ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.20, No.8, August 2009, pp.2015−2025
doi: 10.3724/SP.J.1001.2009.03449

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

ETL的符号化模型检验*

刘万伟+, 王 戟, 王昭飞

(国防科学技术大学 计算机学院 并行与分布处理国家重点实验室,湖南 长沙 410073)

Symbolic Model Checking of ETL

LIU Wan-Wei+, WANG Ji, WANG Zhao-Fei

(National Laboratory for Parallel and Distributed Processing, School of Computer, National University of Defense Technology, Changsha 410073, China)

+ Corresponding author: E-mail: wwliu@nudt.edu.cn

Liu WW, Wang J, Wang ZF. Symbolic model checking of ETL. *Journal of Software*, 2009,20(8):2015−2025.
http://www.jos.org.cn/1000-9825/3449.htm

Abstract: To make symbolic model checking approach applicable to all ω-regular properties, this paper studies symbolic model checking for ETL (extended temporal logic). First, the Tableau approach for LTL (linear temporal logic) is extended to that of ETL, and then the BDD (binary decision diagram)-based encodings of this technique is given. Moreover, the model checking tool ENuSMV is implemented based on NuSMV, which allows users to customize temporal connectives, and hence all ω-regular properties can be verified. Experimental results show that ETL can also be efficiently verified by the symbolic approach.

Key words: symbolic model checking; ETL (extended temporal logic); tableau approach; verification tool; ENuSMV

摘 要: 为使符号化模型检验技术适用于全部ω-正规性质,研究了 ETL(extended temporal logic)的符号化模型检验方法.首先,扩展了 LTL(linear temporal logic)的 Tableau 方法,给出了 ETL 的 Tableau 构造方法,进而给出了该方法基于 BDD(binary decision diagram)的符号化实现.同时,在 NuSMV 的基础上实现了支持 ETL 符号化验证的模型检验工具 ENuSMV.该工具允许用户自定义时序连接子,从而可以检验全部ω-正规性质.实验结果表明,ETL 性质能够被高效地采用符号化技术加以检验.

关键词: 符号化模型检验;扩展时序逻辑;Tableau 方法;验证工具;ENuSMV
中图法分类号: TP301 **文献标识码**: A

随着计算机软、硬件规模的不断提高,其设计、实现的复杂性日益增加.软、硬件行为的正确性越来越难以得到保证.模型检验技术是在上个世纪 80 年代提出来的,其目标为检验某系统的设计或实现是否满足既定的规约性质.系统的设计或实现往往被抽象为一个迁移系统(Kripke 结构);规约性质往往使用某个时序逻辑公式进行描述.从广义上讲,时序逻辑可以分为线性时间的时序逻辑(如 LTL(linear temporal logic)[1]等)和分支时间的

* Supported by the National Natural Science Foundation of China under Grant Nos.60725206, 60621003 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant No.2006AA01Z429 (国家高技术研究发展计划(863))
Received 2008-02-15; Accepted 2008-08-11

时序逻辑(如计算树逻辑 CTL(computation tree logic)等[2]).相对而言,LTL 易于表达人们感兴趣的时序性质;而 CTL 中由于具有路径量词,易于采用高效的符号化检验算法.Vardi 进一步总结了这两类时序逻辑的特点(参见文献[3–5]),指出:许多无法由 CTL 表达的重要时序性质[3]能够由 LTL 表达;LTL 具有更好的可组合性(compositionality)、统一性(uniformity);另外,LTL 非常适合于针对开放系统的模型检验.

早期的 LTL 模型检验算法将该问题归结为一个 Büchi 自动机的判空问题:要验证状态数为 m 的迁移系统是否满足长度为 l 的 LTL 规约性质,所需的状态复杂度为 $O(m2^l)$.当待检验的模型规模较大,或者性质较为复杂时,算法的代价就变得十分高昂.

20 世纪 90 年代,McMillian 等人提出的基于 BDD(binary decision diagram)的符号化模型检验技术[6]极大地促进了 CTL 的模型检验技术.基于 BDD 的高效 CTL 的模型检验器(如 CMU(Carnegie Mellon University)的 SMV(symbolic model verifier))被开发出来,并得到了广泛的应用.

为了将符号化模型检验应用到 LTL 的模型检验中,Clarke 等人提出了新的 LTL 检验技术[7].该技术的核心步骤是:首先为待检验的 LTL 公式构建一个 Tableau(可以看作一个特殊的 Kripke 结构),然后检验该 Tableau 与原模型的乘积在特定公平性约束下是否满足 CTL 公式 EGTrue.这样,就将 LTL 的模型检验问题转化成为 CTL 的验证问题,从而可以使用高效的符号化模型检验技术.基于该思想,构建了新的模型检验工具 NuSMV[8],它同时支持 LTL 和 CTL 的模型检验.

LTL 中包含 next 和 until 两个时序算子,其表达能力严格弱于ω-正规语言.诸如周期采样性(例如,“p 在偶数时刻均成立”[9])等重要性质,无法由 LTL 公式表达.表达能力上的缺陷直接导致 LTL 无法采用组合检验过程(modular model checking[10]).为了增强表达能力,后来提出了许多线性框架下的时序逻辑,如线性μ-演算、PSL(property specification language)[11]、ForSpec[12]、ETL(extended temporal logic)[9,13,14]等.

ETL 是在命题逻辑基础上,通过添加各种ω-自动机作为连接子而获得的时序逻辑.Vardi 和 Wolper 研究了 3 类扩展时序逻辑[13]:ETL_l,ETL_f 和 ETL_r,它们分别采用 looping,finite 和 repeating 接收条件的自动机作为连接子.这 3 种 ETL 的表达能力均等价于ω-正规语言.随后,Kupferman 等人又给出了若干类更加复杂的 ETL 变种[15].本文采用的 ETL 是 ETL_f.之所以选用 ETL_f,是因为目前工业界广泛使用的规约语言(如 ForSpec,PSL)的核心操作算子为有穷字上的正规表达式,它与 finite 接收条件自动机(即 ETL_f 中的时序连接子)存在自然的对应关系.另外,当 finite 接收条件自动机作为连接子时,描述时序性质最为直观.

本文给出了基于 Tableau 的 ETL 公式模型检验算法以及基于 BDD 的符号化实现.进一步地,实现了对 NuSMV 的扩展(称为 ENuSMV),使其支持 ETL 符号化模型检验.该扩展具有下列意义:

1. 从理论角度而言,一种时序逻辑能否使用符号化的方法进行模型检验是人们比较关心的理论问题.最早能够使用符号化技术的规约语言局限在分支时间框架内(如 CTL、模态μ-演算等);1994 年,Clarke 等人给出了基于 BDD 的 LTL 符号化模型检验技术[7];2005 年,苏开乐等人给出了 CTL^*(是 CTL 和 LTL 的公共超集)的符号化模型检验方法[16].
2. 从实际应用的角度而言,SMV/NuSMV 是目前应用较为广泛的模型检验工具.ENuSMV 进一步在 NuSMV 的基础上进行了语法扩充——它向下兼容 NuSMV,同时允许用户自定义时序连接子,从而可以检验许多不能用 LTL 定义的重要时序性质.
3. 目前,包含正规表达式或者自动机连接子的时序逻辑受到工业界的广泛关注,而 ETL 是这些时序逻辑的基础.Intel 采用的 FTL(ForSpec)[12]规约语言,实际上是 ETL_{2a}[15]的一个逻辑片断.FTL 采用有穷正规表达式作为时序连接子,与 ETL 相比,FTL 不允许时序连接子的嵌套.另外,IBM®提出的 PSL 规约语言(原 Sugar 项目)已经成为硬件规约语言的工业标准(1850~2005).值得一提的是,NuSMV 也将准备支持 PSL——目前的 NuSMV 允许用户用 PSL 写规约,但在验证时只支持那些能够转化为 LTL 或 CTL 的性质.因此,研究 ETL 的符号化检验技术对研究 PSL 符号化检验技术具有指导意义(有关显式 PSL 模型检验技术的研究可参见文献[17–19]).

本文第 1 节首先回顾 ETL 的语法、语义及其模型检验问题的定义.第 2 节给出基于 Tableau 的 ETL 模型

检验算法,进而给出该算法基于 BDD 的符号化实现.第 3 节主要介绍 ENuSMV 的语法扩展(相对于 NuSMV),以及使用该工具进行模型检验的若干实验结果.

1 扩展时序逻辑ETL

有穷自动机 A 是一个五元组$\langle \Sigma,Q,\delta,q,F\rangle$.其中:有穷集$\Sigma$为该自动机的字母表;有穷集 Q 为该自动机的状态集合;δ:$Q\times\Sigma\to 2^Q$,为该自动机的迁移函数;$q\in Q$,为该自动机的初始状态;$F\subseteq Q$,为该自动机的终止状态集合.

对于任意的字 w,用 $w(i)$表示 w 中的第 i 个字母;用 w^i 表示 w 以第 i 个字母为起始的后缀(注意,i 从 0 开始).显然,$w^0=w$.若 w 为有穷字,用$|w|$表示 w 的长度.字母表Σ上的有穷字 w 关于自动机 $A=\langle \Sigma,Q,\delta,q,F\rangle$上的一个可接收运行是一个状态序列 $q_0q_1\ldots q_{|w|}$满足:$q_0=q,q_{|w|}\in F$,并且对每个 $0\le i<|w|$,有 $q_{i+1}\in\delta(q_i,w(i))$.$w$ 能够被 A 所接收,当且仅当存在 w 关于 A 的可接收运行.用 $L(A)$表示自动机 A 所接收字的集合.

在本文中,对自动机的语法成分作如下约定:

- 首先,字母表Σ应该看作是一个向量,而不再是一个无序的集合.之所以要强调字母之间的顺序,是因为当使用自动机作为 ETL 连接子时,作为操作子的公式与该自动机的字母之间存在着对应关系.
- 其次,本文中使用的自动机都具有唯一的初始状态.为表述方便,特引入如下规定:设 $A=\langle \Sigma,Q,\delta,q,F\rangle$,则对于任意的 $r\in Q$,记 $A^r=\langle \Sigma,Q,\delta,r,F\rangle$.显然,若 q 为 A 的初始状态,则 A 与 A^q 所指相同.

给定一组原子命题集合 AP,ETL 的合式公式归纳定义如下:

- 对于任意的 $p\in$AP,p 为 ETL 公式.
- 若 f 为 ETL 公式,则$\neg f$ 和$\bigcirc f$ 均为 ETL 公式.
- 若 f_1,f_2 为 ETL 公式,则 $f_1\wedge f_2,f_1\vee f_2$ 都是 ETL 公式.
- 若 A 为以$\{a_1,\ldots,a_n\}$为字母表的自动机,$f_1,\ldots,f_n$ 都是 ETL 公式,则 $A(f_1,\ldots,f_n)$为 ETL 公式.

反复利用关系式$\neg(f\vee g)=\neg f\wedge\neg g,\neg(f\wedge g)=\neg f\vee\neg g,\neg\bigcirc f=\bigcirc\neg f$,可以使得$\neg$操作符只出现在原子命题及自动机连接子之前.这样得到的公式称为原公式的否定范式.

下面归纳定义公式“局部正/负出现”的概念:

- f 在 f 中的(唯一)出现为局部正出现.
- 若$\neg g$ 在 f 中的某次出现为局部正(负)出现,则在该次$\neg g$ 的出现中 g 的出现为局部负(正)出现.
- 若$\bigcirc g$ 在 f 中的某次出现为局部正(负)出现,则在该次$\bigcirc g$ 的出现中 g 的出现为局部正(负)出现.
- 若 $h\wedge g$(或 $g\wedge h,h\vee g,g\vee h$)在 f 中的某次出现为局部正(负)出现,则在该次 $h\wedge g$(或 $g\wedge h,h\vee g,g\vee h$)出现中 g 的出现为局部负(正)出现.
- 无论 $A(f_1,\ldots,g,\ldots,f_n)$在 f 中的某次出现为何种局部出现,在该次 $A(f_1,\ldots,g,\ldots,f_n)$出现中 g 的出现必为局部正出现.

线性结构π是一个以 2^{AP} 为字母表的无穷字.给定线性结构π 以及 $i\in\mathbf{N}$,ETL 公式的语义可归纳定义如下:

- 若 $p\in$AP,则$\pi^i\models p$ 当且仅当 $p\in\pi(i)$.
- $\pi^i\models\neg f$ 当且仅当$\pi^i\models f$ 不成立.
- $\pi^i\models\bigcirc f$ 当且仅当$\pi^{i+1}\models f$.
- $\pi^i\models f_1\wedge f_2$ 当且仅当$\pi^i\models f_1$ 以及$\pi^i\models f_2$ 同时成立;$\pi^i\models f_1\vee f_2$ 当且仅当$\pi^i\models f_1$ 或$\pi^i\models f_2$ 成立.
- $\pi^i\models A(f_1,\ldots,f_n)$当且仅当存在$w\in L(A)$,使得对任意的$0\le j<|w|$,若 $w(j)=a_k$,则$\pi^{i+j}\models f_k$(需要注意的是,如果 A 的初始状态同时是 A 的某个终止状态,则因为 A 能够接收空字,所以此时 $A(f_1,\ldots,f_n)$等价于 True).

一个 Kripke 结构 M 是一个四元组$\langle S,R,S_0,\rho\rangle$.其中,$S$ 是一个有穷集合,称为 M 的状态集合;$R\subseteq S\times S$,称为 M 的迁移关系;$S_0\subseteq S$,为 M 的初始状态集合;ρ:$S\to 2^{AP}$,称为 M 的标记函数.

设 Kripe 结构 $M=\langle S,R,S_0,\rho\rangle$,状态 $s\in S$.称σ是 M 中以 s 为起始状态的路径,如果$\sigma=s_0s_1\ldots$,其中 $s_0=s$,且每个$\langle s_i,s_{i+1}\rangle\in\mathbf{R}$.若线性结构$\pi$满足:对于任意的 $i\in\mathbf{N}$,$\pi(i)=\rho(s_i)$,则称π为σ的路径标记.M 的一个公平性限制是一组状态集$\{F_1,\ldots,F_n\}$,其中每个 F_i 都是 S 的子集.M 中路径σ满足公平性限制$\{F_1,\ldots,F_n\}$是指,对于每个 $1\le i\le n$,σ 无穷多

次经过 F_i 中的状态.

给定 ETL 公式 f,若对于 M 中任何一条以 s 为起始状态且满足公平性约束的路径的标记π都有$\pi\models f$ 成立,则记作 $M,s\models f$.若对于任意的 $s\in S_0$,都有 $M,s\models f$,则称 M 满足 f,记作 $M\models f$.ETL 的模型检验问题就是对给定的 Kripke 结构 M 以及 ETL 公式 f,判断是否有 $M\models f$ 成立.

2 ETL的符号化模型检验

采用线性框架下的时序逻辑作为规约语言,具有许多优势.为了将符号化方法应用于 LTL 的模型检验中,Clarke 等人给出了如下的检验框架:

设 f 为描述规约的 LTL 公式,则首先构造一个关于 $\neg f$ 的 Tableau $T_{\neg f}$.$T_{\neg f}$中的每个节点都是$\neg f$的某些特殊子公式(称作基本子公式)集的子集.接下来,构建 $T_{\neg f}$和待验证迁移系统 M 的乘积.可以证明:M 中存在一条路径标记满足性质$\neg f$(或者说违反性质 f),当且仅当在 $M\times T_{\neg f}$中存在一条在公平性$\{Sat(\neg(g\mathrm{U}h)\vee h)|g\mathrm{U}h$ 为 f 的中的子公式$\}$限制下的一条无穷路径(或者说,满足 CTL 公式 EGTrue).

由于 $T_{\neg f}$能够直接用位变元编码,所以该方法能够非常方便地使用符号化模型检验技术.在本节,我们将该方法推广到 ETL 的符号化模型检验中.

2.1 基于Tableau的ETL验证

给定 ETL 公式 f,归纳定义 f 的基本公式集合 $El(f)$如下:

1. 若 f 为原子命题 p,则 $El(f)=\{p\}$.
2. $El(\neg f)=El(f)$,$El(\bigcirc f)=\{\bigcirc f\}\cup El(f)$,$El(f_1\vee f_2)=El(f_1\wedge f_2)=El(f_1)\cup El(f_2)$.
3. 设 A 的状态集合为 Q,则 $El(A(f_1,\ldots,f_n))=El(f_1)\cup\ldots\cup El(f_n)\cup\{\bigcirc A^q(f_1,\ldots,f_n)|q\in Q\}$.

因此,$El(f)$中的公式或者为原子命题,或者为形如$\bigcirc g$ 的公式.给定 ETL 公式 f,对于 f 的每个子公式或 $El(f)$中的公式 g,归纳定义 $Sat(g)$如下:

1. 若 g 为原子命题或者形如$\bigcirc h$,则 $Sat(g)=\{K|K\subseteq El(f),g\in K\}$.
2. $Sat(\neg g)=2^{El(f)}\backslash Sat(g)$;$Sat(g_1\wedge g_2)=Sat(g_1)\cap Sat(g_2)$;$Sat(g_1\vee g_2)=Sat(g_1)\cup Sat(g_2)$.
3. 设自动机 A 的迁移函数为 δ,字母表为$\{a_1,\ldots,a_n\}$,则:
   1) 对每个非终止状态 q 有:$Sat(A^q(f_1,\ldots,f_n))=\bigcup_{1\le i\le n}(Sat(f_i)\cap\bigcup_{p\in\delta(q,a_i)}Sat(\bigcirc A^p(f_1,\ldots,f_n)))$;
   2) 对每个终止状态 q 有:$Sat(A^q(f_1,\ldots,f_n))=2^{El(f)}$.

给定 ETL 公式 f,关于 f 的 Tableau 是一个特殊的 Kripke 结构$\langle S_f,R_f,S_{0f},\rho_f\rangle$.其中:1) 状态集合 $S_f=2^{El(f)}$.2) 对于任意的 $El(f)$的子集 K 和 V,$\langle K,V\rangle\in R_f$当且仅当$\forall g.\bigcirc g\in El(f)\Rightarrow(\bigcirc g\in K\Leftrightarrow V\in Sat(g))$.3) $S_{0f}=Sat(f)$.4) 对于任意的 $El(f)$的子集 K,$\rho_f(K)=K\cap$AP.

定理 1. 给定迁移系统 M,以及 ETL 公式 f. 若 M 中存在路径标记π, 使得$\pi\models f$, 则在 f 的 Tableau T_f中必然存路径 $s_0s_1\ldots$使得对于任意的 $i\in\mathbf{N}$ 以及 f 的任意子公式 g,$\pi^i\models g$ 当且仅当 $s_i\in Sat(g)$.

证明:令 $s_i=\{h\in El(f)|\pi^i\models h\}$,则显然每个 s_i 都是 T_f 中的状态.现在利用公式结构归纳法证明:对于 f 子公式 g,$\pi^i\models g$ 当且仅当 $s_i\in Sat(g)$.

- 当 $g\in El(f)$时(即当 g 形如 p, $\neg p$ 或$\bigcirc h$ 时),结论显然成立.
- 当 $g=\neg h$ 时,$\pi^i\models g$ 成立当且仅当$\pi^i\models h$ 不成立;由归纳假设,当且仅当 $s_i\notin Sat(h)$,当且仅当 $s_i\in Sat(g)$.
- 当 $g=g_1\vee g_2$ 时,$\pi^i\models g$ 当且仅当$\pi^i\models g_1$ 或$\pi^i\models g_2$;由归纳假设,当且仅当 $s_i\in Sat(g_1)$或 $s_i\in Sat(g_2)$,当且仅当 $s_i\in Sat(g)$.同理,当 $g=g_1\wedge g_2$ 时,$\pi^i\models g$ 当且仅当$\pi^i\models g_1$ 且$\pi^i\models g_2$;由归纳假设,当且仅当 $s_i\in Sat(g_1)$且 $s_i\in Sat(g_2)$,当且仅当 $s_i\in Sat(g)$.
- 当 $g=A^q(f_1,\ldots,f_n)$(其中,$A^q=\langle\{a_1,\ldots,a_n\},Q,\delta,q,F\rangle$)时:若 $q\in F$,则由于此时 g=True,结论显然;若 $q\notin F$,则容易证明 $A^q(f_1,\ldots,f_n)$等价于$\bigvee_{1\le i\le n}(f_i\wedge\bigvee_{p\in\delta(q,a_i)}\bigcirc A^p(f_1,\ldots,f_n))$.所以,由归纳假设$\pi^i\models g$ 成立当且仅当 $s_i\in\bigcup_{1\le i\le n}\left(Sat(f_i)\cap\bigcup_{p\in\delta(q,a_i)}Sat(\bigcirc A^P(f_1,\ldots,f_n))\right)$, 即 $s_i\in Sat(g)$.

由于$\pi\models f$,所以 $s_0\in Sat(f)$,因此,s_0 是 T_f 中的某个初始状态.同时,对于任意的$\bigcirc g\in El(f)$以及 s_i 和 s_{i+1},由上述性质可知:$\bigcirc g\in s_i$ 当且仅当$\pi^i\models\bigcirc g$ 当且仅当 $s_{i+1}\in Sat(g)$.这说明 $s_0,s_1,\ldots$必然是 T_f 中的某条路径. □

采用符号化的方法验证 LTL 性质,需要将其转化成带公平性约束的 CTL 模型检验问题.该公平性约束可以伴随 Tableau 给出,这是由 until 算子的特殊性所决定的.然而,在 ETL 中,时序连接子的种类要丰富得多,原来定义公平性的方法已不再适用.为此,需要引入关于自动机连接子 Tableau 的概念.

设 $A=\langle\Sigma,Q,\delta,q,F\rangle$.对每个 $P,P'\subseteq Q$ 以及$\Pi\subseteq\Sigma$,定义记号 $P\Rightarrow_{\Pi}P'$,其中:对每个 $q\in P\backslash F$,存在 $q'\in P'$以及 $a\in\Pi$,使得 $q'\in\delta(q,a)$.

对自动机连接子 $A=\langle\Sigma,Q,\delta,q,F\rangle$,可以为其构造一个 Tableau,$\langle S,R,S_0,\rho\rangle$,其中:

- $S=S_0=2^Q$.
- $\langle P,P'\rangle\in R$ 当且仅当:或者存在$\Pi\subseteq\Sigma$,使得 $P\Rightarrow_{\Pi}P'$ 成立;或者 $P=\varnothing$.
- 对于任意 $P\subseteq Q,\rho(P)=P$.

现在给出 ETL 检验算法.给定迁移系统 $M=\langle S_M,R_M,S_{0M},\rho_M\rangle$以及 ETL 公式 f,则可以按照下列步骤验证 $M\models f$ 是否成立:

1. 构建关于$\neg f$ 的 Tableau $T_{\neg f}=\langle S_{\neg f},R_{\neg f},S_{0\neg f},\rho_{\neg f}\rangle$.
2. 设 $g_1,\ldots,g_m$ 为$\neg f$ 中所有局部正出现的以自动机连接的子公式.不失一般性,设 $g_i=A_i(f_{(i,1)},\ldots,f_{(i,l(i))})$,其中 A_i 的字母表为Σ_i,$l(i)=|\Sigma_i|$.同时,为每个 g_i 构造关于 A_i 的自动机连接子 Tableau T_i.
3. 按照如下方法构建 $M,T_{\neg f}$ 以及与所有 $T_i(1\le i\le m)$的乘积 $T=\langle S,R,S_0,\rho\rangle$.其中:
   a. $S=\{\langle s_M,s_{\neg f},P_1,\ldots,P_m\rangle|s_M\in S_M,s_{\neg f}\in S_{\neg f},\rho_M(s_M)=\rho_{\neg f}(s_{\neg f});P_i$ 为 T_i 中的状态,并且对任意的 $q\in P_i$,有 $s_{\neg f}\in Sat(A_i^q(f_{(i,1)},\ldots,f_{(i,l(i))}))\}$.
   b. $\langle\langle s_M,s_{\neg f},P_1,\ldots,P_m\rangle,\langle s'_M,s'_{\neg f},P'_1,\ldots,P'_m\rangle\rangle\in R$ 当且仅当$\langle s_M,s'_M\rangle\in R_M,\langle s_{\neg f},s'_{\neg f}\rangle\in R_{\neg f}$,以及对任意的 $1\le i\le m$:
      1) 若 $P_i\neq\varnothing$,则存在$\Pi_i\subseteq\Sigma_i$ 使得 $P\Rightarrow_{\Pi_i}P'$,以及对每个 $a_j\in\Pi_i$,有 $s_{\neg f}\in Sat(f_{(i,j)})$;
      2) 若 $P_i=\varnothing$,则 $q\in P_i'$当且仅当 $s'_{\neg f}\in Sat(A_i^q(f_{(i,1)},\ldots,f_{(i,l(i))}))$.
   c. $\langle s_M,s_{\neg f},P_1,\ldots,P_m\rangle\in S_0$ 当且仅当 $s_M\in S_{0M},s_{0\neg f}\in S_{0\neg f}$.
   d. $\rho(\langle s_M,s_{\neg f},P_1,\ldots,P_m\rangle)=\rho_M(s_M)$.
4. 对每个 $1\le i\le m$ 定义公平集 $Fair_i=\{\langle s_M,s_{\neg f},P_1,\ldots,P_m\rangle|P_i=\varnothing\}$.
5. 在公平性$\{Fair_i|1\le i\le m\}$限制下检验 T 是否满足 CTL 性质 EGTrue.

定理 2. $M\models f$ 不成立当且仅当 T 在公平性$\{Fair_i|1\le i\le m\}$限制下满足 CTL 性质 EGTrue.

定理 2 将 ETL 的模型检验问题转化成为带公平性限制的 CTL 模型检验问题.该定理的证明见附录.

2.2 基于BDD的实现

上一节给出了从 ETL 模型检验到 CTL 模型检验的转化过程.然而,通过构造乘积迁移系统进行显式的模型检验所需的代价是非常大的.本节介绍该检验算法基于 BDD 的实现.

一般而言,可以利用 n 个位变元 $v_1,\ldots,v_n$ 来编码迁移系统中的 2^n 个状态.其中每个位变元有 0 和 1(或者 True 和 False)两种取值.系统的迁移关系,可以用一个 $2n$ 元真值函数 $\Delta(v_1,\ldots,v_n,v'_1,\ldots,v'_n)$ 来刻画,即状态$(b_1,\ldots,b_n)$是$(a_1,\ldots,a_n)$的一个后继当且仅当将Δ中 v_i 的值赋为 a_i,v'_i 的值赋为 b_i 后所得的值为 1.同时,用 n 元真值函数 $I(v_1,\ldots,v_n)$来刻画初始条件,即$(a_1,\ldots,a_n)$为初始状态,当且仅当将 v_i 的值赋为 a_i 后 I 的值为 1.另外,为每个命题变元 p 构造一个 n 元真值函数 $L_p(v_1,\ldots,v_n)$来记录其状态标记,即 p 在$(a_1,\ldots,a_n)$上被标记当且仅当将 v_i 的值赋为 a_i 后 L_p 的值为 1.在通常的编码表示中,每个命题变元对应于一个位变元,因此命题标记函数形如 $L_p=v_k$.

因此,只需存储这些真值函数的 BDD 表示,迁移系统的信息即可完整保留.当获得了 M_1 和 M_2 的 BDD 表示之后,很容易获得 $M_1\times M_2$ 的 BDD 表示——只需取 M_1,M_2 位变元集合的并集作为新的位变元集合;M_1,M_2 初始条件表达式的合取作为新的初始条件表达式;M_1,M_2 迁移表达式的合取作为新的迁移表达式.

对于输入的迁移系统 M 以及 ETL 公式 f,在按照定理 2 中的方法验证 $M\models f$ 是否成立时,需要构造多个 Tableau,最终这些 Tableau 会与 M 做乘积.

1. M 的 BDD 表示可以由输入直接得到.
2. 在构造 Tableau T_f 的 BDD 时,需要为 $El(f)$中每个公式创建一个位变元.进而,对于 f 的每个子公式 g,得到 $Sat(g)$基于位变元的表示.在表示 T_f 的迁移约束$\forall g.\bigcirc g\in El(f)\rightarrow(\bigcirc g\in s\leftrightarrow t\in Sat(g))$时,对每个$\bigcirc g\in El(f)$,若 v 为$\bigcirc g$ 对应的位变元,w 为 $Sat(g)$基于位变元的布尔表示,则只需在迁移函数中引入合取项 $\text{next}(w)\leftrightarrow v$ 即可.其中,next(w)表示将 w 中每个位变元 v 替换为 v'而得到的公式.
3. 对于每个在 f 中局部正出现的子公式 $g=A(f_1,\ldots,f_n)$,设 $A=\langle\{a_1,\ldots,a_n\},Q,\delta,q,F\rangle$,$Q=\{q_1,\ldots,q_m\}$.再设 T_f 中 $Sat(A^{q_i}(f_1,\ldots,f_n))$ 和 $Sat(f_j)$基于位变元的表示为 x_i 和 y_j,$\bigcirc A^{q_k}(f_1,\ldots,f_n)$对应的位变元为 z_k.这样,只需按照如下方式构建 T_g 的 BDD 表示:首先,为每个 $q_i\in Q$,引入一个位变元 v_i;其次,令 $Fair_g=\neg v_1\wedge\ldots\wedge\neg v_m$;而后,在迁移函数中添加下列合取项:

   a) 对每个 v_i,引入项 $Fair_g\rightarrow(v_i'\leftrightarrow z_i)$;

   b) 对每个 v_i,若 $q_i\notin F$,则引入项$\neg Fair_g\rightarrow(v_i\rightarrow(x_i\wedge\bigvee_{1\le j\le n}(y_j\wedge\bigvee_{q_k\in\delta(q_i,a_k)}v_k')))$.

   最后,在公平性限制中添加 $Fair_g$ 即可.

令 G 为 f 中所有形如 $A(f_1,\ldots,f_n)$的局部正出现的子公式,则上述过程构造的符号化迁移系统为 $M\times T_f\times\times_{g\in G}T_g$的符号化表示.因此,若采用基于BDD的符号化检验技术来验证 $M\models f$是否成立,需要的位变元的数目不超过 $m+2\times|El(f)|$.其中,m 为编码迁移系统 M 所需要的位变元数目.

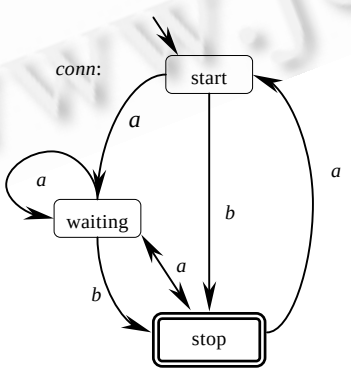
3 ENuSMV

上一节给出了 ETL 的符号化模型检验算法.基于该算法,我们在 NuSMV 的基础上实现了支持 ETL 规约的符号化模型检验器 ENuSMV.本节将介绍 ENuSMV 的新增语法机制以及若干 ETL 的模型检验结果.

3.1 语法扩展

NuSMV 中支持 CTL,LTL 以及 RTCRL 等时序逻辑.为了支持 ETL,我们对 NuSMV 的原有语法[20]进行了如下扩充:为支持用户自定义连接子,在 ENuSMV 中新增加了 CONNECTIVE,STATES 和 TRANSITIONS 这 3 个关键字.

其中,CONNECTIVE 后面跟着连接子的名称以及该连接子的字母表.STATES 后面跟着该自动机连接子的状态列表,状态名用某个标志符定义.前面冠以“>”的状态为初始状态(若状态列表中无初始状态或者初始状态不唯一,则 ENuSMV 会报出连接子定义错误),后面缀以“<”的状态为终止状态(若连接子中无终止状态,则 ENuSMV 不会报错,但会给出警告).关键字 TRANSITIONS 引导自动机的迁移定义.图 1 给出了某自动机连接子的 ENuSMV 语法定义示例.




```
CONNECTIVE conn (a,b)
STATES:
    >start, waiting, stop<
TRANSITIONS (start)
    case
        a: waiting;
        b: stop;
    esac;
TRANSITIONS (stop)
    case a: {start,waiting}; esac;
TRANSITIONS (waiting)
    case
        a: {waiting, stop};
        b: stop;
    esac;
```

Fig.1 Example of automaton connective description in ENuSMV

图 1 ENuSMV 中自动机连接子描述示例

另外,为了书写 ETL 规约,增加了关键字 ETLSPEC.比如.ETLSPEC *conn*(*p*,True)就是一个合法的 ETL 规约声明.除上述扩展语法成分外,NuSMV 原有的语法成分全部保留.因此,模型的描述与 NuSMV/SMV 相兼容.

3.2 实验结果

本节给出几个典型的使用 ENuSMV 的模型检验案例,并给出相应的实验结果.这里主要关心 3 个方面的问题:(1) 对于同时能够用 LTL,CTL 和 ETL 描述的公式,检验效率的比较;(2) 对于只能用 ETL 描述的公式,ENuSMV 的检验效率如何;(3) 使用 ENuSMV 能够检验的模型规模如何.

如非特别声明,本文实验所采用的平台均为:VMWare 5.0+RH Linux 7.2,512M 虚拟内存,外围处理器:Intel P4 3.0GHZ,单核.检验时间由 time 命令获得,按用户时间和系统时间之和计算,为 5 次检验时间的平均值.运行检验的空间开销被认为是以下两部分的乘积:1) 模型本身的状态数(可通过-r 参数取得),该部分与待验证性质无关;2) 运行检验构建的 Tableau 的大小(按相应位变元数的指数计算,随检验结果给出).

第 1 个例子是 NuSMV 中的经典同步模型示例:模 2^n 计数器.该模型由 n 个子模块(module)构成,而每个子模块是一个模 2 计数器.主模块/子模块的 SMV 模型描述如图 2 所示.两条典型性质的检验结果见表 1.其中,为第 1 个性质构建 LTL Tableau 和 ETL Tableau 所需的位变元数分别为 2 和 6;为第 2 个性质构建 LTL Tableau 和 ETL Tableau 所需的位变元数分别为 1 和 2.

```
MODULE main
VAR
    bit_0: counter_cell(1);
    bit_1: counter_cell(bit_0.carry_out);
      ⋮
    bit_n: counter_cell(bit_n−1.carry_out);
```

```
MODULE counter_cell(carry_in)
VAR
    pre_value: boolean;
    value: boolean;
ASSIGN
    init (value):=0;
    init (pre_value):=0;
    next (pre_value):=value;
    next (value):=(value+carry_in) mod 2;
DEFINE
    carry_out:=pre_value & carry_in;
```

Fig.2 ENuSMV code for modulo 2^n counter and counter cells

图 2 模 2^n 计数器及计数器单元 ENuSMV 代码

Table 1 Comparison of LTL, CTL and ETL model checking results for modulo 2^n counter

表 1 模 2^n 计数器 LTL,CTL 和 ETL 模型检验结果比较

Property (with LTL): GF *bit_n.carry_out*				
n	#reachable states/ #total states	# time (s)		
		CTL	LTL	ETL
3	10/64	0.028	0.030	0.029
6	66/4 096	0.030	0.101	0.061
9	514/262 144	0.065	4.830	2.060
Property (with LTL): F (*bit_n.carry_out*∧*bit_n*−1.*carry_out*)				
n	#reachable states/ #total states	# time (s)		
		CTL	LTL	ETL
3	10/64	0.021	0.023	0.024
6	66/4 096	0.025	0.030	0.031
9	514/262 144	0.040	0.040	0.040

在上述检验过程中,采用 ETL 连接子 $A_F=\langle\{a_1,a_2\},\{q_1,q_2\},\delta_F,q_1,\{q_2\}\rangle$来代替 LTL 中的时序算子 F.其中,$\delta_F(q_1,a_1)=\{q_1\},\delta_F(q_1,a_2)=\{q_2\},\delta_F(q_2,a_1)=\delta_F(q_2,a_2)=\varnothing$.显然,若 LTL 公式 f 对应于 ETL 公式 f',则 LTL 公式 Ff 和 Gf 分别对应于 ETL 公式 A_F(True,f')和$\neg A_F$(True,$\neg f'$).

NuSMV 中的一个典型的异步模型是非门环.该模型描述了一个由 n 个首尾相接的非门构成的环形电路.单个非门以及非门环电路的 SMV 描述如图 3 所示.这里主要检验的是系统的活性,即“每个非门无穷多次输出 0 和 1”.用 CTL 描述为$\bigwedge_{1\le i\le n}$(AG AF *cell_i.output*∧AG AF¬*cell_i.output*).实验结果见表 2.由于系统的对称性,只需

对一个非门验证该性质即可.为该性质构建 LTL Tableau 和 ETL Tableau 所需的额外位变元数分别为 3 和 12.

```
MODULE main
VAR
    cell_1: process inverter(cell_n.output);
    cell_2: process inverter(cell_1.output);
      ⋮
    cell_n: process inverter(cell_n−1.output);

MODULE inverter(input)
VAR
    output: boolean;
ASSIGN
    Init (output):=0;
    next (output):=!input;
FAIRNRSS running
```

Fig.3 ENuSMV code for inverters and inverting-ring

图 3 ENuSMV 中反转器和非门环的代码

Table 2 Comparison of CTL, LTL and ETL model checking results for inverting-ring

表 2 基于 CTL,LTL 和 ETL 的非门环模型检验结果比较

n	#reachable states/ #total states	# time (s)		
		CTL	LTL	ETL
6	63/64	0.040	0.067	0.082
9	511/512	0.060	0.090	0.257
12	4 095/4 096	0.080	0.131	0.670
15	32 767/32 768	0.188	0.504	2.309

上述实验结果表明:就相同的模型和相同的时序性质而言,采用 LTL 和 ETL 作为规约语言的检验时间保持一个恒定的比值;并且,随着系统规模的增大,这样的规律逐渐显著.在第 1 个实验中,执行时间比值分别约为 2:1 和 1:1;在第 2 个实验中,这个比值接近于 1:4.这说明,在某些特定的情况下,采用 ETL 作为规约时的检验效率能够高于等价的 LTL 规约的检验效率.造成这种现象的原因可能在于:相对于 LTL 而言,ETL Tableau 的公平性检查要简单一些,当时序算子嵌套层数较多时,ETL 检验就有可能在更短的时间内完成.

下面检验一组仅能被 ETL 描述的性质.对每个自然数 $k\geq 2$,定义自动机 $A_k=\langle\{a_1,a_2\},\{q_0,\ldots,q_{k-1},q_f\},\delta_k,q_0,\{q_f\}\rangle$,其中对每个 $0\leq i<k$,$\delta_k(q_i,a_1)=\{q_{((i+1) \bmod k)}\}$;$\delta_k(q_0,a_2)=\{q_f\}$;$\delta_k$ 在其余处的函数值均为$\varnothing$.易知:$\pi^i\models A_k(\text{True},f)$当且仅当存在某个 $t\in\mathbf{N}$,使得$\pi^{i+k\times t}f;\pi^i\neg A_k(\text{True},\neg f)$当且仅当对任意的 $t\in\mathbf{N}$,都有$\pi^{i+k\times t}\models f$成立.为方便起见,记 $A_k(\text{True},f)$为 $\mathrm{C}^k f$,记$\neg A_k(\text{True},\neg f)$为 $\mathrm{P}^k f$.由文献[9]可知,时序算子 C^k 和 P^k 均无法用 LTL 算子来刻画.表 3 给出了模 2^n 计数器关于该类性质($\bigcirc^k$ 是 k 个连续的$\bigcirc$算子的缩写)的检验结果.

Table 3 Results of model checking properties involving $\mathrm{C}^k/\mathrm{P}^k$ operators

表 3 带有 $\mathrm{C}^k/\mathrm{P}^k$ 算子性质的模型检验结果

n	#reachable states/ #total states	k	Property: C^k(*bit_n.carry_out*)		Property: $\bigcirc^k\mathrm{P}^k$(*bit_0.carry_out*)	
			# time (s)	#tableau bit-variables	# time (s)	#tableau bit-variables
3	10/64	2	0.031	3	0.040	8
6	66/4 096	2	0.040	3	0.048	8
9	514/262 144	4	0.120	5	0.061	14
12	4 098/16 777 216	4	0.860	5	0.167	14

为了测试 ENuSMV 能够处理的模型规模,我们还检验了由 Slat 抽取的 SELinux 的安全策略配置模型.该模型的描述部分有 120 147 行.在 512M 内存,3GHZ 处理器,Fedora 7 Linux 上,对文献[21]中提出的两类关键安全性质(包括两个 event assertion 和一个 order assertion,采用 ETL 书写)进行了检验,实验结果见表 4.注意:该实验中的状态数及可达状态数是针对最终乘积进行的统计.

Table 4 Result of ETL model checking SELinux security policy files

表 4 SELinux 安全策略文件的 ETL 检验结果

Property	#reachable states/#total states	#tableau bit-variables	# time (s)
Event assertion 1	1.41717e+10/2.39638e+10	7	37.969
Event assertion 2	9.06992e+11/1.53368e+12	13	56.048
Order assertion	2.83435e+10/4.79275e+10	8	38.066

4 结 论

本文讨论了 ETL 的符号化模型检验算法,并在 NuSMV 的基础上实现了支持 ETL 符号化模型检验的工具 ENuSMV.ETL 可以看作是线性时序逻辑的扩展,因此,其推理及检验技术是其各种逻辑片断的泛化.ETL 的推理系统可以派生出其他逻辑片断的推理系统[22].ETL 的符号化检验算法同样也可以例化出各种线性时序逻辑片断的符号化模型检验算法.本文给出的 ETL 符号化检验算法主要针对 ETL_f.在 ETL_f 中,形如 $A^q(f_1,\ldots,f_n)$的公式描述的是活性性质,而形如 $\neg A^q(f_1,\ldots,f_n)$的公式描述的是安全性质.在其他类型的 ETL 中,情形稍有不同.比如对于 ETL_l,形如 $A^q(f_1,\ldots,f_n)$的公式描述的是安全性质,而形如$\neg A^q(f_1,\ldots,f_n)$的公式描述的是活性性质.采用 ETL_l 作为规约语言,描述全部的ω-正规性质需要至少两层的时序连接子嵌套[13].

本文将来的工作包括 3 个方面:第一,研究采用交错自动机(alternating automata)作为连接子的 ETL[15]的符号化检验算法.这种 ETL 能够更加简洁地表达时序性质,并且更加接近于 PSL 中的 SERE 表达式.第二,研究采用双向自动机作为连接子的 ETL 符号化检验算法.当前,NuSMV 支持带有过去时态算子的 LTL 检验.研究该问题有助于更好地了解双向时态算子的性质,使用户更加灵活地定义规约性质.第三,研究采用其他类型自动机作为时序连接子的 ETL 的符号化模型检验算法.比如,SMV 模块可以简洁地转换为采用交错 Büchi 自动机作为连接子的 ETL.开发该种逻辑的符号化算法,有可能实现高效的模型等价性检验工具.

致谢 在本工作的研究过程中,得到了陈火旺教授的大力帮助和指导.遗憾的是,陈老师还未看到本文定稿,便永远地离我们而去.在此,谨向陈老师表达我们无限的缅怀和深切的哀思.

References:

[1] Pnueli A. The temporal logic of programs. In: Young P, ed. Proc. of the 18th IEEE Symp. on Foundation of Computer Science. Providence: IEEE Computer Society, 1977. 46−57.

[2] Emerson EA, Clarke EM. Characterizing correctness properties of parallel programs using fixpoints. In: Bakker JW, Leeuween J, eds. Int'l Colloquium on Automata, Languages and Programming. LNCS 85, Berlin, Heildelberg: Springer-Verlag,1980.169−181.

[3] Vardi MY. Linear vs. branching time: A complexity-theoretic perspective. In: Pratt V, ed. Proc. of the 13th Annual IEEE Symp. of Logic in Computer Science. Indianapolis: IEEE Computer Society, 1998. 394−405.

[4] Vardi MY. Branching vs. linear time: Final showdown. In: Margaria T, Wang Y, eds. Proc. of the 7th Int'l Conf. of Tools and Algorithms for the Construction and Analysis of Systems. LNCS 2031, Berlin, Heidelberg: Springer-Verlag, 2001. 1−22.

[5] Nain S, Vardi MY. Branching vs. linear time: Semantical perspective. In: Namjoshi KS, Yoneda T, Higashino T, Okamura Y, eds. Int'l Symp. on Automated Technology on Verification and Analysis. LNCS 4762, Berlin, Heidelberg: Springer-Verlag, 2007. 19−34.

[6] McMillian KL. Symbolic model checking, an approach to the state explosion problem [Ph.D. Thesis]. Boston: Carnegie Mellon University, 1993.

[7] Clarke EM, Grumberg O, Hamaguchi K. Another look at LTL model checking. In: Dill D, ed. Proc. of the 6th Int'l Conf. on Computer-Aided Verification. LNCS 818, Berlin, Heidelberg: Springer-Verlag, 1994. 415−427.

[8] Cimatti A, Clarke EM, Giunchiglia1 F, Roveri M. NuSMV: A new symbolic model verifier. In: Halbwachs N, Peled D, eds. Int'l Symp. on Computer Aided Verification. LNCS 1633, Berlin, Heildelberg: Springer-Verlag, 1999. 495−499.

[9] Wolper P. Temporal logic can be more expressive. Information and Control, 1983,56(1-2):72−99.

[10] Lichtenstein S, Pnueli A, Zuck L. The glory of past. In: Parikh R, ed. Workshop on Logic of Programs. LNCS 193, Berlin, Heildelberg: Springer-Verlag, 1985. 196−218.

[11] Accellera. Accellera property languages reference manual. 2004. http://www.eda.org/vfv/docs/PSL-v1.1.pdf

[12] Armoni R, Fix L, Flaisher A, Gerth R, Ginsburg B, Kanza T, Landver A, Mador-Haim S, Singerman E, Tiemeyer A, Vardi MY, Zbar Y. The ForSpec temporal logic: A new temporal property-specification language. In: Katoen JP, Stevens P, eds. Proc. of the 8th Int'l Conf. on Tools and Algorithms for the Construction and Analysis of Systems. LNCS 2280, Berlin, Heidelberg: Springer-Verlag, 2002. 296−311.

[13] Vardi MY, Wolper P. Reasoning about infinite computations. Information and Computation, 1994,115(1):1−37.

[14] Piterman N. Extending temporal logic with ω-automata [MS. Thesis]. Rehovot: School of the Weizmann Institute of Science, 2000.

[15] Kupferman O, Piterman N, Vardi MY. Extended temporal logic revisit. In: Larsen KG, Nielsen M, eds. Proc. of the 12th Int'l Conf. on Concurrency Theory. LNCS 2154, Berlin, Heidelberg: Springer-Verlag, 2001. 519−535.

[16] Su KL, Luo XY, Lu GF. Symbolic model checking of CTL*. Chinese Journal of Computers, 2005,28(11):1798−1806 (in Chinese with English abstract).

[17] Bloem R, Galler S, Jobstmann B, Piterman N, Pnueli A, Weiglhofer M. Specify, compile, run: Hardware from PSL. Electronic Notes in Theoretical Computer Science, 2007,190(4):3−16.

[18] Boule M, Zilic Z. Efficient automata-based assertion-checker synthesis of SEREs for hardware emulation. In: Onodera H, Matsunaga Y, eds. Proc. of the 12th Conf. on Asia South Pacific Design Automation. Yokohama: IEEE Society, 2007. 324−329.

[19] Jin NY, Shen CJ. Dynamic verifying the properties of the simple subset of PSL. In: Zhu HB, ed. IEEE Symp. of Theoretical Aspects on Software Engineering. Shanghai: IEEE Society, 2007. 173−182.

[20] Cavada R, Cimatti A, Jochim CA, Keighren G, Olivetti E, Pistore M, Roveri M, Tchaltsev A. NuSMV 2.4 User Manual. 2007. http://nusmv.fbk.eu/NuSMV/userman/v24/nusmv.pdf

[21] Guttman J, Herzog A, Ramsdell J. Slat: Information flow analysis of security enhanced Linux. 2005. http://www.nsa.gov/SELinux

[22] Liu WW, Wang J, Dong W, Chen HW. Axiomatizing extended temporal logic fragments via instantiation. In: Jones CB, Liu ZM, Woodcock J, eds. Int'l Colloquium on Theoretical Aspects of Computing. LNCS 4711, Berlin, Heidelberg: Springer-Verlag, 2007. 322−336.

附中文参考文献:

[16] 苏开乐,骆翔宇,吕关锋.符号化模型检测 CTL*.计算机学报,2005,28(11):1798−1806.

附录. 定理2的证明

证明:为方便起见,假设所有的公式均写成了否定范式.并引入下列记号:对 T 中的任一状态 s,用 T^s 表示迁移系统 $\langle S,R,\{s\},\rho\rangle$($T^s$ 与 T 唯一的不同之处在于,前者的初始状态集合变为 $\{s\}$).

I) 下面采用公式结构归纳法证明.对于任意 T 中的状态 $s=\langle s_M,s_{\neg f},P_1,\ldots,P_m\rangle$ 以及 $\neg f$ 中局部正出现的子公式 g,若 $s_{\neg f}\in Sat(g)$,则对于 T^s 中任何一条满足公平性限制 $\{Fair_i|1\le i\le m\}$ 的无穷路径标记 π,都有 $\pi\models g$.

- 当 g 为原子命题 p 或 $g=\neg p$ 时,结论显然成立.
- 当 $g=g_1\wedge g_2$ 时,有 $s_{\neg f}\in Sat(g_1)$ 并且 $s_{\neg f}\in Sat(g_2)$.由归纳假设,有 $\pi\models g_1$ 并且 $\pi\models g_2$.因此,$\pi\models g$.
  若 $g=g_1\vee g_2$,则有 $s_{\neg f}\in Sat(g_1)$ 或者 $s_{\neg f}\in Sat(g_2)$.由归纳假设,有 $\pi\models g_1$ 或者 $\pi\models g_2$.因此,$\pi\models g$
- 当 $g=\bigcirc g'$ 时,设 $\pi=\rho(s_0)\rho(s_1)\rho(s_2)\ldots$,其中 $s_0=s$,以及对任意的 $i\in\mathbf{N}$,有 $\langle s_i,s_{i+1}\rangle\in R$.因为 $s_{\neg f}\in Sat(\bigcirc g')$,由公式 Tableau 的构造可知,$\bigcirc g'\in s_{\neg f}$.设 $s_1=\langle s'_M,s'_{\neg f},P'_1,\ldots,P'_m\rangle$,则由 R_M 定义可知,$s'_{\neg f}\in Sat(g')$.同时,$\pi^1=\rho(s_1)\rho(s_2)\ldots$ 是 T^{s_1} 中满足公平性的路径标记.由归纳假设,$\pi^1\models g'$,所以 $\pi\models g$.
- 当 $g=\neg A^q(f_1,\ldots,f_n)$,其中 A^q 是以 $\{a_1,\ldots,a_n\}$ 为字母表以 F 为终止状态集的自动机,由 $s_{\neg f}\in Sat(g)$ 可知,$q\notin F$.反设 $\pi\models A^q(f_1,\ldots,f_n)$,则由定义,存在有穷字 $w\in L(A)$ 使得对任意的 $0\le j<|w|$,若 $w(j)=a_k$,则 $\pi^j\models f_k$.设 $\pi=\rho(s_0)\rho(s_1)\rho(s_2)\ldots$,其中 $s_0=s$,以及对任意的 $i\in\mathbf{N}$,有 $\langle s_i,s_{i+1}\rangle\in R$.对每个 $0\le j<|w|$,设 $s_j=\langle s_{(M,j)},s_{(\neg f,j)},P_{(1,j)},\ldots,P_{(m,j)}\rangle$,则若 $w(j)=a_k$,有 $s_{(\neg f,j)}\in Sat(f_k)$(否则,$s_{(\neg f,j)}\in Sat(\neg f_k)$,但 π^j 是 T^{s_j} 中的公平路径的标记,由归纳假设,有 $\pi^j\models\neg f_k$,矛盾).不妨设 w 关于 A 的可接收运行为 $q_0,q_1,\ldots,q_{|w|}$,其中,$q_0=q,q_{|w|}\in F$.从而 $s_{|w|}\in Sat(A^{q_{|w|}}(f_1,\ldots,f_n))=2^{El(\neg f)}$.由 Sat 函数的定义,不难得到 $s_{\neg f}=s_0=Sat(A^{q_0}(f_1,\ldots,f_n))=Sat(A^q(f_1,\ldots,f_n))$.这与 $s_{\neg f}\in Sat(\neg A^q(f_1,\ldots,f_n))$ 矛盾.因此,假设 $\pi\models A^q(f_1,\ldots,f_n)$ 是不成立的,故 $\pi\models g$.
- 当 $g=A^q(f_1,\ldots,f_n)$ 时,其中 $A^q=\langle\Sigma,Q,\delta,q,F\rangle$,$\Sigma=\{a_1,\ldots,a_n\}$,如果 $q\in F$,那么显然 $\pi\models g$.下面假定 $q\notin F$.

不妨设 $\pi=\rho(s_0)\rho(s_1)\rho(s_2)\ldots$,其中,$s_0=s$,$\langle s_i,s_{i+1}\rangle\in R$,$s_i=\langle s_{(M,i)},s_{(\neg f,i)},P_{(1,i)},\ldots,P_{(m,i)}\rangle$.如果存在以 Σ 为字母表的有穷字 $w\in L(A^q)$,使得对于任意的 $0\le i<|w|$,若 $w(i)=a_k$,则 $s_{\neg f_i}\in Sat(f_k)$,那么 $\pi\models g$ 成立(因为根据归纳假设,对每个 $0\le i<|w|$,有 $\pi^i\models f_k$ 成立).

反设这样的有穷字 w 不存在.令 $Q_0=\{q\}$,再对每个 $i\in\mathbf{N}$,令 $Q_{i+1}=\{p|\exists r\in Q_i,a_k\in\Sigma$ 使得 $p\in\delta(r,a_k)$ 以及

$s_{(\neg f,i+1)}\in Sat(A^p(f_1,\ldots,f_n))\}$.由 Sat 函数的定义即可得出:对于任意的 $i\in\mathbf{N}$,$Q_i\neq\varnothing$,且 $Q_i\cap F=\varnothing$(否则,必然存在满足前述性质的有穷字 w).不妨设 g 是 f 中第 t 个局部正出现的自动机连接子公式.由 $s_0s_1s_2\ldots$是 T 中的公平路径以及 T 的公平性限制 $Fair_t$ 可知,存在无穷多个 i 使得 $P_{(t,i)}=\varnothing$.现设 $P_{(t,k)}=\varnothing$,则由自动机连接子的迁移关系定义可知:$Q_{k+1}\subseteq P_{(t,k+1)}$,并且对于任意的 $i\geq k+1$ 以及 $p\in Q_i$,若 $p\notin F$,则存在 $p'\in Q_{i+1}\cap P_{(t,i+1)}$.从而,对于任意的 $i\geq k+1$,$Q_i\cap P_{(t,i)}\neq\varnothing$.进而,对于任意的 $i\geq k+1$,$P_{(t,i)}\neq\varnothing$.这样,就与公平性限制 $Fair_t$ 矛盾.所以,这样的有穷字 w 必然存在.因此,$\pi\models g$ 成立.

II) 其次证明:若 M 中存在一条路径标记 π,使得 $\pi\models\neg f$,则在 T 中必然存在一条公平路径.

设 π 对应的路径为 $s_{(M,0)}s_{(M,1)}s_{(M,2)}\ldots$由定理 1,$T_{\neg f}$ 中必然存在路径 $s_{(\neg f,0)}s_{(\neg f,1)}s_{(\neg f,2)}\ldots$使得对于 $\neg f$ 的任意子公式 g 以及任意自然数 i,有 $\pi^i\models g$ 当且仅当 $s_{(\neg f,0)}\in Sat(g)$.由定义,$s_{(M,0)}$和 $s_{(\neg f,0)}$分别是 M 和 $T_{\neg f}$ 中的某个初始状态;并且对每个 $i\in\mathbf{N}$,有$\langle s_{(M,i)},s_{(M,i+1)}\rangle\in R_M$ 及$\langle s_{(\neg f,i)},s_{(\neg f,i+1)}\rangle\in R_M$.下面构造 T 中满足公平性 $\{Fair_i|1\leq i\leq m\}$的路径.对每个 $g_i=A_i(f_{(i,1)},\ldots,f_{(i,l(i))})$(这里 $1\leq i\leq m$),设 $\Sigma_i=\{a_{(i,1)},\ldots,a_{(i,l(i))}\}$,$Q_i$,$\delta_i$ 和 F_i 分别为 A_i 的字母表、状态集、迁移函数和接收状态集,则:

1. 令 $s_0=\langle s_{(M,0)},s_{(\neg f,0)},P_{(1,0)},\ldots,P_{(m,0)}\rangle$,其中,$P_{(i,0)}=\varnothing$.因为 $s_{(\neg f,0)}\in Sat(\neg f)$,所以 s_0 是 T 中的初始状态.
2. 令 $O_0=0$,$P_{(i,O_0+1)}=\{q|\pi^{O_0+1}\models A_i^q(f_{(i,1)},\ldots,f_{(i,l(i))})\}$.由于对于任意的 $q\in P_{(i,O_0+1)}$,都有 $\pi^{O_0+1}\models A_i^q(f_{(i,1)},\ldots,f_{(i,l(i))})$ 成立,由定义:存在 $w_q\in L(A_i^q)$,使得对于任意的 $0\leq j<|w_q|$,若 $w_q(j)=a_k$,则 $\pi^{O_0+1+j}\models f_k$.于是,对每个 $q\in P_{(i,O_0+1)}\setminus F_i$,存在函数 $\lambda_q:\{0,\ldots,|w_q|\}\to Q_i$,使得:$\lambda_q(0)=q$,$\lambda_q(|w_q|)\in F_i$,并且对于任意的 $0\leq j<|w_q|$,$\lambda_q(j+1)\in\delta_i(\lambda_q(j),w_q(j))$;同时,若 $\lambda_q(j)=p$,则 $\pi^{O_0+1+j}\models A_i^p(f_{(i,1)},\ldots,f_{(i,l(i))})$.
   令 $O_1=\max\{|w_q|\,|\,q\in P_{(i,O_0+1)}\setminus F_i\}+1$.对每个 $O_0+1<j<O_1$,令 $P_{(i,j)}=\{\lambda_q(j-O_0-1)|\,q\in P_{(i,O_0+1)}$,且 $O_0+1+|w_q|\geq j\}$,$\Pi_{(i,j)}=\{w_q(j-O_0-2)|\,q\in P_{(i,O_0+1)}$,且 $O_0+2+|w_q|\geq j\}$.
   令 $P_{(i,O_1)}=\varnothing$,显然,在 T_i 中,对任意的 $O_0+1<j<O_1$,有 $P_{(i,j-1)}\Rightarrow_{\Pi_{(i,j)}}P_{(i,j)}$ 成立;并且 $P_{(i,O_1-1)}\subseteq F_i$.
3. 再令 $P_{(i,O_1+1)}=\{q|\pi^{O_1+1}\models A_i^q(f_{(i,1)},\ldots,f_{(i,l(i))})\}$,类似于第 2 步中的构造,可以得到下一个 O_2 使得 $P_{(i,O_2)}=\varnothing$.反复利用次过程,可以得到 T_i 中的一条无穷路径 $P_{(i,O_0)}P_{(i,O_0+1)}\ldots P_{(i,O_1)}P_{(i,O_1+1)}\ldots P_{(i,O_2)}P_{(i,O_2+1)}\ldots$显然,该路径中有无穷多个 $k\in\mathbf{N}$,使得 $P_{(i,j)}=\varnothing$.
4. 对每个 $j\in\mathbf{N}$,令 $s_j=\langle s_{(M,j)},s_{(\neg f,j)},P_{(1,j)},\ldots,P_{(m,j)}\rangle$,容易验证$\langle s_j,s_{j+1}\rangle\in R$.于是,$s_0s_1\ldots$是 T 中的一条无穷路径,并且满足公平性$\{Fair_j|\,1\leq j\leq m\}$.

III) 若 $M\models f$ 不成立,则 M 中必然有某条路径标记 π,满足 $\pi\models\neg f$,由 II)可知,T 中存在一条公平路径,这等价于:T 在公平性$\{Fair_j|1\leq j\leq m\}$限制下满足 CTL 公式 EGTrue.反之,若 T 在公平性$\{Fair_j|1\leq j\leq m\}$限制下满足 CTL 公式 EGTrue,则 T 中必然存在一条公平路径 $s_0s_1\ldots$不妨设 $s_j=\langle s_{(M,j)},s_{(\neg f,j)},P_{(1,j)},\ldots,P_{(m,j)}\rangle$,因为 s_0 是 T 中的初始状态,所以 $s_{(\neg f,0)}\in Sat(\neg f)$.由 I)可知,该路径对应的路径标记 π 必然使得 $\pi\models\neg f$ 成立.注意到 π 同时也是 M 中的路径 $s_{(M,0)}s_{(M,1)}\ldots$对应的标记,因此 $M\models f$ 不成立. □

刘万伟(1980－),男,山东平原人,博士,CCF 会员,主要研究领域为模型检验,定理证明.

王昭飞(1981－),男,博士生,主要研究领域为程序分析及检验.

王戟(1969－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为高可信软件技术.