扩展命题区间时序逻辑公式可满足性判定算法

朱维军[1,2]，邓淼磊[3]，周清雷[1]，张海宾[2]

(1. 郑州大学信息工程学院 郑州 450001；2. 西安电子科技大学计算机学院 西安 710071；3. 河南工业大学信息科学与工程学院 郑州 450001)

【摘要】针对扩展命题区间时序逻辑由于缺少验证算法因而不能用于模型检测问题，提出该逻辑的可满足性判定算法。首先，正则形子算法把带星算子或不带星算子的扩展命题区间时序逻辑公式翻译为其正则形公式；然后，正则图子算法根据正则形公式构造公式的正则图模型；最后，判定子算法在正则图上判定公式的可满足性。如果在正则图上直接加上接受条件，即可得到公式的自动机模型。新算法的提出为带有星算子的扩展命题区间时序逻辑的模型检测解决了核心方法问题。仿真结果表明，与相关方法相比，基于扩展命题区间时序逻辑的新方法在描述与验证循环结构性质方面具有比较优势。

关 键 词 扩展命题区间时序逻辑； 模型检测； 正则图； 可满足性判定

中图分类号 TP301 **文献标识码** A doi:10.3969/j.issn.1001-0548.2011.05.023

Decision Procedure for Extended Propositional Interval Temporal Logic

ZHU Wei-jun[1,2], DENG Miao-lei[3], ZHOU Qing-lei[1], and ZHANG Hai-bin[2]

(1. School of Information Engineering, Zhengzhou University Zhengzhou 450001;

2. School of Computer Science, Xidian University Xi'an 71007;

3. College of Information Science and Technology, Henan University of Technology Zhengzhou 450001)

Abstract Compared with propositional interval temporal logic (PITL), extended propositional interval temporal logic (EPITL) is equipped with infinite models and the chop-star operator additionally. However, there is no algorithm available for model checking EPITL. To this end, we propose an algorithm for EPITL satisfiability checking. The first step is to translate EPITL formulae with or without chop-star operators into their normal forms (NFs). The second step is to construct normal form graphs (NFGs) from the NF formulae. The last step is to check the satisfiability of the EPITL formulae by using the NFGs. Accordingly, we can translate the NFGs into buchi automata. So, the EPITL model checking problem is solved. As shown in the simulation results, our approach is superior to the existing ones based on other logics in terms of specification and verification of some properties of loop structures.

Key words extended propositional interval temporal logic; model checking; normal form graph; satisifiablity checking

模型检测技术是近年来研究的热点，已经在硬件与网络协议领域取得了大量的重要应用。其基本思想是，自动验证有穷状态空间是否满足需求的性质，经常采用自动机、Petri网或进程代数建立动态模型，静态性质则使用时序逻辑来表达。普通时序逻辑只能表达孤立点之间的时序性质，而区间时序逻辑(interval temporal logic，ITL)及其规范可执行子集Tempura语言[1] ，则可通过chop算子实现对数字电路区间内状态之间以及区间之间时序关系性质的描述。ITL逻辑命题部分PITL公式的可有穷满足的判定算法[2]直到2003年才完成，然而由于非终止并发系统大多具无穷模型，因而很难在该判定算法基础上开发相应的模型检测工具。

针对该问题，扩展区间时序逻辑(extended interval temporal logic，EITL)[3-7]把区间从有穷扩展到无穷，与同类扩展ITL模型至无穷区间[8]相比，前者由于只允许最后区间具无穷模型，因而更适合真实的非终止并发系统。而EITL的规范可执行子集扩展Tempura语言[3,5]更使得规范可以通过执行的方式得到结果。然而，目前EITL满足性判定的方法问题

收稿日期：2010－01－18； 修回日期：2010－07－06

基金项目：国家863高技术研究发展计划(2007AA010408)；国家自然科学基金(61003079)；教育部博士点基金(20100203120012)；陕西省工业攻关计划(2009K01-36)；中央高校基本科研业务费(JY10000903014)。

作者简介：朱维军(1976－)，男，博士，主要从事高可信软件与计算机形式化方法方面的研究.

仍未解决，本文对此进行研究。由于一阶部分不可判定，因此只考查命题部分(EPITL)。

1 扩展命题区间时序逻辑EPITL

1.1 语法

定义 1 EPITL基本公式φ被定义为$\varphi ::= Ap \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid O\varphi \mid \varphi_1;\varphi_2 \mid \varphi*$，其中，$Ap$为原子命题；“$O$”为未来算子，表示“下一时刻”；定义中的“;”为区间算子。

1.2 语义

定义 2 在Kripke结构上定义状态s: $Ap \to B$，其中，真值集合B={true,false}。

定义 3 区间σ为状态序列$\langle s_i,\cdots,s_j\rangle$，其中区间从第$i$个状态$s_i$开始，第$j$个状态$s_j$结束，令$N$为整数，$\omega$为无穷大整数，且$i \in N$，$i \in \omega$，$j \in N_\omega = N \cup \{\omega\}$，$|\sigma| = j - i$，区间可为有穷也可为无穷。

定义 4 一个四元组为$I = (\sigma, i, k, j)$，其中k为当前状态，$i \leqslant k \leqslant j$，$(\sigma, i, k, j) \models \varphi$表示公式$\varphi$在区间$\langle s_i,\cdots,s_j\rangle$上被满足。

定义 5 满足关系$\models$被定义为：

$I - Ap$: $I \models p$，当且仅当$s_k[p] = \text{true}$，其中$p \in Ap$；

$I - \text{not}$： $I \models \neg\varphi$，当且仅当$I \not\models \varphi$；

$I - \text{or}$： $I \models \varphi_1 \vee \varphi_2$，当且仅当$I \models \varphi_1$或$I \models \varphi_2$；

$I - \text{next}$：$I \models O\varphi$，当且仅当$k < j$，且$(\sigma, i, k+1, j) \models \varphi$；

$I - \text{chop}$：$I \models \varphi_1;\varphi_2$，当且仅当存在$r$，使得$(\sigma, i, k, r) \models \varphi_1$，$(\sigma, r, r, j) \models \varphi_2$，其中$k \leqslant r \lhd j$；

$I - \text{star}$：$I \models \varphi*$，当且仅当：

1) 存在有穷多个$r_0, r_1, \cdots, r_n \in N_\omega$，使得$k = r_0 \leqslant r_1 \leqslant \cdots \leqslant r_{n-1} \lhd r_n = j$，$(\sigma, i, k, r_0) \models \varphi$，且对任意$1 \leqslant l \leqslant n$，$(\sigma, r_{l-1}, r_{l-1}, r_l) \models \varphi$；或者$j = \omega$，存在无穷多个整数$k = r_0 \leqslant r_1 \leqslant \cdots$，使得$\lim\limits_{i\to\infty} r_i = \omega$，$(\sigma, i, k, r_0) \models \varphi$，且对任意$l > 1$，$(\sigma, r_{l-1}, r_{l-1}, r_l) \models \varphi$。

2) 或者$k = j$。

1.3 导出公式

EPITL导出公式为：

A_0 : $\text{empty} \triangleq \neg O\,\text{true}$

A_1 : $\text{more} \triangleq O\,\text{true}$

A_2 : $O^0\varphi \triangleq \varphi$，$O^n\varphi \triangleq O(O^{n-1}\varphi)$　　$n \in N$

A_3 : $\text{len}(n) \triangleq O^n\ \text{empty}$

A_4 : $\text{skip} \triangleq \text{len}\ 1$

A_5 : $\Diamond\varphi \triangleq \text{true};p$

A_6 : $\Box\varphi \triangleq \neg\Diamond\neg\varphi$

A_7 : $\varphi^+ \triangleq \varphi;\varphi*$

A_8 : $\varphi_1 \wedge \varphi_2 \triangleq \neg(\neg\varphi_1 \vee \neg\varphi_2)$

A_9 : $\varphi_1 \to \varphi_2 \triangleq \neg\varphi_1 \vee \varphi_2$

比较EPITL与命题投影时序逻辑(propositional projection temporal logic，PPTL)[9]的语法和语义，不难发现两种区间逻辑的区别：前者拥有两种区间算子chop(即“;”)和chop star(即“$\varphi*$”)；后者也有两种描述区间语义的算子chop和prj。由于chop star与prj无法互相表示，因而两种逻辑表达能力是相交非包含关系。为了得到EPITL判定算法，应考虑如何在PPTL判定算法[9]的基础上，去掉prj，加上chop star。而PITL判定算法[2]也未实现对chop star的判定。

2 EPITL正则形算法

定义 6 对公式集合$\langle R_i\rangle_i$，如果$V_{i=1}^n R_i \equiv \text{true}$且$V_{i\neq j}(R_i \wedge R_j) \equiv \text{false}$，则称$\langle R_i\rangle_i$是完全的。

定理 1[9] 对完全公式集合$\langle R_i\rangle_i$有$\neg V_{i=1}^n(R_i \wedge Q_i) \equiv V_{i=1}^n(R_i \wedge \neg Q_i)$成立。

定义 7 设$R \in L_{\text{EPITL}}$，如果R可转化为$R \equiv R_e \wedge \text{empty} \vee V_{i=1}^n(\Lambda_{j=1}^m(\dot{p}_{ij} \wedge OR_i))$，则称其为$R$的正则形，其中$\dot{p}_{ij} \in AP$，$\dot{p}_{ij} = p_{ij}$或$\dot{p}_{ij} = \neg p_{ij}$。

引理 1[9] 如果一个公式可以转化为正则形，则其可转化为完全正则形。

引理 2 设$R = p* \in L_{\text{EPITL}}$，则$R$可转化成正则形。

证明 1) 设$p \in L_{\text{EPITL}}$的正则形公式为$p \equiv p_e \wedge \text{empty} \vee V_i(p_{ci} \wedge Op_{ni})$，则：

$$p* \equiv \text{empty} \vee (p; p*) \equiv$$
$$\text{empty} \vee ((p_e \wedge \text{empty} \vee V_i(p_{ci} \wedge Op_{ni}));p*)$$
$$\text{empty} \vee ((p_e \wedge \text{empty});p*) \vee V_i(p_{ci} \wedge O(p_{ni};p*)) \equiv$$
$$\text{empty} \vee ((p_e \wedge \text{empty});(1)) \vee V_i(p_{ci} \wedge O(p_{ni};p*)) \equiv$$
$$\text{empty} \vee ((p_e \wedge \text{empty});p*) \vee V_i(p_{ci} \wedge O(p_{ni};p*))$$

因此：

(1) 如果前缀n个p均为$p_e \wedge \text{empty}$，第n+1个p为$V_i(p_{ci} \wedge Op_{ni})$，$n \in N$时，则：

$$p* \equiv$$
$$((p_e \wedge \text{empty});V_i(p_{ci} \wedge O(p_{ni};p*))) \vee V_i(p_{ci} \wedge O(p_{ni};p*)) \equiv$$
$$V_i(p_{ci} \wedge O(p_{ni};p*))$$

(2) 如果前缀ω个p均为$p_e \wedge \text{empty}$，则无穷个p均为状态模型集结于当前状态k，因此$p* \equiv p_e \wedge \text{empty}$。

(3) 如果$p*$为空，则$p* \equiv \text{empty}$。

考虑到对当前时刻k，有(1)、(2)、(3)共3种情况，

因此有：

$$p^* \equiv \text{empty} \vee p_e \wedge \text{empty} \vee V_i(p_{ci} \wedge O(p_{ni}; p^*)) \equiv \text{empty} \vee V_i(p_{ci} \wedge O(p_{ni}; p^*))$$

2) 把 p^* 的正则形代入 $(q_e \wedge \text{empty}); p^*$，得：

$$(q_e \wedge \text{empty}); p^* \equiv (q_e \wedge \text{empty}); (\text{empty} \vee V_i(p_{ci} \wedge O(p_{ni}; p^*))) \equiv q_e \wedge \text{empty} \vee V_i(q_e \wedge p_{ci} \wedge O(p_{ni}; p^*))$$

综上，对 $R = p^*$，R可转化成正则形。

定理 2　任意公式 $R \in L_{\text{EPITL}}$ 都可转化成正则形。

证明　设P和Q都已转为正则形，即：

$$P \equiv P_e \wedge \text{empty} \vee V_i(P_{ci} \wedge OP_{ni})5 \quad (1)$$

$$Q \equiv Q_e \wedge \text{empty} \vee V_j(Q_{cj} \wedge OQ_{nj}) \quad (2)$$

式中，P_e、Q_e、P_{ci} 和 Q_{cj} 是状态公式，P_{ni}、Q_{nj} 是EPITL公式。

若 $R \equiv r$ 且 $r \in AP$，则 $R \equiv r \wedge (\text{empty} \vee \text{more}) \equiv r \wedge \text{empty} \vee r \wedge O\,\text{true}$

归纳为：

1) $R \equiv P \vee Q$：将式(1)和式(2)代入 $P \vee Q$ 即可；

2) $R \equiv OP$：有 $R \equiv \text{true} \wedge OP$；

3) $R \equiv \neg P$：根据引理1，可把P转化成完全正则形，根据定理1，有 $\neg P \equiv \neg P_e \wedge \text{empty} \vee V_i\,(P_{ci} \wedge \text{O}\neg P_{ni})$；

4) $R \equiv P;Q$，有：

$$R \equiv P_e \wedge Q_e \wedge \text{empty} \vee V_j(P_e \wedge Q_{cj} \wedge OQ_{nj}) \vee V_i(P_{ci} \wedge O(P_{ni}; Q))$$

5) $R \equiv P^*$，由引理2知，结论成立。

根据结构归纳法证明原理，命题得证。

令w表示状态公式，μ表示基本积，将EPITL公式转化为正则形的算法(算法1)如下：

1) 主算法NormalF。

function NormalF(R)

/*前提：$R \in L_{\text{EPITL}}$且公式R已被算法PRE处理过。

/*输出：与公式R等价的正则形NF。

begin function

case

R的形式为true：return empty$\vee\circ$true

R的形式为false：return $p\wedge\neg p\wedge$empty$\vee p\wedge\neg p\wedge\bigcirc$false

R的形式为状态公式：return NFDNF (DNF(R))

R的形式为$\mu\wedge\circ P$：return R

R的形式为$\mu\wedge$empty：return R

R的形式为$P\vee Q$：return NormalF(P)$\vee$NormalF(Q)

R的形式为$P;Q$：return CHOP(R)

R的形式为P^*：return CHOPS(R)

R的形式为$\neg P$：return NEG (CONF(NormalF(P)))

R的形式为$\Diamond P$：return NormalF(P)$\vee\circ\Diamond P$

R的形式为$\Box P$：return NormalF($P\wedge$empty)$\vee$NormalF ($P\wedge\circ\Box P$)

R的形式为$P\wedge Q$：return AND (NormalF(P), NormalF (Q))

end case

end function

2) 子算法CHOP(R) 将chop结构的公式转换为正则形[9]。

3) 子算法CHOPS将chop star结构的公式转换为正则形function CHOPS(R)：

/* 前提：$R \equiv P^* \in L_{\text{EPITL}}$，被算法PRE处理过。

/* 输出：与公式R等价的正则形。

begin function

令 $P_e \wedge \text{empty} \vee V_i(P_{ci} \wedge OP_{ni}) \equiv \text{CHOP}(P)$，则 return $\text{empty} \vee V_i(P_{ci} \wedge O(P_{ni}; P^*))$

end function

4) 子算法CONF(R)将正则形公式R转换为完全正则形[9]。

5) 子算法NEG(R)将公式$\neg R$转换为与之等价的正则形(其中R为完全正则形公式)[9]。

6) 子算法AND (P,Q)将公式$P\wedge Q$转换为与之等价的正则形(其中P和Q均为正则形公式)[9]。

定理2保证了该算法的正确性。

3　EPITL可满足性的判定

3.1　EPITL正则图(NFG)

定义 9　对任意一个公式 $P \in L_{\text{EPITL}}$，表示公式P的NFG定义为二元组 $G = \langle \text{CL}(P), \text{EL}(P) \rangle$，其中顶点集 CL($P$) 和边集 EL($P$) 归纳定义为：

1) $P \in \text{CL}(P)$；

2)对任意 $Q \in \text{CL}(P) \setminus \{\varepsilon, \text{false}\}$，如果 $Q \equiv Q_e \wedge \text{empty} \vee V_{i=1}^{r}(Q_{ci} \wedge OQ_{ni})$，则，对 $\forall i, 1 \leqslant i \leqslant r$，$Q_{ni} \in \text{CL}(P)$，$(Q, Q_{ci}, Q_{ni}) \in \text{EL}(P)$；

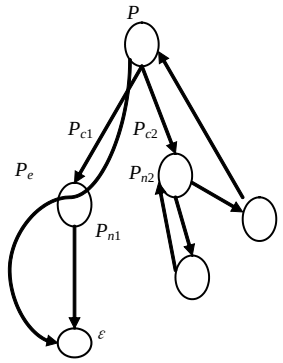
3) CL(P) 和 EL(P) 仅有1)和2)两种情况产生。

例1　P^* 的NFG。$G = \langle \text{CL}(P^*), \text{EL}(P^*) \rangle$，如图1所示，图1a为 $P \in L_{\text{EPITL}}$ 的NFG，图1b为 $P^* \in L_{\text{EPITL}}$ 的 NFG。令 $P \equiv P_e \wedge \text{empty} \vee V_i\,(P_{ci} \wedge OP_{ni})$，则 $P^* \equiv \text{empty} \vee V_i\,(P_{ci} \wedge O(P_{ni}; P^*))$，当 $P;P^*$ 的前缀子区间P用完区间时，有：

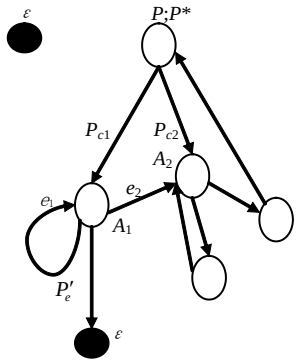
$$(P_e' \wedge \text{empty}); P^* \equiv (P_e' \wedge \text{empty}); \text{empty} \vee V_i\,(P_{ci} \wedge O(P_{ni}; P^*)) \equiv$$

$P_e' \wedge \text{empty} \vee V_i\ (P_e' \wedge P_{ci} \wedge O(P_{ni}; P^*))$

因此，在图1b中，对NFG P的所有有穷模型节点(图中节点A_1)，根据定义9，对从A_1到P节点的所有子节点都引入有向边(图1b中的两条边e_1、e_2)，同时引入边(A_1, P_e', ε)。注意，图中，$A_1 \equiv P_{n1}; P^*$，$A_2 \equiv P_{n2}; P^*$，$e_1 \equiv P_e' \wedge P_{c1}$，$e_2 \equiv P_e' \wedge P_{c2}$。



a. P的NFG



b. P^*的NFG

图1 NFG图

修改PPTL正则图算法[9]，根据正则形公式构造EPITL正则图的算法(算法2)如下：

function NFG(P)

/*precondition: P is a EPITL formula in pre-prepared form

/*postcondition: NFG(P) computes NFG of P, G=(CL(P);EL(P))

begin function

/*initialization

CL(P)={P}; EL(P)=∅; mark[P]=0; Add E=Add N =0;

while there exists $R \in$CL(P)\{ε,false}, and mark[R]==0

do mark[R]=; /*marking R is decomposed

if R is $Q_1;Q_2$ or $(Q_1;Q_2) \wedge Q_3$ or $Q_3 \wedge (Q_1;Q_2)$ then /*computing NFG of Q_1

add label F in node R; G'=(CL(Q_1); EL(Q_1))=NFG(Q_1);

Q=NormalF(R);

case

Q is $V_{j=1}^{h}(Q_{ej} \wedge \text{empty})$: Add E=1;/* first part of NF needs added

Q is $V_{i=1}^{k}(Q_i \wedge \bigcirc Q_i')$: Add N=1; /*second part of NF needs added

Q is $V_{j=1}^{h}(Q_{ej} \wedge \text{empty})$ $\vee V_{i=1}^{k}(Q_i \wedge \bigcirc Q_i')$: Add E =Add N=1;

/*bothparts of NF needs added

end case

if Add E==1 then /*add first part of NF

CL(P)=CL(P) ∪{ }

EL(P)=EL(P) $\cup_{j=1}^{h}\{(R, Q_{ej}, \varepsilon)\}$;

Add E=0;

if Add N==1 then /* add second part of NF

for i=1 to k do

if Q_i' is false then mark[Q_i']=1;/* Q_i' needs not decomposed

else if Q_i' CL(P) then mark[Q_{0i}]=0;

CL(P) = CL(P) $\cup_{i=1}^{k}$ { Q_i' };/* Q_i' needs decomposed

EL(P)= EL(P) $\cup_{i=1}^{k}\{(R, Q_i, Q_i')\}$;

Add N=0;

end while

return G;

end function

引理 3[9] 对任意公式P构造正则图，有$|\text{CL}(O P)| \leqslant |\text{CL}(P)|+1$。

引理 4[9] 设P_e为状态公式，Q为时序公式，构造正则图，有$|\text{CL}(P_e \wedge Q)| \leqslant |\text{CL}(Q)|+1$。

引理 5[9] 对任意公式P、Q构造正则图，有$|\text{CL}(P \vee Q)| \leqslant |\text{CL}(P)|+|\text{CL}(Q)|+1$。

引理 6[9] 对任意公式P、Q构造正则图，有$|\text{CL}(P;Q)| \leqslant |\text{CL}(P)|+|\text{CL}(Q)|+1$。

引理 7[9] 对任意公式P构造正则图，有$|\text{CL}(\neg P)| \leqslant |\text{CL}(P)|+1$。

引理 8 对任意公式P^*构造正则图，有$|\text{CL}(P^*)| \leqslant |\text{CL}(P)|+1$。

证明 对$P; P^*$，对当前状态：

1) 根据正则图算法，若构造P的正则图，则若$Q \equiv V_i\ (Q_{ci} \wedge O Q_{ni})$，添加节点$Q_{ni}$，添加边

(Q,Q_{ci},Q_{ni})；若构造 $P*$ 的正则图，则 $Q'\equiv V_i\ (Q_{ci}\wedge O(Q_{ni};P*))$，可见两图产生相同的边和节点，只是节点标识不同。

2) P 的正则图当前正则形公式若为 $Q\equiv Q_e\wedge\text{empty}$，即$P$用完了子区间，根据正则形算法，有：

$$(Q_e\wedge\text{empty});P*\equiv Q_e\wedge\text{empty}\vee V_i\ (Q_e\wedge P_{ci}\wedge O(P_{ni};P*))$$

根据正则图算法，从当前节点引入一条到 ε 的边，从当前节点引入一条到节点 $P_{ni};P*$ 的边，而节点 $P_{ni};P*$ 在1)中已产生，因而2)并未添加新节点。

3) 因为 $P*\equiv\text{empty}\vee P;P*$，所以若$P$没有有穷模型，即$P$的正则图中无 ε 节点，则 $P*$ 的正则图中将添加 ε 节点。

从步骤1)～步骤3)可知，$P*$ 的NFG最多比P的NFG增加一个 ε 节点。

定理 3　对任意公式 $R\in L_{\text{EPITL}}$，$\exists k\in N$，使得 $|\text{CL}(R)|=k$。

证明　施归纳于公式R的结构。若 $R\in AP$，则：$R\equiv R\wedge\text{empty}\vee R\wedge O\ \text{true}$

因此有：

$\text{CL}(R)=\{R,\text{true},\varepsilon\}$　　$k=3$。

归纳：$|\text{CL}(P)|=k_1,|\text{CL}(Q)|=k_2$，则：

1) 当 $R\equiv OP$ 时，由引理3得 $|\text{CL}(R)|\leqslant k_1+1$；

2) 当 $R\equiv P\vee Q$ 时，由引理5，$|\text{CL}(R)|\leqslant k_1+k_2+1$；

3) 当 $R\equiv\neg P$ 时，由引理7得 $|\text{CL}(R)|\leqslant k_1+1$；

4) 当 $R\equiv P$；Q 时，由引理6得 $|\text{CL}(R)|\leqslant k_1+k_2+1$；

5) 当 $R\equiv P*$ 时，由引理8得 $|\text{CL}(R)|\leqslant k_1+1$；

由结构归纳法原理，R 的NFG节点数有穷。

3.2　EPITL公式可满足性的判定

定理 4　对公式R的正则图G，R有穷可满足当且仅当G中存在有穷路径。

定理 5　对公式R的正则图G，R无穷可满足当且仅当G中存在无穷路径。

定理4、定理5的证明是对正则形公式每个next推进步数归纳构造集合，对有穷情况做数学归纳，对无穷的情况构造不动点，这种证明与公式R的结构无关，因而可以直接把对PPTL的证明[9]用来证明EPITL的结论。

下面的算法CHECK在正则图上判定EPITL公式的可满足性。其中函数SIMPLIFY删除没有后继节点的非终止于 ε 的节点(这样的节点不在有穷模型上也不在无穷模型上)。定理4、定理5保证了判定算法的有效性和正确性。EPITL公式可满足性判定算法(算法3)如下：

function SIMPLIFY (G)

/* precondition: G=(CL(P); EL(P)) is an NFG of EPITL formula P

/* postcondition: SIMPLIFY (G) computes an NFG of P, G'=(CL′(P); EL′(P)), which contains no redundant nodes

begin function

CL′(P)＝CL(P); EL′ (P)=EL(P);

while $\exists R\in\text{CL}'(P)$ and outdegree of R is 0 /*cut finite model

do CL′(P)=CL′(P)\R;

EL′(P)=EL′(P)\$\cup_i$ (R_i;R_e;R);

/* $_i$\(R_i;R_e;R) denotes the set of edges connecting to node R

end while

while $\exists R\in\text{CL}'(P)$ and labeling of R is F /*discard infinite model of prefix interval

do CL′(P)=CL′(P)\R;

EL′(P)=EL′(P)\$\cup_i$ (R_i;R_e;R);

/* $_i$\(R_i;R_e;R) denotes the set of edges connecting to node R

end while

return G';

end function

Decision Procedure CHECK(P)

1) Build the NFG of P, G=(CL(P); EL(P)), by algorithm NFG;

2) If there are nodes without labeling F and with outdegree non-zero in G then P is satisfiable

3) Obtain the simplied NFG, G'=(CL′(P);EL′(P)), by algorithm SIMPLIFY ;

4) If G' is empty then P is unsatisfiable otherwise satisfiable.

例2　在图1中，以 ε 节点为终点的任一路径，均为公式的有穷模型，图中任一无穷路径，均为公式的无穷模型。路径与模型一一对应。

4　复杂度分析

定理 6　算法3最坏情况下的时间复杂度为非初等。

证明　文献[10]证明了对在字母表{0,1}上运行的具“+”“•”“¬”算子的任意正则表达式的判空问题的时间复杂度下限是非初等时间[10]。文

献[1]将上述表达式用PITL公式表示，从而证明了PITL满足性判定问题的时间复杂度是非初等时间[1]。从表达能力上讲，EPITL $\supseteq$ PITL，且本文给出了EPITL 满足性判定算法，因此EPITL满足性判定问题固有时间复杂度仍为非初等时间，即为算法3的时间复杂度。

定理 7 非初等时间复杂度算法的空间复杂度必为非初等。

证明 设算法P的时间复杂度为TIME$\geqslant O(2^{2^{\cdot^{\cdot^{2^n}}}})$，本文用反证法证明其空间复杂度SPACE$\geqslant O(2^{2^{\cdot^{\cdot^{2^n}}}})$。首先假设SPACE$< O(2^{2^{\cdot^{\cdot^{2^n}}}})$，即SPACE$= O(2^n)$，因此图灵机$M$的带空间为$O(2^n)$，$M$最多有$O(2^{2^n})$个格局。由于任何算法必在有穷步内终止，因此图灵机算法P至多在$O(2^{2^n})$步内穷尽所有格局而终止，即P的时间复杂度为TIME$= O(2^{2^n}) < O(2^{2^{\cdot^{\cdot^{2^n}}}})$，与TIME$= O(2^{2^{\cdot^{\cdot^{2^n}}}})$矛盾。因此，SPACE$< O(2^{2^{\cdot^{\cdot^{2^n}}}})$不成立，即SPACE$\geqslant O(2^{2^{\cdot^{\cdot^{2^n}}}})$成立。

推论 1 算法3最坏情况下的空间复杂度为非初等。

5 相关工作比较

在算法1和算法2的基础上，如果对正则图加上接受条件，就可以直接得到自动机，从而得到逻辑与自动机的转换算法。按照模型检测技术的一般原理，结合已经发展成熟的自动机求积算法与自动机语言判空算法[11]，就可直接得到EPITL模型检测算法。

由于EPITL的表达能力高于LTL，因而新算法对模型的验证能力高于已被广泛使用的SPIN等LTL验证工具。由于EPITL与PPTL的表达能力相交非包含，因而与已有的PPTL判定算法相比，本文的NFG算法具有验证chop star区间性质的能力。而这样的性质在表达程序循环结构时特别有效。为了证明新算法对可判定规范程序循环结构描述性质的验证能力，本文分别使用4种不同方法针对4类不同的有穷或无穷循环结构进行仿真验证，实验结果如表1和表2所示。

表1 4种方法对有穷循环结构的描述公式及验证结果

方法	由多条语句组成的循环体被执行n次	由单条语句组成的循环体被执行n次
SPIN(for LTL)[11]	无描述公式，因而不可验证	无描述公式，因而不可验证
NFG(for EPITL)	$\varphi*\wedge$len(n)，可验证	$(p\wedge$empty$)*\wedge$len(n)，验证产生n+2个节点n+1条边
TABLEAU(for PITL)[2]	$\varphi*\wedge$len(n)，无验证算法因而不可验证	$(p\wedge$empty$)*\wedge$len(n)，验证产生3n+4个节点3n+3条边
NFG(for PPTL)[9]	无描述公式，因而不可验证	$(p\wedge$empty$)$; …;$(p\wedge$empty$)$，验证产生n+2个节点n+1条边

表2 4种方法对无穷循环结构的描述公式及验证结果

方法	由多条语句组成的循环体被执行ω次	由单条语句组成的循环体被执行ω次
SPIN(for LTL)[11]	无描述公式，因而不可验证	$\Box p$，验证产生3个节点、4条边
NFG(for EPITL)	$\varphi*$，可验证	$\Box p$，验证产生1个节点、1条边
TABLEAU(for PITL)[2]	$\varphi*$，无验证算法因而不可验证	$(p\wedge$empty$)*$，无验证算法因而不可验证
NFG(for PPTL)[9]	无描述公式，因而不可验证	$\Box p$，验证产生1个节点、1条边

6 结 论

本文工作给出了EPITL逻辑公式可满足性的判定算法，从而为该逻辑的模型检测提供了核心方法。

区间逻辑具非初等复杂度的判定问题仍可在验证实践中被使用[1]。下一步将在EPITL验证算法的基础上，开发基于EPITL的SPIN验证工具，为著名的模型检测器SPIN提高验证能力。

参 考 文 献

[1] MOSZKOWSKI B. Reasoning about digital circuits[D]. Palo Alto, California, USA: Department of Computer Science, Stanford University, 1983.

[2] BOWMAN H, THOMPSON S. A decision procedure and complete axiomatization of finite interval temporal logic with projection[J]. Journal of Logic and Computation, 2003, 13(2):195-239.

[3] DUAN Z. Temporal logic and temporal logic programming [M]. Beijing: Science Press, 2005.

[4] DUAN Z, KOUTNY M, HOLT C. Projection in temporal logic programming[C]//Proceedings of 5th International Conference on Logic Programming and Automated Reasoning. London, UK: Springer, 1994, 333-344.

[5] DUAN Z. An extended interval temporal logic and a framing technique for temporal logic programming[D]. Tyne and Wear, UK: Department of Computing Science, University of Newcastle Upon Tyne, 1996.

[6] DUAN Z, YANG X, KOUTNY M. Framed temporal logic programming[J]. Science of Computer Programming, 2008, 70(1): 31-61.

[7] DUAN Z, KOUTNY M A. Framed temporal logic programming language[J]. Journal of Computer Science and Technology, 2004, 19(3): 341-351,

[8] MOSZKOWSKI B. Compositional reasoning about projected and infinite time[C]//Proceedings of the First IEEE International Conference on Engineering of Complex Computer Systems (ICECCS'95). Fort Lauderdale, Florida, USA: IEEE Computer Society Press, 1995.

[9] DUAN Z, TIAN C, ZHANG L. A decision procedure for propositional projection temporal logic with infinite models[J]. Acta Informatica, 2008, 45(1): 43-78.

[10] STOCKMEYER L. The complexity of decision problems in automata theory and logic[D]. Cambridge, Massachusetts, USA: MIT, 1974.

[11] CLARKE E M, GRUMBERG O, PELED D A. Model checking [M]. Massachusetts: MIT Press, 1999.

编 辑 张 俊