CN 43-1258/TP
ISSN 1007-130X

计算机工程与科学
Computer Engineering & Science

第 45 卷第 12 期 2023 年 12 月
Vol. 45, No. 12, Dec. 2023

文章编号:1007-130X(2023)12-2256-09

基于子句活跃度和复杂度的多元动态演绎算法及应用*

林玲瑜[1],曹　锋[1],易见兵[1],方旺盛[1],李　俊[1],吴贯锋[2]

(1.江西理工大学信息工程学院,江西 赣州 341000;2.西南交通大学数学学院,四川 成都 610031)

摘　要:一阶逻辑自动定理证明是知识表示与自动推理领域重要的研究内容,如何有效选取子句参与演绎是提升自动推理能力和效率的研究热点。基于多元动态演绎良好的演绎特性,通过分析子句的变元项性质和函数项结构,提出了一种子句活跃度和复杂度的度量与计算方法,能很好地对不同项结构的子句进行有效评估;基于该子句评估方法,提出了一种子句充分协同演绎的多元动态演绎算法,能有效优化多元演绎搜索路径。将该多元动态演绎算法应用于国际顶尖证明器 Eprover 2.6 中,以 2021 年国际自动推理 FOF 组竞赛例为测试对象,在标准的 300 s 测试时间内,加入了多元动态演绎算法的 Eprover 2.6 相比原始 Eprover 2.6 多证明定理 4 个,在证明定理总数相同的条件下,平均证明时间减少了 1.12 s;能证明 Eprover 2.6 未证明定理 16 个,占未证明定理总数的 15.1%。实验结果表明,该多元动态演绎算法是一种有效的推理方法,能在一定程度上提升自动定理的证明能力和时间效率。

关键词:一阶逻辑;定理证明;自动推理;多元动态演绎;子句评估

中图分类号:TP311.1　　**文献标志码**:A

doi:10.3969/j.issn.1007-130X.2023.12.017

A multi-clause dynamic deduction algorithm based on clause activity and complexity and its application

LIN Ling-yu[1], CAO Feng[1], YI Jian-bing[1], FANG Wang-sheng[1], LI Jun[1], WU Guan-feng[2]

(1. School of Information Engineering, Jiangxi University of Science and Technology, Ganzhou 341000;
2. School of Mathematics, Southwest Jiaotong University, Chengdu 610031, China)

Abstract: First-order logic automated theorem proving is an important research content in the fields of knowledge representation and automated reasoning. How to effectively select clauses to participate in deduction is a research hotspot to improve the capability and efficiency of automated reasoning. Based on the good de-duction characteristics of multi-clause dynamic deduction, this paper proposes a measure and calculation method of clause activity and complexity by analyzing the properties of the variable terms and the structure of function terms of clauses, which can effectively evaluate clauses with different term structures. Based on the clause evaluation method, a multi-clause dynamic deduction algorithm with fully synergized deduction of clauses is proposed, which can effectively optimize the multi-clause deduction search path. The multi-clause dynamic deduction algorithm is applied to the international top prover Eprover 2.6, and the 2021 international automated reasoning competition problems (FOF group) is used as the test object. Within the standard 300 seconds, Eprover 2.6 with the proposed multi-clause dynamic deduction algorithm proven four more theorems than the original Eprover 2.6, and the average proof time is decreases by 1.12 seconds under the condition of proving the same number of theorems as Eprover 2.6. In addition, it can prove 16 unproven theorems of Eprover 2.6, accounting for 15.1% of the to-

* **收稿日期**:2022-09-09;**修回日期**:2022-10-20
基金项目:国家自然科学基金(62066018,62106206);江西省科技厅项目(20212ACB202003);江西省教育厅项目(GJJ200818,GJJ180482);江西理工大学博士启动基金(205200100060)
通信作者:曹锋(caofeng19840301@163.com)
通信地址:341000 江西省赣州市江西理工大学信息工程学院
Address: School of Information Engineering, Jiangxi University of Science and Technology, Ganzhou 341000, Jiangxi, P. R. China

tal unproven theorems. The experimental results show that the proposed multi-clause dynamic deduction algorithm is an effective inference method, which can improve the capability and time efficiency of automated reasoning to a certain extent.

Key words: first-order logic; theorem proving; automated reasoning; multi-clause dynamic deduction; clause evaluation

1 引言

自动推理是人工智能领域的重要分支,其根据对现实事物的知识表示,通过一系列的推理规则自动地对知识进行推理,最终得出知识内在属性的过程。在常见的知识表示形式中,经典逻辑中的一阶逻辑系统作为最基本、最广泛应用的逻辑系统,具有丰富的现实表达能力,具有严密性和通用性的特点。因此,基于一阶逻辑的自动推理系统可有效应用于数学、形式化验证和程序验证等领域[1,2],对其研究具有重要的现实意义。

当前国际著名的一阶逻辑自动定理证明器(自动推理系统)以推理机制作为核心,仍采用二元演绎方法,例如历年在国际自动推理系统竞赛中获得冠军的证明器 Vampire[3]、亚军证明器 Eprover[4]、著名证明器 iprover[5] 和 Prover9[6] 等。为了提升一阶逻辑自动定理证明器的能力与效率,对于有效选取子句和文字参与演绎的启发式策略研究成为了近十年的研究热点[7]。文献[8]提出了大批的子句和文字选择策略交叉使用的方法,该方法能针对不同的一阶逻辑定理找到适合的策略,从而提升自动定理证明的能力与效率。文献[9]根据子句与目标子句项结构的相似性,提出了数种子句选择策略并设定优先级,实验表明了该方法的有效性。文献[10]通过分析函数项的结构特征,提出函数项的度量计算方法,并将该度量值用于子句的有效选择,实验表明了该方法的有效性。文献[11]对多种子句选择策略对证明器的影响进行了分析,并基于该分析提出了一种动态的子句选择方法,实验表明该方法能有效证明一些未被证明的定理。文献[12]提出了一种松弛的加权路径排序方法,并将该方法用于等词演算过程,实验表明该方法能有效提升策略的使用。文献[13]提出了刻画子句演绎历史和函数项深度的子句度量方法,能有效规划子句参与多元演绎的顺序,实验表明该子句选择方法能有效提升多元演绎的能力和效率。

基于上述文献分析,鉴于多元动态演绎具有多元性、动态性、协同性和导向性等优秀的演绎特性[14],本文基于多元演绎中子句的变元项性质和函数项结构分析,提出了子句活跃度和子句复杂度的度量与计算方法,以评估子句参与多元演绎的复杂性,从而优化多元动态演绎的搜索路径,提升演绎能力和效率。基于子句复杂度和子句活跃度,本文还提出了一种搜索路径由简单到复杂且充分发挥子句协同演绎的多元动态演绎算法,将该算法应用于先进的一阶逻辑自动定理证明器 Eprover,以检验其有效性。

2 一阶逻辑演绎方法

2.1 一阶逻辑知识表示方法

一阶逻辑语言包含常元项(小写英文字母 a, b, c, … 或这些英文字母带下标表示)、变元项(x, y, z, …或这些英文字母带下标表示)、函数项(f, g, … 或这些英文字母带下标表示)、谓词符(P, Q, … 或这些英文字母带下标表示)和逻辑连接词(包括有:$\bot$、$\neg$、$\wedge$、$\vee$、$\rightarrow$和$\forall$和$\exists$,其中,$\forall$ 称为全称量词,$\exists$ 称为存在量词)[15]。一阶逻辑公式中文字由谓词项、函数项、常元项和变元项组成,具有元数的概念。若 P 是 n 元谓词符号,则称 $P(f(a_1),\cdots,f(a_n))$ 为原子。一阶逻辑子句集由子句的合取组成,子句由文字析取组成。若子句中含 n 个文字,则称为 n 元子句。不含文字的子句称为空子句,即当一阶逻辑子句集(由子句的合取组成)通过推理方法演绎生成空子句,则该一阶逻辑公式集不可满足。

2.2 一阶逻辑演绎规则

一阶逻辑自动定理证明器核心为演绎推理机制,即通过一系列的演绎规则进行自动的演绎推理,给出定理的逻辑属性(结论分为不可满足或设定的时间内无法判定)。下面简要介绍一阶逻辑子句集二元演绎和多元演绎 2 种演绎机制。

定义 1(归结原理-二元归结[16]) 设一阶逻辑子句集 $S=\{C_1,C_2,\cdots,C_m\}$,当下面条件成立时:

(1)设 C_i 和 C_j($1\leqslant i,j\leqslant m$)是 2 个无公共变元的子句(若存在不同的子句有相同的变元,需要

进行变元更名)。

(2)对子句 C_i 和 C_j,一个替换 σ(σ 可以是空替换)使得 C_i 和 C_j 中的文字存在互补对(L_1^σ,L_2^σ),则称 $R(C_i,C_j)=(C_i^\sigma-L_1^\sigma)\vee(C_j^\sigma-L_2^\sigma)$ 为二元演绎归结式。

对于一阶逻辑公式集,二元演绎每个演绎步骤有且只有 2 个子句参与,有且只消去 1 组变元合一后的互补对文字(2 个文字),该 2 个子句剩余文字的析取作为二元演绎归结式。二元演绎是一种静态的、局部化推理方法,即当子句被选取参与演绎时,该子句的演绎过程也随之确定下来,且不能反映多子句之间的协同演绎关系。由于一阶逻辑问题的判定拥有无限的演绎搜索路径,为了提升一阶逻辑自动定理判定的能力和效率,一些改进的链式归结演绎方法被提出,即设定演绎条件和终止条件的演绎过程,例如超归结方法[17]、单元结果归结方法[18]和线性归结方法[19]等。这些链式归结方法在每个演绎步骤仍只有 2 个子句参与演绎,有且只消去 1 组互补对文字,因此这些演绎方法本质上仍为二元演绎方法。

定义 2(多元动态演绎-矛盾体分离规则[14])设一阶逻辑子句集 $S=\{C_1,C_2,\cdots,C_m\}$,当下面条件成立时:

(1) $C_1,C_2,\cdots,C_m$ 各子句之间不存在相同的变元(若存在不同的子句有相同的变元,需要进行变元更名)。

(2)对任意的子句 C_i,$i=1,2,\cdots,m$,一个替换 σ_i 可应用于子句 C_i(σ_i 可以是空替换)使得 C_i 相同的文字合并,记做子句 $C_i^{\sigma_i}$。将 $C_i^{\sigma_i}$ 的文字分成 2 个部分 $C_i^{\sigma_i-}$ 和 $C_i^{\sigma_i+}$,其中有:

① $C_i^{\sigma_i}=C_i^{\sigma_i+}\vee C_i^{\sigma_i-}$,$C_i^{\sigma_i-}$ 与 $C_i^{\sigma_i+}$ 不含相同的文字,$C_i^{\sigma_i-}$ 不能为空,$C_i^{\sigma_i+}$ 可以为空。

②对任意 $(x_1,\cdots,x_m)\in\prod_{i=1}^m C_i^{\sigma_i-}$,存在互补对文字,$\wedge_{i=1}^m C_i^{\sigma_i-}$ 称为标准矛盾体 S-SC(Separated Standard Contradiction)。

③ $\vee_{i=1}^m C_i^{\sigma_i+}=C_m^{s\sigma}(C_1,C_2,\cdots,C_m)$,其中 $\sigma=\cup_{i=1}^m\sigma_i$。

则称 $C_m^{s\sigma}(C_1,\cdots,C_m)$ 为矛盾体分离式 S-CSC(Standard Contradiction Separation Clause),该演绎方法称为标准矛盾体分离推理规则,简称 S-CS 规则。

多元动态演绎(矛盾体分离规则)从本质上突破二元归结的限制,每个演绎步骤允许 2 个及 2 个以上的子句进行演绎,通过从参与演绎的子句中选择 1 个或多个文字构建标准矛盾体(具有较强的文字消元能力),并将演绎子句剩余文字的析取作为多元演绎归结式。多元演绎是不断动态选择子句、动态构建标准矛盾体的演绎过程,能有效反映多子句间的协同演绎能力。面对多子句间的多元动态演绎,子句被有效地选取参与演绎是提升多元演绎能力和效率的重要方法[20-21]。

2.3 多元动态演绎过程分析

在矛盾体分离规则中,标准矛盾体的构造前提是参与演绎的多个子句任意选取 1 个文字都存在互补对文字,其中不同的变元进行项替换或相同的变元进行不同的项替换都对演绎产生至关重要的影响。在构建标准矛盾体过程中,寻找变元进行有效的项替换对多元演绎的能力和效率起到重要的作用。通过理论分析表明,不同项结构的文字其进行合一替换往往表现不同的特点,如 $P_1(x_1)$ 与 $P_2(x_2,f_1(x_3,x_4))$,在文字 P_1 中变元 x_1 可替换为任意项,无论是常元、变元或是 n 元的函数项,而 P_2 受到函数项 $f_1(x_3,x_4)$ 的限制,其最右项只能与变元或二元函数项 f_1 进行变元替换。与此同时,项结构复杂的文字与子句中的共享变元(同一子句中不同的文字含有的相同变元称为共享变元)进行演绎,合一替换会导致该子句的项结构变得更加复杂,如文字 $P_2(x_2,f_1(x_3,x_4))$ 与子句 $\sim P_2(a,x_5)\vee P_3(f_2(x_6,x_5))$ 进行演绎,则合一替换项 $\sigma=(x_2/a,x_5/f_1(x_3,x_4))$,则演绎后文字 $P_3(f_2(x_6,x_5))$ 演变为 $P_3(f_2(x_6,f_1(x_3,x_4)))$,其函数嵌套层数增多且在二元函数项 f_2 的基础上又增加了二元函数项 f_1,该文字的项结构更加复杂,加大了其参与演绎的复杂性。当该合一替换后的文字继续参与多元演绎,进一步的项替换将使得该文字的项结构进一步加大。因此,高效地选取和利用好子句进行多元动态演绎,是提升多元演绎能力和效率的关键。

3 基于复杂度评估的子句选择方法

3.1 不同的子句选择方法对多元演绎的影响分析

在多元动态演绎过程中,面对子句集中大量复杂且多样化的子句,子句间不同的演绎顺序以及使用不同的变元替换都可能会导致不同的演绎结果,影响演绎的效率。

例 1 设有多元演绎已选取子句 $C_1=$

$P_1(x_{11},f_1(f_2(a,x_{12}),b))$,$C_3=\sim P_1(x_{31},x_{31})$,待选子句集 $S=\{C_2,C_4,C_5,C_6\}$,其中,$C_2=P_1(x_{21},x_{22})\vee\sim P_1(x_{21},x_{23})$,$C_4=\sim P_1(x_{41},x_{42})\vee\sim P_8(x_{42},f_3(x_{41},x_{42}))$,$C_5=P_8(x_{51},x_{52})\vee P_8(x_{51},x_{53})\vee P_8(x_{54},x_{52})\vee P_8(x_{54},x_{53})\vee\sim P_1((x_{51},x_{54})\vee\sim P_6(x_{52},x_{53})$。

若已选取子句 C_1 和 C_3,优先选取 C_2 进行多元动态演绎,生成矛盾体分离式 C_6 为空子句(如表1和表2所示)。

Table 1 Multi-clause deduction substitution table

表1 多元演绎替换表

i	C_i	σ_i	$C_i^{\sigma_i}$
1	C_1	x_{11}/x_{31}	$P_1(x_{31},f_1(f_2(a,x_{12}),b))$
3	C_3	θ(空替换)	$\sim P_1(x_{31},x_{31})$
2	C_2	x_{21}/x_{31}, x_{22}/x_{31}, $x_{23}/f_1(f_2(a,x_{12}),b)$	$P_1(x_{31},x_{31})\vee\sim P_1(x_{31},f_1(f_2(a,x_{12}),b))$

Table 2 Contradiction separation process of multi-clause deduction

表2 多元演绎的矛盾体分离过程

i	C_i	$C_i^{\sigma_i+}$	$C_i^{\sigma_i-}$
1	C_1		$P_1(x_{31},f_1(f_2(a,x_{12}),b))$
3	C_3		$\sim P_1(x_{31},x_{31})$
2	C_2		$P_1(x_{31},x_{31})\vee\sim P_1(x_{31},f_1(f_2(a,x_{12}),b))$

若已选取子句 C_1 和 C_3,优先选取 C_4 进行多元动态演绎,生成矛盾体分离式 $C_7=\sim P_8(f_1(f_2(a,x_{12}),b),f_3(x_{11},f_1(f_2(a,x_{12}),b)))\vee\sim P_1(x_{31},x_{31})$。面对较为复杂的变元替换项,由于 C_4 含有较为复杂的函数项且该函数项中存在共享变元,增大了多元演绎的复杂性,即一定程度上限制了演绎路径的搜索效率。

若已选取子句 C_1 和 C_3,优先选取 C_5 进行多元动态演绎,生成矛盾体分离式 $C_8=P_8(x_{11},x_{52})\vee P_8(x_{11},x_{53})\vee P_8(f_1(f_2(a,x_{12})),x_{52})\vee P_8(f_1(f_2(a,x_{12})),x_{53})\vee\sim P_6(x_{52},x_{53})\vee\sim P_1(x_{31},x_{31})$,由于子句含有共享变元,且共享变元分布于该子句的多个文字,在多元动态演绎过程中,复杂的变元替换对多元演绎的复杂性以及后续的路径搜索产生了较大的影响。

综上分析,子句的复杂度评估对提升多元动态演绎具有重要的作用。本文提出的子句复杂度从子句中函数项结构与变元项分布情况这2个维度进行刻画,通过基于子句复杂度的度量值,合理选择子句进行演绎,能较好地控制多元演绎过程中的合一替换项复杂度,避免过早生成较复杂的矛盾体分离式,从而有利于多元演绎后续的标准矛盾体构造与演绎路径搜索。

3.2 基于变元项的子句活跃度评估方法

在一阶逻辑公式中,子句由常元、变元、函数构成的文字析取而成。在多元动态演绎合一过程中,设给定子句 C_1 中有3项:a,x,$f(x_1,x_2)$,其中,变元项 x 可与任意项进行合一,而常元项 a 除了与变元项进行合一之外,只能与相同的常元项合一以构建标准矛盾体,函数项则除了变元项外只能与相同二元函数项通过合一构建标准矛盾体。在合一过程中,若变元为共享变元,对变元进行常元项替换将加大含该共享变元的文字合一稳定性,而对该变元进行函数项替换且该共享变元存在于文字的函数项中将进一步加大文字的函数嵌套层数,从而使生成的中间矛盾体分离式结构复杂,影响后续的演绎路径搜索。因此,本文基于子句的变元项和函数项结构分析刻画出子句的复杂度,通过子句复杂度评估其演绎影响程度,并进行有效的多元演绎控制,从而更有效地利用子句进行多元动态演绎,使得演绎搜索路径由简单到复杂,以此优化演绎路径搜索,更利于生成空子句。

定义3(子句共享变元活跃度) 子句共享变元活跃度用于刻画子句中共享变元的分布情况。数值越大,表明该共享变元在子句不同文字中出现次数越多;数值越小,表明该共享变元在子句不同文字中出现次数越少。当该度量值为1时(初始值),表明该变元不是共享变元。

通过该属性可判断子句是否含有共享变元。若含有共享变元,则可刻画该共享变元在该子句各个文字中的分布情况。若子句含有共享变元,演绎合一过程中过早采用复杂变元替换会使简单的变元项变复杂,从而生成较为复杂的中间矛盾体分离式。

针对子句中的某一变元,子句共享变元活跃度为该变元出现在子句各个文字中的次数;若该变元在子句某一文字中出现多次,只记为1次。

定义4(文字共享变元活跃度) 文字共享变元活跃度用于刻画文字中共享变元的分布情况。若该变元为子句共享变元,数值越大,表明该文字共享变元在文字中出现次数越多;若该变元为子句共享变元,数值越小,表明该文字共享变元在子句不同文字中出现次数越少;若该变元不是子句共享变元,其数值大小与演绎复杂性无关。当该度量值为0时,表明变元不存在该文字中。

该属性的判断需考虑该变元是否为子句共享变元,通过该属性可判断文字含有共享变元的分布情况。若文字所含变元同时为子句共享变元,演绎

合一过程中过早采用复杂变元替换会使简单的变元项变复杂,从而生成较为复杂的中间矛盾体分离式。

针对子句文字中的某一共享变元,文字共享变元活跃度为该共享变元出现在该文字中的次数。

多元动态演绎过程中,标准矛盾体分离式的构造需进行变元替换。在变元替换过程中,若子句含共享变元则该变元替换将对含共享变元的文字产生影响,从而影响矛盾体分离式,尤其是当变元同时为文字共享变元时,其产生的影响更大。因此,子句中共享变元越活跃(活跃度越大),对矛盾体分离式的影响越大,在选择子句时,优先选择共享变元活跃度较低的子句,使多元动态演绎的替换从简单到复杂。

子句活跃度 CA(Clause Activity)的影响因素包括子句共享变元和文字共享变元的分布情况,为子句中所有变元活跃度 VA(Variable Activity)之和,如式(1)和式(2)所示:

$$CA=\sum VA(x_i) \tag{1}$$

$$VA(x_i)=2^{CSVA(x_i)}+LSVA(x_i) \tag{2}$$

其中,$CSVA(\cdot)$ 表示子句共享变元活跃度,$LSVA(\cdot)$ 表示文字共享变元活跃度,2 个值都需大于 1,x_i 表示子句中的第 i 个变元。

例 2 设有一阶逻辑子句 $C_1=\sim P_1(x_1,x_2)\vee\sim P_2(x_2,f_1(x_1,x_2))\vee P_3(x_1,x_3)$,计算 C_1 的子句活跃度。

解:根据子句变元分布,得到子句共享变元活跃度和文字共享变元活跃度,如表 3 所示。

Table 3 Distribution of clauses variables

表 3 子句变元分布情况表

	文字共享变元活跃度			子句共享变元活跃度
	$\sim P_1$	$\sim P_2$	P_3	
x_1	1	1	1	3
x_2	1	2	0	2
x_3	0	0	1	1

根据式(1)和式(2):

$$VA(x_1)=2^{CSVA(x_1)}+LSVA(x_1)=2^3=8$$

$$VA(x_2)=2^{CSVA(x_2)}+LSVA(x_2)=2^2+2=6$$

$VA(x_3)$ 中 $CSVA(\cdot)$ 和 $LSVA(\cdot)$ 不满足大于 1 的条件,则 $VA(x_3)=0$。

则有子句活跃度 $CA(C_1)=VA(x_1)+VA(x_2)+VA(x_3)=8+6+0=14$。

3.3 基于函数项结构的子句复杂度评估方法

函数项由常元、变元以及嵌套的函数项构成,因此子句的项复杂程度主要由所含的函数项复杂程度体现出来。基于函数项结构的子句复杂度描述子句中函数项的结构,刻画所含不同项对该函数项复杂度的影响程度不同,即体现多元演绎过程中所含的函数项合一的难易程度,一定程度上反映了该含函数项的文字构建标准矛盾体的能力。函数项复杂度影响因素主要有:

(1)不同项类型对函数项复杂度的影响。

设有一元函数项 $f_1(a)$,$f_1(f_2(a))$,$f_1(x)$ 和 $f_1(f_2(x))$,在项合一过程中,常元项不能合一成为其它项,因此常元项对函数项复杂度的影响不变;变元 x 可合一为任意项,因此变元项对函数项复杂度的影响最灵敏;函数项 $f_1(f_2(x))$ 只能合一成函数深度为 1 的 f_2 项或函数项深度更大的 f_2 项。因此,可行的函数项复杂度排序为 $f_1(f_2(x))>f_1(x)>f_1(f_2(a))>f_1(a)$。此外,函数项深度越大,该函数项复杂度越大。

根据项分布的函数项复杂度刻画方法可得,含变元的函数项复杂度大于只含常元的函数项复杂度,而函数深度越大,函数项的复杂度越大。

(2)函数项元数对函数项复杂度的影响。

设有二元函数项 $f_3(a,b)$,$f_3(f_4(a),f_4(b))$,$f_3(x_1,x_2)$,$f_3(f_4(x_1)$ 和 $f_4(x_2))$,同理,在项合一过程中,常元项不能合一成为其它项,因此常元项对函数项复杂度的影响不变;变元 x_1 和 x_2 可合一为任意项,因此该变元项对函数项复杂度的影响最灵敏;函数项 $f_3(f_4(x_1),f_4(x_2))$ 只能合一成函数深度为 1 的 f_4 项或函数项深度更大的 f_4 项。因此,可行的函数项复杂度排序为 $f_3(f_4(x_1),f_4(x_2))>f_3(x_1,x_2)>f_3(f_4(a),f_4(b))>f_3(a,b)$。

由于一阶逻辑子句集中相同的函数项具有相同的元数,且不同的函数项在合一过程中是相互独立的,因此,不同元数的函数项复杂度,仍可按相同的复杂度排序方法进行评估。

(3)所含的共享变元对函数项复杂度的影响。

由于共享变元在合一过程中可使含共享变元项的其它文字复杂度增加,因此,函数项中的共享变元也是影响其复杂度的重要因素。函数项所含的共享变元越多,其合一过程中的项替换越多,使得构造的标准矛盾体更复杂,增加了多元动态演绎的路径搜索难度。

根据上述函数项复杂度分析,在多元动态演绎过程中,函数项在合一过程中存在 2 种情况:

①变元合一发生在函数项内部,若该变元为子

句共享变元,潜在增加了子句的项复杂度。

②变元合一发生在其它子句中的共享变元,潜在增加了其它子句的项复杂度。

综上所述,函数项复杂度与所含的变元以及函数深度有关,其中含共享变元对函数项复杂度影响最大,函数项的深度影响次之,且含变元的函数项复杂度大于只含常元项的函数项复杂度,因此可将子句函数项复杂度的计算分为3部分:子句函数项复杂度(含共享变元)、子句函数项深度(含非共享变元)和子句函数项深度(含常元)。初始常量 M 和 N 用于区分不同类型的变元对函数项复杂度的影响程度。

(1)子句函数项复杂度(含共享变元)。

子句函数项复杂度 $CFSVC$(Clause Function Shared Variable Complexity)(含共享变元)为子句中所有函数项共享变元复杂度 $FSVC$(Function Shared Variable Complexity)(含共享变元)之和,如式(3)和式(4)所示:

$$CFSVC = M + \sum FSVC(f_i(x_j)) \tag{3}$$

$$FSVC(f(x_j)) = 2^{CSVA(x_j)} \tag{4}$$

其中,$f_i(x_j)$ 表示子句中第 i 个函数项(含共享变元)中的第 j 个共享变元,M 初值为200。

(2)子句函数项深度(含非共享变元)。

子句函数项深度 CFL(Clause Function Layer)(含非共享变元)为子句中所有函数项函数深度 FL(Function Layer)之和,如式(5)和式(6)所示:

$$CFL(f_i(x_k)) = N + \sum FL(f_i(x_k)) \tag{5}$$

$$FL(f(x_k)) = \sum Layer(f(x_k)) \tag{6}$$

其中,$f(x_k)$ 表示含非共享变元 x_k 的函数项,$\sum Layer(f(x_k))$ 表示 f 包含的所有子函数项(含非共享变元)深度之和,$f_i(x_k)$ 表示含非共享变元 x_k 的所有函数项,N 初值为100。

(3)子句函数项深度(含常元)。

子句函数项深度 CFL(Clause Function Layer)(含常元)为子句包含的所有函数项 f_i(含常元)深度之和,如式(7)所示:

$$CFL(f_i) = \sum FL(f_i) \tag{7}$$

那么,子句函数项复杂度 CFC(Clause Function Complexity)的值等于子句函数项复杂度(含共享变元)、子句函数项深度(含非共享变元)和子句函数项深度(含常元)之和,如式(8)所示:

$$CFC = CFSVC + CFL(f_i(x_j)) + CFL(f_i) \tag{8}$$

例3 设有一阶逻辑子句 $C_1 = P_1(x_1, f_1(f_1(a))) \vee \sim P_2(x_1, f_2(x_1, x_2)) \vee P_3(x_1, x_2, f_3(x_3))$,计算 C_1 的子句复杂度。

解:根据子句函数项及变元分布,计算子句共享变元活跃度,如表4所示。

Table 4 Distribution of clauses variables

表4 子句变元分布情况表

变元项	子句共享变元活跃度
x_1	3
x_2	2
x_3	1

由表4可知,子句 C_1 中含有上述所包含的3类函数项类型,则分别按照上述公式代入计算:

根据式(4),有:

$FSVC(f(x_1)) = 2^3 = 8$,$FSVC(f(x_2)) = 2^2 = 4$。

根据式(3),有:

$CFSVC = 200 + \sum FSVC(f_i(x_j)) = 200 + FSVC(f_2(x_1)) + FSVC(f_2(x_2)) = 212$。

根据式(6),有:

$FL(f_3(x_3)) = \sum Layer(f(x_j)) = Layer(f_3(x_3)) = 1$。

根据式(5),则有:

$CFL(f_i(x_j)) = N + \sum FL(f_i(x_j)) = 100 + FL(f_3(x_3)) = 101$。

根据式(7),则有:

$CFL(f_i) = \sum FL(f_i) = FL(f_1) = 2$。

综合上述计算,则有:

$CFC = CFSVC + CFL(f_i(x_j)) + CFL(f_i) = 212 + 101 + 2 = 315$。

4 基于子句活跃度和复杂度的多元动态演绎算法

相比静态的二元演绎方法,多元动态演绎需要考虑多个已参与演绎的子句之间的协同演绎能力。选择不同的演绎子句进行协同演绎,则构建了不同的标准矛盾体,生成了不同的中间矛盾体分离式(多元动态演绎过程中生成的新子句),即多元演绎搜索了不同的演绎路径。因此,选取有效的子句参与演绎,并充分发挥多子句间的协同演绎能力是提升多元动态演绎能力与效率的重要途径。本文基于子句活跃度和复杂度的度量方法,提出了一种较优子句选取和充分发挥已参与演绎子句间协同演

绎能力的多元动态演绎算法,其具体步骤如下所示:

(1)将一阶逻辑子句集的所有子句根据子句复杂度从小到大进行排序。当子句复杂度相同时,则按子句活跃度从小到大排序。

(2)按顺序提取排序子句集中所有的单元子句。若排序子句集中没有单元子句,则选取排序子句集中第1个二元子句参与演绎。

(3)将排序子句集中未被选取的子句集作为候选子句集 Q 。

(4)遍历候选子句集 Q,依次选取子句参与多元动态演绎,构建标准矛盾体,得到矛盾体分离式。

(5)若矛盾体分离式为空子句,则得出不可满足结论,算法结束。若矛盾体分离式为单元子句,则重构该子句(备份当前演绎路径,清除该子句所有的变元替换项),继续参与多元动态演绎,构建新的标准矛盾体,得到新的矛盾体分离式,重复步骤(5),直到该子句充分发挥协同演绎,即不再生成有效的矛盾体分离式。若候选子句集 Q 为第1次遍历,则当矛盾体分离式为非单元子句,则清除该子句参与本次多元演绎的演绎记录,转到步骤(4)。若候选子句集 Q 为第2次遍历,则当矛盾体分离式为非单元子句,若矛盾体分离式文字数达到阈值,则算法结束,否则转到步骤(6)。

矛盾体分离式有效性的评估方法:

①矛盾体分离式为恒真式。

②矛盾体分离式为包含冗余子句。

③矛盾体分离式不满足设定的子句复杂度阈值。

(6)当候选子句集 Q 第1次遍历完成时,则对 Q 中未参与演绎的子句进行第2次遍历,依次选取子句参与多元动态演绎,构建标准矛盾体,得到矛盾体分离式,转到步骤(5)。

在演绎算法执行过程中,设定的运行时间达到时,算法结束执行。为了提升多元演绎算法效率,当子句选取参与演绎时,在给定阈值的演绎次数下,都无法得到有效的矛盾体分离式,结束该子句的演绎,选择下一个子句继续参与演绎。

(7)当1次多元动态演绎过程结束后,记录子句参与有效演绎的次数,并将演绎次数作为下一次候选子句集排序的首选条件,即排序方式变为:①子句有效演绎次数从小到大;②子句复杂度从小到大;③子句活跃度从小到大,转到步骤(2)。

5 实验及结果分析

5.1 实验准备

为了体现本文提出的基于子句活跃度和复杂度的多元动态演绎算法对于一阶逻辑自动定理证明的有效性,将该演绎算法应用于国际先进的开元一阶逻辑自动定理证明器 Eprover 2.6 中,记作 mcs_Eprover 证明器(该证明器包含了多元演绎路径搜索能力)。实验选取 2021 年国际一阶逻辑自动定理证明器 CADE(Conference on Automated DEduction) FOF 组竞赛例进行测试(共 500 个),硬件环境为一台计算机(3.2 GHz Intel® Core™ i7-8700 处理器,8 GB 内存),操作系统采用 Ubuntu1 5.04 64 位,每个定理判定时间限定为 300 s(标准时间)。为了将 mcs_Eprover 和 Eprover 2.6 进行有效的实验对比,在相同的硬件环境下对 Eprover 2.6 进行测试并获取数据,其中 Eprover 2.6 能证明总共 500 个定理中的 394 个定理。为了进一步测试本文提出的基于子句活跃度和复杂度的多元动态演绎算法的有效性,针对 Eprover 2.6 不能证明的 106 个定理进行了单独测试。

5.2 mcs_Eprover 证明国际竞赛例(共 500 个)判定情况

mcs_Eprover 证明器将本文提出的基于子句活跃度和复杂度的多元动态演绎算法加入到 Eprover 2.6 证明器中,使 Eprover 2.6 证明器在原有的二元推理机制上增加了多元演绎推理方法,且该多元演绎推理方法具有较优的路径搜索能力。图1为 mcs_Eprover 与 Eprover 2.6 的性能对比图。从定理证明能力上看,mcs_Eprover 共证明 398 个定理,在 Eprover 2.6 已证明的 394 个定理的基础上新增了 4 个定理,新增定理的平均难度等级为 0.83,最大值为 0.94。从定理证明效率上看,mcs_Eprover 证明 398 个定理平均用时 30.538 s,Eprover 2.6 证明 394 个定理平均用时 29.080 2 s,在证明定理总数相同的条件下 mcs_Eprove 证明 394 个定理平均用时 27.963 s。从图1的分布曲线总体可看出,mcs_Eprover 分布曲线总体位于 Eprover 2.6 分布曲线的右侧。

实验表明,mcs_Eprover 的定理证明能力和时间效率相比 Eprover 2.6 都在一定程度上得到了提升,本文提出的基于子句活跃度和复杂度的多元动态演绎算法能有效提高一阶逻辑自动定理证明

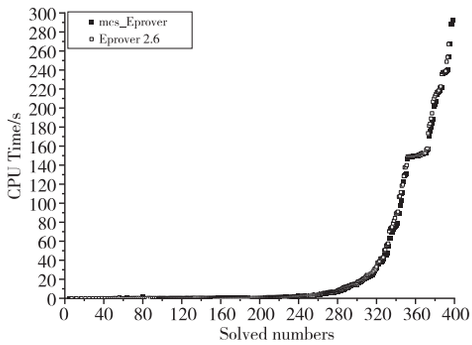

Figure 1 Performance comparison of mcs_Eprover and Eprover 2.6

图 1 mcs_Eprover 与 Eprover 2.6 的性能对比图

器的能力和效率。

5.3 mcs_Eprover 证明 Eprover 2.6 未证明定理(106 个)的判定情况

mcs_Eprover 证明器在 Eprover 2.6 中加入了本文提出的基于子句活跃度和复杂度的多元动态演绎算法,使 Eprover 2.6 具备了多元演绎协同推理能力,能有效应用于难度系数较高的定理证明。实验可得,mcs_Eprover 能有效证明 Eprover 2.6 证明器均未证明的 16 个定理,占其未证明定理的 15.1%,其中表 5 列出了 mcs_Eprover 证明的 16 个定理的情况。

Table 5 Details of 16 theorems solved by mcs_Eprover

表 5 mcs_Eprover 证明的 16 个定理的情况

定理名称	难度等级	最大函数项深度	变元项个数	证明时间/s
AGT013+2	0.72	5	70	279.855
COM148+1	0.86	5	323	236.555
GEO514+1	0.81	3	708	205.142
GEO532+1	0.83	3	973	205.783
ITP004+4	0.83	22	10 434	9.443
NUM736+4	1.00	8	1 458	83.109
NUM924+4	0.44	21	11 690	207.826
SWB016+1	0.75	2	973	152.110
SWB025+1	0.64	2	978	14.740
SWB090+1	0.97	2	973	149.074
SWB093+1	0.89	2	973	19.844
SWB094+1	0.89	2	973	27.073
SWW095+1	0.44	2	148	212.166
SWW470+5	0.64	16	501	205.241
SWW470+6	0.72	16	2 150	205.386
SWW470+7	0.81	19	3 700	205.575
平均	0.765	8	2 314	151.183

由表 5 可分析得出这些定理难度系数较高,平均难度等级为 0.765,其中难度等级最高为 1(难度等级为 1 表示所有的证明器均未证明的定理)。在 mcs_Eprover 证明的 16 个定理中,大部分的定理含有较多的变元项或较复杂的函数项,变元项个数大于 500 的定理有 13 个,最大函数项深度大于 16 的定理有 5 个,其中定理 ITP004+4 最大函数项深度为 22,变元项个数为 10 434,难度等级为 0.83,其证明时间为 9.443 s。面对含有函数项结构较复杂与变元项较活跃的定理,本文提出的子句选择方法能有效地对其子句进行评估,较好地控制多元演绎过程中的合一替换项复杂度,从而有效选择子句进行演绎,充分利用子句在多元演绎中的消文字能力,生成对空子句路径搜索更有利的矛盾体分离式,进而提高证明器的自动推理能力。mcs_Eprover 证明这些较为复杂的定理平均用时 151.183 s,这些定理难度系数较大,证明效率较高。

6 结束语

针对当前一阶逻辑自动定理证明器仍采用二元演绎作为核心推理机制,本文基于多元演绎所具有的良好演绎特性,通过不同项结构类型的子句参与多元演绎进行分析,以子句中对演绎产生重要影响的变元项和函数项进行刻画与度量(记作子句变元活跃度和子句函数复杂度),通过该度量值能有效对子句参与演绎的复杂性进行评估,即简单的子句具有参与演绎的优先选择权。基于该子句评估方法,本文提出了一种多元动态演绎算法,使得多元演绎过程中的变元替换从简单逐步到复杂,同时算法具有演绎评估机制,能有效地评估每个多元演绎路径,并通过回溯机制进行重新演绎。为了验证本文提出的基于子句活跃度和复杂度的多元动态演绎算法有效性,将该算法应用于先进的一阶逻辑自动定理证明器 Eprover 2.6 中,实验表明,加入本文提出的多元动态演绎算法的 Eprover 2.6 的证明能力和时间效率都有一定程度的提升,且能有效证明一批 Eprover 2.6 无法证明的定理。

一阶逻辑自动定理证明属于无限问题,其定理判定存在无限的演绎搜索路径。由于不同的定理存在不同的证明路径,且证明路径相差巨大,因此针对不同的定理如何进一步地有效选择子句是接下来的重要研究内容。同时,不同的定理进行多元动态演绎,演绎路径的有效评估对于定理的证明至关重要,如何基于不同的定理进行演绎路径的动态

评估也是接下来研究的主要内容。

参考文献:

[1] Kovács L. Symbolic computation and automated reasoning for program analysis[C]//Proc of the 12th International Conference on Integrated Formal Methods,2016:20-27.

[2] Cao Feng, Xu Yang, Chen Shu-wei, et al. Application of multi-clause synergized deduction in first-order logic automated theorem proving[J]. Journal of Southwest Jiaotong University,2020,55(2):401-408. (in Chinese)

[3] Kovács L, Voronkov A. First-order theorem proving and Vampire[C]//Proc of the 25th International Conference on Computer Aided Verification,2013:1-35.

[4] Schulz S,Cruanes S,Vukmirovi P. Faster,higher,stronger:E 2.3[C]//Proc of the 27th International Conference on Automated Deduction,2019:495-507.

[5] Korovin K. iProver—An instantiation-based theorem prover for first-order logic[C]//Proc of the 4th International Joint Conference on Automated Reasoning,2009:292-298.

[6] McCune W. Release of Prover9[C]//Proc of the Mile High Conference on Quasigroups, Loops and Nonassociative Systems,2005:1.

[7] Schulz S,Möhrmann M. Performance of clause selection heuristics for saturation-based theorem proving[C]//Proc of the 8th International Joint Conference on Automated Reasoning, 2016:330-345.

[8] Reger G,Tishkovsky D,Voronkov A. CoopeRating proof attempts[C]//Proc of the 25th International Conference on Automated Deduction,2015:339-355.

[9] Jakubuv J,Urban J. Extending E prover with similarity based clause selection strategies[C]//Proc of the 9th International Conference on Intelligent Computer Mathematics,2016:151-156.

[10] Cao F,Xu Y,Zhong J,et al. Holistic deductive framework theorem proving based on standard contradiction separation for first-order logic[C]//Proc of the 12th International Conference on Intelligent Systems and Knowledge Engineering, 2017:389-393.

[11] Rawson M, Reger G. Old or heavy? Decaying gracefully with age/weight shapes[C]//Proc of the 27th International Conference on Automated Deduction,2019:462-476.

[12] Jakubuv J,Kaliszyk C. Relaxed weighted path order in theorem proving[J]. Mathematics in Computer Science,2020,14(1):657-670.

[13] Cao F, Xu Y,Liu J,et al. A multi-clause dynamic deduction algorithm based on standard contradiction separation rule [J]. Information Sciences,2021,566:281-299.

[14] Xu Y,Liu J,Chen S W, et al. Contradiction separation based dynamic multi-clause synergized automated deduction[J]. Information Sciences,2018,462(1):93-113.

[15] Cao Feng, Xu Yang, Wu Guan-feng, et al. Application of multi-clause dynamic deduction in Prover9[J]. Computer Engineering & Science, 2019, 41(9): 1686-1692. (in Chinese)

[16] Robinson J A. A machine-oriented logic based on the resolution principle[J]. Journal of the ACM,1965,12(1):23-41.

[17] Robinson J A. Automatic deduction with hyper-resolution [J]. International Journal of Computing & Mathematics, 1965,1(3):227-234.

[18] Overbeek R,Mccharen J,Wos L. Complexity and related enhancements for automated theorem-proving programs[J]. Computers & Mathematics with Applications,1976,2(1): 1-16.

[19] Loveland D W. A linear format for resolution[C]//Proc of the International IRIA Symposium on Automatic Demonstration,1970:147-162.

[20] Cao F, Xu Y, Liu J, et al. CSE_E 1.0:An integrated automated theorem prover for first-order logic[J]. Symmetry, 2019,11(9):1142.1-1142.15.

[21] Cao F,Xu Y,Chen S W,et al. A contradiction separation dynamic deduction algorithm based on optimized proof search [J]. The International Journal of Computational Intelligence Systems,2019,12(2):1245-1254.

附中文参考文献:

[2] 曹锋,徐扬,陈树伟,等. 多元协同演绎在一阶逻辑 ATP 中的应用[J]. 西南交通大学学报,2020,55(2):401-408.

[15] 曹锋,徐扬,吴贯锋,等. 多元动态演绎在 Prover9 证明器中的应用[J]. 计算机工程与科学,2019,41(9):1686-1692.

作者简介:

林玲瑜(1999-),女,广东汕头人,硕士生,研究方向为智能信息处理、自动推理和一阶逻辑自动定理证明。**E-mail**:812774058@qq.com

LIN Ling-yu, born in 1999, MS candidate, her research interests include intelligent information processing, automated reasoning, and first-order logic automated theorem proving.

曹锋(1984-),男,江西上饶人,博士,讲师,研究方向为智能信息处理、自动推理和一阶逻辑自动定理证。**E-mail**:caofeng19840301@163.com

CAO Feng, born in 1984, PhD, lecturer, his research interests include intelligent information processing, automated reasoning, and first-order logic automated theorem proving.

易见兵(1980-),男,江西宜春人,博士,副教授,研究方向为人工智能。**E-mail**:yijianbing8@163.com

YI Jian-bing, born in 1980, PhD, associate professor, his research interest includes artificial intelligence.