带测试动作的动态时序逻辑扩展*

孙永新[1,2]，赵希顺[2]

(1. 仲恺农业工程学院 计算机科学与工程学院，广州 510225；2. 中山大学 逻辑与认知研究所，广州 510275)

摘 要：作为一种动态知识表示形式，动态时序逻辑(DLTL)尤适用于正规程序验证，然而它不直接支持测试动作，这使得其应用受到一定限制。为支持测试动作，提出一个 DLTL 扩展 $DLTL^+$ 和一个判定 $DLTL^+$ 公式可满足性的 tableau 算法，并给出了算法的正确性以及其时间复杂度为 $2^{O(n)}$ 的证明。分析表明，$DLTL^+$ 提供了一种直接的、有效的测试动作支持方式，该方式比已知的其他方式更具有实际应用价值。

关键词：测试动作；动态时序逻辑；扩展；tableau 算法；计算复杂性

中图分类号：TP301 **文献标志码**：A **文章编号**：1001-3695(2012)09-3269-05

doi:10.3969/j.issn.1001-3695.2012.09.018

Extending dynamic linear time temporal logic to support test actions

SUN Yong-xin[1,2], Zhao Xi-shun[2]

(1. *College of Computer Science & Engineering, Zhongkai University of Agriculture & Engineering, Guangzhou* 510225, *China*; 2. *Institute of Logic & Cognition, Sun Yat-sen University, Guangzhou* 510275, *China*)

Abstract: As a dynamic knowledge representation formalism, dynamic linear time temporal logic (DLTL) is especially suitable for being applied in verifying regular programs. But for the reason of no direct supporting for test actions, the application of DLTL is quite restricted. To support test actions, this paper proposed an extension of DLTL, $DLTL^+$, presented a tableau algorithm for determining $DLTL^+$ formulas' satisfiability, and showed that the algorithm was correct and its time complexity was $2^{O(n)}$. Analysis results demonstrate that $DLTL^+$ provides a direct and efficient way for supporting test actions, which has more practical application value than other known ways.

Key words: test action; dynamic linear temporal logic (DLTL); extension; tableau algorithm; computational complexity

0 引言

在人工智能领域，动态知识的表示和推理一直受到极大的关注，研究人员为此也提出了不少基于逻辑的动态知识表示形式。其中，动态时序逻辑(DLTL)[1]整合了动态逻辑(dynamic logic)[2]和时序逻辑(linear temporal logics)[3]的表达特性，可以显示地表达动作组合形式和动态系统的时间特性，适用于正规程序的自动验证[4,5]。笔者也曾提出一类 DLTL 和描述逻辑的组合形式 $DLTL_{DL}$，用于基于本体的动态系统中以验证组合动作(正规程序)的特性[6]。然而，由于 DLTL 和 $DLTL_{DL}$ 直接支持的动作形式不包含测试动作，这在一定程度上影响了 DLTL 和 $DLTL_{DL}$ 的实际应用。

Giordano 等人[4]的基于 DLTL 的动作理论中提出一种支持测试动作的方法，不过笔者认为该方法可能会因引入太多的公理而导致其失去实际应用价值。例如，设某问题可以用含测试动作的 $DLTL^+$ 公式 φ 描述($DLTL^+$ 参见下文 1.1 节)，则按 Giordano 等人的方法，该问题需用按以下方式构造的 DLTL 公式 φ' 描述：

a) 为 φ 中出现的每个测试动作？φ 引入一族公理：

(a) $\Box(\neg\varphi\rightarrow[?\ \varphi]\neg T)$；

(b) 对所有的流文字 L，$\Box(\langle?\ \varphi\rangle T\rightarrow(L\leftrightarrow[?\ \varphi]L))$。

b) 令 φ' 为 φ 与以上所有公理的合取(φ' 中的每个？φ 被当做一个 DLTL 的原子动作)。可以简单分析 Giordano 方法的计算复杂度：令 φ 的公式长度为 n，在最差情况下，φ 中出现命题符号个数和测试动作个数均与 n 呈线性关系(如 $\varphi=[?\ p_1;?\ p_2;\cdots;?\ p_m]T$，所有 p_i 均为命题符号，流文字个数等于 $2m$，测试动作个数等于 m，而 m 约为 φ 长度的 1/3)，此时 φ' 的公式长度约为 Kn^2(K 为常量)。由 DLTL 公式可满足性问题的时间复杂性为 $2^{O(|\varphi'|)}$ 可推知[1]，判定 φ' 的可满足性需要耗费的时间属于 $2^{O(n^2)}$。

为了更有效地解决测试动作支持问题，本文提出了扩展的动态时序逻辑 $DLTL^+$。

1 基本定义

定义 1 命题公式。给定一个原子命题集合 $Pr=\{p_1,p_2,\cdots,p_n\}$，$Pr$ 上的命题公式归纳定义如下：

a) $p_i\in Pr$ 和 T 都是命题公式；

b) 如果 α 是命题公式，则 $\neg\alpha$ 是命题公式；

c) 如果 α 和 β 是命题公式，则 $\alpha\wedge\beta$ 是命题公式。

$DLTL^+$ 用允许测试动作出现的正规程序[2]来表达组合动作。测试动作的直观含义是：运行一个含？φ 的正规程序，执行？φ 动作实际上就是检测 φ 是否成立。当 φ 成立时，继续执行下边的动作；否则？φ 所在的程序分支将挂起，且执行？φ 前后机器状态不变。利用测试动作，一般程序设计语言中的 if

收稿日期：2012-03-15；**修回日期**：2012-04-26 **基金项目**：国家自然科学基金资助项目(60970040)

作者简介：孙永新(1972-)，男，江西吉安人，工程师，博士研究生，主要研究方向为描述逻辑、语义 Web、软件工程、人工智能逻辑(syx2000@tom.com)；赵希顺(1964-)，男，教授，博导，主要研究方向为可计算性与计算复杂性理论、数理逻辑、人工智能逻辑.

条件分支语句“if φ then a_1 else a_2”可以表示为正规程序“$?\ \varphi; a_1 + ?\ \neg\varphi; a_2$”,while 循环语句“while (φ) do a;”可以表示为正规程序“$(?\ \varphi; a)^* ; ?\ \neg\varphi$”。

定义 2 正规程序和正规集。$\mathrm{Act} = \{a_1, \cdots, a_m\}$ 表示原子动作集合,$\Gamma = \{?\ \varphi \mid \varphi$ 是一个测试$\}$(文中限定 φ 为命题公式),表示测试动作集合,$\Sigma = \mathrm{Act} \cup \Gamma$。$\Sigma$ 上的正规程序 π 及其表示的正规集合 $RS(\pi)$ 定义如下:

a)ε 表示空串,$a \in (\Sigma \cup \{\varepsilon\})$ 为原子正规程序,它表示的正规集合 $RS(a) = \{a\}$;

b)如果 π_0、π_1 是正规程序,则 π_0 与 π_1 的选择 $\pi_0 + \pi_1$ 也是正规程序,它表示的正规集合 $RS(\pi_0 + \pi_1) = RS(\pi_0) \cup RS(\pi_1)$;

c)如果 π_0、π_1 是正规程序,则 π_0 与 π_1 的连接 $\pi_0;\pi_1$ 也是正规程序,它表示的正规集合 $RS(\pi_0;\pi_1) = \{\tau_0\tau_1 \mid \tau_0 \in RS(\pi_0),\ \tau_1 \in RS(\pi_1)\}$;

d)如果 π 是正规程序,则 π 的闭包 π^* 也是正规程序,它表示的正规集合 $RS(\pi^*) = \cup_{i\in\omega} RS(\pi^i)$,其中,$\pi^i$ 表示 π 的 i 次连接,$\pi^0 \equiv \varepsilon$ 且对任意 $i \geqslant 1$,$\pi^i \equiv \pi^{i-1};\pi$。

本文用 $\mathrm{prg}(\Sigma)$ 表示 Σ 上的所有正规程序集合,用 Σ^ω 表示 Σ 上所有无穷长的动作序列集合,用 $|\Sigma|$ 表示动作序列 Σ 的字符串长度,用 $<$ 表示字符串的严格前缀关系,用 $\mathrm{prf}(\tau)$ 表示字符串 τ 的所有前缀的集合。

定义 3 DLTL^+ 语法。$\mathrm{DLTL}^+(\Sigma)$ 符号包含原子命题集合 $Pr = \{p_1, p_2, \cdots, p_n\}$。$\mathrm{DLTL}^+(\Sigma)$ 公式归纳定义如下:

a)$p_i \in Pr$ 和 T 都是 $\mathrm{DLTL}^+(\Sigma)$ 公式;

b)如果 α 是 $\mathrm{DLTL}^+(\Sigma)$ 公式,则 $\neg\alpha$ 是 $\mathrm{DLTL}^+(\Sigma)$ 公式;

c)如果 α 和 β 是 $\mathrm{DLTL}^+(\Sigma)$ 公式,则 $\alpha \wedge \beta$ 是 $\mathrm{DLTL}^+(\Sigma)$ 公式;

d)如果 $\pi \in \mathrm{prg}(\Sigma)$,$\alpha$ 和 β 是 $\mathrm{DLTL}^+(\Sigma)$ 公式,则 $\alpha U^\pi \beta$ 是 $\mathrm{DLTL}^+(\Sigma)$ 公式,文中将该类公式称为预测。

显然,Pr 上的所有命题公式都是 $\mathrm{DLTL}^+(\Sigma)$ 公式。

定义 4 $\mathrm{DLTL}^+(\Sigma)$ 语义。$\mathrm{DLTL}^+(\Sigma)$ 模型是一个三元组 $M = \langle\sigma, <, I\rangle$,其中 $\sigma \in \Sigma^\omega$;$<$ 是 $\mathrm{prf}(\sigma)$ 上的严格前缀关系,文中把 $\mathrm{prf}(\sigma)$ 中的每个元素称为一个点(状态);I 是一个解释函数,将每个 $p_i \in Pr$ 解释为 $\mathrm{prf}(\sigma)$ 的一个子集,即 $I(p_i) \subseteq \mathrm{prf}(\sigma)$,而 T 则被解释为 $\mathrm{prf}(\sigma)$,即 $I(T) = \mathrm{prf}(\sigma)$。扩展以上函数 I,将 Pr 上的任意命题公式 γ 解释为 $\mathrm{prf}(\sigma)$ 的一个子集:当 γ 形如 $\neg\alpha$ 时,$\mathrm{prf}(\sigma)$ 中点 $\tau \in I(\neg\alpha)$ 当且仅当 $\tau \notin I(\alpha)$;当 γ 形如 $\alpha \wedge \beta$ 时,$\mathrm{prf}(\sigma)$ 中点 $\tau \in I(\alpha \wedge \beta)$ 当且仅当 $\tau \in I(\alpha)$ 且 $\tau \in I(\beta)$。而且,I 符合测试动作条件:对任意 $\tau ?\ \varphi \in \mathrm{prf}(\sigma)$,$\tau \in I(\varphi)$ 且对 Pr 上的任意命题公式 γ,$\tau \in I(\gamma)$ 当且仅当 $\tau ?\ \varphi \in I(\gamma)$。

$\mathrm{DLTL}^+(\Sigma)$ 模型 $M = \langle\sigma, <, I\rangle$,$\tau \in \mathrm{prf}(\sigma)$ 和 $\mathrm{DLTL}^+(\Sigma)$ 公式 α 的满足关系 $\vDash$ 归纳定义如下:

a)对任意 $p_i \in Pr$,$M,\tau \vDash p_i$ 当且仅当在 $\tau \in I(p_i)$;

b)$M,\tau \vDash T$;

c)$M,\tau \vDash \neg\alpha$ 当且仅当 非 $M,\tau \vDash \alpha$;

d)$M,\tau \vDash \alpha \wedge \beta$ 当且仅当 $M,\tau \vDash \alpha, M,\tau \vDash \beta$;

e)$M,\tau \vDash \alpha U^\pi \beta$ 当且仅当 $\exists \tau_1 \in RS(\pi)(\tau\tau_1 \in \mathrm{prf}(\sigma),\ M, \tau\tau_1 \vDash \beta, \forall \tau_2(\varepsilon \leqslant \tau_2 < \tau_1 \Rightarrow M,\ \tau\tau_2 \vDash \alpha))$。

定义 5 $\mathrm{DLTL}^+(\Sigma)$ 公式可满足性。如果存在模型 $M = \langle\sigma, <, I\rangle$ 和 τ 使得 $M,\tau \vDash \alpha$,则称公式 α 在基于 σ 的模型 M 中可满足。

为了便于表述且不失一般性,后文假设形如 $\neg\neg\alpha$ 的公式均自动转换为 α。$\mathrm{DLTL}^+(\Sigma)$ 公式可以表达动态和时序约束:给定动作集合 $\Sigma = \{a_1, \cdots, a_m\} \cup \{?\ \varphi_0, \cdots, ?\ \varphi_n\}$,动态算子 $\langle\pi\rangle$,$[\pi]$ 和时态算子 U(until),$\bigcirc$(next),$\Diamond$(sometimes),$\Box$(always)可以分别定义为:$\langle\pi\rangle\alpha \equiv TU^\pi\alpha$;$[\pi]\alpha \equiv \neg\langle\pi\rangle\neg\alpha$;$\alpha U\beta \equiv \alpha U^{\Pi^*}\beta$,其中 $\Pi = (a_0 + \cdots + a_m + ?\ \varphi_0 + \cdots + ?\ \varphi_n)$;$\bigcirc\alpha \equiv \vee_{a\in\Sigma}\langle a\rangle\alpha$;$\Diamond\alpha \equiv TU\alpha$;$\Box\alpha \equiv \neg\Diamond\neg\alpha$。

2 Tableau 算法

输入 $\mathrm{DLTL}^+(\Sigma)$ 公式 φ,tableau 算法根据 tableau 规则,构造出一个称为 tableau 结构类型的对象。2.1 节描述了与 tableau 相关的主要数据结构,2.2 节描述了 tableau 规则,2.4 节描述了 tableau 算法执行过程,saturate($\underline{S}$)是算法中用到的一个重要函数,笔者将在 2.3 节单独描述它。

2.1 约束系统和 tableau

用 Φ 表示所有 $\mathrm{DLTL}^+(\Sigma)$ 公式的集合。一个约束系统是 Φ 的一个有限子集,文中一般用 S 表示;一个标记约束系统是 $\Phi \times \{0,1\}$ 的一个有限子集,文中一般用 $\underline{S}$ 表示($\underline{S}$ 中元素 (α,x) 可直观地看做是带标记的 α,如 $(\alpha,1)$ 可直观地看成 α,即在 α 上加一个下画线标记,而 $(\alpha,0)$ 可直观地看做是不带下划线的 α,故称 $\underline{S}$ 为标记约束系统)。

称约束系统 S 有冲突当且仅当以下条件之一成立:a)$\neg T \in S$;b)$\{\alpha, \neg\alpha\} \subseteq S$;c)$\{\alpha_1 U^a \beta_1, \alpha_2 U^b \beta_2\} \subseteq S$ 且 $a, b \in \Sigma$,$a \neq b$。

给定标记约束系统 $\underline{S}$,文中用 $CS(\underline{S})$ 表示集合 $\{\alpha \mid (\alpha,x) \in S,\ x \in \{0,1\}\}$($CS(\underline{S})$ 可直观理解为一个将 $\underline{S}$ 中元素的标记抹去后得到的约束系统)。

Tableau 为二元组 $G = (G,L)$,$G = (N,E)$ 为有向图,L 为 G 上的标签函数:节点 n_1 到 n_2 的有向边 $(n_1,n_2) \in E$ 的标签为 $L((n_1,n_2)) \in \Sigma$;节点 $n \in N$ 的标签为二元组 $L(n) = (L(n).\underline{S}, L(n).f)$,其中 $L(n).\underline{S}$ 为一个标记约束系统,$L(n).f \in \{\surd, \downarrow\}$。用 $L(n).S$ 表示 $CS(L(n).S)$,称其为节点 n 的约束系统,用 $L(n).U$ 表示 $\{(\alpha,1) \mid (\alpha,1) \in L(n).\underline{S}\}$,用 $L(n).F$ 表示 $\{(\alpha,0) \mid (\alpha,0) \in L(n).\underline{S}\}$。

2.2 Tableau 规则

Tableau 规则分为两大类:一类是表 1 中的规则,应用其中规则后的结果只影响 tableau 的单个节点,因此把这类规则称为局部规则;另一类是表 2 中的规则,应用其中规则时,总是要创建新的 tableau 节点和有向边,对 tableau 的整体结构产生影响,因此把这类规则称为全局规则。

需要说明的是,在表 1 局部规则描述中,$\underline{S}$ 为节点的标记约束系统,$S = CS(\underline{S})$。另外,文中算法保证 $\Box\bigcirc T \in S$,故应用局部规则 $\rightarrow_{\neg a}$ 前,必存在 $\theta U^b\gamma \in S$。

2.3 Saturate($\underline{S}$)

给定一个标记约束系统 $\underline{S}$,如果没有任何局部规则可以应用到 $\underline{S}$ 中的任意元素,则称 $\underline{S}$ 为饱和标记约束系统。

```
define procedure saturate(S)
{  if (非→¬a局部规则 r 可应用到S中的元素) {
```

for($\underline{S_i}$ = 应用 r 后的某个可能$\underline{S}$)

$\underline{Ss_i}$ = saturate($\underline{S_i}$);

return $\underline{Ss}$ = $\cup_i$($\underline{Ss_i}$); }

if ($\rightarrow_{\neg a}$规则 r 可应用到 S 中的元素) {

for($\underline{S_i}$ = 应用 r 后的某个可能$\underline{S}$)

$\underline{Ss_i}$ = saturate($\underline{S_i}$);

return $\underline{Ss}$ = $\cup_i$($\underline{Ss_i}$);}

$\underline{S}$ = $\underline{S}$\{(α,0) | (α,0),(α,1) ∈ $\underline{S}$};

return $\underline{Ss}$ = {$\underline{S}$};}

表 1　DLTL$^+$(Σ)的局部规则

规则	条件	动作
$\rightarrow_\wedge$	$\alpha\wedge\beta\in S,\alpha\notin S$ 或 $\beta\notin S$	$\underline{S}=\underline{S}\cup\{(\alpha,0),(\beta,0)\}$
$\rightarrow_{\neg\wedge}$	$\neg(\alpha\wedge\beta)\in S$, $\{\neg\alpha,\neg\beta\}\cap S=\varnothing$	$\underline{S}=\underline{S}\cup\{(\neg\alpha,0)\}$ 或 $\underline{S}=\underline{S}\cup\{(\neg\beta,0)\}$
$\rightarrow_a$	$\alpha U^a\beta\in S, a\in Act, \nexists b\in\Sigma(a\neq b,\ \theta U^b\gamma\in S)$ 且 $\alpha\notin S$	$\underline{S}=\underline{S}\cup\{(\alpha,0)\}$
$\rightarrow_?$	$\alpha U^?\varphi\beta\in S, \nexists b\in\Sigma(b\neq?\ \varphi,\ \theta U^b\gamma\in S)$ 且 $\{\alpha,\varphi\}\not\subseteq S$	$\underline{S}=\underline{S}\cup\{(\alpha,0),(\varphi,0)\}$
$\rightarrow_{\neg a}$	$\neg(\alpha U^a\beta)\in S,\ a\in\Sigma,$ $\nexists b\in\Sigma(a\neq b,\ \theta U^b\gamma\in S)$ 且 $\{\neg\alpha,\ \alpha U^a\neg\beta\}\cap S=\varnothing$	$\underline{S}=\underline{S}\cup\{(\neg\alpha,0)\}$ 或 $\underline{S}=\underline{S}\cup\{(\alpha U^a\neg\beta,0)\}$
$\rightarrow_;$	$(\alpha U^{\pi1;\pi2}\beta,x)\in\underline{S}$ 且 $(\alpha U^{\pi1}\alpha U^{\pi2}\beta,x)\notin\underline{S}$	$\underline{S}=\underline{S}\cup\{(\alpha U^{\pi1}\alpha U^{\pi2}\beta,x)\}$
$\rightarrow_{\neg;}$	$\neg(\alpha U^{\pi1;\pi2}\beta)\in S$ 且 $\neg(\alpha U^{\pi1}\alpha U^{\pi2}\beta)\notin S$	$\underline{S}=\underline{S}\cup\{(\neg(\alpha U^{\pi1}\alpha U^{\pi2}\beta),0)\}$
$\rightarrow_+$	$(\alpha U^{\pi1+\pi2}\beta,x)\in\underline{S}$ 且 $\{(\alpha U^{\pi1}\beta,x),(\alpha U^{\pi2}\beta,x)\}\cap\underline{S}=\varnothing$	$\underline{S}=\underline{S}\cup\{(\alpha U^{\pi1}\beta,x)\}$ 或 $\underline{S}=\underline{S}\cup\{(\alpha U^{\pi2}\beta,x)\}$
$\rightarrow_{\neg+}$	$\neg(\alpha U^{\pi1+\pi2}\beta)\in S$ 且 $\neg(\alpha U^{\pi1}\beta),\neg(\alpha U^{\pi2}\beta)\}\not\subseteq S$	$\underline{S}=\underline{S}\cup\{(\neg(\alpha U^{\pi1}\beta),0),$ $(\neg(\alpha U^{\pi2}\beta),0)\}$
$\rightarrow_*$	$(\alpha U^{\pi*}\beta,x)\in\underline{S}$ 且 $(\alpha U^\pi\alpha U^{\pi*}\beta,x)\notin\underline{S},\ \beta\notin S$	$\underline{S}=\underline{S}\cup\{(\beta,0)\}$ 或 $S=\underline{S}\cup\{(\alpha U^\pi\alpha U^{\pi*}\beta,x)\}$
$\rightarrow_{\neg*}$	$\neg(\alpha U^{\pi*}\beta)\in S$ 且 $\{\neg(\alpha U^\pi\alpha U^{\pi*}\beta),\neg\beta\}\not\subseteq S$	$\underline{S}=\underline{S}\cup\{(\neg(\alpha U^\pi\alpha U^{\pi*}\beta),0),$ $(\neg\beta,0)\}$

表 2　DLTL$^+$(Σ)的全局规则

规则	条件	动作
$\rightarrow_{ua}$	存在 $a\in\Sigma$, $\theta U^a\gamma\in cn.\ \underline{S}$	构造新节点 n。令 $L(n).\ \underline{U}=\{(\alpha U^\pi\beta,1)\mid(\alpha U^a\alpha U^\pi\beta,1)\in cn.\ \underline{U}\}$,如 $L(n).\ \underline{U}=\varnothing$,则令 $L(n).f=\surd$,否则 $L(n).f=\downarrow$;如 $a\in Act$,则令 $L(n).\ F=\{(\beta,0)\mid\alpha U^a\beta\in cn.\ S$ 且$(\beta,1)\notin L(n).\ \underline{U}\}$,否则,令 $L(n).F=\{(\neg p,0)\mid p\in Pr,\neg p\in cn.\ S\}\cup\{(p,0)\mid p\in Pr,p\in cn.\ S\}\cup\{(\beta,0)\mid\alpha U^a\beta\in cn.\ S$ 且$(\beta,1)\notin L(n).\ \underline{U}\}$。最后令 $pn=cn$, $act=a$ (注:pn、cn、act 的说明见 2.4 节)

在 saturate($\underline{S}$)中,tableau 算法反复按优先应用非$\rightarrow_{\neg a}$规则的策略(即保证在应用$\rightarrow_{\neg a}$规则时没有其他规则可应用)对标记约束系统中的元素应用局部规则,逐步扩张标记约束系统,直至标记约束系统饱和。由于有些局部规则是析取规则,如$\rightarrow_{\neg\wedge}$等,有多个可能输出,因此最终由一个标记约束系统经扩张可得到的饱和标记约束系统可能有多个。下文用$\underline{Ss}$表示 saturate($\underline{S}$)返回的饱和标记约束系统集合。

值得说明的是,$\underline{Ss}$中元素并非严格意义的饱和标记约束系统,而是饱和约束系统经去冗余后的产物,即如果存在(α,0)和(α,1)均属于$\underline{S}$的某个饱和标记约束系统$\underline{S_i}$,则从$\underline{S_i}$中去除(α,0)。这样处理的目的是为了优化 tableau 算法,减少 tableau 节点数。无论是否去冗余,都不影响算法正确性。

由局部 tableau 规则和 saturate($\underline{S}$)可看出,以下命题 1 ~ 3 均成立。

命题 1　对任意$\underline{S_i}\in\underline{Ss}$,如果$(\alpha U^\pi\beta,1)\in\underline{S_i}$,则当 $\varepsilon\in RS(\pi)$时,以下 a)或 b)成立,否则 b)或 c)成立:a)$\beta\in CS(\underline{S_i})$;b)$\alpha\in CS(\underline{S_i})$,且存在$(\alpha U^a\alpha U^{\pi1}\cdots\alpha U^{\pi m}\beta,1)\in\underline{S_i}$满足 $m\geqslant1$, $RS(a;\pi_1;\cdots;\pi_m)\subseteq RS(\pi)$, $\pi_i\neq\varepsilon(0<i\leqslant m)$;c)$\alpha\in CS(\underline{S_i})$且存在$(\alpha U^a\beta,1)\in\underline{S_i}$满足 $a\in RS(\pi)$。

命题 2　对任意$\underline{S_i}\in\underline{Ss}$,如果 $\alpha U^\pi\beta\in CS(\underline{S_i})$,则当 $\varepsilon\in RS(\pi)$时,以下 a)或 b)成立,否则 b)或 c)成立:a)$\beta\in CS(\underline{S_i})$;b)$\alpha\in CS(\underline{S_i})$,且存在 $\alpha U^a\alpha U^{\pi1}\cdots\alpha U^{\pi m}\beta\in CS(\underline{S_i})$ 满足 $m\geqslant1$, $RS(a;\pi_1;\cdots;\pi_m)\subseteq RS(\pi)$, $\pi_i\neq\varepsilon(0<i\leqslant m)$;c)$\alpha\in CS(\underline{S_i})$且存在 $\alpha U^a\beta\in CS(\underline{S_i})$ 满足 $a\in RS(\pi)$。

命题 3　如果$\square\bigcirc T\in CS(\underline{S})$,则 $M,\tau\vDash\wedge_{\alpha\in CS(\underline{S})}\alpha$ 当且仅当 $M,\tau\vDash\vee_{\underline{S_i}\in\underline{Ss}}\wedge_{\alpha\in CS(\underline{S_i})}\alpha$。

2.4　Tableau 过程

类似文献[7]中的 tableau 算法,算法执行分两个阶段:第一阶段的主要任务是利用 tableau 规则为输入公式 φ 构造一个完全 tableau $G^c=(G^c,L^c)$;第二个阶段的主要任务是删除 G^c 中那些被阻塞、约束系统有冲突或含不可实现预测的节点(关于阻塞节点和预测可实现的说明见本节下文),用 $G^\varphi=(G^\varphi,L^\varphi)$表示最终得到的 tableau。

2.4.1　第一阶段

Tableau 算法执行的第一阶段记做 con(φ)。用 *cn* 表示当前节点,用 *pn* 表示当前前驱节点,用 *act* 表示 *pn* 出边的标签,并为每个节点 *n* 增加一个辅助工作标志 *n*. Ext,表示是否已扩展 *n*(节点扩展过程见本节下文),*n*. Ext 的值默认为 0。con(φ)说明如下:

a)执行初始化。构造初始节点 n_0,令 $L(n_0).S=\{(\varphi,0),(\square\bigcirc T,0)\}$, $L(n_0).f=\surd$, $cn=n_0$, pn = NULL, act = NULL。

b)执行步骤(a) ~ (c)。

(a)本步骤称为节点 *cn* 的渗透过程。令 $\underline{Ss}$ = saturate($L(cn).\underline{S}$),对每个$\underline{S_i}\in\underline{Ss}$,构造节点 n_i,将 n_i 加入 G^c,$L^c(n_i).f=L(cn).f$,如果 $L(cn).f=\surd$,则令 $L^c(n_i).\underline{U}=\{(\alpha,1)\mid\alpha\in CS(S_i)$为预测$\}$, $L^c(n_i).F=\{(\alpha,0)\mid\alpha\in CS(S_i)$非预测$\}$,否则 $L^c(n_i).\underline{S}=\underline{S_i}$,把 n_i 称为 *cn* 的一个饱和节点;如果 $pn\neq$ NULL,将(pn,n_i)加入 G^c,令 $L^c((pn,n_i))=act$。以上过程可得到一个 *cn* 的饱和节点的集合,从中选择一个节点 n_i,置 n_i. Ext = 1,令 $cn=n_i$,转(b)。

(b)本步骤称为节点 *cn* 的扩展过程。如果存在节点 $n\in G^c$ 满足 $L(n).\underline{S}=L(cn).\underline{S}$,$L(n).f=L(cn).f$,则称 *cn* 被 n_1 阻塞,将(pn,n)加入 G^c,令 $L^c((pn,n))=act$,转(c);否则,如果 $L(cn).S$ 中有冲突,转(c);否则,对 $L^c(cn).\underline{S}$应用全局规则生成新节点 *n*,令 $cn=n$,转(a)。

(c)如果存在 $n\in G^c$ 满足 *n*. Ext = 0,则令 $cn=n$,转(b);否则,第一阶段结束。

对 tableau $G=(G,L)$, $G=(N,E)$, $n_1\in N$ 和 $\alpha U^\pi\beta\in L(n_1).S$,如果存在一条满足以下两个条件的路径 $P:n_1,\cdots,n_m,\cdots(m\geqslant1)$,则 $L(n_1).S$ 中的预测 $\alpha U^\pi\beta$ 可在路径 *P* 上实现。a)从 n_1 到 n_m 有向边上的动作序列 $\sigma\in RS(\pi)$;b)$\beta\in L(n_m).S$且对任意 $i<m$,$\alpha\in L(n_i).S$。

由命题 1、2,以及全局规则和 con(φ)可以看出,以下命题 4 成立。

命题 4　n_2 为 G^c 中节点 n_1 的后续节点:a) $CS(L^c(n_1).\underline{U})$中所有的预测均可在 $P:n_1,n_2$ 上实现当且仅当 $L^c(n_2).f=$

$\surd$;b)当 $L^c(n_2).f=\surd$ 时,如果 $CS(L^c(n_2).\underline{U})$ 中所有预测均可在路径 P_1 上实现,则 $CS(L^c(n_1).F)$ 中所有预测均可在路径 $P_2:n_1,P_1$ 上实现;c)当 $L^c(n_2).f=\downarrow$ 时,那么(a) $CS(L^c(n_2).\underline{U})$ 中的所有预测均可在路径 $P_1:n_2,\cdots$ 上实现当且仅当 $CS(L^c(n_1).\underline{U})$ 中的所有预测均可在路径 $P_2:n_1,P_1$ 上实现,(b) $L^c(n_2).S$ 中的所有预测均可在路径 $P_1:n_2,\cdots$ 上实现当且仅当 $L^c(n_1).S$ 中的所有预测均可在路径 $P_2:n_1,P_1$ 上实现。

下文用 $n\rightsquigarrow_P n_1$ 表示路径 P 从节点 n 出发,途径节点 n_1。由节点的浸透过程可看出,对 G^c 中任意节点 n,$CS(L^c(n).\underline{U})$ 非空,结合命题4中a)和c)的(a)可以看出,以下命题5成立。

命题5 对 G^c 中节点 n,$CS(L^c(n).\underline{U})$ 中的所有预测均可在路径 P 上实现当且仅当 $\exists n_1(n\rightsquigarrow_P n_1,L(n_1).f=\surd)$。

由命题4b)和命题5可推得:对 G^c 中节点 n,如 $\exists n_1,n_2$(n_1 为 n 后续节点,$n_1\rightsquigarrow_P n_2$,$L(n_1).f=\surd$,$L(n_2).f=\surd$),则 $L^c(n).F$ 中的所有预测均可在路径 $P_1:n,P$ 上实现;结合命题4中a)可进而推得:$L^c(n).S$ 中的所有预测均可在路径 P_1 上实现;结合命题4中的c)(b)又可以看出,以下命题6成立。

命题6 对 G^c 中节点 n,如果存在 n_1,n_2($n\rightsquigarrow_{P1} n_1$,$n_1\rightsquigarrow_{P2} n_2$,$L(n_1).f=\surd$,$L(n_2).f=\surd$),则 $L^c(n).S$ 中的所有预测均可在一条由 n 出发,途径 n_1、n_2 的路径上实现。

2.4.2 第二阶段

Tableau 算法第二个阶段说明如下:

迭代遍历 G^c,对 G^c 中节点 n,如果不存在(n_1,P)($n\rightsquigarrow_P n_1$,$L(n_1).f=\surd$),则删除 n 及其相关有向边,直至 G^c 没有变化为止,最终得到 tableau $G^\varphi=(G^\varphi,L^\varphi)$。

如果 G^φ 非空,则判定 φ 可满足;否则判定 φ 不可满足。

2.5 Tableau 算法分析

2.5.1 算法正确性

命题7 $M,\tau\vDash\varphi$ 当且仅当 $M,\tau\vDash\varphi\wedge\Box\bigcirc T$。

命题8 如果存在模型 M 和 τ 满足 $M,\tau\vDash\varphi$,则必存在模型 M'满足 $M',\varepsilon\vDash\varphi$,反之亦然。

引理1 如果 G^φ 非空,则存在模型 M 满足 $M,\varepsilon\vDash\varphi$。

证明 通过从 G^φ 构造一个使得 φ 在 ε 点可满足的模型 M 证明本引理。

如果 G^φ 非空,由 tableau 算法第二阶段执行过程可以看出,对 G^φ 中任意节点 n,$\exists(n_1,P)(n\rightsquigarrow_P n_1,L(n_1).f=\surd)$,因此可以从 G^φ 中选择满足$\{\varphi,\Box\bigcirc T\}\subseteq L(n_0).S$ 的节点 n_0,由 n_0 开始可以构造一条无穷长的路径 $n_0,\cdots,n_m,\cdots$,且对路径上的任意节点 n_i,$\exists n_{i+k}(k>0,L(n_{i+k}).f=\surd)$,由命题6可推知,$L(n_i).S$ 中的所有预测均可在路径 $n_i,n_{i+1},\cdots$ 上可实现。设路径 $n_0,\cdots,n_m,\cdots$ 上的字符串为 σ,路径 $n_0,\cdots,n_i$ 上的字符串为 τ_i,构造 $M=\langle\sigma,<,I\rangle$,其中 I 按以下方式定义:

对任意原子命题 p,如果 $p\in L(n_i).S$,则 $\tau_i\in I(p)$,如果 $\neg p\in L(n_i).S$,则 $\tau_i\notin I(p)$;如果 $p,\neg p$ 均不属于 $L(n_i).S$,则对任意 $\tau_i?\varphi\in\mathrm{prf}(\sigma)$,$\tau_i$ 和 $\tau_i?\varphi$ 同时属于或不属于 $I(p)$,否则任取 $\tau_i\in I(p)$ 或 $\tau_i\notin I(p)$。

a)显然,I 符合测试动作条件,可以证明:对任意 $\tau_{i+1}=\tau_i?\varphi$ 和对任意命题公式 α,以下(a)(b)成立:(a) $\tau_i\in I(\alpha)$ 当且仅当 $\tau_i?\varphi\in I(\alpha)$;(b) $\tau_i\in I(\varphi)$。由以上 I 的定义可以很容易推得(a)。(b)的证明如下:由 $\tau_i?\varphi\in\mathrm{prf}(\sigma)$ 可推得存在 $\theta U^{?\varphi}\gamma\in L(n_i).S$,再由 $\rightarrow_?$ 规则可知 $\varphi\in L(n_i).S$。假如 φ 为 p 或 $\neg p$,可由 I 定义直接推得 $\tau_i\in I(\varphi)$。假设当 φ 形如 $\theta,\gamma,\neg\theta$ 或 $\neg\gamma$ 且 $\varphi\in L(n_i).S$ 时,$\tau_i\in I(\varphi)$。当 φ 为 $\theta\wedge\gamma$ 时,则依 $\rightarrow_\wedge$ 规则可知 $\theta,\gamma\in L(n_i).S$,依归纳假设 $\tau_i\in I(\theta)$,$\tau_i\in I(\gamma)$ 可推得 $\tau_i\in I\wedge\theta(\gamma)$;当 φ 为 $\neg(\theta\wedge\gamma)$ 时,则依 $\rightarrow_{\neg\wedge}$ 规则可知 $\neg\theta\in L(n_i).S$ 或 $\neg\gamma\in L(n_i).S$,依归纳假设 $\tau_i\in I(\neg\theta)$ 或 $\tau_i\in I(\neg\gamma)$,可推得 $\tau_i\in I(\neg(\theta\wedge\gamma))$。综上可知(b)成立。

b)还可依公式结构归纳证明:如果 $\lambda\in L(n_i).S$,则 $M,\tau_i\vDash\lambda$。当 λ 为 p 或 $\neg p$,结论是显然的。λ 结构为其他情形的情况下,证明方法相似但过程比较繁琐,在此仅给出其中两种情形的证明。假设当 λ 为 $\alpha,\beta,\neg\alpha$ 或 $\neg\beta$ 且 $\lambda\in L(n_j).S(j\geqslant 0)$ 时,$M,\tau_j\vDash\lambda$,即:(a)如果 λ 为 $\alpha U^\pi\beta$,因 λ 在路径 $n_i,\cdots,n_m,\cdots$ 上可实现,故存在 $m\geqslant 0$,使得 $\alpha\in L(n_{i+k}).S(k<m)$,$\beta\in L(n_{i+m}).S$ 且 n_1 到 n_m 有向边上的动作序列 $\tau\in RS(\pi)$。依归纳假设可知 $M,\tau_{i+k}\vDash\alpha(k<m)$,$M,\tau_{i+m}\vDash\beta,\tau_i\tau=\tau_{i+m}$,即推得 $M,\tau_i\vDash\alpha U^\pi\beta$。(b)如果 λ 为 $\neg(\alpha U^a\beta)$,则由 $\rightarrow_{\neg a}$ 规则可知:①当存在 $b\in\Sigma(a\neq b,\theta U^b\gamma\in L(n_i).S)$ 时,依归纳假设 $M,\tau_i\vDash\theta U^b\gamma$ 可直接推得 $M,\tau_i\vDash\neg(\alpha U^a\beta)$;②否则,$\neg\alpha\in L(n_i).S$ 或 $\alpha U^a\neg\beta\in L(n_i).S$,依归纳假设 $M,\tau_i\vDash\neg\alpha$ 或 $M,\tau_i\vDash\alpha U^a\neg\beta$,可推得 $M,\tau_i\vDash\neg(\alpha U^a\beta)$。

由 a)和 b)及 $\varphi\in L(n_0).S$ 可知,M 满足测试动作条件且 $M,\varepsilon\vDash\varphi$。

引理2 如果存在模型 $M=\langle\sigma,<,I\rangle$ 满足 $M,\varepsilon\vDash\varphi$,则可通过 tableau 过程构造一个 tableau $G^\varphi=(G^\varphi,L^\varphi)$,$G^\varphi$ 非空。

证明 假设 φ 的完全 tableau 为 $G^c=(G^c,L^c)$,证明策略是根据 M 构造一个 G^c 的一个子图 G,并证明 G 中节点都不会在 tableau 过程的第二阶段删除。

a)令 $G=\varnothing$,构造初始节点 n_0',令 $L(n_0').\underline{S}=\{(\varphi,0),(\Box\bigcirc T,0)\}$,$L(n_0').f=\surd$。经过 n_0' 的浸透过程后,可以得到 n_0' 的饱和节点集合。用 τ_i 表示字符串 $\tau(\tau\in\mathrm{prf}(\sigma),|\tau|=i)$,可以根据 I 在 τ_0 点的解释从 n_0'的饱和节点集合中 $\mathrm{saturate}(L(n_0').\underline{S})$ 选择一个满足以下两个条件的节点 n_0:(a)对任意 $\alpha\in L(n_0).S$,$M,\tau_0\vDash\alpha$;(b)对任意 $\alpha U^{\pi^*}\beta\in L(n_0).S$,如果 $M,\tau_0\vDash\beta$,则 $\beta\in L(n_0).S$。由命题3可知,这样的 n_0 必然存在。令 $L(n_0).\underline{S}=\{(\alpha,1)\mid\alpha\in L(n_0).S\text{ 为预测}\}\cup\{(\alpha,0)\mid\alpha\in L(n_0).S\text{ 非预测}\}$,至此得到 G 的第一个节点 n_0,令 n_0^t 为 n_0。

b)对 $i=1,2,\cdots$ 分别重复以下过程,可以得到 G 的其他节点和有向边:首先,由 tableau 构造过程可知,必然存在$(\alpha U^a\beta,1)$属于 $L(n_{i-1}^t).\underline{S}$。因此,通过扩展 n_{i-1}^t,可以得到节点 n_i',由全局规则可知,对任意 $\alpha\in L(n_i').S$,$M,\tau_0 a\vDash\alpha$,即 $M,\tau_i\vDash\alpha$。经过 n_i' 的浸透过程后,又可以选择一个满足以下两个条件的 n_i' 的饱和节点 n_i:(a)对任意 $\alpha\in L(n_i).S$,$M,\tau_i\vDash\alpha$;(b)对任意 $\alpha U^{\pi^*}\beta\in L(n_i).S$,如果 $M,\tau_i\vDash\beta$,则 $\beta\in L(n_i).S$。如果不存在 $n(n\in G,L(n_i).\underline{S}=L(n).\underline{S})$,则令 n_i^t 为 n_i,将 n_i,(n_{i-1},n_i)加入 G,$L((n_{i-1},n_i))=a$,否则令 n_i^t 为 n,将(n_{i-1},n)加入 G,$L((n_{i-1},n))=a$。

通过以上构造过程,可以得到一个无穷节点序列 $n_0^t,n_1^t,\cdots$ 和一个 tableau $G=(G,L)$,显然 G 为 G^c 的一个子图,G 中所有节点都在上述序列中出现,而且所有 n_i^t 均为 G 中节点。

对任意 n_i^t,可证明以下命题9。

命题9 对任意 $\alpha U^\pi\beta\in L(n_i^t).S$,存在 $m\geqslant 0$,使得 $\alpha\in L(n_{i+j}^t).S(j<m)$,$\beta\in L(n_{i+m}^t).S$ 且路径 $n_i^t,\cdots,n_{i+m}^t$ 上的字符

串 $\tau \in RS(\pi)$。

证明　假设 $\alpha U^{\pi}\beta \in L(n_i^t).S$,由节点 n_i^t 的构造过程可知,$M,\tau_i \vDash \alpha U^{\pi}\beta$。下面依 π 结构归纳证明命题。

a)基本步。当 π 为 $a \in \Sigma$ 时,因 n_i^t 为饱和节点,故如果 $\alpha U^{a}\beta \in L(n_i^t).S$,则必有 $\alpha \in L(n_i^t).S$,而 n_{i+1}^t 是 n_i^t 扩展出的节点 $n_{i+1}{}'$的饱和节点,故 $\beta \in L(n_{i+1}^t).S, L((n_i^t, n_{i+1}^t)) = a$,命题 9 成立,此时 $m=1$。

b)假设当 π 为 π_1、π_2 时,命题 9 成立。

c)分以下三种不同情形证明归纳步:

(a)当 π 为 $\pi_1;\pi_2$ 时,由 $\alpha U^{\pi_1;\pi_2}\beta \in L(n_i^t).S$ 和 n_i^t 为饱和节点可知 $\alpha U^{\pi_1}\alpha U^{\pi_2}\beta \in L(n_i^t).S$。由归纳假设可知,存在 m_1 满足 $\alpha \in L(n_{i+j}^t).S(j<m_1)$,$\alpha U^{\pi_2}\beta \in L(n_{i+m_1}^t).S$ 且路径 $n_i^t,\cdots,n_{i+m_1}^t$ 上的字符串 $\tau_1 \in RS(\pi_1)$。由 $\alpha U^{\pi_2}\beta \in L(n_{i+m_1}^t).S$ 和归纳假设又可进一步推知,存在 m_2 满足 $\alpha \in L(n_{i+m_1+j}^t).S(j<m_2)$,$\beta \in L(n_{i+m_1+m_2}^t).S$ 且路径 $n_{i+m_1}^t,\cdots,n_{i+m_1+m_2}^t$ 上的字符串 $\tau_2 \in RS(\pi_2)$,显然命题 9 成立,此时 $m=m_1+m_2$。

(b)当 π 为 $\pi_1 + \pi_2$ 时,由 $\alpha U^{\pi_1+\pi_2}\beta \in L(n_i^t).S$ 和 n_i^t 为饱和节点可知 $\alpha U^{\pi_1}\beta \in L(n_i^t).S$ 或 $\alpha U^{\pi_2}\beta \in L(n_i^t).S$,由归纳假设易推得命题 9 成立。

(c)当 π 为 π_1^* 时,$M,\tau_i \vDash \alpha U^{\pi_1^*}\beta$。由于固定 M,$M,\tau_i \vDash \alpha U^{\pi_1^*}\beta$ 当且仅当存在 $\tau \in RS(\pi')$(π'为 π_1 的 n 次连接),使得 $M,\tau_i \vDash \alpha U^{\pi'}\beta$,因此只要证明当 π'为 π_1 的 n 次连接这种情形时,命题 9 即可成立。当 n 为 0 时,即 $M,\tau_i \vDash \beta$,由节点 n_i^t 的构造过程可知必有 $\beta \in L(n_i^t).S$,命题 9 成立。假设当 n 为 $m(m \geqslant 0)$时,命题 9 成立。当 n 为 $m+1$ 时,$\beta \notin L(n_i^t).S$,由 $\alpha U^{\pi_1^*}\beta \in L(n_i^t).S$ 和 n_i^t 为饱和节点可知 $\alpha U^{\pi_1}\alpha U^{\pi_1^*}\beta \in L(n_i^t).S$,由归纳假设,参照前面的 π 为 $\pi_1;\pi_2$ 时的证明过程,易推得命题 9 成立。

至此,本文证明了命题 9,即对任意 $\alpha U^{\pi}\beta \in L(n_i^t).S$,$\alpha U^{\pi}\beta$ 可在路径 $n_i^t, n_{i+1}^t,\cdots$上实现。由命题 5 和 9 易进一步推得,存在路径 $n_i^t, n_{i+1}^t,\cdots$上的节点 n_{i+m}^t 满足 $L(n_{i+m}^t).f=\surd$。因此,在序列 $n_0^t, n_1^t,\cdots$中出现的节点均不满足 tableau 算法第二阶段执行过程中的节点删除条件,而 G 中所有节点都在序列 $n_0^t, n_1^t,\cdots$中出现,故 G 中所有节点都不会从 G^c 中删除,G^{φ} 非空。

定理 1　φ 可满足当且仅当 G^{φ} 非空。

证明　由命题 7、8 和引理 1 可推得,当 G^{φ} 非空时,存在模型 M 和 τ 满足 $M,\tau \vDash \varphi$,即 φ 可满足;由命题 8 和引理 2 可推得,当 φ 可满足时,G^{φ} 非空。

2.5.2　算法复杂性

DLTL$^+(\Sigma)$公式 φ 的公式闭包 sub(φ)为满足以下条件的最小集合:

$\varphi \in \text{sub}(\varphi)$;

如果 $\alpha \in \text{sub}(\varphi)$,则$\neg\alpha \in \text{sub}(\varphi)$;

如果 $\alpha \wedge \beta \in \text{sub}(\varphi)$,则 $\alpha \in \text{sub}(\varphi), \beta \in \text{sub}(\varphi)$;

如果 $\alpha U^{\gamma}\varphi \beta \in \text{sub}(\varphi)$,则 $\varphi \in \text{sub}(\varphi)$;

对于 $a \in \Sigma$,如 $\alpha U^{a}\beta \in \text{sub}(\varphi)$,则 $\alpha \in \text{sub}(\varphi), \beta \in \text{sub}(\varphi)$;

对于 $a \in \Sigma$,如$\neg(\alpha U^{a}\beta) \in \text{sub}(\varphi)$,则 $\alpha U^{a}\neg\beta \in \text{sub}(\varphi)$;

如果 $\alpha U^{\pi_0;\pi_1}\beta \in \text{sub}(\varphi)$,则 $\alpha U^{\pi_0}\alpha U^{\pi_1}\beta \in \text{sub}(\varphi)$;

如果 $\alpha U^{\pi_0+\pi_1}\beta \in \text{sub}(\varphi)$,则 $\alpha U^{\pi_0}\beta \in \text{sub}(\varphi), \alpha U^{\pi_1}\beta \in \text{sub}(\varphi)$;

如果 $\alpha U^{\pi^*}\beta \in \text{sub}(\varphi)$,则 $\alpha U^{\pi}\alpha U^{\pi^*}\beta \in \text{sub}(\varphi)$;

用#A 表示集合 A 的基数,依 φ 的结构归纳,易证得#sub(φ)$\leqslant 2|\varphi|$。

定理 2　Tableau 算法的时间复杂度为 $2^{O(n)}$。

证明　设 DLTL$^+(\Sigma)$公式 φ 的长度$|\varphi|=n$,φ 的完全 tableau 为 $G^c=(G^c,L^c)$。由 tableau 规则可看出,G^c 构造过程中引入的所有公式都属于 sub(φ),G^c 中的节点用 $2^{\text{sub}(\varphi)\times|0,1|}\times\{\surd,\downarrow\}$中的元素作标签,由#sub($\varphi$)$\leqslant 2|\varphi|$可推得不同节点标签的个数属于 $2^{O(n)}$,因此 G^c 中未被阻塞的节点个数属于 $2^{O(n)}$。而且对任意节点 $n \in G^c$,#$L^c(n)$.Σ不多于 $4n$,故 n 浸透至饱和需要应用 tableau 规则次数不超过 $4n$ 次。令 d 为 tableau 规则的最大可选输出个数,故 n 的不同饱和节点数不超过 d^{4n},即一个节点的后续节点个数不超过 d^{4n}。因此,G^c 中所有节点个数属于 $2^{O(n)}$。构造 G^c 只需最多运用 $2^{O(n)}$次 tableau 规则以及 $2^{O(n)}$次节点比较操作。在 tableau 过程的第二阶段,除最后一次外,每次遍历 G^c 中至少删除一个节点,每次删除最多只需访问 G^c 中其他节点一次,因此,tableau 过程的第二阶段也可在 $2^{O(n)}$时间内结束。

3　结束语

本文提出 DLTL 的扩展 DLTL$^+$ 以支持测试动作。相比于文献[4]的方法,本文支持测试动作方法从语言根源上解决了测试动作支持问题,没有显著提高公式可满足性问题的计算复杂性,因而更直接有效。

文中参考了文献[1]中的部分思想,提出了一个 tableau 算法,利用该算法可在 $2^{O(n)}$时间内判定 DLTL$^+$公式的可满足性。与文献[1]的基于自动机的 DLTL 公式可满足性算法不同,本文基于 tableau 的算法用于判定 DLTL$^+$公式可满足性不需要将 DLTL$^+$公式中出现的正规程序转换为有限自动机,计算效率应相对更高。

文中 DLTL 扩展的方式和 tableau 算法设计思路对 DLTL$_{\text{DL}}$ 后续研究有较大帮助。首先,本文的测试动作扩展方式从理论上讲也适用于 DLTL$_{\text{DL}}$,因而为 DLTL$_{\text{DL}}$的测试动作支持问题指出了一条可行的解决之道;其次,文献[6]中的 DLTL$_{\text{DL}}$公式可满足性算法同样基于自动机,参照本文 tableau 算法,笔者相信可以设计出更高效的判定 DLTL$_{\text{DL}}$公式可满足性的算法。

参考文献:

[1] HENRIKSEN J G, THIAGARAJAN P S. Dynamic linear time temporal logic[J]. Annals of Pure and Applied logic, 1999, 96(1-3): 187-207.

[2] HAREL D, KOZEN D, TIUR N J. Dynamic logic[M]. Cambridge: The MIT Press, 2000.

[3] KRÖGER F, MERZ S. Temporal logic and state systems[M]. New York: Springer-Verlag, 2008.

[4] GIORDANO L, MARTELLI A, SCHWIND C. Reasoning about actions in dynamic linear time temporal logic[J]. Logic Journal of IGPL, 2001, 9(2): 273-287.

[5] GIORDANO L, MARTELLI A, SCHWIND C. Specifying and verifying interaction protocols in a temporal action logic[J]. Journal of Applied Logic, 2007, 5(2): 214-234.

[6] 孙永新,赵希顺,符志强. 描述逻辑的动态时序扩展[J]. 计算机应用研究, 2012, 29(2): 536-541.

[7] WOLPER P, The tableau method for temporal logic: an overview[J]. Logique et Analyse, 1985, 28(110-111): 119-136.