另一运行序列 $r'=s_ibs_i'a_{i+1}s'_{i+1}\cdots a_ns_n'$，使得 $view_L(r)=view_L(r')$。

该引理的证明是简单的，由定义 5 可知，一个 NSLKS M 满足前向可修正属性，M 上任意运行序列 $r=r_1\circ r_2$，若 $Act(r_1)=\alpha$，$Act(r_2)=\alpha'$ 其中 $\alpha'\in A_L^*$，则对于任意高级别动作 $b\in A_H$，存在另一运行序列 $r'=r_1'\circ r_2'$ 使得 $r_1\equiv r_1'$，$Act(r_2')=b\cdot\alpha'$ 且 $view_L(r)=view_L(r')$。也就是说运行序列 r 和 r' 的前半段序列 r_1 和 r_1' 是相等的，而后半段序列 r_2 和 r_2' 在低级别观察上是一致的，即 $view_L(r_2)=view_L(r_2')$。可令 $r=s_0a_1s_1a_2s_2\cdots a_is_ia_{i+1}s_{i+1}\cdots a_ns_n$（其中 $r_1=s_0a_1s_1a_2s_2\cdots a_is_i$，$r_2=s_ia_{i+1}s_{i+1}\cdots a_ns_n$），$r'=s_0a_1s_1a_2s_2\cdots a_is_ibs_i'a_{i+1}s'_{i+1}\cdots a_ns_n'$（其中 $r_1'=s_0a_1s_1a_2s_2\cdots a_is_i$，$r_2'=s_ib\ s_i'a_{i+1}s'_{i+1}\cdots a_ns_n'$）。由于 r_1 和 r_1' 在 s_i 之前是相等的，而 s_i 是 M 上初始状态 s_0 的一个可达状态，因此 r 和 r' 可以直接在 s_0 的可达状态 s_i 之后进行比较判定，因此原命题可证。

3.1　不确定型系统的确定型构造

定义 6　令 $M=(S,s_0,next,obs,dom,A)$ 为一 NSLKS，定义确定型系统 $M^D=(S^D,s_0^D,A,next^D)$，这里

- $S^D=2^S$；
- $s_0^D=\{s_0\}$；
- $next^D:S^D\times A\to S^D$ 是状态转换函数，这里 $s_1^D=next^D(s^D,\alpha)$ 当且仅当 $s_1^D=\{s'\mid\exists s\in s^D(s'\in next(s,\alpha))\}$。

定义 6 说明如何将一个 NSLKS M 演绎为一个行为等价的确定型系统。这种演绎使我们可以将前向可修正属性的验证归约为确定型系统上相应性质的验证。下面以图 3 中的 M_1 为例来说明如何构造 M_1 上的确定型系统 M_1^D。由定义 6，M_1^D 中的初始状态 $s_0^D=\{s_0\}$。首先计算在初始状态下输入 l 触发的状态转换：s_0^D 中只包含 M_1 的初始状态 s_0，M_1 中在 s_0 下输入 l 会产生两个后继 s_1 和 s_2，因此在 S_0^D 下输入 l 系统状态变为 $s_1^D=\{s_1,s_2\}$。对于 s_1^D，因为在 s_1 下输入 h 会产生两个后继 s_0，s_2，在 s_2 下输入 h 会产生后继 s_3，所以在 h 的触发下系统状态变为 $s_2^D=\{s_0,s_2,s_3\}$。同理可得其它转换关系，最终得到的系统如图 4 所示。

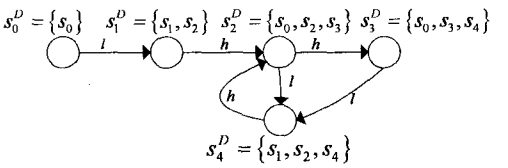

图 4　M_1 上的确定型构造 M_1^D

3.2　确定型系统的双构造

定义 7　令 $M^D=(S^D,s_0^D,A,next^D)$，定义 M^D 上的双构造系统 $M^{D^2}=(S^{D^2},s_0^{D^2},A,next^{D^2})$，这里

- $S^{D^2}=S^D\times S^D$；
- $s_0^{D^2}=(s_0^D,s_0^D)$；
- $next^{D^2}:S^{D^2}\times A\to S^{D^2}$ 是状态转换函数，这里对于域为 L 的动作 α，$(s_3^D,s_4^D)=next^{D^2}((s_1^D,s_2^D),\alpha)$ 当且仅当 $s_3^D=next^D(s_1^D,\alpha)$，$s_4^D=next^D(s_2^D,\alpha)$，对于域为 H 的动作 β，$(s_3^D,s_4^D)=next^{D^2}((s_1^D,s_2^D),\beta)$ 当且仅当 $s_3^D=(s_1^D,\beta)$，$s_4^D=next^D(s_2^D,\beta)\vee s_4^D=s_2^D$。

注意到在 M^{D^2} 中对于域为 H 的动作 β，我们增加了一个额外的局部转换关系，即状态自身到自身的转换。定义 M^{D^2} 的主要目的在于将 M^D 中的输入动作序列 r 和它的扰乱 r' 合并为一条路径，从而将前向可修正属性的验证归约为 M^{D^2} 上的状态之间的可达性问题。以图 4 中的 M_1^D 为例来说明双构造系统的计算，重点说明转换关系的计算。首先对于初始状态 $s_0^{D^2}=(s_0^D,s_0^D)$，当动作 l 发生时，由定义 7，$s_0^{D^2}$ 中的两个元素均要发生转化。$s_1^D=next^D(s_0^D,l)$，所以 $s_1^{D^2}=(s_1^D,s_1^D)$，$s_{11}^{D^2}=next^{D^2}(s_0^{D^2},l)$。对于状态 $s_1^{D^2}$，动作 h 发生时一定会引起 $s_1^{D^2}$ 中第一个元素 s_1^D 按照 M_1^D 中的转换关系发生变化，而对于第二个元素有两种可能：状态不变或者按照 M_1^D 中的转换关系发生变化。因此 $s_1^{D^2}$ 在输入 h 下有两个后继，即 $s_2^{D^2}=(s_2^D,s_2^D)$，$s_{11}^{D^2}=(s_2^D,s_1^D)$。最终系统 M^{D^2} 如图 5 所示。

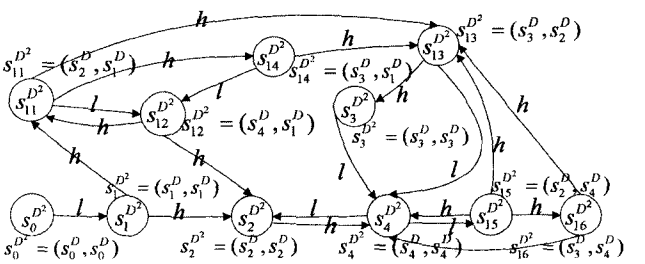

图 5　M_1^D 上的双构造

3.3　前向可修正属性的验证方法

定义 8（一步 H 可达）　令 M 为一 NSLKS，M^D 为 M 上

的确定型构造系统，M^{D^2} 为 M^D 上的双构造。在 M^{D^2} 称 (r^D, t^D) 是从 (s_i^D, s_i^D) 一步 H 可达的当且仅当存在一个域为 H 的动作 $b, b\in A_H$，使得 $r^D = s_i^D, t^D = next(s_i^D, b)$。

定义 9（L 可达） 令 M 为一 NSLKS，M^D 为 M 上的确定型构造系统，M^{D^2} 为 M^D 上的双构造。(u^D, v^D) 是从 (r^D, t^D) L 可达的当且仅当存在域为 L 的动作序列 σ，使得 $u^D = r^D \cdot \sigma$，$v^D = t^D \cdot \sigma$。

定义 10（双构造上的前驱状态与后继状态） 如在双构造 M^{D^2} 上有一状态转换关系 $(r^D, t^D) \xrightarrow{a} (u^D, v^D), a\in A$，则称 (u^D, v^D) 为 (r^D, t^D) 动作 a 的后继状态，(r^D, t^D) 为 (u^D, v^D) 动作 a 的前驱状态。

定义 11 定义状态转换低级别上的观察函数 O_L^z：

· 对于不确定型系统 M 上的状态转换，$O_L^z: S(AS) \to O(AO)$；

· 对于 M 的确定型构造系统 M^D 上的状态转换，$O_L^z: S^D(AS^D) \to \{O(AO)\}$；

如 M 上有状态转换 $s_1 \xrightarrow{a} s_4, s_1 \xrightarrow{a} s_5, s_2 \xrightarrow{a} s_3$，且 $obs_L(s_1) = obs_L(s_5) = p_1, obs_L(s_2) = obs_L(s_4) = p_2, obs_L(s_3) = p_3$，则 $O_L^z(s_1 \xrightarrow{a} s_4) = p_1 \xrightarrow{a} p_2, O_L^z(s_1 \xrightarrow{a} s_5) p_1 \xrightarrow{a} p_1, O_L^z(s_2 \xrightarrow{a} s_3) = p_2 \xrightarrow{a} p_3$。$M^D$ 为 M 上的确定型构造系统，$s^D = \{s_1, s_2\}, r^D = \{s_3, s_4, s_5\}$，则 $O_L^z(s^D \xrightarrow{a} r^D) = \{p_1 \xrightarrow{a} \{p_1, p_2\}, p_2 \xrightarrow{a} p_3\}$。$M^{D^2}$ 为 M^D 的双构造，$(r^D, t^D) \xrightarrow{a} (u^D, v^D)$ 为 M^{D^2} 上的一个状态转换，若 $O_L^z(r^D \xrightarrow{a} u^D) = O_L^z(t^D \xrightarrow{a} v^D)$，则称 $(r^D, t^D) \xrightarrow{a} (u^D, v^D)$ 在双构造上的状态转换低级别观察结果相等。

引理 2 令 M 为一 NSLKS，s_0 是 M 上的初始状态，S^R 为 s_0 经过任意动作序列后的可达状态集合，M^D 为 M 上的确定型构造系统，M^{D^2} 为 M^D 上的双构造。若 M 满足前向可修正属性，当且仅当对于所有的 $s_i, s_i \in S^R$，(r^D, t^D) 为 (s_i, s_i) 的一步 H 可达状态，(u^D, v^D) 为 (r^D, t^D) L 可达的，则所有的 (u^D, v^D) 与其前驱状态在双构造上的状态转换观察结果相等。

该引理的证明是平凡的，直接由 M^D、M^{D^2}、双构造上状态转换观察的定义以及引理 1 可得。在引理 2 的基础上我们提出前向可修正属性的判断算法，该算法的思想是将前向可修正属性的验证问题归约为状态可达性问题。遍历所求系统初始状态的可达状态集 S^R，求出 S^R 中每个元素的一步 H 可达状态并进一步求出该一步 H 可达状态的 L 可达状态集，判断所有的 L 可达状态是否与其前驱状态在双构造上状态转换观察结果相等。若是，则满足前向可修正属性，否则不满足。

算法 1 前向可修正属性的判断

输入：NSLKS M

输出：系统 M 是否满足前向可修正属性

步骤 1 构造 M 的初始状态 s_0 经过任意动作序列后的可达状态集合 $S^R = \{s \mid s_0 \xrightarrow{a} \cdots s_i \xrightarrow{b} s_j \cdots \xrightarrow{c} s, \{a, b, c\} \in A\}$。

步骤 2 while($S^R \neq \emptyset$)

{ 从 S^R 集合中选择一个 s_i；

构造以 s_i 为出发状态，系统 M 的确定型构造系统 M_i^D；

构造 M_i^D 上的双结构 $M_i^{D^2}$；

求出 $M_i^{D^2}$ 上 (s_i, s_i) 的一步 H 可达状态 $(s_k^D, s_m^D) = \{s_k^D = \{s_i\}, s_m^D = next(\{s_i\}, h)\}$；

构造 $(s_k^D, s_m^D) L$ 可达集合 $S^L = \{(u^D, v^D) \mid ((s_k^D, s_m^D) \xrightarrow{a} \cdots \xrightarrow{b} (u^D, v^D)) \vee (s_k^D, s_m^D), \{a, b\} \in A_L\}$；

while($S^L \neq \emptyset$)

{ 从 S^L 中取出一个 (u^D, v^D)；

求出 (u^D, v^D) 在 S^L 中的前项可达状态集 $S^Q = \{(r^D, t^D) \mid (r^D, t^D) \xrightarrow{l} (u^D, v^D), l \in A_L\}$；

while($S^Q \neq \emptyset$)

{ 从 S^Q 中取出一个 (r^D, t^D)；

判断 $(r^D, t^D) \xrightarrow{l} (u^D, v^D)$ 在双结构上的状态转换低级别观察结果是否一致；（即判断 $O_L^z(r^D \xrightarrow{l} u^D) = O_L^z(t^D \xrightarrow{l} v^D)$）；

若判断结果为真，则继续往下运行；

若结果为假，则输出系统 M 不满足前向可修正属性并且跳出循环；

}

}

}

步骤 3 输出系统满足前向可修正属性。

该算法不仅可以判断一不确定型系统是否满足前向可修正属性，还可以通过增加相关操作来得到当系统不满足前向可修正属性时的一个反例。在步骤 1，求得每一个可达状态，同时记录下每个可达状态 s 从 s_0 出发第一次到达的路径。通过增加标记位来标识状态转换之间的前驱与后继关系，在步骤 2 中如果判断出所求系统不满足前向可修正属性，可一步步地找出当前状态的前驱状态，一直到当前的 S^R 中的可达状态。0 结合步骤 1 记录的可达状态的路径，则可返回一个系统不满足前向可修正的反例。

下面用上述验证方法来判断图 3 中的系统 M_1 是否满足前向可修正属性。

步骤 1

求出 M_1 的初始状态 s_0 的可达状态 $S^R = \{s_0, s_1, s_2, s_3, s_4\}$；并记录各可达状态的一条路径 $\{s_0 \xrightarrow{l} s_1, s_0 \xrightarrow{l} s_2, s_0 \xrightarrow{l} s_1 \xrightarrow{h} s_2 \xrightarrow{h} s_3, s_0 \xrightarrow{l} s_1 \xrightarrow{h} s_2 \xrightarrow{h} s_3 \xrightarrow{l} s_4\}$；

步骤 2

(1)在 S^R 中取出一个元素 s_0；

构造以 s_0 为出发状态时系统 M_1 的确定型构造系统 M^D（见图 4）；

构造 M^D 上的双结构 M^{D^2}（见图 5）；

求出 (s_0, s_0) 的一步 H 可达状态为 (s_0^D, s_0^D)；

求出 $(s_0^D, s_0^D) L$ 可达状态集为 (s_1^D, s_1^D)；

因为 $O_L^z(s_0^D \xrightarrow{l} s_1^D) = O_L^z(s_0^D \xrightarrow{l} s_1^D)$，所以继续往下运行；

(2)在 S^R 中取出一个元素 s_1；

构造以 s_1 为出发状态时系统 M_1 的确定型构造系统 M_1^D（这里图不具体画出）；

构造 M_1^D 上的双构造 $M_1^{D^2}$（这里图不具体画出）；

求出 (s_1, s_1) 的一步 H 可达状态为 $(\{s_1\}, \{s_0, s_2\})$；

求出 $(\{s_1\}, \{s_0, s_2\}) L$ 可达状态集为 $(\{s_1\}, \{s_1, s_2\})$；

因为 $O_L^z(s_1 \xrightarrow{l} s_1) = O_L^z(\{s_0, s_2\} \xrightarrow{l} \{s_1, s_2\})$，所以继续往下运行；

(3)在 S^R 中取出一个元素 s_2；

构造以 s_2 为出发状态时系统 M_1 的确定型构造系统 M_2^D（这里图不具体画出）；

构造 M_2^D 上的双构造 $M_2^{D^2}$（这里图不具体画出）；求出($\{s_2\},\{s_2\}$)的一步 H 可达状态为($\{s_2\},\{s_3\}$)；

求出($\{s_2\},\{s_3\}$)L 可达状态集为($\{s_2\},\{s_4\}$)；

由于 $O_L^x(s_2 \xrightarrow{l} s_2) \neq O_L^x(s_3 \xrightarrow{l} s_4)$，因此输出系统 M_1 不满足前向可修正属，并根据步骤 1 记录的路径可返回不满足前向可修正属性的反例 $s_0 \xrightarrow{l} s_2 \xrightarrow{l} s_2$（在序列中第二个状态 s_2 后插入一个高级别动作 h，在系统 M_1 中找不到序列 $s_0 \xrightarrow{l} s_2 \xrightarrow{h} s'_2 \xrightarrow{l} s''_2$，使得 $view_L(s_0 \xrightarrow{l} s_2 \xrightarrow{h} s'_2 \xrightarrow{l} s''_2) = view_L(s_0 \xrightarrow{l} s_2 \xrightarrow{l} s_2)$。

步骤 3　跳出循环，算法结束。

根据得到的反例，对图 3 的系统 M_1 稍作修改，如图 6 所示的系统 M_2，增加两个状态 s_5，s_6 后该系统满足前向可修正属性。

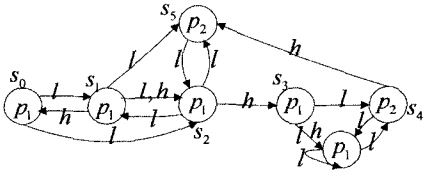

图 6　非确定安全标记 Kripke 结构 M_2

4　实例分析

磁盘在读写的过程中由于磁臂运动会产生隐通道，这种类型的隐通道称为磁臂隐通道[18]。下面给出一个磁臂隐通道的例子，将提出的前向可修正属性的验证算法应用在磁臂隐通道问题上。*High* 表示高级别进程，*Low* 表示低级别进程，假定 *Low* 发出一个异步的磁盘读请求 l_1，l_1 读柱面 1。*Low* 立即放弃 CPU，并等待请求得到响应。接着进程 *High* 占有 CPU，它定位于柱面 2 的请求 h_2（信息 0）或柱面 3 的请求 h_3（信息 1），在此之后立即放弃 CPU，并等待请求得到响应。当 *Low* 读柱面 1 的请求得到响应后，*Low* 立即同时发出两个异步请求，定位于柱面 2 的请求 l_2 和柱面 3 的请求 l_3。按照“电梯算法”，若进程 *High* 请求定位的是柱面 2（信息 0），即 h_2，则 *Low* 异步请求的响应次序为 $l_2 \rightarrow l_3$，l_2 比 l_3 先得到响应。反之，若进程 *High* 请求定位柱面 3（信息 1），即 h_3。则 *Low* 异步请求的响应次序为 $l_3 \rightarrow l_2$，l_3 比 l_2 先得到响应。根据请求 l_2 和 l_3 得到响应的次序，低级别进程 *Low* 得知高进程 *High* 发出的定位请求的柱面号，即泄漏了一个比特的信息。

我们用状态机模型来描述磁盘读写过程，用系统 N 来表示（见图 7）。该系统有两个进程 *High* 和 *Low*，$Domain=\{H,L\}$。$A=\{h_1,h_2,h_3,l_{12},l_{13},l_{23}\}$，其中 $A_H=\{h_1,h_2,h_3\}$，分别表示高进程请求定位到柱面 1、柱面 2 和柱面 3。$A_L=\{l_{12},l_{13},l_{23}\}$，分别表示低级别进程同时发出两个异步请求定位到柱面 1 和 2，柱面 1 和 3，柱面 2 和 3。状态集为 $S=\{s_0,s_1,s_2,s_3,s_4,s_5,s_6,s_7,s_8,s_9,s_{10},s_{11},s_{12},s_{13},s_{14},s_{15},s_{16},s_{17}\}$，其中 s_0 表示当前磁臂定位在柱面 1，磁臂方向朝左。s_1 表示当前磁臂定位在柱面 1，磁臂方向朝右。s_2 表示当前磁臂定位在柱面 2，磁臂方向朝左。以此类推，同样定义状态 s_3，s_4，s_5。而 s_6，s_7，s_8，s_9，s_{10}，s_{11}，s_{12}，s_{13}，s_{14}，s_{15}，s_{16}，s_{17} 则表示在 6 个状态 s_0，s_1，s_2，s_3，s_4，s_5 时，高级别进程执行操作后到达的状态。对于 s_0，高进程执行 h_2 后到达 s_6，即 $s_0 \xrightarrow{h_2} s_6$。由于这个状态转换过程，低级别进程是无法观察到的，即低级别进程不知道高进程执行了哪种动作且执行后的结果，因此对于状态 s_6，高进程观察到的是磁臂在柱面 2，方向朝右，而低进程观察到的仍是磁臂在柱面 1，方向朝左。以此类推，有 $s_0 \xrightarrow{h_2} s_7$，$s_1 \xrightarrow{h_2} s_8$，$s_1 \xrightarrow{h_3} s_9$，$s_2 \xrightarrow{h_1} s_{10}$，$s_2 \xrightarrow{h_3} s_{11}$，$s_3 \xrightarrow{h_1} s_{12}$，$s_3 \xrightarrow{h_3} s_{13}$，$s_4 \xrightarrow{h_1} s_{14}$，$s_4 \xrightarrow{h_2} s_{15}$，$s_5 \xrightarrow{h_1} s_{16}$，$s_5 \xrightarrow{h_2} s_{17}$。这里状态观察结果包括当前磁臂的位置以及磁臂的方向，因此 $O_L(s_6)=O_L(s_7)=O_L(s_0)=${在柱面 1，朝左}，$O_L(s_8)=O_L(s_9)=O_L(s_1)=${在柱面 1，朝右}，$O_H(s_6)=O_H(s_8)=O_H(s_3)=${在柱面 2，朝右}，$O_H(s_7)=O_H(s_9)=O_H(s_5)=${在柱面 3，朝右}，$O_L(s_{10})=O_L(s_{11})=O_L(s_2)=${在柱面 2，朝左}，$O_L(s_{12})=O_L(s_{13})=O_L(s_3)=${在柱面 2，朝右}，$O_H(s_{10})=O_H(s_{12})=O_H(s_0)=${在柱面 1，朝左}，$O_H(s_{11})=O_H(s_{13})=O_H(s_5)=${在柱面 3，朝右}，$O_L(s_{14})=O_L(s_{15})=O_L(s_4)=${在柱面 3，朝左}，$O_L(s_{16})=O_L(s_{17})=O_L(s_5)=${在柱面 3，朝右}，$O_H(s_{15})=O_H(s_{17})=O_H(s_2)=${在柱面 2，朝左}，$O_H(s_{14})=O_H(s_{16})=O_H(s_0)=${在柱面 1，朝左}。各状态间转换过程如图 7 所示。

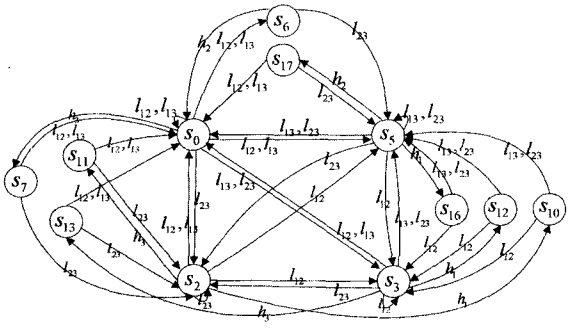

图 7　非确定安全标记 Kripke 结构 N

下面用算法 1 来分析系统 N。

步骤 1　初始状态 s_0 的可达状态集为 $S^R=\{s_0,s_2,s_3,s_5,s_6,s_7,s_{10},s_{11},s_{12},s_{13},s_{16},s_{17}\}$；

步骤 2　在 S^R 中取出一个元素 s_0；

构造以 s_0 为出发状态时系统 N 的确定型构造系统 N^D（这里不具体画出）；

构造 N^D 上的双结构 N^{D^2}（这里不具体画出）；

求出(s_0,s_0)的一步 h_2 可达状态为($\{s_0\},\{s_6\}$)，一步 h_3 可达状态为($\{s_0\},\{s_7\}$)；

求出($\{s_0\},\{s_6\}$)L 可达状态集 S^L 包含($\{s_0,s_3\},\{s_0\}$)，($\{s_0,s_5\},\{s_0\}$)，($\{s_2,s_5\},\{s_5\}$)，($\{s_2\},\{s_2\}$)，($\{s_2\},\{s_2\}$)，($\{s_0\},\{s_0\}$)，($\{s_2,s_5\},\{s_2\}$)；

在 S^L 中取出($\{s_0,s_3\},\{s_0\}$)，($\{s_0,s_3\},\{s_0\}$)的前驱可达状态集 S^Q 包含($\{s_0\},\{s_6\}$)，因为($\{s_0\},\{s_6\}$)$\xrightarrow{l_{12}}$($\{s_0,s_3\},\{s_0\}$)，并且 $O_L^x(\{s_0\} \xrightarrow{l_{12}} \{s_0,s_3\}) \neq O_L^x(\{s_6\} \xrightarrow{l_{12}} \{s_0\})$，因此输出系统 N 不满足前向可修正属性；

步骤 3　跳出循环，算法结束。

从上面的分析结果可得出系统 N 不满足前向可修正属

性，该系统存在非法信息流。因为低级别进程 Low 在状态 s_0 同时发出的两个异步请求 l_{12} 响应的顺序有两种情况(情况一先位置 1 再到位置 2，情况二先位置 2 再到位置 1)，但高级别进程 $High$ 先对状态 s_0 进行高级别操作 h_1 后，会使得低级别进程 Low 同时发出的两个异步请求 l_{12} 响应的顺序确定下来(先位置 1 再到位置 2)，也就是说低级别进程可以根据请求响应的顺序来推断出高级别进程所进行的操作，即高级别进程向低级别进程泄漏信息。

结束语 本文首先回顾了已有前向可修正属性的研究成果，然后在基于状态转换系统的前向可修正属性定义的基础上，分析前向可修正属性的具体性质，提出了一种可靠且完备的前向可修正属性验证方法。该方法将不确定型系统前向可修正属性的验证问题归约为确定型系统上相应性质的验证，然后进一步归约为确定型系统的双构造上的可达性问题，进而借助于可达性检测技术完成验证。和已有的前向可修正属性“展开定理”相比，我们的方法有两方面的优点：(1)避免了“展开方法”中局部条件的构造和验证，即“展开定理”的构造，是一种完备的方法。(2)系统在不满足前向可修正属性时，能够给出属性失效的反例，反例的给出对非法信息流的消除和控制具有直接的帮助。最后，本文通过一个磁臂隐通道的例子说明了如何将该方法应用在实际的隐通道分析中。

将来的工作主要包括：(1)将本文提出的算术验证方法拓展到其他信息流安全属性(如不可演绎属性、限制属性、隔离属性等)，提出各属性具体验证算法。(2)研究模型检测中的谓词抽象技术，并利用其提炼出系统的有限状态机模型。

参 考 文 献

[1] 刘文清，韩乃平，陶喆. 一个安全操作系统 Slinux 隐蔽通道标识与处理[J]. 电子学报，2007，35(1)：153-156

[2] 王昌达，鞠时光，周从华，等. 一种隐通道威胁审计的度量方法[J]. 计算机学报，2009，32(4)：751-762

[3] 卿斯汉，沈昌祥. 高等级安全操作系统的设计[J]. 中国科学 E，2007，37(2)：238-253

[4] Denning D E. A Lattice Model of Secure Information Flow[J]. Communication of the ACM，1976，19(5)：236-242

[5] Millen J K. Security Kernel Validation in Practice[J]. Communication of the ACM，1976，19(5)：243-250

[6] Goguen J A，Meseguer J. Security policies and security models [C]// Proceedings of the 1982 IEEE Computer Society Symposium on Research in Security and Privacy. Los Alamitos：IEEE Computer Society Press，1982：11-20

[7] D'Souza D，Raveendra H，Janardhan K. On the decidability of modelchecking information flow properties[C]// 4th International Conference on Information Systems Security. ICISS 2008，December 2008：26-40

[8] 张原，史浩山. 信息安全模型研究[J]. 小型微型计算机系统，2003，24(10)：1878-1881

[9] 赵保华，陈波，陆超. 概率信息流安全属性分析[J]. 计算机学报，2006，129(8)：1447-1452

[10] Zhou Cong-hua，Chen li，Ju Shi-guang. Petri nets based noninterference analysis[J]. Journal of Computational Information Systems，2009，15(4)：1231-1239

[11] Millen J K. Hookup Security for Synchronous Machines[C]// Proceedings of The Computer Security Foundations Workshop III. IEEE Computer Society，1990：84-90

[12] McCoullough D. Specifications for multi-level security and a hookup property[C]// Proceedings of the Symposium on Research in Security and Privacy. New York：IEEE Computer Society，1987：161-166

[13] McCoullough D. Noniterference and the composability of security properties[C]// Proceedings of the IEEE Symposium on Research in Security and Privacy. New York：IEEE&Technical Committee on Security and Privacy，1988：177-186

[14] Johnson D，Thayer F. Security and the compositon of machines [C]// Proceedings of the Computer Security Foundations Workshop. IEEE Press，1988：14-23

[15] Jonathan K M. Unwinding forward correctability [C]// Proceedings of Computer Security Foundations Workshop VII. Franconia，1994：2-10

[16] 周伟，尹青，郭金庚. 计算机安全中的无干扰模型[J]. 计算机科学，2005，32(2)：159-165

[17] van der Heydena R，Zhanga C. Algorithmic Verification of Noninterference Properties[J]. Electronic notes in Theoretical Computer Science，2007，168(SPEC)：61-75

[18] 刘志峰，鞠时光，李沛. 基于操作语义的磁臂隐通道分析[J]. 计算机应用研究，2007，24(11)：157-160

(上接第 92 页)

[12] Yin Jin-yong，Guo Guo-chang，Wu Yan-xia. A Hybrid Fault-tolerant Scheduling Algorithm of Periodic and Aperiodic Real-time Tasks to Partially Reconfigurable FPGAs[J]. Intelligent Systems and Applications. ISA 2009：International Workshops，2009，6：1-5

[13] Qin Xiao，Han Zong-feng，Pang Li-ping，et al. Design and Performance Analysis of a Hybrid Real-Time Scheduling Algorithm with Fault-Tolerance[J]. Journal of Software，2000，11(5)：686-693

[14] Liu Huai，Lin Qiu-shi，Huang Jian-xin，et al. A Novel Fault-Tolerant Scheduling Algorithm for Periodic Tasks of Distributed Control Systems[J]. 2009 Chinese Controland Decision Conference，2009，8：1584-1588

[15] Qin Xiao，Han Zong-feng，Pang Li-ping. Real-Time Scheduling with Fault-Tolerance in Heterogeneous Distributed Systems [J]. Chinese Journal of Computers，2002，25(1)：49-56

[16] Wei Luo，Xiao Qin，Bellam K. Reliability-Driven Scheduling of Periodic Tasks in Heterogeneous Real-Time Systems[C]// the 7th ACIS International Conference on Software Engineering，Artificial Intelligence，Networking and Parallel/Distributed Computing(SNPD '07). China，2007：640-645

[17] Guo Hui，Wang Zhi-guang，Zhou Jian-li. Load Balancing Based Process Scheduling with Fault-Tolerance in Heterogeneous Distributed Systems[J]. Chinese Journal of Computers，2005，28(11)：1807-1816

[18] Luo Wei，Yang Fu-min，Pang Li-ping，et al. A Real-Time Fault-Tolerant Scheduling Algorithm of Periodic Tasks in Heterogeneous Distributed Systems[J]. Chinese Journal of Computers，2007，30(10)：1740-1749

[19] Garey R，Johnson D S. Computers and Intractability：a Guide to the Theory of NP-Completeness[M]. New York：W. H. Freeman Company，1979

[20] Zhang Kunlong，Qin Xiao，Han Zong-fen，et al. Study of the Model for Fault-Tolerant Scheduling in Heterongeneous Distributed Real-Time Systems [J]. J. Huazhong Univ. of Sci. & Tech.，2000，28(8)：17-19