

基于目标演绎距离的一阶逻辑子句集预处理方法



曹 锋^{1,3} 徐 扬^{2,3} 钟 建^{1,3} 宁欣然^{1,3}

1 西南交通大学信息科学与技术学院 成都 611756

2 西南交通大学数学学院 成都 611756

3 系统可信性自动验证国家地方联合工程实验室 成都 610031

摘 要 一阶逻辑定理证明是人工智能的核心基础,研究一阶逻辑自动定理证明器的相关理论和高效的算法实现具有重要的学术意义。当前一阶逻辑自动定理证明器首先通过子句集预处理约简子句集规模,然后通过演绎方法对定理进行判定。现有的应用于证明器中的子句集预处理方法普遍只从与目标子句项符号相关性角度出发,不能很好地从文字的互补对关系中体现子句间的演绎。为了在子句集预处理时从演绎的角度刻画子句间的关系,定义了目标演绎距离的概念并给出了计算方法,提出了一种基于目标演绎距离的一阶逻辑子句集预处理方法。首先对原始子句集进行包含冗余子句约简并应用纯文字删除规则,然后根据目标子句计算剩余子句集中的文字目标演绎距离、子句目标演绎距离,并最终通过设定子句演绎距离阈值来实现对子句集的进一步预处理。将该预处理方法应用于顶尖证明器 Vampire,以 2017 年国际一阶逻辑自动定理证明器标准一阶逻辑问题组竞赛例为测试对象,在标准的 300 s 内,加入提出的子句集预处理方法的 Vampire4.1 相比原始的 Vampire4.1 多证明 4 个定理,能证明 10 个 Vampire4.1 未证明的定理,占其未证明定理总数的 13.5%;在证明的定理中,提出的子句集预处理方法能对 77.2%的子句集进行约简,最大子句集约简规模达到 51.7%。实验结果表明,提出的一阶逻辑子句集预处理方法是一种有效的方法,能有效地约简一阶逻辑子句集的规模,提高一阶逻辑自动定理证明器的证明能力。

关键词: 一阶逻辑;人工智能;子句集预处理;演绎距离;冗余子句

中图法分类号 TP311

First-order Logic Clause Set Preprocessing Method Based on Goal Deduction Distance

CAO Feng^{1,3}, XU Yang^{2,3}, ZHONG Jian^{1,3} and NING Xin-ran^{1,3}

1 School of Information Science and Technology, Southwest Jiaotong University, Chengdu 611756, China

2 School of Mathematics, Southwest Jiaotong University, Chengdu 611756, China

3 National-Local Joint Engineering Laboratory of System Credibility Automatic Verification, Chengdu 610031, China

Abstract The first-order logic theorem proving is the core foundation of artificial intelligence, it has great academic significance to study the theory and efficient algorithm implementation for first-order logic automated theorem provers. The current provers adopt clause set preprocessing approaches to reduce clause set scale, then apply inference rules to prove theorems. The existing clause set preprocessing method used in provers is generally only from the perspective of the semantic relevance to axioms and conjecture, and it can't reflect the deduction between clauses well from the complementary pairs of literals point of view. In order to describe the relationship between clauses from the perspective of deduction, the definition of goal deduction distance and the calculation method are proposed, and a first-order logic clause set preprocessing method is presented based on the proposed goal deduction distance. Firstly, the original clause set is applied with redundant clause simplification and pure literal deletion rule. Then, taking the goal clauses into consideration, the literal goal deduction distance and the clause goal deduction distance are calculated, and finally further clause set preprocessing is realized by the setting threshold of clause deduction distance. We implement the proposed clause set preprocessing method in Vampire that is the winner of the 2017 CADE ATP System Competition (CASC-26) FOF division, and apply it to solve the CASC-26 problem. Within the standard 300 seconds, the top prover Vampire4.1 with the proposed clause set preprocessing method outperforms Vampire4.1 by solving 4 theorems more than Vampire4.1, and 10 out of the 74 problems unsolved by Vampire4.1, accounting for 13.5% of the total. The proposed clause set preprocessing method

到稿日期:2019-01-02 返修日期:2019-03-05 本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。

基金项目:国家自然科学基金(61673320);中央高校基本科研业务费专项资金(2682017ZT12,2682018CX59,2682018ZT25)

This work was supported by the National Natural Science Foundation of China (61673320) and Fundamental Research Funds for the Central Universities (2682017ZT12,2682018CX59,2682018ZT25).

通信作者:曹锋(caofeng19840301@163.com)

has affected 77.2% of the solved theorems, and the largest reduction proportion is 51.7%. Experimental results show that the proposed first-order logic clause set preprocessing method is an effective method, which can effectively reduce the clause set scale and improve the ability of first-order logic automated theorem prover.

Keywords First-order logic, Artificial intelligence, Clause set preprocessing, Deduction distance, Redundant clause

1 引言

自动推理是人工智能的重要领域,一阶逻辑其中证明器是其中的重要研究方向。经过十多年的发展,一阶逻辑自动定理证明器(如 Vampire^[1-2], Eprover^[3-4], iProver^[5]等)已具有较强的能力和较高的效率。相比命题逻辑,一阶逻辑具有丰富的表达能力,现实中的许多问题更容易用一阶逻辑表示,因此一阶逻辑自动定理证明器已被广泛应用于数学定理证明、程序验证、电路设计与科学管理^[6-7]等领域。对于一阶逻辑表述的大理论问题和大规模的实际问题,预处理技术已成为一阶逻辑自动定理证明的研究热点。

2 相关工作

当前的一阶逻辑自动定理证明器中,子句集预处理技术主要包括冗余子句处理和基于目标子句的子句约简方法。冗余子句处理是一种不改变子句集属性的方法。例如,文献[8]定义了纯文字的概念,即该文字在当前子句集中没有可以归结的文字,删除含有纯文字的子句不影响判定的完备性;文献[9-10]给出了一种简化子句集的方法,即利用当前子句集对当前选定的子句进行包含冗余处理,利用当前子句对整个子句集进行包含冗余子句约简;文献[11]提出了一种谓词消去方法,通过定义自参考谓词和非自参考谓词实现子句消去的目的。基于目标子句的子句集约简是一种可能会改变子句集属性,但能有效提高自动演绎效率的方法。文献[12]通过标记与目标子句符号相关的子句,并只使用已标记的子句进行推理;文献[13]表明证明器在判定实际问题时,其效率与目标子句紧密相关,并介绍了一种基于软件规范子句集约简方法;文献[14]介绍了一种根据子句与目标子句之间的语义进行相关性度量的子句集约简方法;文献[15]介绍了一种描述子句与目标子句中项符号的相关性度量方法,并通过设置阈值对子句集进行约简。

上文介绍的基于目标子句的子句约简方法有效地提高了一阶逻辑自动定理证明器的能力,但这些方法都是基于项符号来度量子句与目标子句的关系,项符号相同或相似的子句之间并不表示存在文字间的合一互补,不能较好地体现子句之间的演绎关系。本文从子句演绎的角度出发,为了较大幅度地对子句集进行约简,首先对子句集进行冗余处理,通过度量子句演绎距离并设置阈值,来对剩余子句集做进一步的预处理;将该预处理方法应用于顶级证明器 Vampire4.1 中,有效地提高了 Vampire4.1 证明器的能力。

3 一阶逻辑与目标演绎距离

3.1 一阶逻辑

一阶逻辑原子公式被称为文字,包括谓词项、函数项、变元项、常元项。子句由文字的析取构成,只含有 1 个文字的子

句称为单元子句;不含文字的子句称为空子句,记作 $\emptyset$ 。多个子句的合取构成了一阶逻辑公式的子句集,子句集中包含了前提和结论的否定,结论的否定称为目标子句。证明一阶逻辑公式不可满足,即应用推理规则寻找演绎到空子句的证明路径。

3.2 目标演绎距离

定义 1 目标子句中的文字称为目标文字,子句集中的其他文字称为前提文字。除目标子句外,子句集中的其他子句称为前提子句。

定义 2 文字与目标文字演绎的距离称为文字目标演绎距离。

文字目标演绎距离度量文字与目标文字之间的演绎关系,用于刻画文字与目标文字演绎的难易程度,文字目标演绎距离越小则越容易与目标文字进行演绎。一阶逻辑子句集在判定过程中是通过文字进行演绎的,且演绎目标生成空子句(不含文字)。为了有效体现一阶逻辑子句演绎的特点,按以下两种方式计算文字目标演绎距离。

(1)文字 P_2 和文字 P_1 (P_1 已计算出目标演绎距离)可形成合一互补对。若文字 P_1 的目标演绎距离为 k ,则演绎文字 P_2 的目标演绎距离为 $k+1$ 。

(2)同一子句中若有文字已计算出目标演绎距离值,则该子句中的其他未计算出目标演绎距离的文字也赋予相同的目标演绎距离值。

计算文字演绎距离产生如下冲突:文字计算目标演绎距离满足上述两种方法,即存在文字可以与已计算出目标演绎距离的文字直接演绎,同时同一子句的其他文字也已计算出目标演绎距离。

为了使目标演绎距离小的文字具有优先选取权,取两种计算方法的最小值作为该文字的目标演绎距离值。

定义 3 子句与目标子句演绎的距离称为子句目标演绎距离。

子句目标演绎距离用于度量与目标子句的演绎关系,演绎距离值越小则越容易与目标子句进行演绎。

4 基于目标演绎距离的一阶逻辑子句集预处理方法与应用

4.1 冗余子句处理

当前一阶逻辑自动定理证明器普遍采用 Given-clause 浸透演绎框架。判断子句是否为冗余子句时需要遍历子句集,算法上执行冗余检测需要消耗一定的时间;但从提升证明器能力方面考虑,能以较小的时间代价在标准时间内证明出更多的定理,冗余检测的时间消耗是可取的。本文以提高定理证明能力为目标,为了较大幅度地实现原始子句集的约简,并减小演绎的搜索空间,首先使用完备性方法对原始子句集进行包含冗余子句处理和纯文字处理。

包含冗余子句表示该子句在语法上被子句集中的其他子句包含^[3],删除该子句不影响子句集判定的完备性。包含冗余子句处理算法的描述如下:

(1)设子句 C_A 的文字数为 M ,记录文字列表下标值 i (i 从 0 开始),取第 i 个文字 P_i 。

(2)在子句集中查找与文字 P_i 有相同谓词的文字,并将查找到的文字加入文字列表 L_1 中。

(3)遍历文字列表 L_1 ,记录文字列表下标值 j (j 从 0 开始),取第 j 个文字 Q_j ,标记文字 Q_j 所在的子句 C_B 。

(4)判定文字 P_i 是否被子句 C_B 中的文字包含。若存在包含关系,则记录子句 C_B 中的文字列表 L_2 ,并转到步骤(5);否则文字列表下标值 i 加 1,重复步骤(4),直到子句 C_A 中所有的文字被遍历完。

(5)判定文字列表 L_2 是否包含子句 C_B 的所有文字。若包含所有文字,则算法退出,子句 C_A 为冗余子句,删除该子句;否则文字列表下标值 j 加 1,转到步骤(3)。

纯文字表示的是该文字在子句集中不含合一互补的文字^[8],即该文字不在演绎路径上,删除该文字所在的子句不影响子句集判定的完备性。纯文字处理算法的描述如下:

(1)遍历子句集,记录子句列表下标值 i (i 从 0 开始),取第 i 个子句 C_i ,转到步骤(2);若子句集已遍历完,则算法退出。

(2)记录文字列表下标值 j (j 从 0 开始),依次取 C_i 中的第 j 个文字 Q_j 。若 C_i 中文字已遍历完,则转到步骤(4),否则转到步骤(3)。

(3)在子句集中判断 Q_j 是否存在互补对文字。若存在,则下标值 j 加 1,转到步骤(2);否则该文字为纯文字,子句 C_i 按纯文字规则进行删除操作,下标值 i 加 1,转到步骤(1)。

(4)该子句不含纯文字,下标值 i 加 1,转到步骤(1)。

4.2 演绎距离算法设计

4.2.1 文字目标演绎距离算法

一阶逻辑子句集的演绎通过子句中的文字寻找合一互补文字,最终得到演绎归结式的过程。文字是演绎的基本单位,因此有必要从演绎的角度刻画文字的演绎属性。文字目标演绎距离算法首先提取目标文字加入到文字列表 L_1 中,通过遍历 L_1 中的文字,在子句集中寻找该文字可合一互补的文字,计算合一互补文字及所在子句中其他文字的目标演绎距离,并将已计算目标演绎距离的所有文字加入 L_1 中,直到文字列表 L_1 遍历完成。文字目标演绎距离计算算法如算法 1 所示。

算法 1 文字目标演绎距离计算算法

输入:存放目标文字的文字列表 unifyLit

输出:unifyLit 中 goalDepth 字段

```
1. int resLen=0; //相同演绎距离的文字总数
2. int resLength=0; //文字演绎距离变量
3. for(int i=0; i<unifyLit.size(); ++i){
    //计算文字的目标演绎距离
4.    unifyLit[i]->goalDepth=resLength;
5.    if(i==resLen){ //距离加 1
6.        ++resLength;
7.        resLen=unifyLit.size();
8.    }
    ...
```

```
9.    vCandLit=unifyLit[i]->getPairLst();
10.   for (UINT32 j=0; j< vCandLit->size(); ++j) {
11.       candLitP=vCandLit->at(pasLitInd);
12.       candClaP=candLitP->claPtr;
13.       //保存互补对文字对应子句中的所有文字
14.       for(int m=0; m<candClaP->uLitNum; ++m){
15.           if(notInVec(candClaP->LitPtr[m])){
16.               saveVec(candClaP->LitPtr[m]);
17.           }
18.       }
19.   }
20. }
```

4.2.2 子句目标演绎距离算法

根据文字目标演绎距离,使得含文字目标演绎距离小的子句能优先进行演绎。子句目标演绎距离算法的描述如下:

(1)遍历原始子句集,记录子句列表下标值 i (i 从 0 开始),取第 i 个子句 C_i 。

(2)依次取 C_i 中文字的目标演绎距离值,得到最小文字目标演绎距离值 d_1 和最小正整数文字目标演绎距离值 d_2 。若 d_2 不存在,即 C_i 中所有文字的目标演绎距离值为负值,则转到步骤(3)。

1)若 d_1 为正数,将 d_1 的值作为该子句的目标演绎距离值,表明该含有最小目标演绎距离的文字存在于基于目标子句演绎的搜索路径上;

2)若 d_1 为负数,表明该含有最小目标演绎距离的文字不再在基于目标子句演绎的搜索路径上,将 d_2 的值作为该子句的目标演绎距离值。

(3)子句 C_i 中所有文字都不在基于目标子句演绎的搜索路径上,此时 d_1 为负值,将 d_1 的值作为该子句的目标演绎距离值。

4.2.3 基于目标演绎距离的一阶逻辑子句集预处理算法

区别于当前一阶逻辑子句集预处理采用项符号进行相关性度量,基于目标演绎距离的子句集预处理从子句演绎的角度进行刻画。为了更大程度实现对子句集的约简,本文提出的预处理方法包含了完备的方法(包含冗余子句删除与纯文字处理)和不完备的方法(基于子句目标演绎距离)。其子句集预处理算法的描述如下:

(1)将一阶逻辑公式句子化,对其解析并进行有效存;

(2)对子句集应用包含冗余子句处理;

(3)对剩余子句集应用纯文字规则进行处理;

(4)根据文字目标演绎算法计算剩余子句集中所有的文字演绎距离;

(5)根据子句目标演绎算法计算剩余子句集中所有的子句演绎距离;

(6)设置子句演绎距离阈值;

(7)遍历剩余子句集,基于子句目标演绎距离阈值对子句集做进一步的预处理。

4.2.4 基于目标演绎距离的一阶逻辑子句集预处理的应用

为了验证本文提出的子句集预处理方法的有效性,将该预处理方法应用于 Vampire——一款先进的基于归结方法的一阶逻辑自动定理证明器,即在标准的判定时间内,首先利用

本文提出的子句集预处理算法对原始子句集进行约简,并利用 Vampire 对约简后的剩余子句集进行判定。将加入了本文提出的子句集预处理算法的 Vampire4.1 证明器记作 Vampire4.1_P(Vampire4.1 Preprocessing)。

5 实验与分析

5.1 实验准备

对 2017 年 CADE 竞赛例(FOF 组,使用标准格式转换工具 TPTP2X 成功转换 443 个 TPTP 格式)进行测试,测试硬件环境为 3.6GHz Inter(R) Core(TM) i7-4790 处理器,16GB 内存,操作系统为 Ubuntu15.04 64 位,每个定理的判定时间限定为 300s(竞赛标准时间)。设置的子句目标演绎距离阈值为 7,并将本文提出的预处理方法应用于 Vampire4.1。为了与 Vampire4.1 进行比较,在同一软硬件环境下对 Vampire4.1 进行测试并得到了实验数据(Vampire4.1 证明定理总数为 369 个)。

5.2 Vampire4.1_P 判定定理情况

5.2.1 Vampire4.1_P 能力分析

不同于 Vampire4.1, Vampire4.1_P 加入了本文提出的子句集预处理方法,能从演绎的角度对原始子句集进行较大程度的约简。图 1 为 Vampire4.1_P 与 Vampire4.1 的判定情况对比图。从图 1 可得,在 100 s 时, Vampire4.1_P 和 Vampire4.1 证明定理总数相同,为 351 个;在 100 s 后, Vampire4.1_P 证明定理总数超过了 Vampire4.1;在 200 s 时, Vampire4.1_P 证明定理为 366 个,超过 Vampire4.1 证明定理总数 5 个;在 300 s 时, Vampire4.1_P 证明定理 373 个,超出 Vampire4.1 证明定理总数 4 个。实验表明, Vampire4.1_P 证明定理的能力优于 Vampire4.1,本文提出的一阶逻辑子句集预处理方法是一种有效的方法,能有效提高顶尖证明器 Vampire4.1 的能力。

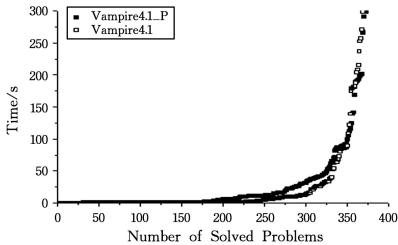


图 1 Vampire4.1_P 与 Vampire4.1 的判定情况对比
Fig. 1 Comparison on solved theorems by Vampire4.1_P and Vampire4.1

在方法的有效性上,相比 Vampire4.1, Vampire4.1_P 提高了 116 个定理的证明效率,占 Vampire4.1 证明定理总数的 31.4%。对于 Vampire4.1 已证明的 369 个定理,在子句目标演绎距离阈值设定为 7 时,只有 6 个定理 Vampire4.1_P 无法判定,占 Vampire4.1 证明定理总数的 1.6%,因此本文提出的子句集预处理方法错误约简子句的比例非常低,具有较满意的约简效果。

在证明定理的时间效率上, Vampire4.1_P 证明定理的平均时间为 23.4 s, Vampire4.1 证明定理的平均时间为 17.3 s。从图 2 可以得出,在 100 s 内, Vampire4.1_P 证明定理的总数少于 Vampire4.1,这是因为在预处理阶段的 Vampire4.1_P

消耗了一定的时间,由于子句目标距离的计算时间复杂度为多项式时间,因此预处理阶段的时间主要消耗在冗余子句的处理上。后续的研究需要提高冗余子句的处理效率,例如有针对性地对较小规模的子句集进行冗余处理。

5.2.2 Vampire4.1_P 证明 Vampire4.1 无法判定的定理情况

Vampire4.1_P 证明 Vampire4.1 无法判定的定理 10 个,占 Vampire4.1 无法判定定理总数的 13.5%。表 1 列出了 Vampire4.1_P 证明 Vampire4.1 无法判定的定理情况。由表 1 可得, Vampire4.1_P 证明的这 10 个定理的难度系数较高(难度系数为 1 表示所有证明器未证明的定理),平均难度系数为 0.86,最高难度系数为 0.97,平均使用时间为 121s,判定效率较为满意;其中证明难度系数为 0.9 的定理 SCT170+2,使用时间为 3.88s。实验表明,基于演绎角度的子句集预处理方法能很好地反映子句的演绎情况, Vampire4.1_P 能高效地证明一些较难的定理,进一步表明了本文预处理方法的有效性。

表 1 Vampire4.1_P 证明 Vampire4.1 无法判定的定理

Table 1 Solved theorems by Vampire4.1_P, but unsolved by Vampire4.1

定理名称	难度系数	证明时间/s	定理名称	难度系数	证明时间/s
SET745+4	0.66	24.8	GEO341+1	0.79	105.4
GEO330+1	0.86	89	HAL001+1	0.9	190
LCL557+1	0.9	298.4	SCT170+2	0.9	3.88
SCT124+1	0.93	115.7	LAT258+1	0.86	70.7
GEO338+1	0.97	111.2	SWW340+1	0.83	200.5

5.3 子句集预处理分析

在证明的 373 个定理中,有 254 个定理通过包含冗余子句处理对子句集进行了约简,有 115 个定理通过纯文字规则对子句集进行了约简,有 98 个定理通过子句目标距离对子句集进行了约简,其中定理进行子句集约简子句的最大个数为 1544,占子句集总数的 51.7%,相比 Vampire4.1 证明,该定理所需时间少了 0.26s;对于定理 SCT124+1, Vampire4.1 无法判定, Vampire4.1_P 的判定时间为 115.7 s,约简子句为 141 条,子句约简的比例达到 10.6%;对于定理 SCT170+2, Vampire4.1 无法判定, Vampire4.1_P 的判定时间为 3.88 s,约简子句为 131 条,子句约简的比例达到 3.8%。

图 2 给出了每个已证明定理约简子句的总个数,图 3 列出了每个已证明定理约简子句的比例。

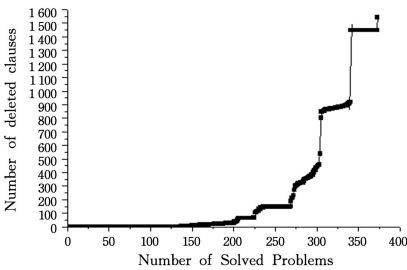


图 2 已证明定理约简子句的总个数
Fig. 2 Total number of reduced clauses in solved theorems

从图 2 可得,共有 288 个定理进行了子句集的约简,占证明定理总数的 77.2%;有 33 个定理子句集约简子句达到 1000 个以上,占证明定理总数的 8.8%;有 70 个定理子句集

约简子句达到 500 个以上,占证明定理总数的 18.8%;有 148 个定理子句集约简子句达到 100 个以上,占证明定理总数的 39.7%。

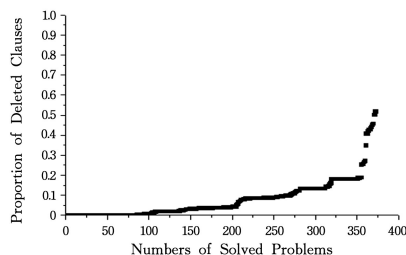


图 3 已证明定理约简子句的比例

Fig. 3 Proportion of reduced clauses in solved theorems

从图 4 可得,定理子句约简的比例最高达到 51.7%;有 12 个定理子句集约简子句的比例达到 40%以上,占证明定理总数的 3.2%;有 103 个定理子句集约简子句的比例达到 10%以上,占证明定理总数的 27.6%;有 279 个定理子句集约简子句的比例达到 5%以上,占证明定理总数的 74.8%。实验表明,本文提出的子句集预处理方法能对子句集进行较大程度的约简,有效减少了原始子句集的规模,提高了证明定理的能力。

结束语 针对当前一阶逻辑子句集预处理采用目标子句项符号相关性进行子句集的约简,不能有效表达子句间的演绎情况,本文从子句间的演绎角度出发,首先对原始子句集进行冗余子句处理,并通过刻画子句与目标子句的演绎关系提出了一种基于目标演绎距离的子句集预处理方法。实验表明,该方法对 77.2%的已证明定理进行了子句集的约简,使定理约简子句的最大比例高达 51.7%,能较大幅度地对子句集进行约简,应用该预处理方法的 Vampire4.1 证明器的能力有所提升,表明基于目标演绎距离的子句集预处理是有效的。

由于一阶逻辑子句演绎具有复杂性,子句集预处理中子句目标演绎距离的阈值合理性设定非常重要。本文阈值设定为固定值 7,但并不表示这种设定适合所有的一阶逻辑问题,因此目标演绎距离阈值的动态设定是后续重要研究内容。此外,目前已实现的基于目标演绎距离的子句集预处理算法虽具有一定的效率,但子句的冗余处理需要遍历整个子句集并对子句中的文字包含关系进行判断,导致了 Vampire4.1_P 的判定效率略低于 Vampire4.1,其子句冗余处理实现的效率还需进一步提高,算法的高效率实现也是接下来的重要研究内容。

参 考 文 献

- [1] RIAZANOV A, VORONKOV A. The design and implementation of VAMPIRE [J]. AI Communications, 2002, 15(2): 91-110.
- [2] VORONKOV A. First-Order theorem proving and vampire [C]//Proceedings of the 27th International Conference on Computer Aided Verification, London: Springer Press, 2013: 1-35.
- [3] SCHULZ S. E-a brainiac theorem prover [J]. AI Communications, 2002, 15(2): 111-126.
- [4] SCHULZ S. System Description: E 1.8 [C]//Proceedings of the

19th International Conferences on Logic for Programming, Artificial Intelligence and Reasoning. Stellenbosch: Springer Press, 2013: 735-743.

- [5] KOROVIN K. iProver-An Instantiation-Based Theorem Prover for First-Order Logic (System Description) [J]. Lecture Notes in Computer Science, 2008, 5195(1): 292-298.
- [6] PAVLOV V, SCHUKIN A, CHERKASOVA T. Exploring Automated Reasoning in First-Order Logic: Tools, Techniques and Application Areas [J]. Communications in Computer & Information Science, 2013, 394(1): 102-116.
- [7] HÖFNER P, STRUTH G, SUTCLIFFE G. Automated verification of refinement laws [J]. Annals of Mathematics & Artificial Intelligence, 2009, 55(1/2): 35-62.
- [8] ROBINSON J A. A machine oriented logic based on the resolution principle [J]. Journal of the Acm, 1965, 12(1): 23-41.
- [9] MCCUNE W. OTTER 3.3 Reference Manual [J]. Office of Scientific & Technical Information Technical Reports, 2007, 11(4): 217-220.
- [10] DENZINGER J, KRONENBURG M, SCHULZ S. Distributed and Learning Equational Prover [J]. Journal of Automated Reasoning, 1997, 18(2): 189-198.
- [11] KHASIDASHVILI Z, KOROVIN K. Predicate Elimination for Preprocessing in First-Order Theorem Proving [C]//International Conference on Theory and Applications of Satisfiability Testing. Bordeaux: Springer Press, 2016: 361-372.
- [12] LOVELAND D W, REED D W, WILSON D S. SATCHMORE: SATCHMO with Relevancy [J]. Journal of Automated Reasoning, 1995, 14(2): 325-351.
- [13] BIBEL W, SCHMITT P H. Automated Deduction—A Basis for Applications, volume III [M]. Dordrecht: Kluwer Academic Publishers, 1998: 119-124.
- [14] SUTCLIFFE G, PUZIS Y. SRASS-A Semantic Relevance Axiom Selection System [C]//Proceedings of the 21th International Conference on Automated Deduction. Bremen: Springer Press, 2007: 295-310.
- [15] VORONKOV A. Sine Qua non for large theory reasoning [C]//Proceedings of the 23th International Conference on Automated Deduction. Wroclaw: Springer Press, 2011: 299-314.



CAO Feng, born in 1984, Ph.D candidate. His main research interests include intelligent information processing and automated reasoning.



XU Yang, born in 1956, Ph.D supervisor, is member of China Computer Federation. His main research interests include uncertainty reasoning and automated reasoning.