Software Engineering and Applications 软件工程与应用, 2015, 4(6), 121-128
Published Online December 2015 in Hans. http://www.hanspub.org/journal/sea
http://dx.doi.org/10.12677/sea.2015.46016

Hans汉斯

Interpolation Algorithm for Quantifier Formulas in LIUF Theory

Qianhui Li, Jianguo Jiang, Wenxiu Liu

School of Mathematics, Liaoning Normal University, Dalian Liaoning
Email: lqh274243982@sina.com, jjgbox@sina.com, 1070957463@qq.com

Received: Dec. 8th, 2015; accepted: Dec. 22nd, 2015; published: Dec. 28th, 2015

Abstract

The problem of interpolation for quantifier formulas is still unsolved in LIUF theory. For the problems of how to eliminate quantifiers and compute the interpolation after eliminating the quantifiers, a new algorithm based on quantifier-free theory interpolation algorithm is proposed. First, the skolemization was used to eliminate existential quantifiers and universal quantifiers were instantiated with free individual variables to create quantifier-free formulas; Then, the quantifier-free theory interpolation algorithm was used to compute the formulas; Finally, the new variables were eliminated by quantifying universally or existentially, and interpolation for quantifier formulas was obtained. The algorithm example shows that the new interpolation algorithm can solve the problem of quantifier formulas in LIUF theory.

Keywords

Theory Interpolation, LIUF Theory, Eliminate Quantifier, Skolemization

LIUF理论量词公式的插值算法

李千卉，江建国，刘文秀

辽宁师范大学数学学院，辽宁 大连
Email: lqh274243982@sina.com, jjgbox@sina.com, 1070957463@qq.com

收稿日期：2015年12月8日；录用日期：2015年12月22日；发布日期：2015年12月28日

文章引用：李千卉，江建国，刘文秀. LIUF 理论量词公式的插值算法[J]. 软件工程与应用, 2015, 4(6): 121-128.
http://dx.doi.org/10.12677/sea.2015.46016

摘　要

量词公式的插值是LIUF理论中一个未解决的问题。针对如何消去量词、消去量词后如何求出公式的插值等问题，提出了一种基于无量词公式理论插值的新算法。首先利用斯科拉姆化消去存在量词，并通过引入新变量消去全称量词，使量词公式变为无量词公式；然后运用已有的LIUF无量词理论公式插值算法求出变换后公式的插值；最后将插值中含有的新变量用存在量词或全称量词替换，从而得到LIUF理论中量词公式的插值。实例表明新算法可以解决LIUF理论中量词公式的插值问题。

关键词

理论插值，LIUF理论，量词消去，斯科拉姆化

1. 引言

1957 年，Craig 首次提出了插值的概念[1]。在此后的研究中，研究者们发现 Craig 插值在命题逻辑、命题可满足性(SAT)问题[2] [3]、无量词一阶逻辑问题[4]、可满足性模理论(SMT)问题[5]、软件模型检测[6]和抽象优化[7] [8]等领域有广泛的应用，因此，其中所涉及的插值的求法成为研究的热点。当前最常用的求插值的方法为 DPLL 法[9]，它因易操作、高效等优点被广泛使用。由 DPLL 的研究可知，它可与理论 T 结合，因此插值的研究可扩展到理论中去。

近些年来，由于实际研究的需要，理论插值的研究越来越广泛。研究的理论包括线性不等式理论[10]、数组理论[11]、未解释函数理论[12]等。但这些都局限于对单个理论的研究，当面对两个或两个以上理论时，就无法高效地求出插值了。本文研究的是两种理论结合下的插值，即线性不等式理论(Linear Inequality，简称 LI)和未解释函数理论(Uninterpreted Functions，简称 UF)结合下的插值，这里将这种结合理论称为 LIUF 理论。LIUF 理论中非协调公式对 (A,B) 的插值 Φ 满足：(1) $A\vdash_T \Phi$，(2) $\Phi\vdash_T \neg B$，(3) Φ 中的符号都是 A,B 公有的。

LIUF 理论在软件模型检测中有十分重要的应用。同时，当已知 LI 理论的插值时，可通过结合理论下的插值算法求出 UF 理论的插值，反之亦然，这大大减少了工作量。然而，现有的工作仅研究了无量词公式在 LIUF 理论中的插值算法，量词公式的插值算法仍是一个未解决的问题。

LIUF 理论量词公式插值求解面临两大问题，即如何消去量词以及如何消去插值中由量词消去引入的新变量。针对所面临的问题，本文提出了 LIUF 理论量词公式的插值算法，并通过实例详细地说明了如何在 LIUF 理论中求出量词公式的插值。

2. LIUF 理论

在 LIUF 理论中，项是单个变量或 $f(t_1,\cdots,t_n)$，这里 $t_1,\cdots,t_n$ 是项，f 是 n 元函数符号。形如 $c_0+c_1t_1+\cdots+c_nt_n$ 的线性组合称为算术项，其中 $t_1,\cdots,t_n$ 是项，$c_0,c_1,\cdots,c_n$ 为整数常量，且 $c_i\neq 0$，$i=1,\cdots,n$。原子谓词指单个命题变量或不等式 $0\le t$ (这里 t 是算术项)或等式 $t=s$ (这里 t,s 是项)。原子谓词本身或原子谓词的否定构成文字，文字的析取构成子句，子句通常用“$\langle\rangle$”表示，将包含文字集 Γ 的子句记作 $\langle\Gamma\rangle$，空子句记作 $\langle\rangle$。

序列是有限公式集的序列对，记为 $\Gamma\vdash\Delta$，其中 Γ,Δ 为公式集(本文中为文字集或子句集)，Γ 为公式的合取，Δ 为公式的析取。现设本文所讨论的插值均有如下形式：$(A,B)\vdash\psi[X]$，其中 (A,B) 为公式对，

括号中的 A,B 也可表示为 $A\wedge B$； X 代表与 ψ 有关的插值，当 ψ 为空子句时， X 就是 (A,B) 的插值，这里的插值以及下文所提到的插值均为 Craig 插值。通常， X 满足： $A\vdash X$ ， $X\wedge B\vdash\psi$ 。需要说明，文中所有的小写字母都代表公式，大写字母都代表公式集。

现给出证明规则，使用这些规则可得到 $(A,B)\vdash\psi$ 的证明过程。

HYP $\dfrac{}{\Gamma\vdash\varphi}\ \varphi\in\Gamma$

COMB $\dfrac{\Gamma\vdash 0\le t\quad\Gamma\vdash 0\le s}{\Gamma\vdash 0\le c_1t+c_2s},\ c_{1,2}>0$

RES $\dfrac{\Gamma\vdash\langle p,\Delta\rangle\quad\Gamma\vdash\langle\neg p,\Delta'\rangle}{\Gamma\vdash\langle\Delta,\Delta'\rangle}$

TRANS $\dfrac{\Gamma\vdash t=s\quad\Gamma\vdash s=m}{\Gamma\vdash t=m}$

CONG $\dfrac{\Gamma\vdash t_1=s_1\cdots\Gamma\vdash t_n=s_n}{\Gamma\vdash f(t_1,\cdots,t_n)=f(s_1,\cdots s_n)}$ (*)

EQNEQ $\dfrac{\Gamma\vdash t=s}{\Gamma\vdash\bot}\ \neg(t=s)\in\Gamma$

LEQEQ $\dfrac{\Gamma\vdash t=s}{\Gamma\vdash 0\le t-s}$

EQLEQ $\dfrac{\Gamma\vdash 0\le t-s\quad\Gamma\vdash 0\le s-t}{\Gamma\vdash t=s}$

本文中(*)表示出现在“$\vdash$”右侧的符号也出现在其左侧。

例 设 $A=\{0\le x-a,0\le y-b\}$， $B=\{0\le a-y,0\le b-x\}$， 试写出 $(A,B)\vdash x=y$ 的证明过程。

1 $A\vdash 0\le x-a$ (HYP)　　2 $A\vdash 0\le y-b$ (HYP)

3 $B\vdash 0\le a-y$ (HYP)　　4 $B\vdash 0\le b-x$ (HYP)

5 $A\wedge B\vdash 0\le x-y$ (COMB1, 3)　　6 $A\wedge B\vdash 0\le y-x$ (COMB2, 4)

7 $A\wedge B\vdash x=y$ (EQLEQ5, 6)

3. LIUF 无量词理论公式的插值

上节介绍了 $(A,B)\vdash\psi$ 的证明过程，而插值正是从这个证明过程中推出的，那么如何从中提取出插值呢？这正是本节所要讨论的内容：介绍 LIUF 理论中三个无量词公式理论插值的定义，引入插值规则，讨论插值的求法。

定义 1(不等式插值)将形如 $(A,B)\vdash 0\le t[t',\rho,\gamma]$ 序列的称为不等式插值，其中 A,B 为文字集， t,t' 为项， ρ,γ 为公式。它是有效的，满足：

(1) $A\wedge\rho\vdash 0\le t'\wedge\gamma$；

(2) $B\vdash\rho$ 且 $B\wedge\gamma\vdash 0\le t-t'$；

(3) $\rho,\gamma\prec B$， $t',\rho,\gamma\prec A$， $(t-t')\prec B$ 。

这里，t' 为插值。符号“$\prec$”在 $\varphi\prec\Gamma$ 中表示 φ 中所有的变量和未解释函数符号都出现在 Γ 中；在 $x\prec B$ 中表示 x 中所有的变量和未解释函数符号都出现在 B 中，此时称 x 是全局项；否则，称 x 是局部项。在不等式理论中，ρ,γ 总为 $\top$。$0\le t$ 表示 A,B 中不等式的线性组合，t' 表示对 A 中不等式相加后得到的结果。

定义 2(子句插值)将形如 $(A,B)\vdash\langle\Delta\rangle[\varphi]$ 的序列称为子句插值，这里 A,B 是子句集，Δ 是文字集，ϕ 是公式。它是有效的，满足：

(1) $A\vdash\varphi\vee\langle\Delta\backslash B\rangle$；

(2) $B\wedge\varphi\vdash\langle\Delta\downarrow B\rangle$；

(3) $\varphi\prec B,\varphi\prec A$。

这里， $\Delta\downarrow B$ 表示 Δ 中文字的原子谓词出现在 B 中， $\Delta\backslash B$ 表示 Δ 中文字的原子谓词不出现在 B 中。在子句插值中， φ 就是插值。

定义 3(等式插值)将形如 $(A,B)\vdash t=s[t',s',\rho,\gamma]$ 的序列称为等式插值，其中 A,B 是文字集，t,s,t',s' 为

项， ρ,γ 为公式。它是有效的，满足：

(1) $A\wedge\rho\vdash t=t'\wedge s=s'\wedge\gamma$；

(2) $B\vdash\rho$

①$t'\approx s, s'\approx t$；

②$B\wedge\gamma\vdash t'=s'$；

(3) $\rho,\gamma\prec B$， $\rho,\gamma\prec A$，如果 $t\prec B$，那么 $t'\approx t$，否则 $t'\prec A$， s,s' 同理。

这里，γ 为插值。为了得到 $t=s$，首先建立 A,B 的等式链 $(t=x_1)(x_1=x_2)\cdots(x_n=s)$，等式链中的等式来自于 A 或 B，设 t' 代表等式中最左侧的全局项，s' 代表等式中最右侧的全局项。当等式中有全局项时，由 A 可推出 $t=t'$，$s=s'$；当等式中无全局项时，有 $t'\approx s$，$s'\approx t$。在 TRANS 规则中，ρ 总为⊤。

下面介绍几种插值规则，运用这些规则，就可以得到插值。

首先介绍前提(HYP)规则：

HYP-A $\dfrac{}{(A,B)\vdash 0\le t[t,\top,\top]}\ (0\le t)\in A$ **HYP-B** $\dfrac{}{(A,B)\vdash 0\le t[0,\top,\top]}\ (0\le t)\in B$

HYPC-A $\dfrac{}{(A,B)\vdash\langle\Delta\rangle[\langle\Delta\downarrow B\rangle]}\ \langle\Delta\rangle\in A$ **HYPC-B** $\dfrac{}{(A,B)\vdash\langle\Delta\rangle[\top]}\ \langle\Delta\rangle\in B$

HYPEQ-A $\dfrac{}{(A,B)\vdash t=s['(t,s),(t,s)',\top,(t=s)|_B]}\ (t=s)\in A$

其中 $'(t,s)=\begin{cases}t, & t\prec B\\ s, & \text{否则}\end{cases}$ $(t,s)'=\begin{cases}s, & s\prec B\\ t, & \text{否则}\end{cases}$，$(t=s)|_B=\begin{cases}t=s, & (t=s)\in B\\ T, & \text{否则}\end{cases}$

HYPEQ-B $\dfrac{}{(A,B)\vdash t=s[t,s,\top,\top]}\ (t=s)\in B$

HYP 规则应用在插值算法的最开始，主要作用是将 A,B 中的公式写成 $(A,B)\vdash\psi[X]$ 的形式，方便下面的运算。

下面介绍其余插值规则，这些规则的使用并无固定顺序，可根据实际需要选择恰当的规则计算插值。

COMB $\dfrac{(A,B)\vdash 0\le t[t',\rho,\gamma]\quad(A,B)\vdash 0\le s[s',\rho',\gamma']}{(A,B)\vdash 0\le c_1t+c_2s[c_1t'+c_2s',\rho\wedge\rho',\gamma\wedge\gamma']}$， $c_{1,2}>0$

RES-A $\dfrac{(A,B)\vdash\langle p,\Delta\rangle[\varphi]\quad(A,B)\vdash\langle\neg p,\Delta'\rangle[\varphi']}{(A,B)\vdash\langle\Delta,\Delta'\rangle[\varphi\vee\varphi']}$， p 不出现在 B 中

RES-B $\dfrac{(A,B)\vdash\langle p,\Delta\rangle[\varphi]\quad(A,B)\vdash\langle\neg p,\Delta'\rangle[\varphi']}{(A,B)\vdash\langle\Delta,\Delta'\rangle[\varphi\wedge\varphi']}$， p 出现在 B 中

TRANS1 $\dfrac{\begin{array}{c}(A,B)\vdash t=s[t',s',\rho,\gamma]\\ (A,B)\vdash s=m[s'',m',\rho',\gamma']\end{array}}{(A,B)\vdash t=m[t',m',\rho\wedge\rho',\gamma\wedge\gamma'\wedge s'=s'']}$， $t'\not\approx s$ 且 $m'\not\approx s$

TRANS2 $\dfrac{\begin{array}{c}(A,B)\vdash t=s[t',s',\rho,\gamma]\\ (A,B)\vdash s=m[s'',m',\rho',\gamma']\end{array}}{(A,B)\vdash t=m[t'(s''/s),m'(s'/s),\rho\wedge\rho',\gamma\wedge\gamma']}$， $t'\approx s$ 或 $m'\approx s$

其中 $t'(s''/s)=\begin{cases}s'', & t'\approx s\\ t', & \text{否则}\end{cases}$， $m'(s'/s)=\begin{cases}s', & m'\approx s\\ m', & \text{否则}\end{cases}$

CONG1 $\dfrac{(A,B)\vdash t=s[t',s',\rho,\gamma]}{(A,B)\vdash f(t)=f(s)[f(t'),f(s'),\rho,\gamma]}$ (*)， $f(t)\prec B$ 或 $f(s)\prec B$

CONG2 $\dfrac{(A,B)\vdash t=s[t',s',\rho,\gamma]}{(A,B)\vdash f(t)=f(s)[f(s),f(t),\rho\wedge(\gamma\Rightarrow(t'=s')|_B),\gamma]}$ (*)， $f(t)\not\prec B, f(s)\not\prec B$

LEQEQ1 $\dfrac{(A,B)\vdash t=s[t',s',\rho,\gamma]}{(A,B)\vdash 0\le t-s[t-t'-s+s',\rho,\gamma]}$， $t'\not\approx s$ 或 $s'\not\approx t$

LEQEQ2 $\dfrac{(A,B)\vdash t=s[s,t,\rho,\gamma]}{(A,B)\vdash 0\le t-s[t-s,\rho,\gamma]}$， $t'\approx s$， $s'\approx t$

EQLEQ-BB $\dfrac{\begin{array}{c}(A,B)\vdash 0\le t-s[t',\rho,\gamma]\\ (A,B)\vdash 0\le s-t[s',\rho',\gamma']\end{array}}{(A,B)\vdash t=s[t,s,\rho\wedge\rho',\gamma\wedge\gamma'\wedge 0\le t'\wedge 0\le s']}$ (*)， $t\prec B$， $s\prec B$

EQLEQ-AB $\dfrac{\begin{array}{c}(A,B)\vdash 0\le t-s[t',\rho,\gamma]\\ (A,B)\vdash 0\le s-t[s',\rho',\gamma']\end{array}}{(A,B)\vdash t=s[t+s',s,\rho\wedge\rho'\wedge 0\le -t'-s',\gamma\wedge\gamma'\wedge 0\le t'+s']}$ (*)， $t\not\prec B$， $s\prec B$

EQLEQ-AA $\dfrac{\begin{array}{c}(A,B)\vdash 0\le t-s[t',\rho,\gamma]\\ (A,B)\vdash 0\le s-t[s',\rho',\gamma']\end{array}}{(A,B)\vdash t=s[s,t,\rho\wedge\rho'\wedge 0\le s-t-s'\wedge 0\le t-s-t',\gamma\wedge\gamma']}$ (*)， $t\not\prec B$， $s\not\prec B$

4. LIUF 理论量词公式的插值算法

4.1. 面临的挑战

通过上一节的介绍可以很容易证明出 LIUF 理论无量词公式插值系统的完备性和可靠性。但在某些情况下，无量词公式插值通常是不存在的。例如：$A=\{x=3y+1\}$，$B=\{x=z\}$，(A,B) 的插值为“存在 x”，这是量词的表达方法。因此，对于单纯的整数域上的 LIUF 理论无量词公式，并不能得到完备的系统，这就需要研究带有量词公式的 LIUF 理论插值系统。

LIUF 理论量词公式的插值求解过程中，如何消去量词成为首个要面临的问题。量词包括存在量词和全称量词。存在量词可用斯科拉姆化法[13]消去，这是很容易做到的；但消去全称量词就有些复杂了，全称量词的消去过程会引入新的变量，那么新变量的选择就变得至关重要，如果选择不当，就会增加不必要的计算量，而且当公式中同时含有存在量词和全称量词时，必须考虑先消去哪种量词，这关系到整个插值过程的工作量。

当消去量词后，原公式变为无量词公式，然后求出无量词公式的插值。应注意：消去量词后得到的公式可用无量词公式插值算法计算。但此时求出的插值并不是所要求的量词公式的插值，因为其中含有由量词消去所引入的新变量，这时需要考虑如何消去新变量，且保证不会影响插值的整体形式。

针对上述问题，要求出 LIUF 理论量词公式的插值，需要满足：

1) 确定消去存在量词和全称量词的顺序和方法；

2) 找到无量词公式插值中引入的新变量的消去方法。

4.2. LIUF 理论量词公式插值算法

对于上述问题，我们给出 LIUF 理论量词公式插值算法。该算法可以解决上节所提出的两个问题，并能成功求出其插值。对于量词公式对 (A,B)，若仅出现一种量词，就用相应的方法消去量词，若两种

量词同时出现，则先消去存在量词，然后再消去全称量词，并将公式变为无量词的情况(A',B')，然后通过已有的插值算法先求出(A',B')的插值Φ'，再由文献[1]，将Φ'中由量词消去引入的新变量用存在或全称量词替换，就得到了(A,B)的插值。算法的具体过程如算法 1 所示。

算法 1. LIUF 理论量词公式的插值算法

输入：量词公式 A,B。

输出：量词公式 A,B 的插值 Φ。

步骤 1 对量词公式 A,B，如果公式中仅出现存在量词，那么用斯科拉姆化消去存在量词；如果公式中仅出现全称量词，那么先化为全称前束范式；如果公式中既出现存在量词又出现全称量词，那么先消去存在量词，然后再把全称量词变为全称前束范式。

步骤 2 引入新变量 v_t 消去全称量词，并适当在 A 或 B 中加入等式 $v_t=t$，从而将出现在 A 或 B 中带有量词的项 t 实例化，把公式对 (A,B) 化为不含量词的公式对 (A',B')。

步骤 3 运用 LIUF 理论无量词公式插值算法计算实例化公式对 (A',B') 的插值 Φ'。

步骤 4 将 Φ' 中的 v_t 用量词替换。若 v_t 不在 A 中，则用全称量词替换；若 v_t 不在 B 中，则用存在量词替换。替换后的结果 Φ 则为 (A,B) 的插值。

4.3. 算法实例

使用 LIUF 理论量词公式插值算法可以求出量词公式的插值。下面以具体例子进行说明。

例 设 $A=\{(\forall x)[(0\le x-a)\wedge(\exists y)(0\le y-b-1)\wedge(\exists y)(0\le -y)]\}$，

$B=\{(\forall x)[(\exists y)(0\le a-y)\wedge(0\le b-x)\wedge(\exists y)(0\le y)]\}$，求 (A,B) 的插值 Φ。

1) 用斯科拉姆化法消去存在量词，并把公式变为全称前束范式。

$$A=\{(\forall x)[(0\le x-a)\wedge(0\le f(x)-b-1)\wedge(0\le -f(x))]\};$$

$$B=\{(\forall x)[(0\le a-g(x))\wedge(0\le b-x)\wedge(0\le g(x))]\}$$

2) 消去全称量词，将 A,B 中的项 x 实例化，分别引入新变量 v_1,v_2 和等式 $x=v_1,v_2=x$，将量词公式对 (A,B) 化为无量词公式对 (A',B')。

$$A'=\{0\le v_1-a,0\le f(v_1)-b-1,0\le -f(v_1),x=v_1\};$$

$$B'=\{0\le a-g(v_2),0\le b-v_2,0\le g(v_2),v_2=x\};$$

3) 由 LIUF 理论无量词公式的插值算法求出 (A',B') 的插值 Φ'。

具体步骤如下：

(1) $(A',B')\vdash 0\le v_1-a[v_1-a,\top,\top]$ (HYP-A)

(2) $(A',B')\vdash 0\le f(v_1)-b-1[f(v_1)-b-1,\top,\top]$ (HYP-A)

(3) $(A',B')\vdash 0\le -f(v_1)[-f(v_1),\top,\top]$ (HYP-A)

(4) $(A',B')\vdash x=v_1[x,x,\top,\top]$ (HYPEQ-A)

(5) $(A',B')\vdash 0\le a-g(v_2)[0,\top,\top]$ (HYP-B)

(6) $(A',B')\vdash 0\le b-v_2[0,\top,\top]$ (HYP-B)

(7) $(A',B')\vdash 0\le g(v_2)[0,\top,\top]$ (HYP-B)

(8) $(A',B')\vdash v_2=x[v_2,\mathrm{x},\top,\top]$ (HYPEQ-B)

(9) $(A',B')\vdash 0\le v_1-g(v_2)[\mathrm{v}_1-\mathrm{a},\top,\top]$ (COMB1, 5)

(10) $(A', B') \vdash 0 \le f(v_1) - v_2 - 1[f(v_1) - b - 1, \top, \top]$ (COMB2, 6)

(11) $(A', B') \vdash 0 \le x - v_1[x - v_1, \top, \top]$ (LEQEQ2 4)

(12) $(A', B') \vdash 0 \le v_2 - x[0, \top, \top]$ (LEQEQ1 8)

(13) $(A', B') \vdash 0 \le x - g(v_2)[x - a, \top, \top]$ (COMB9, 11)

(14) $(A', B') \vdash 0 \le f(v_1) - x - 1[f(v_1) - b - 1, \top, \top]$ (COMB10, 12)

(15) $(A', B') \vdash 0 \le f(v_1) - g(v_2) - 1[f(v_1) - b + x - a - 1, \top, \top]$ (COMB13, 14)

(16) $(A', B') \vdash 0 \le g(v_2) - f(v_1)[-f(v_1), \top, \top]$ (COMB3, 7)

(17) $(A', B') \vdash 0 \le -1[x - a - b - 1, \top, \top]$ (COMB15, 16)

所以 Φ' 为 $0 \le x - a - b - 1$。

4) 因为 Φ' 中不含有新变量，所以不需要将变量替换为量词。

所以，(A, B) 的插值 Φ 为 $0 \le x - a - b - 1$。

有了上述算法，无量词公式的插值规则都可以应用在带有量词公式上，这就使插值有了更广泛的应用。

5. 结语

本文在无量词理论插值算法的基础上，介绍了 LIUF 理论量词公式的插值算法，解决了之前无法求解量词公式插值的问题，从而使插值有了更为广泛的应用。但这种方法只能应用在已知的无量词插值算法的基础上，并不能独立使用，进一步的工作是要改进出适用范围更广的插值算法，这将使插值计算更加智能化和高效化。

参考文献 (References)

[1] Craig, W. (1957) Three Uses of the Herbrand-Gentzen Theorem in Relating Model Theory and Proof Theory. *The Journal of Symbolic Logic*, **22**, 269-285. http://dx.doi.org/10.2307/2963594

[2] McMillan, K.L. (2003) Interpolation and SAT-Based Model Checking. In: *CAV* 2003: 2003 *Proceedings of the* 15*th Conference on Computer Aided Verification, Lecture Notes in Computer Science*, Springer, Berlin, 1-13.

[3] Weissenbacher, G. (2010) Program Analysis with Interpolants. Magdalen College, Oxford University, Oxford, England, 277.

[4] Henzinger, T.A., Jhala, R., Majumdar, R., *et al.* (2004) Abstractions from Proofs. *ACM SIGPLAN Notices*, **39**, 232-244.

[5] Bruttomesso, R., Rollini, S.F., Sharygina, N., *et al.* (2010) Flexible Interpolation Generation in Satisfiability Modulo Theories. *ICCAD* 2010: 2010 *Proceedings of the* 14*th International Conference on Computer-Aided Design*, IEEE, Piscataway, 770-777.

[6] 陈祖希, 徐中伟, 霍伟伟, 等. 基于 Craig 插值的线性混成系统符号化模型检测[J]. 电子学报, 2014(7): 1338-1346.

[7] McMillan, K.L., Ball, T. and Jones, R.B. (2006) Lazy Abstraction with Interpolations. In: *CAV* 2006: 2006 *Proceedings of the* 18*th Conference on Computer Aided Verification*, Springer, Berlin, 123-136.

[8] 屈婉霞, 李暾, 郭阳, 杨晓东. 模型检验中抽象技术研究综述[J]. 计算机工程与应用, 2006, 42(33): 15-19.

[9] Nieuwenhuis, R., Oliveras, A. and Tinelli, C. (2006) Solving SAT and SAT Modulo Theories: From an Abstract Davis-Putnam-Logemann-Loveland procedure to DPLL(T). *Journal of the ACM*, **53**, 937-977.

[10] Pudlak, P. (1997) Lower Bounds for Resolution and Cutting Plane Proofs and Monotone Computations. *Mathematical Institute Academy of Sciences of Czech Republic*, **62**, 1-20. http://dx.doi.org/10.2307/2275583

[11] Alberti, F., *et al.* (2014) An Extension of Lazy Abstraction with Interpolation of Programs with Arrays. *Formal Methods in System Design*, **45**, 63-109. http://dx.doi.org/10.1007/s10703-014-0209-9

[12] Lopes, N.P. and Monteiro, J. (2015) Automatic Equivalence Checking of Programs with Uninterpreted Functions and Integer Arithmetic. *International Journal on Software Tools for Technology Transfer*, Springer, Science, 1-16.

[13] Fitting, M. (1996) First-Order Logic and Automated Theorem Proving. Springer, Berlin. http://dx.doi.org/10.1007/978-1-4612-2360-3