ANITA：解析语义树证明助手

Davi Romero
Vasconcelos塞阿拉联邦大学
基沙达校区，巴西 计算机科学
学院 塞阿拉联邦大学 巴西基沙
达 daviromero@ufc.br

本文介绍了系统 ANITA（解析语义树证明助手），该系统旨在帮助计算机科学专业学生学习解析语义树。该工具使用 Python 编写，可作为桌面应用程序使用，也可通过 Web 平台使用。本文介绍了该工具的逻辑系统，说明了工具的使用方法，并将其与若干类似工具进行了比较。ANITA 已用于逻辑课程，本文还介绍了对该工具的评估。

1 引言

计算机科学中的逻辑课程是大多数信息与通信技术专业课程的一部分，例如塞阿拉联邦大学基沙达校区的信息系统、软件工程、计算机科学和计算机工程专业都将其列为必修课。该课程的挂科率很高。为了更好地掌握课程内容，学生必须进行练习，并获得关于其证明正确性的反馈。

许多演绎系统都用于教授证明的形式推理，例如公理系统（希尔伯特式）、自然演绎系统 [7, 8], 和解析语义树 [12, 2]。

解析语义树系统广泛用于教授证明，并出现在许多逻辑教科书中，例如 [12, 11, 13, 4]。本文介绍一款证明助手 ANITA（解析语义树证明助手），旨在辅助本科生和研究生的教与学。为了教授演绎系统，我们在设计 ANITA 时考虑了以下特点：学生编写证明的方式应尽可能接近教科书中的写法，也应符合学生通常在纸上书写的习惯；工具应易于使用，并尽量减少点击次数，因为频繁点击鼠标可能令人厌烦；工具应允许学生犯错，并指出证明中的错误。

本文其余部分安排如下：第2节简要定义解析表系统；第3节提出以 Fitch 风格在解析语义树中书写证明。第4节介绍 ANITA。第5节将 ANITA 与其他证明助手进行比较。第6节介绍对 ANITA 的评估。最后，我们在第7节总结本文。

2 解析语义树

现在我们来介绍命题逻辑的解析语义树，之后将把它扩展到一阶逻辑。

解析语义树是一种基于反证的推理方法：为了证明 $\Gamma \vdash \varphi$，我们断言 Γ 中的每个公式都为真，而 φ 为假，以推导出矛盾。另一方面，如果没有得到矛-

J. Marcos、W. Neuper 和 P. Quaresma（编）：教育软件的定理证明组件 2022（ThEdu'22），EPTCS 375，2023，第 38–53 页，
doi:10.4204/EPTCS.375.4

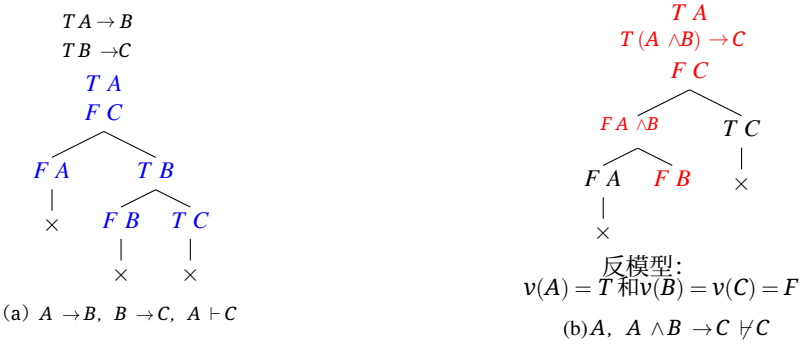
© D.R. Vasconcelos 本作品依据知识共享署名许可协议授权。

盾，那么我们就构造一个反模型，也就是一个满足 Γ 且不满足 φ 的赋值 [1] 。

在解析语义树方法中，我们将$T\ \varphi$和$F\ \varphi$定义为带符号公式，分别表示 φ 为真、 φ 为假。构造语义树的第一步是给 Γ 中的所有公式标上T，并给公式φ 标上F。从初始语义树出发，可以使用语义树扩展规则来：在一个分支末尾添加新公式（α型规则）；或将一个分支拆分为两个分支（β型规则）。语义树的构造规则如下：

α 规则	$T\ \varphi\wedge\psi$ $T\ \varphi$ $T\ \psi$	$F\ \varphi\vee\psi$ $F\ \varphi$ $F\ \psi$	$F\ \varphi\to\psi$ $T\ \varphi$ $F\ \psi$	$T\ \neg\varphi$ $F\ \varphi$ $\quad$ $F\ \neg\varphi$ $T\ \varphi$
β 规则	$F\ \varphi\wedge\psi$ $F\ \varphi$ $\quad$ $F\ \psi$	$T\ \varphi\vee\psi$ $T\ \varphi$ $\quad$ $T\ \psi$	$T\ \varphi\to\psi$ $F\ \varphi$ $\quad$ $T\ \psi$	

在每个分支中，一个公式只能展开一次。没有更多公式可供展开的分支称为**饱和的**。包含一对公式$T\ \varphi$和$F\ \varphi$的分支称为**封闭的**。封闭分支无需再展开。若语义树的所有分支都已封闭，即 $\Gamma\vdash\varphi$，则称该语义树是封闭的。饱和且未封闭的分支提供一个反模型，即 $\Gamma\nvdash\varphi$。图1a展示了$A\to B$，$B\to C$，A 蕴含C，这是一个解析语义树证明。我们使用 $\times$ 作为符号，通过蓝色节点中的带符号公式来封闭一个分支。图1b展示了一个证明，其中有一个分支（红色节点）是饱和的。因此，我们可以根据该分支中原子的真值提取一个反模型。



(a) $A\to B,\ B\to C,\ A\vdash C$ (b)$A,\ A\wedge B\to C\nvdash C$

图 1：解析表中的证明示例

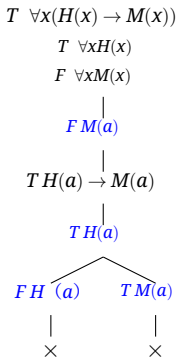
我们扩展解析表系统，以纳入一阶逻辑的证明。该系统包含命题逻辑的所有规则，并增加以下规则：

γ 规则	$T\ \forall x\varphi$ $T\ \varphi_t^x$ t 可代入x 于 φ	$F\ \exists x\varphi$ $F\ \varphi_t^x$ t 可代入x 于 φ
δ 规则	$F\ \forall x\varphi$ $F\ \varphi_a^x$ a 是一个新变量	$T\ \exists x\varphi$ $T\ \varphi_a^x$ a 是一个新变量

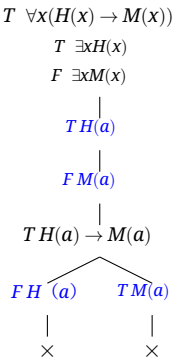
[1]赋值函数 v 是从原子到集合 $\{T,\ F\}$的映射。

这里， φ_t^x是通过将公式 φ 中每个自由出现的变量x替换为项t后得到的表达式。例如，($H(x) \to \forall x M(x))_y^x = (H(y) \to \forall x M(x))$ 。如果项t 中的变量x 不会被 φ 中的 $\forall y$ （或 $\exists y$）量词捕获，那么就称项t 可以代入 φ 中的x 。例如，项z 可以代入 $\forall x P(x,y)$中的y 。另一方面，项x 不能代入 $\forall x P(x,y)$中的y 。

上述规则可以在每个分支中出现多次，因为我们可以对变量进行任意替换。因此，在一般情况下，我们将无法生成反模型。图 2 展示了解析语义树证明的示例。



(a) $\forall x(H(x) \to M(x))\ \forall x H(x) \vdash \forall x M(x)$



(b) $\forall x(H(x) \to M(x))\ \exists x H(x) \vdash \exists x M(x)$

图 2：解析语义树中的证明示例

3 Fitch 风格的解析语义树

（带符号的）语义树是一种特定类型的二叉标记有序树，其中每个节点都标记有一个带符号公式。不过，我们可以用 Fitch 风格呈现解析语义树。证明按线性顺序书写，为所有行编号，并写出一条语句（带符号公式）及其依据，该依据可以是前提、证明的结论，或应用某条推理规则。分支的每次分叉都由 { 和 }界定。在给定位置，只有当某个公式此前已在该分支中出现时，才能在证明中使用该公式。接下来，我们将以 Fitch 风格介绍所有规则。

初始解析树规则： 证明 $\varphi_1, \varphi_2, \ldots, \varphi_n \vdash \psi$从图 3a 所示的初始语义树开始，其中：

- 前提 φ_1， φ_2，……， φ_n分别占一行，按顺序编号，并标记为T （真），理由标注为“前提”。
- 结论 ψ 位于最后一个前提之后的一行，标记为F （假），理由标注为“结论”。

闭分支规则： 如果某个公式φ 在第m 行被标记为T ，并在另一行n 被标记为F （且两行都在第p行之前），那么我们称第p 行的**分支已闭合**（包含一个矛盾 $\perp$）。闭合的分支无法再继续扩展。图 4a 展示了该规则的示意图。图 4b 展示了$A \vdash A$的证明，其中



图 3：初始解析树规则

我们根据第 1 行和第 2 行中分别标记为T 和F的公式A ，在第 3 行闭合了唯一的分支。

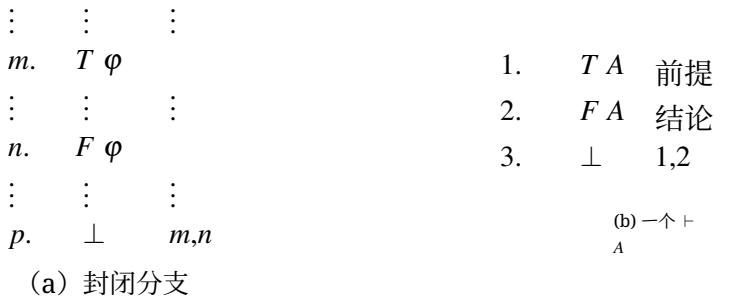


图 4：闭分支规则

图 5a 展示了**否定真规则 (**$\neg T$**)** ，其中，带符号公式$F\ \varphi$ 可根据第m 行的带符号公式$T\ \neg\varphi$ 在第n行得出。类似地，**否定假规则 (**$\neg F$**)**（见图 5b）可用于根据第m行的$F\ \neg\varphi$ 在第n 行得出$T\ \varphi$ 。如图 5c 所示，我们通过第 2 行的否定假规则，在第 3 行得出$T\ \neg A$ 。因此，我们应用 $\neg T$ 规则，在第 4 行得到$F\ A$ 。于是，我们在第 5 行关闭该分支（矛盾），因为第 1 行有$T\ A$ ，第 4 行有$F\ A$ 。

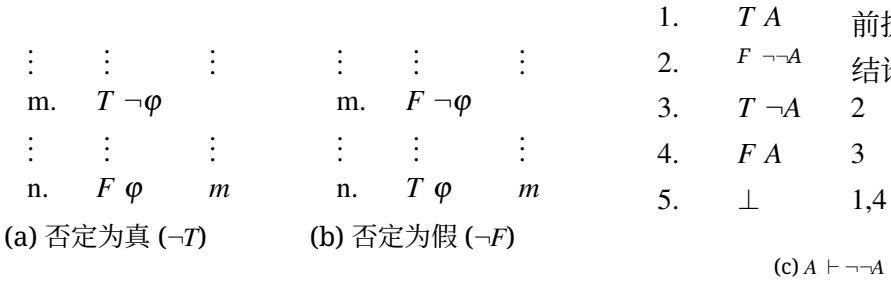


图5：否定规则

合取真规则（$\wedge T$）如图6a所示，其中第n行和第$n+1$行的带符号公式$T\ \varphi$ 和$T\ \psi$，分别由第m行的带符号公式$T\ \varphi\wedge\psi$得到。例如，在图6b中，我们将规则 $\wedge T$应用于第1行的$TA\wedge B$，并在第3行和第4行分别推出TA和TB。

该**与-假　　se rule (**$\wedge F$**)** 如图 7a 所示。此规则应用于第 m 行的$F\ \varphi\wedge\psi$，并将该分　　ts

⋮	⋮	⋮
m.	$T\ \varphi \wedge \psi$	
⋮	⋮	⋮
n.	$T\ \varphi$	m
n+1.	$T\ \psi$	m

(a) 合取真值 ($\wedge T$)

1.	$T\ A \wedge B$	前提
2.	$F\ A$	结论
3.	$T\ A$	1
4.	$T\ B$	1
5.	$\perp$	2,3

(b) $A \wedge B \vdash A$

图6：合取真规则

支分成两部分：一部分从第n 行的$F\ \varphi$开始；另一部分从第p 行的$F\ \psi$开始。我们用符号 {和 }来界定各自的分支。例如，在图7b中，我们将 $\wedge F$规则应用于第 3 行的$F\ A \wedge B$，并将该分支拆分：

1. 在第 4 行开始的分支中，$F\ A$ 与第 1 行的$T\ A$ 配合使用，从而在第 5 行关闭该分支。2. 在第 6 行开始的分支中，$F\ B$ 与第 2 行的$T\ B$ 配合使用，从而在第 7 行关闭该分支。

值得注意的是，我们只能引用同一分支中的公式。因此，例如，第 4 行的公式$F\ A$不能在第 6 行开始的分支中引用。

⋮	⋮	⋮
m.	$F\ \varphi \wedge \psi$	
⋮	⋮	⋮
n.	{ $F\ \varphi$	m
⋮	⋮	⋮
	}	
p.	{ $F\ \psi$	m
⋮	⋮	⋮
	}	

(a) 合取为假 ($\wedge F$)

1.	$T\ A$	前提
2.	$T\ B$	前提
3.	$F\ A \wedge B$	结论
4.	{ $F\ A$	3
5.	$\perp$	1,4 }
6.	{ $F\ B$	3
7.	$\perp$	2,6 }

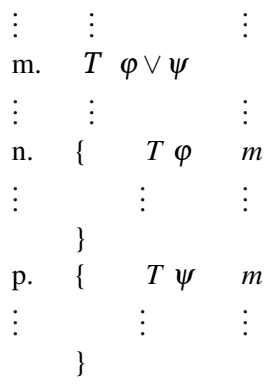
(b) A、$B \vdash A \wedge B$

图 7：合取式假规则

或真规则（$\vee$真 ）见图 8a。我们将此规则应用于第 m行的 真 $\varphi \vee \psi$，并将此分支拆分为两个新分支：一个从第n 行的 真 φ开始；另一个从第 p 行的 真 ψ开始。例如，在图 8b 中，规则 $\vee$真 应用于第 1 行的 真 $A \vee B$，并将此分支拆分为：

1. 在从第 5 行的 真 A 开始的分支中，使用它与第 3 行的 假 A 关闭该分支，结束于第 6 行。
2. 在从第 7 行的 真 B 开始的分支中，使用它与第 4 行的 假 B 关闭该分支，结束于第 8 行。

图 9a 展示了**析取为假规则**（$\vee F$），其中第n行和第$n+1$行的带符号公式$F\ \varphi$和$F\ \psi$分别由带符号公式$F\ \varphi \vee \psi$推导得出。例如，在图 9b 中，我们对第 2 行的$F\ A \vee B$应用规则 $\vee F$，推导出$F\ A$和$F\ B$，分别写在第 3 行和第 4 行。

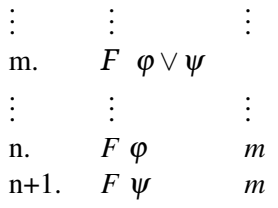


(a) 或真（$\vee T$）

1.	$T\ A \vee B$	前提
2.	$T\ \neg B$	前提
3.	$F\ A$	结论
4.	$F\ B$	2
5.	{ $T\ A$	1
6.	$\bot$	3,5 }
7.	{ $T\ B$	1
8.	$\bot$	4,7 }

(b) $A \vee B\ \neg B \vdash A$

图 8：析取真规则



(a) 析取为假 ($\vee F$)

1.	$T\ A$	前提
2.	$F\ A \vee B$	结论
3.	$F\ A$	2
4.	$F\ B$	2
5.	$\bot$	1,3

(b) $A \vdash A \vee B$

图 9：析取假规则

图 10a 展示了**蕴涵真规则**（$\rightarrow T$）。我们对第m 行的$T\ \varphi \rightarrow \psi$ 应用此规则，并将此分支拆分为两个新分支：一个从第n 行开始，包含$F\ \varphi$；另一个从第p 行开始，包含$T\ \psi$。例如，在图 10b 中，规则 $\rightarrow T$ 应用于第 1 行的$T\ A \rightarrow B$，我们将此分支拆分：

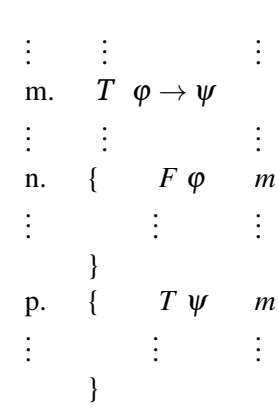
1. 在从第 5 行的 $F\ \neg A$ 开始的分支中，我们应用 $\neg F$，得到第 6 行的 $T\ A$，然后使用第 3 行的$F\ A$，在第 7 行关闭此分支。2. 在从第 8 行的 $T\ B$ 开始的分支中，使用第 4 行的 $F\ B$，在第 9 行关闭此分支。

图11a展示了**蕴含为假规则**（$\rightarrow F$）。其中，第n 行和第$n+1$行的带符号公式$T\ \varphi$ 和$F\ \psi$分别由带符号公式 $F\ \varphi \rightarrow \psi$得到。例如，在图11b中，我们将规则 $\rightarrow F$ 应用于第2行的$F\ A \rightarrow B$，并在第3行和第4行分别推导出$T\ A$ 和$F\ B$。

图12a展示了**全称真规则**（$\forall T$）。其中，我们将 $\forall T$规则应用于第m 行的带符号公式$T\ \forall x\varphi(x)$，并在第n行得到带符号公式$T\ \varphi_t^x$，其中t 可代入 φ中的x。图12b展示了将该规则应用于第1行的$T\ \forall x(H(x) \rightarrow M(x))$，并在第4行得到$T\ H(s) \rightarrow M(s)$。

图13a展示了**全称否定规则**（$\forall F$）。其中，我们将 $\forall F$规则应用于第m 行的带符号公式$F\ \forall x\varphi(x)$，并在第n行得到带符号公式$F\ \varphi_a^x$，其中a 是一个新变量。图13b展示了将该规则应用于第3行的$F\ \forall x M(x)$，并在第4行得出$F\ M(a)$。

存在量词真规则 ($\exists T$**)** 如图14a所示，其中我们对第m 行的带符号公式$T\ \exists x\varphi(x)$应用 $\exists T$规则，并在第n行得到带符号公式$T\ \varphi_a^x$，其中a 是一个新变量。图14b展示了如何将此规则应用于第2行的$T\ \exists x\ H(x)$，从而在第4行得到$T\ H(a)$。

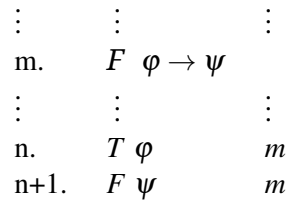


(a) 蕴涵为真 ($\to T$)

1.	$T\ \neg A \to B$	前提
2.	$F\ A \vee B$	结论
3.	$F\ A$	2
4.	$F\ B$	2
5.	{ $F\ \neg A$	1
6.	$T\ A$	5
7.	$\bot$	6,3 }
8.	{ $T\ B$	1
9.	$\bot$	8,4 }

(b) $\neg A \to B \vdash A \vee B$

图 10：蕴涵为真规则

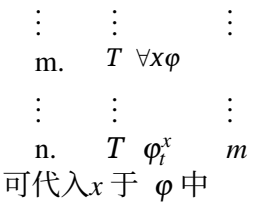


(a) 蕴含为假（$\to F$）

1.	$T\ B$	前提
2.	$F\ A \to B$	结论
3.	$T\ A$	2
4.	$F\ B$	2
5.	$\bot$	1,4

(b) $B \vdash A \to B$

图11：蕴含为假规则



t 可代入x 于 φ 中

(a) 全称真值 ($\forall T$)

1.	$T\ \forall x(H(x) \to M(x))$	前提
2.	$T\ H(s)$	前提
3.	$F\ M(s)$	结论
4.	$T\ H(s) \to M(s)$	1
5.	{ $F\ H(s)$	4
6.	$\bot$	2,5 }
7.	{ $T\ M(s)$	4
8.	$\bot$	7,3 }

(b) $\forall x(H(x) \to M(x)),\ H(s) \vdash M(s)$

图 12：全称真规则

存在量词为假规则 ($\exists F$) 如图15a所示，其中我们对第m 行的带符号公式$F\ \exists x\varphi(x)$应用 $\exists F$规则，并在第n行得到带符号公式$F\ \varphi_t^x$，其中t 可代入 φ 中的x 。图15b展示了如何将此规则应用于第4行的$F\ \exists xP(x)$，从而在第5行推出$P(a)$。

⋮	⋮	⋮
m.	$F\ \forall x\varphi$	
⋮	⋮	⋮
n.	$F\ \varphi^x_a$	m

a 是新变量

(a) 全称假 ($\forall F$)

1.	$T\ \forall x(H(x) \to M(x))$	前提
2.	$T\ \forall x H(x)$	前提
3.	$F\ \forall x M(x)$	结论
4.	$F\ M(a)$	3
5.	$T\ H(a)$	2
6.	$T\ H(a) \to M(a)$	1
7.	$\{\quad F\ H(a)$	6
8.	$\quad\ \bot$	5,7 $\}$
9.	$\{\quad T\ M(a)$	6
10.	$\quad\ \bot$	9,4 $\}$

(b) $\forall x(H(x) \to M(x))\ \forall x H(x) \vdash \forall x M(x)$

图 13：全称否定规则

⋮	⋮	⋮
m.	$T\ \exists x\varphi$	
⋮	⋮	⋮
n.	$T\ \varphi^x_a$	m

a 是新变量

(a) 存在真 ($\exists T$)

1.	$T\ \forall x(H(x) \to M(x))$	前提
2.	$T\ \exists x H(x)$	前提
3.	$F\ \exists x M(x)$	结论
4.	$T\ H(—)$	2
5.	$F\ M(a)$	3
6.	$T\ H(a) \to M(a)$	1
7.	$\{\quad F\ H(a)$	6
8.	$\quad\ \bot$	4,7 $\}$
9.	$\{\quad T\ M(a)$	6
10.	$\quad\ \bot$	5,9 $\}$

(b) $\forall x(H(x) \to M(x))\ \exists x H(x) \vdash \exists x M(x)$

图14：存在真规则

4 ANITA解析语义树证明助手

ANITA[2] 证明助手——解析语义树证明助手，是一款使用 Python 编写的工具，可作为桌面 应用程序使用，也可通过 Web平台[3]使用。其主要理念是让学生尽可能按照教科书中的形式以及平时在纸上书写的方式来编写证明。ANITA 可让学生自动检查解析语义树中的证明是否有效。如果证明不正确，该工具会显示证明中的错误。这样，学生便可以从错误中学习。该 Web 界面非常易于使用，并具有：

- 用于以纯文本形式编辑证明的区域。学生应按照 Fitch 风格编写证明，格

[2]ANITA 的源代码可在 https://github.com/daviromero/anita 获取，采用 MIT 许可证。[3]ANITA 可通过以下网址访问：https://sistemas.quixada.ufc.br/anita/en/

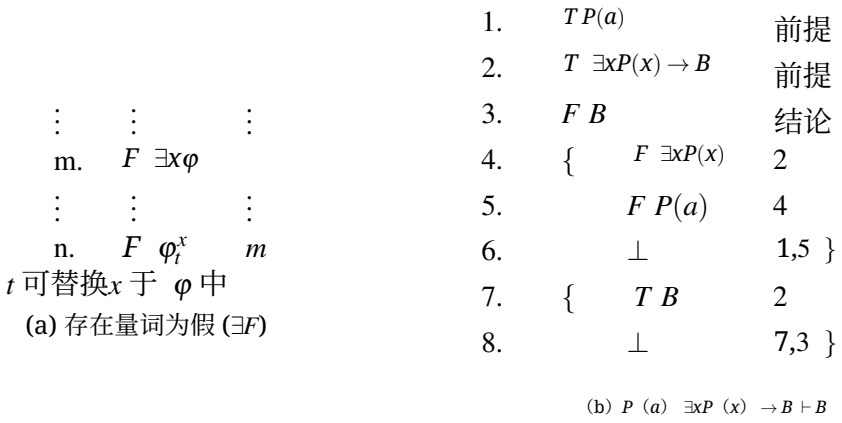


图 15：存在量词为假规则

式见第3节。

• 用于显示证明是否有效、反模型或证明中错误的消息区域。

• 以及以下链接：检查，用于检查证明的正确性；手册，用于查看包含推理规则和示例的文档；*LaTeX*，用于生成有效证明中树的 LaTeX 代码[4]；在 *Overleaf* 中使用 *Latex*，用于直接在 Overleaf 中打开证明源代码[5]。

为了便于编写证明，我们在 ANITA 中采用了以下约定：

• 原子[6] 用大写字母表示（例如 A、B、H(x)）；

• 变量以小写字母开头，后接字母和数字（例如 x、x0）；

• 含有 $\forall x$ 和 $\exists x$ 的公式分别表示为 Ax 和 Ex （“A”和“E”后接变量 x）。例如，Ax (H(x)−> M(x)) 表示 $\forall x\,(H(x) \to M(x))$。

• 图16展示了逻辑符号与 ANITA 中所用符号之间的对应关系。

• 量词和逻辑联结词的优先顺序由 $\neg, \forall, \exists, \wedge, \vee, \to$ 定义，并采用右结合方式。例如：–公式 ~A&B−>C 表示公式 $(((\neg A) \wedge B) \to C)$；–定理 ~A|B |− A−>C 表示 $((\neg A) \vee B) \vdash (A \to C)$。

• 每条推理规则都以相应的联结词和带符号公式的真值命名。例如，&T 表示合取真值规则。规则名称也可以省略。

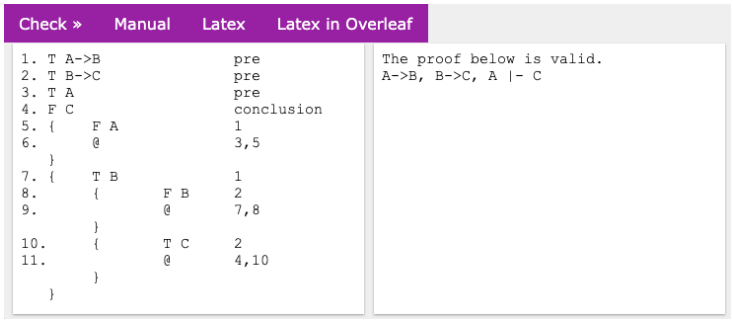
• 前提和结论的证明依据分别使用保留字 pre 和 conclusion。

符号	¬	∧	∨	→	∀x	∃x	⊥	分支	⊢
LaTeX	\否定	\合取	\lor	\蕴含	\全称量词 x	\存在量词 x	\bot	[.]	\推导符号
ANITA	~	&	|	−>	Ax	Ex	@	{}	|−

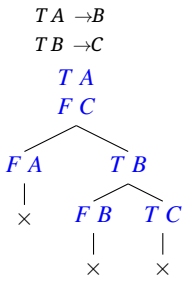
图16：逻辑、ANITA 与 LaTeX 符号之间的对应关系

[4]在 LaTeX 代码中使用*qtree* 宏包。[5]Overleaf 是一个用于协作编辑 LaTeX 的平台。网址：http://overleaf.com/[6]原子公式（或原子）就是将谓词应用于一个项元组；也就是说，原子公式的形式为$P(t1,\ldots,t_n)$，其中P 是谓词，而t_n 是项。

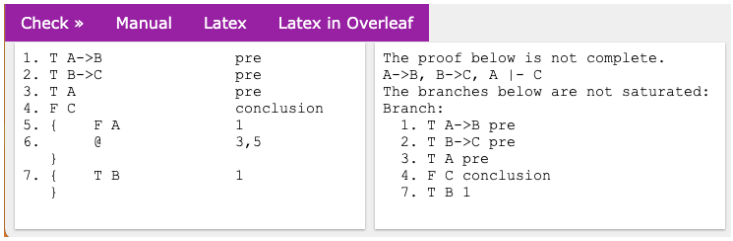
图17a展示了在 ANITA 中对 $A \to B, B \to C, A \vdash C$ 的一个有效证明，图17b展示了 ANITA 生成的树，其中蓝色节点（带符号公式）指出了闭合分支。图17c展示了在 ANITA 中对 $A \to B, B \to C, A \vdash C$ 的不完整证明示例，而图17d则以红色显示了 ANITA 生成的解析表中的开放分支。



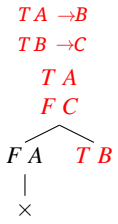
(a) ANITA: $A \to B$, $B \to C$, $A \vdash C$



(b) 解析语义树证明



(c) ANITA 中的开放分支: $A \to B$, $B \to C$, $A \vdash C$



(d) 解析语义树证明

图 17: ANITA: $A \to B$, $B \to C$, $A \vdash C$

图 18 显示了一条提示，指出第 1 行中的带符号公式$T\,Ex\,H(x)$未能正确应用存在量词真规则，因此无法在第 4 行得到$T\,H(a)$，因为项a 不是新变量（见第 3 行）。

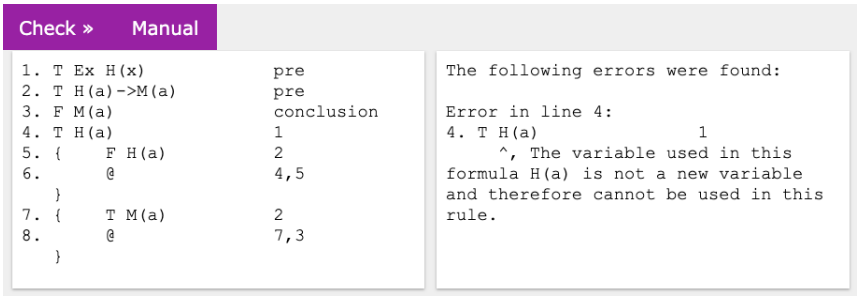
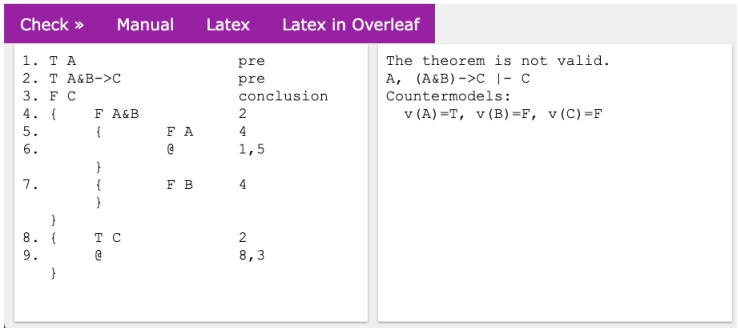
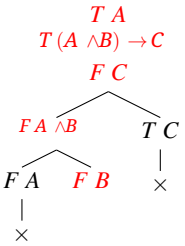


图 18: ANITA: 证明中的错误

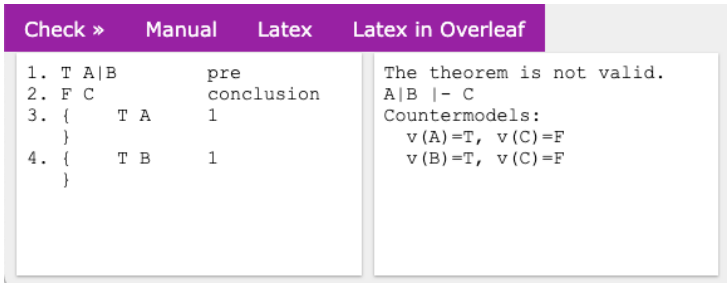
图 19a 展示了一个证明，说明$A, A \wedge B \to C$不蕴涵C，而 ANITA 展示了该证明的反模型。图 19b 用红色显示了由 ANITA 生成的解析语义树中的饱和分支，该分支提供了一个反模型。图 19d 用红色显示了由 ANITA 生成的解析语义树中的两个饱和分支，它们为证明$A \vee B \nvdash C$提供反模型，见图 19c。注意，在开放分支（第 1,2 和 3 行）中，原子公式B 并未出现，因此$v(B)$可以取T 或假值F，反模型表示为$v(A) = T, v(C) = F$。



(a) ANITA：A，$A \wedge B \rightarrow C \nvdash C$



(b) 解析语义树证明



(c) ANITA：$A \vee B \nvdash C$



(d) 解析语义树证明

图 19：ANITA：反模型示例

5 相关工作

本文聚焦于自然演绎和解析语义树证明助手。尽管也存在用于其他系统的证明助手，例如 SeCaV [5]。我们总结了这些证明助手的特性，并着重介绍这些工具与本文所提方案之间的相似之处和差异。我们还给出了各个证明助手中$A \rightarrow B, B \rightarrow C, A \vdash C$的证明。

- Jape[7] [1] 是一款桌面证明助手，用于以自然演绎中的 Fitch 风格编写证明。用户通过其图形用户界面插入推理规则来完成证明。
- ProofWeb [10] 是一个 Web 界面，旨在成为 Jape 的演进版本，并使用 Coq[8] ——这是一款先进的证明助手，可用于编写数学证明。用户必须在文本区域中编写证明，或使用图形用户界面添加推理规则。ProofWeb 可以用 Fitch 或 Gentzen 风格显示证明。

- Panda [6] 也是一款桌面证明助手，与前述工具不同的是，它允许用户通过其图形用户界面以 Gentzen 风格编写证明。
- NaDeA[9] [15] 是一款用于自然演绎的 Web 证明助手，其形式化实现基于 Isabelle。用户必须通过其基于点击操作的用户界面编写证明。
- NADIA[10][11] [14] 是一款采用 Fitch 风格的自然演绎网页证明助手。NADIA 允许

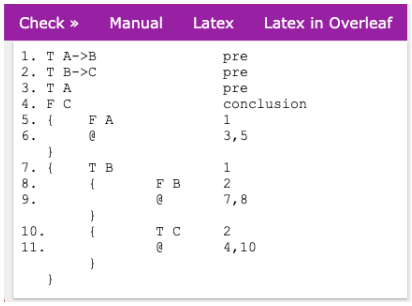
[7]Jape 的源代码可在 https://github.com/RBornat/jape/ 获取[8]Coq 可在 https://coq.inria.fr/ 获取[9]可在 https://nadea.compute.dtu.dk/ 获取[10]可在 https://sistemas.quixada.ufc.br/nadia/ 获取 [11]NADIA 的源代码可在 https://github.com/daviromero/nadia 获取，采用 MIT 许可证。

学生使用类似于 [8] 的输入语法代码，尽可能按照在纸上书写的方式编写证明。NADIA 会以 Fitch 或 Gentzen 风格显示证明。

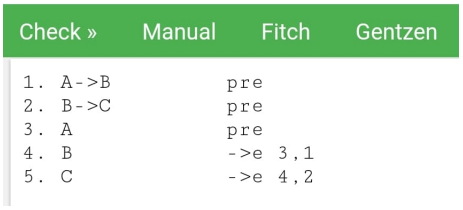
• Carnap.io[12] [9] 是一个免费的开源 Haskell 框架，用于创建和探索形式语言、逻辑和语义学。[13] 还提供了一个用于解析语义树的 Web 证明助手，可通过图形用户界面构建证明。

• Tree Proof Generator Tableau[14] 是一个语义树证明器，适用于经典命题逻辑和一阶逻辑，以及某些模态逻辑。该证明器使用 JavaScript 编写，完全在浏览器中运行。用户可以输入标准命题逻辑、谓词逻辑或模态逻辑公式，证明器会自动尝试寻找反模型或树形证明。

ANITA 与 NADIA 非常相似。两个系统都接收一段包含定理证明的文本作为输入，并检查该证明是否正确。如果不正确，工具会显示发现的错误。这两种证明助手的主要区别在于，ANITA 接受解析语义树证明（见图 20a），而 NADIA 接受自然演绎证明（见图 20b）。ANITA 的证明解析器与 NADIA 的解析器完全不同，因为二者实现的推理规则集截然不同。



(a) ANITA: $A \to B,\ B \to C,\ A \vdash C$

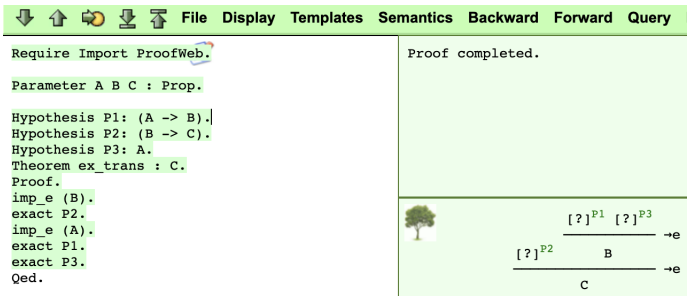


(b) NADIA: $A \to B,\ B \to C,\ A \vdash C$

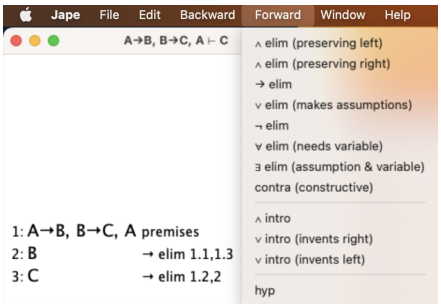
图 20：ANITA 和 NADIA 证明助手

图 21a 展示了 ProofWeb 中的一个证明。需要注意的是，学生必须学习一种新语法，与他们在纸上书写的方式有很大不同。另一方面，Jape（图 21b）、Panda（图 21c）、NaDeA（图 21d）和 Carnap.io（图 22a）中的证明都通过图形用户界面完成，用户需要点击菜单来添加每条推理规则。事实上，证明生成器是一种定理证明器，而非证明助手。因此，用户只能与该工具交互，输入定理，并获得一个反模型或一棵树形证明；这对于教授如何使用推理规则并不是很有用。

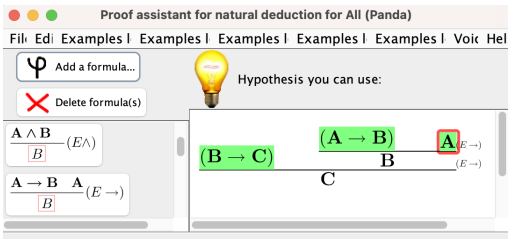
[12]可访问：https://carnap.io[13]可访问：https://carnap.io/srv/doc/truth-tree.md[14]可访问：https://www.umsu.de/trees



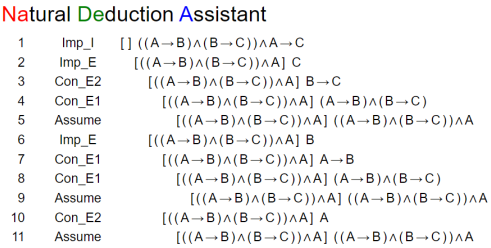
(a) ProofWeb：$A \to B,\ B \to C,\ A \vdash C$



(b) Jape：$A \to B,\ B \to C,\ A \vdash C$

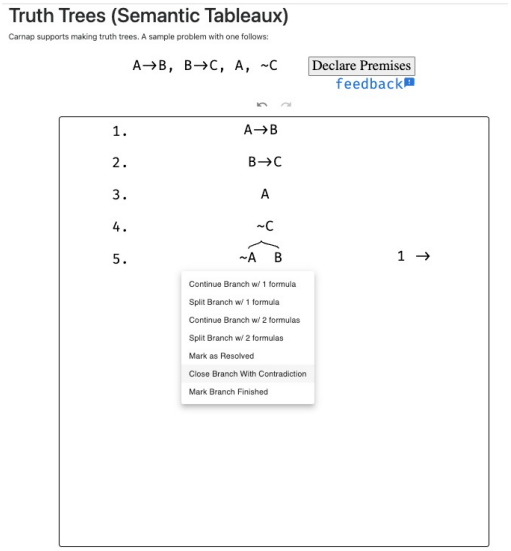


(c) Panda：$A \to B,\ B \to C,\ A \vdash C$

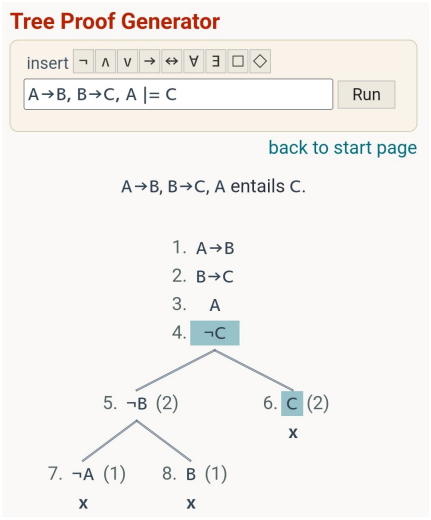


(d) NaDeA：$A \to B,\ B \to C,\ A \vdash C$

图 21：自然演绎表列证明助手



(a) Carnap.io：一个 $A \to B,\ B \to C,\ A \vdash C$



(b) 证明生成器：$A \to B,\ B \to C,\ A \vdash C$

图 22：分析性表列证明助手

下面我们从以下方面总结证明助手：演绎系统（自然演绎的 ND、解析语义树的 AT）；证明风格的呈现方式（F 表示 Fitch 风格，G 表示 Gentzen风格）；输入证明写作的方式（GUI 表示基于点击图形用户界面，PT 表示纯文本）。

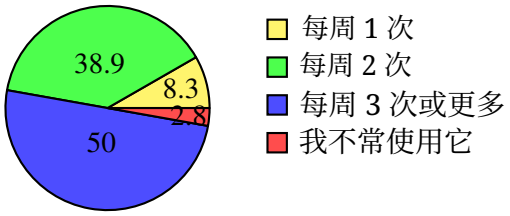
	ProofWeb	Jape	Panda	NaDeA	NADIA	ANITA	Carnap.io	证明生成
演绎系统	ND	ND	ND	ND	ND	AT	AT	AT
显示证明风格	F, G	F	G	F	F, G	F	T	T
输入证明写作	图形用户界面，PT	GUI	GUI	GUI	PT	PT	GUI	GUI

6 ANITA 评估

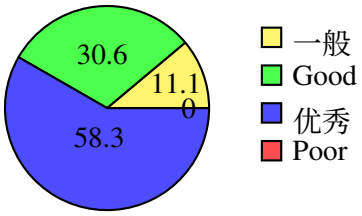
本节介绍 ANITA 评估的结果。这项评估于 2022 年在塞阿拉联邦大学基沙达校区的两个计算机科学中的逻辑班级中开展。每个班级每周上课 4 小时，共 16 周，总计有 74 名学生注册。

6.1 ANITA 学生评估

74 名注册学生中共有 36 人填写了匿名在线表单（49%）。100% 的学生表示他们使用 ANITA 作为学习工具，并认为 ANITA 有助于练习所学内容。91.7% 的学生认为 ANITA 非常易于使用。图 23a 显示学生使用 ANITA 的频率，图 23b 显示学生对 ANITA 错误消息的评价。



(a) 你多久使用一次 ANITA?



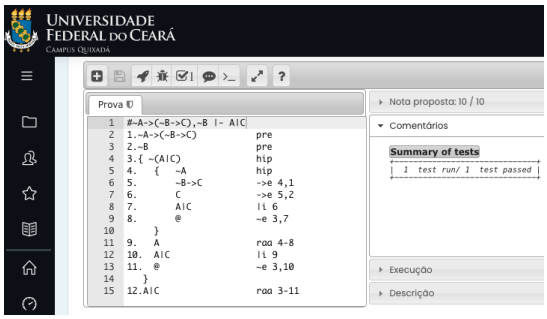
(b) 你如何评价 ANITA 的错误消息?

图 23：评估

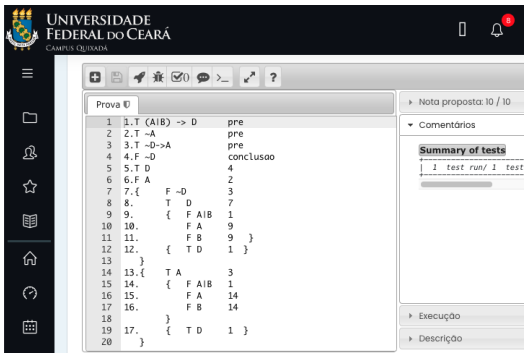
6.2 评估

我们在第二次部分评估（AP2）中使用了集成于 Moodle 平台[15], 的 NADIA 和 ANITA。该评估在实验室进行，包含四道需要用自然演绎（ND）证明的定理和四道需要用解析语义树（AT）证明的定理，每道题分值为 1.25。学生在 Moodle 平台上写下每道定理的证明，并由 ANITA 和 NADIA 自动检查证明是否正确。例如，图 24a 展示了某位学生在 Moodle 平台上对自然演绎问题 1 的作答。图 24b 展示了某位学生在 Moodle 平台上对解析语义树问题 4 的作答。

[15]Moodle 可通过 https://moodle.com/ 访问



(a) Moodle（NADIA）- 问题 1 的答案：
$\neg A \rightarrow (\neg B \rightarrow C) \ \neg B \vdash A \vee C$



(b) Moodle（ANITA）- 问题 4 的答案：
$(A \vee B) \rightarrow D \ \neg A \ \neg D \rightarrow A \not\vdash \neg D$

图24：AP2 - Moodle 平台中 NADIA 和 ANITA 问题示例 m

74 名注册学生中共有 48 人参加了测试（65%）。学生的平均分（M）为 6.80，标准差（SD）为 3.28。自然演绎题目的平均分为 2.76（MND），解析语义树题目的平均分为 4.04（MAT）。63% 的学生（AND）回答了自然演绎题目，其中 87%（RND）答对；92% 的学生（AAT）回答了解析语义树题目，其中 88%（RAT）答对。表1按班级呈现了结果。

班级	学生	M	SD	MND	SDND	AND	RND	MAT	SDAT	AAT	RAT		
A	20	7.31	2.93	2.94	2.08	65%	90%	4.38	1.43	94%	93%		
B	28	6.44	3.53	2.64	2.03	62%	85%	3.80	1.82	91%	84%		
A+B	**48**	**6.80**	3.28	**2.76**	2,05	63%	87%	**4.04**	1.66	92%	88%		

表1：各班级的结果

7 结论与未来工作

ANITA 已用于向计算机科学专业学生教授解析语义树。我们将 ANITA 与其他逻辑教学工具进行了比较。从评估的角度来看，ANITA 在我们的课程中取得了成功。49% 的学生填写了匿名在线表单，其中：100% 认为该工具有助于练习课程内容；91% 认为该工具易于使用（评价为优秀或良好）；90% 每周使用该工具两次或更多次；90% 认为对消息的理解程度为优秀或良好。我们在部分评估中使用了集成于 Moodle 平台的 ANITA。74 名注册学生中共有 48 名参加了测试（65%）。92% 的学生提交了针对 4 个定理的证明，其中 88% 答对了题目。

作为未来工作，我们考虑为 ANITA 开发更多教学材料，并进一步评估 ANITA 作为逻辑教学工具的效果。

致谢 ： 本研究部分受到项目 04772314/2020/FUNCAP 的资助。

参考文献

[1] Richard Bornat & Bernard Sufrin（1996）：*Jape* 的简洁界面。定理证明器用户界面 *UITP'98*。

[2] M. D’Agostino、D.M. Gabbay、R. Hähnle & J. Posegga（1999）：表列方法手册。施普林格·荷兰，doi:10.1007/978-94-017-1754-0。[3] Dirk van Dalen（2013）：逻辑与结构（第*5*版）。施普林格·伦敦，伦敦，doi:10.1007/978-1-4471- 4558-5。[4] Melvin Fitting（1996）：一阶逻辑与自动定理证明（第*2*版）。施普林格出版社，柏林，海德堡，doi:10.1007/978-1-4612-2360-3。[5] Asta Halkjær From、Frederik Krogsdal Jacobsen & Jørgen Villadsen（2022）：*SeCaV*：*Isabelle/HOL* 中的相继演算验证器。收录于 Mauricio Ayala-Rincon & Eduardo Bonelli（编）：第16届逻辑与语义框架及其应用会议论文集，布宜诺斯艾利斯，阿根廷（线上），2021年7月23日至24日，理论计算机科学电子会议论文集357，开放出版协会，第38–55页，doi:10.4204/EPTCS.357.4。[6] Olivier Gasquet、François Schwarzentruber & Martin Strecker（2011）：*Panda*：面向所有人的自然演绎证明助手。本科生用的 *Gentzen* 风格证明助手。收录于 Patrick Blackburn、Hans van Ditmarsch、María Manzano & Fernando Soler-Toscano（编）：逻辑教学工具，施普林格·柏林·海德堡，柏林，海德堡，第85–92页，doi:10.1007/978-3-642-21350-2_11。[7] Gerhard Gentzen（1969）：论文集。北荷兰出版公司，doi:10.2307/2272429。[8] Michael Huth & Mark Ryan（2004）：计算机科学中的逻辑：系统建模与推理（第*2*版）。剑桥大学出版社，doi:10.1017/CBO9780511810275。[9] Graham Leach-Krouse（2018）：*Carnap*：浏览器中的开放式形式推理框架。收录于 Pedro Quaresma & Walther Neuper（编）：第6届国际研讨会论文集教育软件的定理证明组件，哥德堡，瑞典，2017年8月6日，理论计算机科学电子会议论文集267，开放出版协会，第70–88页，doi:10.4204/EPTCS.267.5。[10] Hendriks Maxim、Cezary Kaliszyk、Femke van Raamsdonk & Freek Wiedijk（2010）：使用先进的证明助手进行逻辑教学。《纳波卡教学学报》3。可访问：https://files.eric.ed.gov/ fulltext/EJ1056118.pdf。[11] F.S.C. da Silva、M. Finger & A.C.V. de Melo（2006）：计算机逻辑。圣智学习。可访问：https://books.google.com.br/books?id=w27uOgAACAAJ。[12] R.M. Smullyan（1995）：一阶逻辑。多佛高级数学丛书，多佛出版社，doi:10.1007/978- 3-642-86718-7。[13] J.N. de Souza（2008）：计算机科学逻辑。Campus SBC，爱思唯尔。可访问：https: //books.google.com.br/books?id=Y8GEsUoRKiEC。[14] D. R. Vasconcelos、R. Paula & M. V. Menezes（2022）：*NADIA* - 自然演绎证明助手。收录于：第*XXX*届计算机教育研讨会论文集，巴西计算机学会，阿雷格里港，南里奥格兰德州，巴西，第427–438页，doi:10.5753/wei.2022.222875。[15] Jørgen Villadsen、Andreas Halkjær From & Anders Schlichtkrull（2018）：自然演绎与 *Isabelle* 证明助手。收录于 Pedro Quaresma & Walther Neuper（编）：第6届国际研讨会论文集教育软件的定理证明组件，哥德堡，瑞典，2017年8月6日，理论计算机科学电子会议论文集267，开放出版协会，第140–155页，doi:10.4204/EPTCS.267.9。