基于线性时序逻辑的实时系统模型检查*

李广元, 唐稚松

(中国科学院 软件研究所 计算机科学重点实验室,北京 100080)
E-mail: {ligy,cst}@ios.ac.cn
http://www.ios.ac.cn

摘要: 模型检查是一种用于并发系统的性质验证的算法技术.LTLC(linear temporal logic with clocks)是一种连续时间时序逻辑,它是线性时序逻辑LTL的一种实时扩充.讨论实时系统关于LTLC公式的模型检查问题,将实时系统关于LTLC公式的模型检查化归为有穷状态转换系统关于LTL公式的模型检查,从而可以利用LTL的模型检查工具来对 LTLC 进行模型检查.由于 LTLC 既能表示实时系统的性质,又能表示实时系统的实现,这就使得时序逻辑LTLC的模型检查过程既能用于实时系统的性质验证,又能用于实时系统之间的一致性验证.

关 键 词: 实时系统;时间自动机;线性时序逻辑;模型检查;性质验证
中图法分类号: TP301 文献标识码: A

文献[1]提出了一个带有时钟变量的线性时序逻辑 LTLC(linear temporal logic with clocks).LTLC 是一种实时逻辑[2],它与现有的实时逻辑的一个主要不同之处在于,LTLC 既能作为规范语言用来表示实时系统的性质,又能作为系统描述语言来表示实时系统的实现模型.它能在统一的框架下表示实时系统的模型和性质,并进行性质验证.而其它实时逻辑主要是作为规范语言用于表示系统的性质,它们缺乏表示状态转换的能力,因而不适合作为系统描述语言.LTLC 的这个特点使得它有助于系统性质的形式验证,并有助于系统的逐步求精.

模型检查(model checking)[3~6]是一种关于系统性质验证的算法方法,它通常采用状态空间搜索的方法来检查一个给定的计算模型(程序)是否满足某个(用时序逻辑公式表示的)特定性质.由于模型检查方法已在通信协议验证和硬件系统验证等方面的实际应用中取得了成功,所以该方法近年来得到广泛关注,随着计算机硬件速度的提高和算法的不断优化,模型检查方法的应用前景被普遍看好.

本文将讨论基于 LTLC 的实时系统的模型检查.在 LTLC 中,实时系统是用时间模块[1]来表示的.对于一个给定的时间模块 M 和一个用于表示系统性质的 LTLC 公式 φ,我们要给出检验 φ 是否是 M 的性质的算法方法.由于 M 也是一个 LTLC 公式,故要检验 φ 是否是 M 的性质只需要检验 $M \Rightarrow \varphi$ 在 LTLC 中是否是永真的.由于表示实时模型的公式 M 相当于一个时间自动机(timed automaton[7]),故利用从时间自动机构造域自动机(region automaton[6,7])的思想,我们可从公式 M 出发构造出一个 LTL(linear temporal logic[8,9])公式 $\hat{M}$,公式 $\hat{M}$ 表示了一个有穷状态转换系统,这个有穷状态转换系统本质上相当于文献[6,7]中构造的域自动机.由于可以证明 $M \Rightarrow \varphi$ 在 LTLC 中是永真的当且仅当 $\hat{M} \Rightarrow \overline{\varphi}$ 在 LTL 中是永真的(见本文定理 3.1,其中 $\overline{\varphi}$ 是一个由 φ 转换而来的 LTL 公式).而 $\hat{M} \Rightarrow \overline{\varphi}$ 由于其每个变量的取值域都是有限的,故其永真性在 LTL 中是可判定的[10,11],判定过程可使用命题 LTL 的可满足性判定算法[12]或 LTL 的模型检查算法[5]来完成.这样我们就将实时系统关于 LTLC 公式的

* 收稿日期: 2000-07-20; 修改日期: 2001-07-30
基金项目: 国家自然科学基金资助项目(60073020);国家"九五"重点科技攻关项目(98-780-01-07-01);国家863高科技发展计划资助项目(863-306-ZT02-04-1)

作者简介: 李广元(1962－),男,陕西乾县人,博士,副研究员,主要研究领域为形式化方法,时序逻辑,实时系统;唐稚松(1925－),男,湖南长沙人,研究员,博士生导师,中国科学院院士,主要研究领域为时序逻辑,形式化方法,软件工程,软件体系结构.

模型检查问题归结为有穷状态转换系统关于 LTL 公式的模型检查问题.

由于 LTLC 具有与其它实时逻辑不同的表达特征,故基于 LTLC 的模型检查方法可以验证其它模型检查方法所不能表示或不容易表示的一些性质.例如,如果要检查的公式 φ 表示的是另外一个时间模块(这在 LTLC 中是可能的,而在其他实时逻辑中则不一定),则本文的模型检查方法可以用来检查实时系统之间的求精关系,而其它模型检查方法却很少能用于检查实时系统的求精关系.

本文第 1 节给出了几个必要的定义和记号.第 2 节简要地给出了 LTLC 的语法和语义,并定义了时间模块的基本概念和语义.第 3 节给出了将 LTLC 公式的模型检查问题归结为 LTL 公式的模型检查问题的转换过程.第 4 节是一个简短的总结.

1 预备概念

在本文中,$\Re$,$\Re^+$,Z 和 N 将分别表示实数集、非负实数集、整数集和非负整数集.

定义 **1.1**. 称无穷序列 $\{t_i\}_{i\in N}$ 是一个时间序列,若

(1) $0=t_0<t_1<t_2<\ldots<t_n<\ldots$;

(2) 序列 $\{t_i\}_{i\in N}$ 是一个发散序列,即 $\lim_{n\to\infty}t_n=\infty$.

定义 **1.2**. 称 $\Re^+$ 上的实函数 $f(t)$ 是一个步函数,若存在时间序列 $\{t_i\}_{i\in N}$,使得对任意有 $i\in N$,$f(t)$ 在区间 (t_i,t_i+1) 上都是常函数(即取值为一个常数).若步函数 $f(t)$ 的取值域为集合 $\{0,1\}$ 的子集,则称 $f(t)$ 是一个布尔值步函数.

定义 **1.3**. 称 $\Re^+$ 上的实函数 $f(t)$ 是一个时钟函数,若存在时间序列 $\{t_i\}_{i\in N}$,使得

$$f(t)=\begin{cases}0 & \text{若 } t=0\\ t-t_i & \text{若 } t\in(t_i,t_{i+1}]\end{cases},$$

或者存在有穷序列 $t_0,t_1,t_2,\ldots,t_n$,使得 $0=t_0<t_1<t_2<\ldots<t_n$,且

$$f(t)=\begin{cases}0 & \text{若 } t=0\\ t-t_i & \text{若 } t\in(t_i,t_{i+1}]\\ t-t_n & \text{若 } t>t_n\end{cases}.$$

定义 **1.4**. 设 f 是 $\Re^+$ 上的一个步函数或时钟函数.定义一个伴随于 f 的函数 $f'\colon \Re^+\mapsto\Re$ 如下:任意 $t_0\in\Re^+$ 有:

$$f'(t_0)=\lim_{t\to t_0+}f(t).$$

2 LTLC 的语法和语义

LTLC 的字母包括以下符号和括号:

(1) 布尔型变量: $p,p_0,p_1,p_2,\ldots$;

(2) 时钟变量: $c,c_0,c_1,c_2,\ldots$;

(3) 常元: 0,1,2,3,…(非负整数);

(4) 关系符号:=,≤;

(5) 联结词: $\neg$, $\wedge$;

(6) 时序符: $'$(撇,新值符号),[](总是),U (直到).

定义 **2.1**. LTLC 的公式归纳定义如下:

$$\varphi:=p\mid p'\mid(c=m)\mid(c\le m)\mid(c'=c)\mid(c'=0)\mid(\neg\varphi)\mid(\varphi_1\wedge\varphi_2)\mid[]\varphi\mid(\varphi_1U\varphi_2)$$

这里,p 是一个布尔型变量,c 是一个时钟变量,m 是常元.

其他一些常见的联结词,如 $\vee$ (析取), $\Rightarrow$ (蕴涵)和 $\Leftrightarrow$ (等价)等可通过 $\wedge$ 和 $\neg$ 定义出来;时序算子 $<>$ (最终)可通过 [] 和 $\neg$ 定义出来.一些常用的函数和关系符号(如 $\ge$,<,>等)也可以作为已有公式的缩写在 LTLC 中定义出来.

注：本文定义的 LTLC 实际只是文献[1,11]中的同名逻辑的一个子部分,但为方便起见,本文仍用 LTLC 称呼这个子部分.

我们用 var(φ)来表示公式φ中出现的所有变量的集合.如果 var(φ)$\subseteq V$,则称φ是变量集 V 上的公式.

此外,我们称由下面的规则所生成的 LTLC 公式为状态公式,状态公式的特征是它不含时序符'[]'和'U'.

$$\varphi := p \mid p' \mid (c = m) \mid (c \le m) \mid (c' = c) \mid (c' = 0) \mid (\neg\varphi) \mid (\varphi_1 \wedge \varphi_2)$$

这里,p,c 和 m 等的含义同上.

2.1 LTLC的语义

在 LTLC 中,时间是用非负实数来表示的,布尔型变量被解释为时间域$\Re^+$上的一个布尔值步函数,时钟变量被解释为时间域$\Re^+$上的一个时钟函数,刚性变量被解释为一个实数.若布尔型变量 p 和时钟变量 c 分别被解释为布尔值步函数 f_p 和时钟函数 f_c,则和 c'将分别被解释为与 f_p 和 f_c 相伴随定义的两函数 f'_p 和 f'_c.常元 m 被解释为非负整数 m.≤被解释为实数间的'小于或等于'关系,=被解释为'等于'关系.

定义 **2.2**. 设 V 是一个有穷的变量集合,称 J 是变量集 V 上的一个 LTLC-模型(或解释),若

(1) 对 V 中的任一布尔型变量 p,J 指派$\Re^+$上的一个布尔值步函数 f_p 作为 p 的解释;

(2) 对 V 中的任一时钟变量 c,J 指派$\Re^+$上的一个时钟函数 f_c 作为 c 的解释.

设 V 是一个有穷的变量集,φ是 V 上的一个公式.如果 J 是 V 上的一个 LTLC-模型,则对于任意 $t_0 \in \Re^+$,类似于在文献[1]中的做法,可以归纳定义当 $t=t_0$ 时φ在模型 J 下的值 $J(\varphi,t_0)$.例

(1) $J(p,t_0) := f_p(t_0)$;

(2) $J(p',t_0) := f'_p(t_0)$;

(3) $J(c \le m, t_0) := \begin{cases} 1 & \text{若 } f_c(t_0) \le m; \\ 0 & \text{否则.} \end{cases}$

(4) $J(c' = c, t_0) := \begin{cases} 1 & \text{若 } f'_c(t_0) = f_c(t_0); \\ 0 & \text{否则.} \end{cases}$

(5) $J([]\varphi, t_0) := \begin{cases} 1 & \text{若对于任意 } t_1 \ge t_0 \text{ 有 } J(\varphi,t_1) = 1\text{；} \\ 0 & \text{否则.} \end{cases}$

如果$J(\varphi,t_0)=1$,则称φ在时刻$t=t_0$时为真.如果φ在时刻$t=0$时为真,则称J是φ的一个LTLC-模型,记为 $J \models \varphi$.

定义**2.3**. 设φ和ϕ是变量集V上的两个公式.如果φ的所有(在变量集V上的)模型都是ϕ的模型,则称ϕ是φ的一个逻辑结论,记作 $\varphi \models \phi$.

我们以时间模块(timed modules)[1]来模拟实时系统.每个时间模块只能有有限个变量(这里的变量可以是布尔型变量,其类型记为boolean;也可以是时钟变量,其类型记为clock).每个时间模块由有限个转换所组成.变量的值是通过转换的作用来改变的.时间模块有两种类型的转换:跳跃转换(jump transition)和延迟转换(delay transition).跳跃转换的发生不占用时间段,在其作用的瞬间(零时段内)它可以改变所有变量(无论命题变量或时钟变量)的值.但每个延迟转换的发生具有一个正的时间段,在其作用的这个时段内,命题变量的值保持不变,但时钟变量的值随时间流逝而同步增长,其增加量为所流逝的时间.在任意一个有限时间段中,跳跃转换只能发生有限次,相邻发生的两次跳跃转换之间为一个延迟转换的作用时段.

跳跃转换通常被写成形如vertex∧guard→new_vertex∧assignment的卫式命令的形式.这里vertex是由布尔型变量组成的表达式,用于表示跳跃转换发生时系统所处的顶点;new_vertex表示跳跃转换发生后系统将位于的新顶点;guard是跳跃转换的使能条件(enabling condition),它用于表示跳跃转换发生时系统的变量应满足的约束条件;assignment是跳跃转换的命令部分,它相当于赋值语句,它使模块中每个时钟变量c要么重置为0(即$c'=0$),要么维持原值不变(即$c'=c$),延迟转换通常被写成下面的形式:vertex→invariant.其中vertex表示延迟转换作用时所处

的顶点;invariant是延迟转换的不变量部分,用于表示在该转换作用的时间段内时钟变量必须满足的约束条件.

一般而言,一个时间模块具有如下的形式：

module module_name
var {variable_name: boolean | clock }*
init init_cond
jump {vertex ∧ guard → new_vertex ∧ assignment}*
delay {vertex → invariant }*

这里,init_cond是模块的初始条件,用于表示模块在初始时刻(即t=0时)其变量所应满足的条件.由于初始时刻时钟变量的值皆为0,故有 init_cond⇒c=0.关于时间模块的具体例子可见文献[1].

对于跳跃转换α:vertex∧guard→new_vertex∧assignment,称LTLC公式vertex∧guard∧new_vertex∧assignment为转换α所对应的时序逻辑公式,记作 TLF(α).对于延迟转换β:vertex→invariant,称 LTLC 公式 vertex∧invariant为延迟转换β所对应的时序逻辑公式,记作 TLF(β).

定义 **2.4**. 设 M 是一个时间模块, $\alpha_0, \alpha_1, ..., \alpha_{n-1}$ 是 M 的所有跳跃转换, $\beta_0, \beta_1, ..., \beta_{k-1}$ 是 M 的所有延迟转换.称LTLC公式 $\text{init\_cond} \wedge ([](V = V' \vee (\vee_{i<n} \text{TLF}(\alpha_i)))) \wedge ([](\vee_{j<k} \text{TLF}(\beta_j)))$ 为时间模块 M 所对应的时序逻辑公式;这里 init_cond 是 M 的初始条件,V 是 M 中所有变量的集合, $V = V'$ 是时序公式 $\wedge_{v \in V} (v = v')$ 的缩写形式.以下以TLF(M)来表示这个对应于模块 M 的时序逻辑公式.并以 TLF(M)的语义模型作为 M 的语义模型.

定义 **2.5**. 设M是一个时间模块,φ是一个LTLC公式.如果 $\text{TLF}(M) \models \varphi$,则称$\varphi$是$M$的一个性质,记作 $M \models \varphi$.

3 LTLC 的模型检查

3.1 线性时序逻辑LTL

根据本文的需要,我们介绍线性时序逻辑LTL[8,9]的一个子部分如下,但为方便起见,本节仍用LTL来称呼原LTL的这个子部分

LTL 的字母包括以下符号和括号：

(1) 命题变量： $p, p_0, p_1, p_2, \cdots$;
(2) 整型变量： $x, c, c_0, c_1, c_2, \cdots$;
(3) 常元：0,1,2,3,…(非负整数);
(4) 函数符号：–(取负),+(加),*(乘)；
(5) 关系符号：≤(小于或等于)；
(6) 联结词： $\neg, \wedge$;
(7) 时序符： $^+$(下一时刻), [](总是), U (直到).

LTL 的表达式和公式定义如下：

$$e := x \mid x^+ \mid m \mid (-e) \mid (e_1 + e_2) \mid (e_1 * e_2)$$
$$\varphi := p \mid p^+ \mid (e_1 \le e_2) \mid (\neg\varphi) \mid (\varphi_1 \wedge \varphi_2) \mid [] \varphi \mid (\varphi_1 U \varphi_2)$$

这里,x 是整型变量,m 是常元,p 是命题变量,e 为表达式, φ 为公式.

其他一些常用联结词、时序算子和关系符(如∨,⇒,◇,≥,=等)可按通常的做法在LTL中定义出来.

设 V_d 是 LTL 中变量(包括命题变量和整型变量)的一个有穷集合, V_d 上的一个状态π是指 V_d 上这样的一个指派:对于 V_d 中的命题变量 p 有 $\pi(p) \in \{0,1\}$,对于 V_d 中的整型变量 x 有 $\pi(x) \in Z$. V_d 上的一个 LTL-解释 Π 是 V_d 上状态的一个无穷序列.

定义 **3.1**. 设 $\Pi := \langle \pi_0, \pi_1, \pi_2, ... \rangle$ 是 V_d 上的一个 LTL-解释.

(1) 对于 V_d 上的任一表达式 e 及 $i \in \mathrm{N}$,归纳定义 e 在 Π 的第 i 个状态下的值 $\Pi(e,i) \in Z$ 如下：

$\Pi(x,i) ::= \pi_i(x)$, $\Pi(x^+,i) ::= \pi_{i+1}(x)$, $\Pi(m,i) ::= m$ (这里 m 是常元),其余略.

(2) 对于 V_d 上的任一公式 φ 及 $i \in \mathrm{N}$,归纳定义 φ 在 Π 的第 i 个状态下的值 $\Pi(\varphi,i) \in \{0,1\}$ 如下:

$\Pi(p,i) ::= \pi_i(p)$, $\Pi(p^+,i) ::= \pi_{i+1}(p)$, $\Pi([]\varphi,i) ::= \mathrm{Min}\{\Pi(\varphi,j) \mid j \in N \text{且} j \geq i\}$, 其余略.这里 Min 表示取集合中的最小值.

若 $\Pi(\varphi,0)=1$,则称解释 Π 是 φ 的一个 LTL-模型,记为 $\Pi \models_{ltl} \varphi$.

一个状态转换系统 P 通常可用形如 $\varphi_0 \wedge ([]Next) \wedge L$ 的一个 LTL 公式表示出来(参见文献[8]),其中 φ_0 表示系统的初始条件,$Next$ 表示系统从当前状态到下一状态的转换关系,L 表示系统需满足的活性(liveness)要求.设 ϕ 是一个 LTL 公式,若 $\varphi_0 \wedge ([]Next) \wedge L \Rightarrow \phi$ 在 LTL 中永真,则称 ϕ 是系统 P 的性质,记为 $P \models_{ltl} \phi$.当 P 和 ϕ 中的所有变量的取值域都为事先给定的有穷集时,$\varphi_0 \wedge ([]Next) \wedge L \Rightarrow \phi$ 的永真性是可判定的[10,11].

3.2 域等价

设 $M ::= init \wedge ([](V' = V \vee \alpha_1 \vee \alpha_2 \vee ... \vee \alpha_m)) \wedge ([](\beta_1 \vee \beta_2 \vee ... \vee \beta_k))$ 是一个时间模块,这里 V 是 M 中所有变量的集合,$\alpha_1, \alpha_2, ..., \alpha_m$ 分别是 M 的 m 个跳跃转换所对应的时序逻辑公式,$\beta_1, \beta_2, ..., \beta_k$ 分别是 M 的 k 个延迟转换所对应的时序逻辑公式,$init$ 是 M 的初始条件.设 n 是一个正整数,且时间模块 M 中出现的常元皆不超过 n.在本节以下的讨论中我们固定上面给出的这个模块 M 和正整数 n.

对于上面固定的正整数 n,定义函数 $h_n : \Re \mapsto \mathrm{Z}$ 如下:

$$h_n(a) ::= \begin{cases} 2a & \text{若 } a = \lfloor a \rfloor \text{且} |a| \leq n \\ 2\lfloor a \rfloor + 1 & \text{若} a \neq \lfloor a \rfloor \text{且} |a| \leq n \\ 2n+1 & \text{若} a > n \\ -2n-1 & \text{若} a < -n \end{cases}$$

这里,$\lfloor a \rfloor$ 和 $|a|$ 分别表示实数 a 的整数部分和 a 的绝对值.为方便起见,下面常以 $[a]_n$ 记 $h_n(a)$.

不难验证,$h_n(a)$ 是单增的,且对于不超过 n 的非负整数 m 有: $h_n(a) \leq 2m$ 当且仅当 $a \leq m$.

设 $V = B \cup C$,其中 B 是命题变量集,C 是时钟变量集,令 $V' ::= \{v' \mid v \in V\}$.称 $V \cup V'$ 上的映射 σ 为变量集 V 上的一个 LTLC-状态,若对任意 $p \in B$,有 $\sigma(p) \in \{0,1\}$ 和 $\sigma(p') \in \{0,1\}$;且对任意 $c \in C$,有 $\sigma(c) \in \Re^+$ 和 $\sigma(c') = 0 \vee \sigma(c') = \sigma(c)$.设 σ 是变量集 V 上的一个状态,对任意状态公式 φ,我们可按通常的做法定义 φ 在 σ 下的真值 $\sigma(\varphi)$.

对于正整数 n,定义 V 上的状态之间的等价关系 $\equiv_n$ 如下:

定义 **3.2**. 设 σ 和 τ 是 V 上的任意两个 LTLC-状态,$\sigma \equiv_n \tau$ 当且仅当

(1) 对任意 $p \in B$ 有 $\sigma(p) = \tau(p)$, $\sigma(p') = \tau(p')$;

(2) 对任意 $c \in C$ 有 $[\sigma(c)]_n = [\tau(p)]_n$, $[\sigma(c')]_n = [\tau(c')]_n$;

(3) 对任意 $c, d \in C$ 有 $[\sigma(c) - \sigma(d)]_n = [\tau(c) - \tau(d)]_n$.

我们记 σ 所在的等价类为 $[\sigma]_n$.

对于上面的正整数 n 及变量集 V,下面以 $Form(V,n)$ 表示 LTLC 公式集 $\{\varphi \mid \mathrm{var}(\varphi) \subseteq V$ 且 φ 中的常元皆不超过 $n\}$.

引理 **3.1.** 设 σ, τ 是 V 上的两个等价状态,$\varphi \in Form(V,n)$ 且 φ 是一个状态公式,则 $\sigma(\varphi) = \tau(\varphi)$.

证明:对公式 φ 进行归纳即可得到,此处只给出情况 $\varphi ::= (c' = c)$ 时的证明作为示例.

假设 $\sigma(c' = c) = 1$,而 $\tau(c' = c) = 0$.于是 $\sigma(c') = \sigma(c)$,且 $\tau(c') \neq \tau(c)$.由状态的定义知 τ 满足 $\tau(c') = 0 \vee \tau(c') = \tau(c)$,于是 $\tau(c') = 0$,再由 $\sigma \equiv_n \tau$ 得出 $\sigma(c') = 0$,进一步有 $\sigma(c) = 0$ 和 $\tau(c) = 0$,但这与 $\tau(c') \neq \tau(c)$ 矛盾. □

定义 **3.3**. 设 σ 是 V 上的一个 LTLC-状态, t 是一个正实数,定义 $\sigma+t$ 是这样一个状态:对于任意 $p\in B$ 有 $(\sigma+t)(p)=(\sigma+t)(p')=\sigma(p')$,对于任意 $c\in C$ 有 $(\sigma+t)(c)=(\sigma+t)(c')=\sigma(c')+t$.

定义 **3.4**. 设 σ 是 V 上的一个 LTLC-状态.

(1) 称 σ 是不连续状态(记为 discontin(σ)),若存在 $p\in B$ 使得 $\sigma(p')\neq\sigma(p)$,或存在 $c\in C$ 使得 $\sigma(c')=0$;否则,称其为连续状态,记为 contin(σ).

(2) 称 σ 是边界状态(记为 boundary(σ)),若存在 $c\in C$ 使得 $\sigma(c')\in\{0,1,2,...,n\}$.

定义 **3.5**. 设 σ 和 τ 是 V 上的任意两个 LTLC-状态;称 $\sigma\triangleright\tau$,若 contin(σ)且对于任意 $v\in V$ 有 $\tau(v)=\sigma(v)$ (注意这里 $\tau(v')$ 不一定等于 $\tau(v)$).

定义 **3.6**. 设 σ 和 τ 是 V 上的任意两个状态, t 是一个正实数.

(1) 称$\sigma \overset{t}{\rightsquigarrow} \tau$,若 $((\sigma+t)\triangleright\tau)\wedge\forall s((s>0\wedge s<t)\Rightarrow(\sigma+s\equiv_n\sigma\vee\sigma+s\equiv_n\tau))$.

(2) 称 $\sigma \rightsquigarrow \tau$,若存在正实数 t,使得$\sigma \overset{t}{\rightsquigarrow} \tau$ 成立.

定义 **3.7**. 设 J 是 V 上的一个 LTLC-解释, $t\in\Re^+$, 我们用 $J(t)$ 来表示 V 上如下定义的一个 LTLC-状态:对任意的 $v\in V$,变量 v 的值为 $f_v(t)$, v' 的值为 $f_v'(t)$.

(1) 若 discontin($J(t)$),则称 t 是 J 的一个不连续点.

(2) 若 boundary($J(t)$),则称 t 是 J 的一个边界点.

由定义 1.2 和 1.3 可知, $\Im$ 的不连续点和边界点均为 $\Re^+$ 上的孤立点.

定义 **3.8**. 设 J 是 V 上的一个 LTLC-解释, $0=t_0<t_1<t_2<...<t_n<...$ 是一个时间序列,若对于任意 $i\in$ N 有 $\mathcal{J}(t_i)\overset{\delta_i}{\rightsquigarrow}\mathcal{J}(t_{i+1})$ (这里 $\delta_i=t_{i+1}-t_i$),则称 $J(t_0),J(t_1),J(t_2),...$ 是 J 的一个执行序列.

引理 **3.2**. 设 J 是 V 上的一个 LTLC-解释,则存在时间序列 $0=t_0<t_1<t_2<...<t_n<...$ 使得 $J(t_0),J(t_1),J(t_2),...$ 是 J 的一个执行序列.

证明:设 G 是 J 的所有边界点和不连续点构成的集合,则 G 是 $\Re^+$ 上的一个孤立点集.

情况 1.当 G 是有限集时,设 G 中的元素(从小到大)为 $s_0,s_1,s_2,...,s_m$.

对于任意 $i\in$ N,定义 t_i 如下

$$t_i:=\begin{cases}s_{(i/2)} & \text{若}i\text{是偶数且}i\le 2m\\ (s_{((i-1)/2)}+s_{((i+1)/2))})/2 & \text{若}i\text{是奇数且}i<2m\\ s_m+(i-2m) & \text{若}i>2m\end{cases},$$

情况 2.当 G 是无限集时,由于 G 是孤立点集,故可设 G 中的元素可以从小到大排列为 $s_0,s_1,s_2,...,s_k,...$.

对于任意 $i\in$ N,定义 t_i 如下:

$$t_i:=\begin{cases}s_{(i/2)} & \text{若}i\text{是偶数}\\ (s_{((i-1)/2)}+s_{((i+1)/2))})/2 & \text{若}i\text{是奇数}\end{cases}'$$

不难验证这样定义的 t_i 满足引理的要求. □

定义 **3.9**. 设 $\Im:=\langle J(t_0),J(t_1),J(t_2),...\rangle$ 是解释 J 的一个执行序列,定义函数 $\ell_\Im:\Re^+\mapsto$ N 如下:

$$\ell_\Im(s):=\begin{cases}i & \text{若}s=t_i\vee(s\in(t_i,t_{i+1})\wedge\neg(\text{discontin}(\Im(t_i))\vee\text{boundary}(\Im(t_i))))\\ i+1 & \text{若}s\in(t_i,t_{i+1})\wedge(\text{discontin}(\Im(t_i))\vee\text{boundary}(\Im(t_i)))\end{cases}.$$

不难看出,函数 $\ell_\Im$ 是满的和单增的,且对于任意 $s\in\Re^+$ 有 $\Im(s)\equiv_n\Im(t_{\ell_\Im(s)})$.

3.3 构造有限状态变换系统 $\hat{M}$

设 $V=B\cup C$,其中 B, C 分别为命题变量集和时钟变量集.我们将构造一个有限状态变换系统 $\hat{M}$,它是一个LTL公式,它的变量集合为 $V_d=B\cup C\cup D\cup B'\cup C'\cup D'$,其中 $D:=\{\Delta_{cd}|c,d\in C\}$,$B':=\{p'|p\in B\}$,$C':=\{c'|c\in C\}$,$D':=\{\Delta'_{cd}|c,d\in C\}$.$B\cup B'$中的变量是命题变量,取值 0 和 1;$C\cup C'$中的变量为整型变量,取值域为 $\{0,1,2,...,2n+1\}$;$D\cup D'$中的变量也是整型变量,取值域为 $\{-2n-1,-2n,...,-2,-1,0,1,2,...,2n+1\}$.

定义 **3.10**. 对于 V 上的一个状态 σ,定义其在 V_d 上的投影状态 $\bar{\sigma}$ 如下:

(1) 对任意 $p\in B$,$\bar{\sigma}(p)=\sigma(p),\sigma(p')=\sigma(p')$;

(2) 对任意 $c\in C$,$\bar{\sigma}(c)=[\sigma(c)]_n,\bar{\sigma}(c')=[\sigma(c')]_n$;

(3) 对任意 $c,d\in C$,$\bar{\sigma}(\Delta_{cd})=[\sigma(c)-\sigma(d)]_n,\bar{\sigma}(\Delta'_{cd})=[\sigma(c')-\sigma(d')]_n$.

显然,对于 V 上的任意两个状态 σ 和 τ 有:$\bar{\sigma}=\bar{\tau}$ 当且仅当 $[\sigma]_n=[\tau]_n$ (即当且仅当 $\sigma\equiv_n\tau$).

定义 **3.11**. 对于 $\varphi\in Form(V,n)$,归纳定义 V_d 上的 LTL 公式 $\overline{\varphi}$ 如下:

(1) 对 $p\in B$,$\overline{p}:=p$,　$\overline{p'}:=p'$;

(2) 对 $c\in C$,$\overline{c\le m}:=c\le 2m,\overline{c=m}:=c=2m,\overline{c'=0}:=c'=0,\overline{c'=c}:=c'=c$;

(3) $\overline{\neg\varphi}:=\neg\overline{\varphi},\overline{\varphi_1\wedge\varphi_2}:=\overline{\varphi_1}\wedge\overline{\varphi_2},\overline{[]\varphi}:=[]\overline{\varphi},\overline{\varphi_1 U\varphi_2}:=\overline{\varphi_1}U\overline{\varphi_2}$.

引理 **3.3**. 设 σ 是 V 上的一个状态,$\varphi\in Form(V,n)$ 且 φ 是一个状态公式,则 $\sigma(\varphi)=\bar{\sigma}(\overline{\varphi})$.

证明:对 φ 用归纳即可,我们只给出情况 $\varphi:=(c\le m)$ 时的证明作为示例.

$\bar{\sigma}(\overline{\varphi})=1\Leftrightarrow\bar{\sigma}(c\le 2m)=1\Leftrightarrow\bar{\sigma}(c)\le 2m\Leftrightarrow[\sigma(c)]_n\le 2m\Leftrightarrow\sigma(c)\le m\Leftrightarrow\sigma(\varphi)=1$. □

引理 **3.4.**设 $\Im:=\langle J(t_0),J(t_1),J(t_2),...\rangle$ 是解释 J 的一个执行序列,$\overline{\Im}:=\langle\overline{J(t_0)},\overline{J(t_1)},\overline{J(t_2)},...\rangle$ 是序列 $\Im$ 在 V_d 上的投影序列.对于任意 $\varphi\in Form(V,n)$,有:$J|=\varphi$ 当且仅当 $\overline{\Im}|=_{ltl}\overline{\varphi}$.

证明:只需证对任意 $s\in\Re^+$ 有:$J(\varphi,s)=\overline{\Im}(\overline{\varphi},\ell_{\Im}(s))$ 即可.

对 φ 进行归纳,归纳基础可由引理 3.3 得到,下面给出情况 $\varphi:=[]\phi$ 时的证明,其余情况略.

(1) 若 $J([]\phi,s)=1$,则对于任意 $r\ge s$ 有 $J(\phi,r)=1$,由归纳假设得,对任意 $r\ge s$ 有 $\overline{\Im}(\overline{\phi},\ell_{\Im}(r))=1$.由于 $\ell_{\Im}$ 是满射,且是单增的,于是可得 $\overline{\Im}(\overline{[]\phi},\ell_{\Im}(s))=1$.

(2) 若 $J([]\phi,s)=0$,则存在 $r\ge s$ 使得 $J(\phi,r)=0$,由归纳假设知,存在 $r\ge s$ 使.$\overline{\Im}(\overline{\phi},\ell_{\Im}(r))=0$.由于 $\ell_{\Im}(r)\ge\ell_{\Im}(s)$,于是 $\overline{\Im}(\overline{[]\phi},\ell_{\Im}(s))=0$. □

定义 **3.12**. 定义 V_d 上的 LTL 公式 *Next* 如下:

$$
\begin{aligned}
Next:=&(B^+=B')\wedge(D^+=D')\wedge(\\
&(boundary\Rightarrow(((B')^+=B')\wedge((D')^+=D')\wedge next1))\wedge\\
&((discontin\wedge\neg boundary)\Rightarrow(((B')^+=B')\wedge(C^+=C')\wedge((C')^+=C')\wedge((D')^+=D')))\wedge\\
&((\neg discontin\wedge\neg boundary)\Rightarrow(next2\vee(C^+=C'))))
\end{aligned}
$$

$B^+ = B' \quad := \wedge_{p\in B}(p^+ = p')$

$(B')^+ = B' \quad := \wedge_{p\in B}((p')^+ = p')$

$C^+ = C' \quad := \wedge_{c\in C}(c^+ = c')$

$(C')^+ = C' \quad := \wedge_{c\in C}((c')^+ = c')$

$(C')^+ = C^+ \quad := \wedge_{c\in C}((c')^+ = c^+)$

$D^+ = D' \quad := \wedge_{v\in D}(v^+ = v')$

$(D')^+ = D' \quad := \wedge_{v\in D}((v')^+ = v')$

$next1 \quad := \wedge_{c\in C}((boundary(c) \Rightarrow c^+ = c'+1) \wedge (\neg boundary(c) \Rightarrow c^+ = c'))) \wedge ((C')^+ = C^+)$

$next2 \quad := \wedge_{c\in C}((critical(c) \Rightarrow c^+ = c'+1) \wedge (\neg critical(c) \Rightarrow c^+ = c'))$

$discontin \quad := (\vee_{c\in C}(c' = 0)) \vee \vee_{p\in B} \neg(p = p')$

$boundary(c) \quad := (c' = 0 \vee c' = 2 \vee c' = 4 \vee c' = 2n)$

$boundary \quad := \vee_{c\in C} boundary(c)$

$critical(c) \quad := (c' \le 2n) \wedge \wedge_{d\in C}((d' \le 2n) \Rightarrow (c' \le (d' + \Delta'_{cd})))$

LTL 公式 *Next* 刻画了当[illegible]时,两个投影 $\overline{\sigma}$ 和 $\overline{\tau}$ 应满足的约束关系.若 J 是 V 上的一个 LTLC-解释, $J(t_0), J(t_1), J(t_2), \ldots$ 是 J 的一个执行序列,则此执行序列在 V_d 上的投影序列 $\overline{J(t_0)}, \overline{J(t_1)}, \overline{J(t_2)}, \ldots$ 将是 LTL 公式 $[]Next$ 的模型.

定义 **3.13**. 对于时间模块 M 和正整数 n ,定义 V_d 上的 LTL 公式 $\hat{M}$ 如下:

$\hat{M} := \overline{M} \wedge clock \wedge liveness \wedge Next$

$\overline{M} := \overline{init} \wedge [](\overline{\alpha_1} \vee \overline{\alpha_2} \vee \ldots \vee \overline{\alpha_m} \vee V' = V) \wedge [](\overline{\beta_1} \vee \overline{\beta_2} \vee \ldots \vee \overline{\beta_k})$

$clock := (\wedge_{c\in C}(c = 0 \wedge c' = 0)) \wedge (\wedge_{c,d\in C}(\Delta'_{cd} = 0 \wedge \Delta'_{cd} = 0)) \wedge ([] \wedge_{c\in C}(c = c' \vee c' = 0)) \wedge$
$[](\wedge_{c,d\in C}((c' = c \wedge d' = d \Rightarrow \Delta'_{cd} = \Delta_{cd}) \wedge (c' = 0 \Rightarrow \Delta'_{cd} = -d') \wedge (d' = 0 \Rightarrow \Delta'_{cd} = c')))$

$liveness := \wedge_{c\in C} []<> (c' = 0 \vee c > 2n)$

引理 **3.5**. 设 $\Im := \langle J(t_0), J(t_1), J(t_2), \ldots \rangle$ 是解释 J 的一个执行序列, $\overline{\Im}$ 是序列 $\Im$ 在 V_d 上的投影序列;则 $J|=M$ 当且仅当 $\overline{\Im}|=_{ltl} \hat{M}$.

证明:利用引理 3.4 并注意到公式 *Next* 的直观含义则不难得到此引理的结论. □

引理 **3.6**. 若 $\Pi := \langle \pi_0, \pi_1, \pi_2, \ldots \rangle$ 是 $\hat{M}$ 的一个模型,则存在 M 的模型 J 使得 Π 是 J 的某个执行序列在 V_d 上的投影序列.

本引理的证明较长,详细证明过程可见文献[11],此处从略.

定理 **3.1**. 对于时间模块 M 和正整数 n ,设 φ 是一个 LTLC 公式且 $\varphi \in Form(V, n)$,则 $M|=\varphi$ 当且仅当 $\hat{M}|=_{ltl}\overline{\varphi}$.

证明: (1) 假设 $M|=\varphi$.

设 $\Pi := \langle \pi_0, \pi_1, \pi_2, \ldots \rangle$ 是 $\hat{M}$ 的任意一个模型,由引理 3.6 可知,存在 M 的模型 J 使得 Π 是 J 的某个执行序列 $\Im$ 在 V_d 上的投影序列.于是 $J|=M$,由假设 $M|=\varphi$ 便得 $J|=\varphi$,应用引理 3.4 便可得 $\overline{\Im}|=_{ltl}\overline{\varphi}$,即 $\Pi|=_{ltl}\overline{\varphi}$,于是 $\hat{M}|=_{ltl}\overline{\varphi}$.

(2) 假设 $\hat{M}|=_{ltl}\overline{\varphi}$.

设 J 是 M 的一个模型,由引理 3.5 得 $\overline{\Im}|=_{ltl}\hat{M}$,其中 $\Im$ 是 J 的一个执行序列;应用假设条件 $\hat{M}|=_{ltl}\overline{\varphi}$ 便得到 $\overline{\Im}|=_{ltl}\overline{\varphi}$.这样由引理 3.4 便可推出 $J|=\varphi$,于是 $M|=\varphi$. □

由于 V_d 中的所有变量的取值域都是有限的(不超出域 $\{-2n-1,-2n,\ldots,-2,-1,0,1,2,\ldots,2n+1\}$),故 $\hat{M}\models_{ltl}\overline{\varphi}$ 的永真性在 LTL 中是可判定的,这样利用 $\hat{M}\models_{ltl}\overline{\varphi}$ 的永真性判定过程就可判定 $M\models\varphi$ 的永真性.

4 结 论

本文给出了一种判定时间模块是否满足某个 LTLC 公式的算法检查方法,由于 LTLC 具有与其他它实时逻辑不同的表达能力,它既能表示实时系统的性质(如安全性,活性等),又能表示实时系统本身,这就使得我们给出的LTLC的模型检查方法不但可以用于检查时间模块是否具有某些性质,而且还可用于检查两个时间模块之间的求精关系.而后者在别的实时模型检查方法中至今还很少考虑.本文的工作也进一步说明了 LTLC 能在统一框架下表示实时系统的模型和性质这一特征所具有的好处.由于实时系统的模型检查比较复杂,实现起来比较困难,而且效率较低.所以我们还未将本文的模型检查方法实现为一模型检查工具.若条件许可,且有较好的实现思路时我们将做这一工作.

References:

[1] Li, Guang-yuan, Tang, Zhi-song. A linear temporal logic with clocks for verification of real-time systems. Journal of Software, 2002,13(1):33~41 (in Chinese).

[2] Alur, R., Henzinger, T.A. Logics and models of real time: a survey. In: Rozenberg, G; Roever, W.P., Huizing, C., eds. Real Time: Theory in Practice. Lecture Notes in Computer Science 600, New York: Springer-Verlag, 1992. 74~106.

[3] Clarke, E.M, Grumberg, O, Peled, D. Model Checking. Cambridge, MA: MIT Press, 1999.

[4] Clarke, E.M, Emerson, E.A., Sistla, A.P. Automatic verification of finite-state concurrent systems using temporal-logic specifications. ACM Transactions on Programming Languages and Systems, 1986,8(2):244~263.

[5] Vardi, M.Y., Wolper, P. An automata-theoretic approach to automatic program verification. In: Proceedings of the 1st Symposium on Logic in Computer Science. Cambridge: IEEE Computer Society Press, 1986. 322~331. http://www.cs.rice.edu/~vardi/papers/index.html

[6] Courcoubetis, A.C., Dill, D.L. Model-Checking in dense real-time. Information and Computation, 1993,104(1):2~34.

[7] Alur, R., Dill, D.L. A theory of timed automata. Theoretical Computer Science, 1994,126(2):183~235.

[8] Lamport, L. The temporal logic of actions. ACM Transactions on Programming Languages and Systems, 1994,6(3):872~923.

[9] Manna, Z., Pnueli, A. The Temporal Logic of Reactive and Concurrent Systems: Specification. New York: Springer-Verlag, 1992.

[10] Sistla, A.P., Clarke, E.M. The complexity of propositional linear temporal logics. Journal of the Association for Computing Machinery, 1985,32(3):733~749.

[11] Li, Guang-yuan. LTLC: a continuous-time temporal logic for real-time and hybrid systems [Ph.D. Thesis]. Beijing: Institute of Software, the Chinese Academy of Sciences, 2001 (in Chinese).

[12] Kesten, Y., Manna, Z., McGuire, H., *et al*. A decision algorithm for full propositional temporal logic. In: Courcoubetis, C., ed. Proceedings of the 5th Conference on Computer Aided Verification. Lecture Notes in Computer Science 697, New York: Springer-Verlag, 1993. 97~109.

附中文参考文献:

[1] 李广元,唐稚松.带有时钟变量的线性时序逻辑与实时系统验证.软件学报,2002,13(1):33~41.

[11] 李广元.LTLC:面向实时与混成系统的连续时序逻辑[博士学位论文].北京:中国科学院软件研究所,2001.

A Model Checking of Real-Time Systems in Linear Temporal Logic with Clocks*

LI Guang-yuan, TANG Zhi-song

(*Key Laboratory of Computer Science*, *Institute of Software*, *The Chinese Academy of Sciences*, *Beijing* 100080, *China*)

E-mail: {ligy,cst}@ios.ac.cn

http://www.ios.ac.cn

Abstract: Model checking is an algorithmic technique for checking if a concurrent system satisfies a given property expressed in an appropriate temporal logic. LTLC(linear temporal logic with clocks) is a continuous-time temporal logic proposed for the specification of real-time systems. It is a real-time extension of the temporal logic LTL. In this paper, the model checking problem for LTLC is discrssed and a reduction from LTLC model checking to LTL model checking is presented. This reduction will enable us to use the existing LTL model checking tools for LTLC model checking. Owing to the fact that LTLC can express both the properties and the implementations of real-time systems, the LTLC model checking procedures can be used for both the property verification and the refinement verification for real-time systems with finite locations.

Key words: real-time system; timed automaton; linear temporal logic; model checking; property verification

* Received July 20, 2000; accepted July 30, 2001

Supported by the National Natural Science Foundation of China under Grant No.60073020; the Key Sci-Tech Project of the National 'Ninth Five-Year-Plan' of China under Grant No.98-780-01-07-01; the National High Technology Development 863 Program of China under Grant No.863-306-ZT02-04-1

敬告作者

《软件学报》创刊以来,蒙国内外学术界厚爱,收到许多高质量的稿件,其中不少在发表后读者反映良好,认为本刊保持了较高的学术水平.但也有一些稿件因不符合本刊的要求而未能通过审稿.为了帮助广大作者尽快地把他们的优秀研究成果发表在我刊上,特此列举一些审稿过程中经常遇到的问题,请作者投稿时尽量予以避免,以利大作的发表.

1. 读书偶有所得,即匆忙成文,未曾注意该领域或该研究课题国内外近年来的发展情况,不引用和不比较最近文献中的同类结果,有的甚至完全不列参考文献.

2. 做了一个软件系统,详尽描述该系统的各个方面,如像工作报告,但采用的基本上是成熟技术,未与国内外同类系统比较,没有指出该系统在技术上哪几点比别人先进,为什么先进.一般来说,技术上没有创新的软件系统是没有发表价值的.

3. 提出一个新的算法,认为该算法优越,但既未从数学上证明比现有的其他算法好(例如降低复杂性),也没有用实验数据来进行对比,难以令人信服.

4. 提出一个大型软件系统的总体设想,但很粗糙,而且还没有(哪怕是部分的)实现,很难证明该设想是现实的、可行的、先进的.

5. 介绍一个现有的软件开发方法,或一个现有软件产品的结构(非作者本人开发,往往是引进的,或公司产品),甚至某一软件的使用方法.本刊不登载高级科普文章,不支持在论文中引进广告色彩.

6. 提出对软件开发或软件产业的某种观点,泛泛而论,技术含量少.本刊目前暂不开办软件论坛,只发表学术文章,但也欢迎材料丰富,反映现代软件理论或技术发展,并含有作者精辟见解的某一领域的综述文章.

7. 介绍作者做的把软件技术应用于某个领域的工作,但其中软件技术含量太少,甚至微不足道,大部分内容是其他专业领域的技术细节,这类文章宜改投其他专业刊物.

8. 其主要内容已经在其他正式学术刊物上或在正式出版物中发表过的文章,一稿多投的文章,经退稿后未作本质修改换名重投的文章.

本刊热情欢迎国内外科技界对《软件学报》踊跃投稿.为了和大家一起办好本刊,特提出以上各点敬告作者.并且欢迎广大作者和读者对本刊的各个方面,尤其是对论文的质量多多提出批评建议.