arXiv:2603.07322v1 [cs.LO] 7 Mar 2026

一种自动生成证明复杂度相近的证明练习的方法

若昂·门德斯[1],若昂·马科斯[2],和 Patrick Terrematte[3]

[1]巴西利亚大学信息学研究生项目 [2]圣卡塔琳娜联邦大学哲学系[3]北里奥格兰德联邦大学大都会数字学院

摘要

自动生成练习可以大幅减少教育工作者用于手动设计练习的时间。然而，将此类自动化技术融入教学实践的一大障碍，在于如何控制机器生成练习的难度。本文提出一种自动生成证明复杂度相近的证明练习的方法。该方法以一道证明练习及一组可用于证明该练习的规则为输入，并生成一组证明复杂度与输入练习相近的证明练习。该方法聚焦于以一阶语言表述的数学证明练习，涵盖离散数学本科课程通常涉及的主题。我们通过考察用非形式证明解答这些练习所需的努力来评估其证明复杂度，并指出，这种努力可以通过不含逻辑符号的基于切规则的语义树证明来形式化刻画。支配这类证明的规则通过本文提出的一种可机械化提取程序获得。利用这些规则的分析性以及通过语义树规则构造的证明所固有的结构，我们推导出一种实现该方法的计算程序。

关键词： 自动问题生成、逻辑教学、基于切割的语义树、一阶经典逻辑、离散数学

1. 引言

除了课堂教学，教育工作者还经常承担大量工作，包括课程规划、提供课外学生支持、布置练习以及设计评估。在这些任务中，手动编写练习尤其耗时，也是教学工作量的重要组成部分。这一挑战推动了自动问题生成（AQG）工具的发展，其目标是减少制作教育材料所需的精力。AQG 系统从结构化（如数据库和本体）或非结构化（如连续文本）知识源中生成问题，同时允许教育工作者指定所需的问题属性，例如选择题中干扰项的数量 [10]。尽管取得了这些进展，确保

自动生成的练习符合教学要求，仍然是一项核心挑战。

缺乏能够精细控制证明复杂度的 AQG 工具，仍是其在教学场景中得到有效应用的一大障碍 [1, 3,10]。为解决这一局限，我们提出一种方法，用于自动生成证明复杂度相近的证明练习。本文所说的证明练习，是指陈述中包含可证猜想的练习。此类练习的证明复杂度通过本文提出的理论特定表格法框架进行评估。

理论特定的语义树的一个关键性质是，其证明中出现的公式不含逻辑符号。构造这些语义树所用的规则，可通过一种可机械化的程序，从某些封闭的一阶公理中提取出来。这些公理旨在刻画本科离散数学课程中通常讲授的一阶数学理论片段，例如集合论和数论，从而能够系统地推导出适用于课堂证明练习的可证猜想。

理论特定的语义树可视为所谓“KE 方法”的一种实例；本文将其称为基于切割的演算方法。该方法最初由 D’Agostino 和 Mondadori 为经典逻辑提出 [7, 8]。这一方法的一个核心特点是，基于切割的语义树能够以多项式复杂度模拟真值表程序，而 Smullyan 风格表格演算则不具备这一性质 [6]。这种计算优势源于将分支限制为解析切割规则的应用，从而更严格地控制证明的扩展。在解析切割中，切割公式通常必须是语义树中已有公式的子公式。事实上，在这一研究脉络的基础上， [5]，探讨了子公式性质的推广形式，以及更广义的解析性；其中的研究成果为本文所提出的进展提供了动机和依据。

本文的组织结构如下。第 2 节回顾了 AQG 中现有的难度控制方法，并指出本方法所要解决的不足。第 3 节介绍全文所采用的句法、演绎和语义概念。第 4 节介绍用于提取构造特定理论语义树所需规则的程序。接着，第 5 节定义证明复杂度的概念，并阐述采用特定理论语义树来确定证明复杂度的教学动机。第 6 节描述生成证明复杂度相当的练习题的方法。最后，第 7 节总结全文，并概述未来的研究方向。

2. 相关工作

多篇关于 AQG 的文献综述已经指出，缺乏具备难度控制功能的自动题目生成工具。在 [10], 中，作者对 2015 年至 2019 年间发表的 93 篇关于教育用途 AQG 的论文进行了综述。在这 93 篇论文中，只有 14 篇介绍了具备控制生成内容难度机制的方法。正如 [3], 所指出的，这一挑战并非新问题，因为对 1969 年至 2015 年间发表的 81 篇论文进行的类似分析也发现，能够生成难度等级可控问题的方法十分匮乏。另一项较新的研究 [1], 考察了将深度学习用于教育用途的 AQG 工具，并指出按难度等级定制这些工具是“未来研究的又一机会”。

从教学角度来看，难度控制是自动题目生成工具的一项关键功能。首先，它能确保生成的练习符合考生预期的熟练程度——

。其次，配备难度控制机制的 AQG 工具能够帮助教师构建个性化的形成性评估——换言之，即在课程期间进行的评估，旨在持续提供与每位学生掌握程度相匹配的反馈。这类工具还可以帮助开发计算机自适应测试，也就是根据考生对先前题目的作答表现来选择后续题目的动态个性化评估。第三，对于个性化评估，难度控制可以避免因给某些学生分配比其他同学更简单的问题而使他们受益。

尽管缺乏具有难度控制的AQG技术，文献中已经提出了一些与领域无关的方法。文本问题难度估计（QDET） [4, 2] 是自然语言处理中近期日益兴起的趋势，旨在自动估计从文本中抽取的问题的难度。在QDET中，通常的做法是提供一个机器学习模型，利用先前由人工做出的分类来预测问题的难度。

QDET以及一般机器学习模型的一个缺点是，依据人工判断对问题进行分类可能导致不一致，因为即使是专家也可能存在分歧，例如某个问题对学生而言是否难以回答。另一个缺点是，这些模型给出的难度分类可能并不附带解释。如第5节和第6节将详细说明的那样，这些缺点并不影响我们的方法，因为它不依赖于任何先前的人工判断，并且证明练习的难度分类由其最小证明加以解释。

关于与我们的研究范围更为密切相关、并提供可解释机制来控制题目难度的工具，一种用于判定逻辑相关证明练习难度的指标是其句法结构，如[14, 15]所示。在这些研究中，作者提出了一种生成命题逻辑公式的方法，通过命题原子的数量、联结词的数量以及公式的深度等参数，调节由此生成的练习的证明复杂度。这些参数是根据公式的抽象表示而非具体表示来衡量的。具体表示通常将公式表达为符号序列，而抽象表示则明确呈现数学对象的内部结构。例如，命题公式 φ 的抽象表示可以建模为一棵树，其根节点对应于 φ的主要联结词。尽管选择这类参数的理由可能是它们会影响相关真值表的规模，但这一思路并不容易推广到一阶语言。更重要的是，两个可证的命题逻辑公式 φ_1 和φ_2 可能具有相同的句法结构，但它们的证明结构可能大不相同。因此，学生证明 φ_1 时可能比证明 φ_2更困难，反之亦然。为了控制所生成练习的难度， [14, 15] 提出的方法仅考虑那些先于练习解答而存在、且不一定受解答影响的因素。

同样，在 [12], 中，作者利用涉及等式推理的代数证明练习的句法结构，生成范围相同且难度等级相近的其他证明练习。该系统还允许用户在生成过程中添加约束，以此控制难度。例如，如果输入练习是一个右边含有既约分数的等号，用户可以约束方法，使其只生成右边为既约分数的等号练习。在某些情况下，作者认为

其方法生成的练习可以通过“类似的证明策略”来证明。但作者没有对此提供进一步的细节。同样，由形式语言描述的练习的句法结构未必能决定其解答的规模，因此，赋予练习的难度等级可能无法反映解题所需的实际努力。

从某种意义上说，我们的方法与 [16], 中提出的方法更为接近。该方法生成数学文字题——也就是说，将练习编写成简短叙事，其中包含待求的未知数，且只能利用文本中提供的信息来确定其值——其难度等级可通过改变解出相应方程所需的算术运算次数来控制。一项针对 30 名学生开展的实验结果显示：增加练习中的运算次数会提高错误率，表明练习难度随之增加；而运算次数相同的练习，其错误率则没有显著差异。尽管这些结果涉及的是另一种类型的练习，但它们表明，在逻辑等形式学科中，以解答结构为基础来刻画练习难度，或许是一条更有前景的方向。

3. 预备知识

我们的方法旨在处理以一阶语言表述的证明练习。按照惯例，一阶语言由一阶基调指定，该基调包含一族函数符号和一族谓词符号，以及一个可数无穷的变元集合。命题联结词为 $\bot$、$\top$、$\neg$、$\wedge$、$\vee$、$\rightarrow$和 $\leftrightarrow$，一阶量词为 $\forall$ 和 $\exists$。原子公式，也称为理论特定公式，不包含量词或联结词。按照惯例，文字要么是原子公式或零元联结词，要么是它们的否定；子句则是文字的析取。如果文字是原子公式，则称其为正文字；否则称其为负文字。正文字的相反文字 l 是 $\neg l$ ，而负文字的相反文字 $\neg l$ 是 l。我们用 l^{op}表示文字 l 的相反文字。理论特定公式 sf 的抽象表示是一棵树，其根节点是 sf的谓词符号。对于任何其他标记为函数符号 f的节点，其第 i个子节点对应于 sf中 f 的第 i个参数。理论特定公式的深度sf 等于其抽象表示的高度。

一个带符号公式是前面带有“$+$”符号或“$-$”符号的公式。对于带符号公式 sf，我们用 $fmla(sf)$表示其基础公式，用$sign(sf$表示其符号。带符号公式的共轭是一个带符号公式 sf_1 ，满足 sf_2 ，使得$fmla(sf_1) = fmla(sf_2)$，且 $sign(sf_1) \neq sign(sf_2)$。

分析表 [13] 是一种演绎系统，其中证明表示为树，树的节点由公式标记。这些树通过表扩展规则进行扩展。表扩展规则定义为一种模式，由前提集和结论集组成，表示由该模式生成的所有规则实例。如果对于 T 的某个分支 θ，其中出现的一组公式 θ 与 r 的前提实例相匹配，则规则 r 适用于语义树 T。应用该规则时，通过添加 r 的备选结论所对应的实例来扩展分支 θ 。应用规则可能会增加语义树中的分支数量。本研究主要考虑线性扩展规则，即应用时不会引入额外分支的规则。由于第 4 节中介绍的程序，这些线性规则均为单一结论。

设 SF 为一个有限的带符号公式集， ER 为一组语义树扩展规则。若某个分支的节点恰好标记为 SF 中的带符号公式，则该分支构成一个.....

..... ER-语义树（针对 SF）；并且，如果 T 是针对 SF 的 ER-语义树，而 T* 是通过对 T 应用 ER 中的任一规则得到的，那么 T* 也是针对 SF 的 ER-语义树。ER-语义树中的分支若包含 $+\bot$ 或 $-\top$，或同时包含某个带符号公式及其共轭公式，则称为闭分支；否则称为开放分支。闭分支下方用符号 $*$ 标记。这些概念自然也适用于整个语义树：若一个 ER-语义树的所有分支均为闭分支，则称其为闭语义树；否则称为开放语义树。

公式集 $\Delta := \{\delta_1, \ldots, \delta_n\}$ 由公式集 $\Gamma := \{\gamma_1, \ldots, \gamma_m\}$ 得出的一个 ER-证明，是针对 $P := \{+\gamma_1, \ldots, +\gamma_m, -\delta_1, \ldots, -\delta_n\}$ 的一个闭合 ER-语义树。若存在由 Γ 得出 Δ 的 ER-证明，则称 Δ 可由 Γ ER-可证。我们将其记为 $\Gamma \vdash_{ER} \Delta$，并称之为 ER 所导出的后承关系。此外，称集合 P 是 ER-可反驳的。

语义树 T 中的节点 n 是节点 m 的后代节点，当 n 是因应用一条扩展规则而被加入 T 的结果，且该规则的前提要么包含 m 中的带符号公式，要么包含节点 m' 中的带符号公式，而 m' 本身是 m 的后代节点。在前一种情形中，n 称为 m 的直接后代节点。在语义树的分支 θ 中，带符号公式互为共轭的节点称为 θ 的闭合节点。若其中每个节点都是某个闭合节点的后代节点，则称该语义树是干净的。如第 6 节将表明的那样，由一组带符号公式构造证明的蛮力过程可能会自然地生成不干净的语义树。此类语义树与我们的目的无关。

我们采用的可满足性和有效性概念是经典一阶逻辑中的标准概念。此外，对于带符号公式，若模型满足 $+\varphi$，则称其满足 φ；若模型不满足 $-\varphi$，则称其满足 φ。若两个（带符号）公式由相同的模型满足，则称它们可满足性等价；该定义自然地扩展到（带符号）公式集合的情形。带符号公式 sf 满足$fmla(sf) = \varphi$时，可通过函数 $eq\ fmla$定义的转换 $eq\ fmla(sf) = \varphi$若 $sign(sf) = +$，则 $eq\ fmla(sf) = \neg\varphi$；否则，转换为与之可满足性等价的无符号公式。 -

范式用于将公式转换为具有特定形式且与之可满足性等价的公式。本文使用一种称为规则蕴含范式(RINF)的范式。RINF 中的公式具有形式 $(l_1 \wedge \cdots \wedge l_{n-1}) \rightarrow l_n$，其中 l_1、.....、l_n 均为文字。通过使用经典逻辑中的有效等价关系，例如德摩根律和替换定理，可以验证命题经典逻辑中的每个公式都能化为 RINF。此外，若集合 Γ 中的所有公式都是带符号理论特定公式，则称带符号公式集合 Γ 处于带符号理论特定范式(STSNF)。请注意，有些带符号公式集合无法转换为与之可满足性等价的 STSNF 集合。例如，集合 $S := \{+\varphi_1 \vee \varphi_2\}$无法化为 STSNF，因为任何集合 S'（其中 $sf \in S'$仅当 $fmla(sf) \in \{\varphi_1, \varphi_2\}$ 时），都与 S可满足性等价。

如果语义树的某个分支上的所有带符号公式构成的集合至少有一个是可满足的，则称该语义树是可满足的 。经典逻辑语义树系统的可靠性要求：对可满足的语义树应用某条规则后，所得语义树仍然可满足。这意味着，对于线性扩展规则 r，如果前提 r 可满足，那么其（单个）结论 r也可满足。因此，以一组带符号公式 SF 为前提、以一个带符号公式 sf 为结论的线性扩展规则 r 是可靠的，当且仅当 $\varphi :=$ $\bigwedge_{s \in SF} eq\ fmla(s) \rightarrow eq\ fmla(sf)$ 可满足。请注意，φ位于RINF。在这种情况下，规则 r 被称为 φ 的对应规则，而 φ 是 r的对应公式。因此，如果某条规则的对应公式可满足，我们也可以说该规则可满足。

4. 理论特定证明

我们打算借助节点标记为理论特定带符号公式的、基于切割的语义树证明来计算练习的证明复杂度。为实现这一目标，需要先给出一些定义。特别是，我们将此类语义树证明称为理论特定证明，将支配它们的规则称为理论特定规则。

理论特定规则取自定义理论。定义理论则是一种结构 $\langle L, Ax, < \rangle$，其中 L 是一阶语言， $Ax \subseteq L$是不包含 $\bot$或 $\top$的有限闭合公式集，我们将其称为定义公理，而 $<$ 是 L 的谓词符号上的良基关系。

例1。 这里，我们介绍一种定义理论，整篇论文都将以它作为贯穿始终的例子：集合的定义理论，简称为集合论 。它旨在让我们能够从集合论中推导出部分定理。集合的定义理论的语言由签名 $\Sigma := \langle FS, PS \rangle$定义，其中：

- Σ的唯一常量符号（零元函数符号）是“$\emptyset$”。

Σ的唯一一元函数符号是“$\complement$”、“fst”和“snd”。

- Σ 的唯一二元函数符号是“$\cup$”、“$\cap$”、“$\setminus$”、“$\times$”和“$\triangle$”。
- FS_i 对于任意 $i > 2$，均为空。
- Σ的唯一二元谓词符号是“$\in$”、“$\subseteq$”和“$\between$”。
- PS_i 对于任意 $i \neq 2$，均为空。

对于二元符号，我们采用中缀记法，并在没有歧义时省略括号。定义公理集 Ax 由以下公式给出：

- $[ax\emptyset]$ $\emptyset$的定义公理：$(\forall x)(\neg x \in \emptyset)$。
- $[ax\complement]$ $\complement$的定义公理：$(\forall x, y)(x \in \complement(y) \leftrightarrow \neg x \in y)$。
- $[ax\cup]$ $\cup$的定义公理：$(\forall x, y, z)(x \in y \cup z \leftrightarrow (x \in y \vee x \in z))$。
- $[ax\cap]$ $\cap$的定义公理：$(\forall x, y, z)(x \in y \cap z \leftrightarrow (x \in y \wedge x \in z))$。
- $[ax\setminus]$ $\setminus$的定义公理：$(\forall x, y, z)(x \in y \setminus z \leftrightarrow (x \in y \wedge \neg x \in z))$。
- $[ax\times]$ $\times$的定义公理：$(\forall x, y, z)(x \in y \times z \leftrightarrow (fst(x) \in y \wedge snd(x) \in z))$。
- $[ax\triangle]$ $\triangle$的定义公理：$(\forall x,\ y,\ z)(x \in y \triangle z \leftrightarrow ((\neg x \in y \rightarrow x \in z) \wedge (x \in y \rightarrow \neg x \in z)))$.

- $[ax\subseteq]$ $\subseteq$的定义公理：$(\forall x,\ y)(x \subseteq y \leftrightarrow (\forall z)(z \in x \rightarrow z \in y))$.
- $[ax\between]$ $\between$的定义公理：$(\forall x,\ y)(x \between y \leftrightarrow (\forall z)\neg(z \in x \wedge z \in y))$.

唯一包含一元符号 fst and snd 的定义公理是$[ax\times]$。直观而言，我们希望当 x 是一个序对（a，b），且满足 $a \in y$ 和 $b \in z$ 时， $x \in y \times z$ 成立。为了避免涉及序对的公理，我们使用 fst 来还原序对的第一元素，并使用 snd 来还原序对的第二元素。最后，集合论的良基关系 $<$ 定义如下：当且仅当 x 是隶属关系的谓词符号（即 $\in$），且 y 是其他谓词符号之一（即 $\subseteq$ 或 $\between$）时，$x < y$ 成立。

有人可能会疑惑，为什么 $\times$ 的定义公理不是公式 $(\forall x_1,\ x_2,\ y,\ z)\ ((x_1,\ x_2) \in y \times z \leftrightarrow (x_1 \in y \wedge x_2 \in z))$。正如第 5 节所解释的，这里采用的证明可比性概念是基于不含逻辑符号的证明之间的同构。此类证明由从相应定义公理中提取的规则构造。通过表述原子子公式彼此同构的定义公理，我们提高了获得同构证明的可能性。

给定定义理论 $\langle L,\ Ax,\ <\rangle$，我们希望能够以定义公理为基础假设，证明关于语言 L 的猜想。从这个意义上说，定义公理可以被视为零前提规则。然而，出于第 5 节讨论的教学原因，证明最好不包含逻辑符号。因此，我们需要提取一些规则，使其作用与这些定义公理完全相同，并且其前提和结论中都不含逻辑符号。在说明如何提取这些规则之前，我们先详细介绍它们的结构。

在 Smullyan 风格表格演算中，解析性通常依据直接子公式来理解：公式通过分解其直接组成部分进行组合式分析。由于理论特定公式是原子的，这种标准的解析性概念不足以适用于理论特定证明，因此必须加以扩展。在 [5], 中，提出了一种广义的解析性概念：公式 φ 的句法复杂度由与 φ相关的公式决定；这些公式可以说比 φ 的句法复杂度更低，但不一定是 φ 的直接子公式。实际上，对于理论特定证明，我们也采用类似的思路：规则的单一结论由前提集中已经出现的项构造而成；通过定义理论中谓词符号上的良基关系与理论特定公式的句法深度之间的相互作用，确保句法复杂度得以降低。下文将定义线性扩展规则（我们称之为理论特定规则），并说明如何从定义公理中提取这些规则。基于这些线性扩展规则，我们还将引入一族适用于理论特定证明的切割规则，作为一般割模式的实例。

定义理论 $Th := \langle L$ 的理论特定规则 $Ax, <\rangle$,是由一组理论特定带符号公式（前提）和一个理论特定带符号公式（结论）组成的序对，并须满足以下三项理论特定分析性限制：

1. 结论中的变元必须出现在前提集中。2. 根据关系 $<$，结论中的谓词符号必须小于至少一个前提中的谓词符号。3. 如果前提集和结论中的所有谓词符号都相同，那么结论中公式的深度必须小于至少一个前提中公式的深度。

由于第一项限制，对语义树 T 应用理论特定规则时，每次应用只会产生有限多个实例，因为添加到 T 中的结论受到约束，其中只能包含 T 中已出现的符号。第二项和第三项限制确保我们以某种方式化简证明中的公式。图1展示了集合论中的三个规则模式的应用。在每种情况下，横线上方的带符号公式表示规则的前提，横线下方的带符号公式表示结论。其中，只有模式 (c) 符合理论特定规则的要求：模式 (a) 违反了第一项和第三项限制，而模式 (b) 违反了第二项限制。

$$\text{(a)}\ \frac{+x \in y}{+x \in y \cup z} \qquad \text{(b)}\ \frac{+z \in x \quad -z \in y}{-x \subseteq y} \qquad \text{(c)}\ \frac{-z \in y \quad +x \subseteq y}{-z \in x}$$

图1：集合的定义理论的规则模式。

在经典逻辑中，要证明由规则集$ER \cup \{r_1\}$ 所诱导的后承关系与由规则集 $ER \cup \{r_2\}$所诱导的后承关系相同，只需证明 r_1 和 r_2 可满足性等价。对于每个定义公理，我们的策略是将其转换为规则蕴含范式中的一个等可满足公式集合 Γ。这一转换首先将公理转换为前束范式 (PNF)，然后进行斯科伦化，再将所得结果转换为合取范式 (CNF)，最后将所得公式集合改写为 RINF。

设 ax 是一个定义公理。我们首先将 ax 转换为 PNF；如有必要，对所得公式进行斯科伦化，然后将其矩阵——也就是移除所有前置量词后得到的公式的无量词部分——转换为 CNF。由此，我们得到一个子句集 Γ，它与 ax 可满足性等价。每个子句 $\varphi := l_1 \vee \cdots \vee l_n$ 含有 $n > 1$ 的 n 个文字，可转换为 RINF 中的 n 个公式，即 $\varphi_1 := (l_2^{\mathrm{op}} \wedge \cdots \wedge l_n^{\mathrm{op}}) \rightarrow l_1$，……，$\varphi_n := (l_1^{\mathrm{op}} \wedge \cdots \wedge l_{n-1}^{\mathrm{op}}) \rightarrow l_n$，使得 φ、φ_1、……、φ_{n-1} 和 φ_n 均可满足性等价。从 RINF 中的这些公式 φ_i（对于 $1 \leq i \leq n$）提取出的规则集，包含与这些公式对应的规则。这些规则的前提集和结论中均不含逻辑符号，并且满足理论特定分析性限制。

仅含一个负文字的子句 $\neg l$ 可以转换为两个在 RINF 中等可满足的公式，即：$\varphi_1 := l \rightarrow \bot$ 和 $\varphi_2 := \top \rightarrow \neg l$。$\varphi_1$ 的对应规则为 $r_1 := \langle \{+l\}, +\bot \rangle$，该规则指出，只要某个分支包含带符号公式 $+l$，就可以将其闭合。因此，规则 r_1 表示为 $\langle \{+l\}, * \rangle$，并加入提取的规则集合中。$\varphi_2$ 的对应规则为 $r_2 := \langle \{+\top\}, -l \rangle$。该规则允许在任意阶段将 $-l$ 引入任何分支，从而产生无限多个实例。因此，提取的规则中不包含 r_2。如果该子句中的单个文字为正，则提取的规则为 $\langle \{-l\}, * \rangle$。

在提取过程中，我们本可以采用比 RINF 更一般的范式，即：$(l_1 \wedge \cdots \wedge l_i) \rightarrow (l_{i+1} \wedge \cdots \wedge l_n)$，其中 l_1、……、l_i、……、l_n 是文字。在这种情况下，我们将能够提取具有多个结论的线性规则。然而，在这种一般范式中，能够转换为包含 n 个文字的子句的公式数量大约为 $n!$ 这就是我们将理论特定规则限定为线性规则的原因。

在本文余下部分，每当我们提到“定义理论 Th 的理论特定规则集合”时，指的都是按照上述方法，从 Th 的定义公理中提取出的理论特定规则集合。

示例 2. 考虑集合论中的定义公理 $[ax \subseteq]$。将其转换为前束范式并进行斯科伦化后，结果为 $(\forall x,\ y,\ z)((f(x,\ y) \in x \rightarrow f(x,\ y) \in y) \rightarrow x \subseteq y) \wedge (x \subseteq y \rightarrow (z \in x \rightarrow z \in y))$，其中 f 是一个斯科伦函数符号。随后，将该公式的矩阵转换为合取范式，我们得到以下子句集合：$\{f(x,\ y) \in x \vee x \subseteq y, \neg f(x,\ y) \in y \vee x \subseteq y, \neg x \subseteq y \vee (\neg z \in x \vee z \in y)\}$。从 $[ax \subseteq]$以及集合论中的其他定义公理中提取出的理论特定规则集合见图 2。我们将此集合称为 TS_{sets}。请注意，函数符号 g 也是一个斯科伦

函数符号。

$$\frac{+x \in \emptyset}{*}\ +\emptyset \qquad \frac{+x \in \complement(y)}{-x \in y}\ +\complement\mathbf{E} \qquad \frac{-x \in y}{+x \in \complement(y)}\ -\complement\mathbf{E}$$

$$\frac{-x \in y}{+x \in y \cup z}\ +\cup\mathbf{E}_1 \qquad \frac{-x \in z}{+x \in y \cup z}\ +\cup\mathbf{E}_2 \qquad \frac{-x \in y \cup z}{-x \in y}\ -\cup\mathbf{E}_1 \qquad \frac{-x \in y \cup z}{-x \in z}\ -\cup\mathbf{E}_2$$

$$\frac{+x \in y \cap z}{+x \in y}\ +\cap\mathbf{E}_1 \qquad \frac{+x \in y \cap z}{+x \in z}\ +\cap\mathbf{E}_2 \qquad \frac{\begin{array}{c}+x \in y\\ -x \in y \cap z\end{array}}{-x \in z}\ -\cap\mathbf{E}_1 \qquad \frac{\begin{array}{c}+x \in z\\ -x \in y \cap z\end{array}}{-x \in y}\ -\cap\mathbf{E}_2$$

$$\frac{+x \in y \setminus z}{+x \in y}\ +\setminus\mathbf{E}_1 \qquad \frac{+x \in y \setminus z}{-x \in z}\ +\setminus\mathbf{E}_2 \qquad \frac{\begin{array}{c}+x \in y\\ -x \in y \setminus z\end{array}}{+x \in z}\ -\setminus\mathbf{E}_1 \qquad \frac{\begin{array}{c}-x \in z\\ -x \in y \setminus z\end{array}}{-x \in y}\ -\setminus\mathbf{E}_2$$

$$\frac{+x \in y \times z}{+x \in fst(y)}\ +\times\mathbf{E}_1 \qquad \frac{+x \in y \times z}{+x \in snd(z)}\ +\times\mathbf{E}_2 \qquad \frac{\begin{array}{c}+x \in fst(y)\\ -x \in y \times z\end{array}}{-x \in snd(z)}\ -\times\mathbf{E}_1 \qquad \frac{\begin{array}{c}+x \in snd(z)\\ -x \in y \times z\end{array}}{-x \in fst(y)}\ -\times\mathbf{E}_2$$

$$\frac{-x \in y}{+x \in y \triangle z}\ +\triangle\mathbf{E}_{1a} \qquad \frac{-x \in z}{+x \in y \triangle z}\ +\triangle\mathbf{E}_{1b} \qquad \frac{+x \in y}{+x \in y \triangle z}\ +\triangle\mathbf{E}_{2a} \qquad \frac{+x \in z}{+x \in y \triangle z}\ +\triangle\mathbf{E}_{2b}$$

$$\frac{-x \in y}{-x \in y \triangle z}\ -\triangle\mathbf{E}_{1a} \qquad \frac{-x \in z}{-x \in y \triangle z}\ -\triangle\mathbf{E}_{1b} \qquad \frac{+x \in y}{-x \in y \triangle z}\ -\triangle\mathbf{E}_{2a} \qquad \frac{+x \in z}{-x \in y \triangle z}\ -\triangle\mathbf{E}_{2b}$$

$$\frac{\begin{array}{c}+z \in x\\ +x \subseteq y\end{array}}{+z \in y}\ +\subseteq\mathbf{E}_1 \qquad \frac{\begin{array}{c}-z \in y\\ +x \subseteq y\end{array}}{-z \in x}\ +\subseteq\mathbf{E}_2 \qquad \frac{-x \subseteq y}{+f(x,y) \in x}\ -\subseteq\mathbf{E}_1 \qquad \frac{-x \subseteq y}{-f(x,y) \in y}\ -\subseteq\mathbf{E}_2$$

$$\frac{\begin{array}{c}+z \in x\\ +x \between y\end{array}}{-z \in y}\ +\between\mathbf{E}_1 \qquad \frac{\begin{array}{c}+z \in y\\ +x \between y\end{array}}{-z \in x}\ +\between\mathbf{E}_2 \qquad \frac{-x \between y}{+g(x,y) \in x}\ -\between\mathbf{E}_1 \qquad \frac{-x \between y}{+g(x,y) \in y}\ -\between\mathbf{E}_2$$

图 2：从集合论中提取出的理论特定规则集合。

有些情况下，无法从给定的定义公理中提取出理论特定规则。例如，考虑定义公理 $(\forall x,\ y,\ z)(x\ \in y \cup z \rightarrow x \in z \cup y)$。与该公理可满足性等价的 RINF 公式有 $x \in y \cup z \rightarrow x \in z \cup y$ 和 $\neg x \in z \cup y \rightarrow \neg x \in y \cup z$。在这两种情况下，对应规则都违反了第三条理论特定分析性限制。尽管如此，对于此处给出的集合论定义公理，都至少可以提取出一条理论特定规则。

现在，我们为理论特定证明介绍一种替代版本的切割。为此，我们首先给出两个辅助定义。对于第一个定义，设 r 是至少有两个前提的理论特定规则。若一个前提 $prem$ 属于 r，且包含主前提属于 r，如果它包含 r的前提中出现的所有变元，则称该前提为其主前提。例如，$-x \in y \cap z$ 是规则 $-\mathbf{E}_1$ 和 $-\mathbf{E}_2$ 中的主前提，这些规则来自 $TS\ sets$。尽管 $TS\ sets$ 中没有任何规则包含多个主前提，但原则上这种情况可能出现在其他理论中。

对于第二个定义，设 sf_1 和 sf_2 是两个带符号公式，ER 是一个扩展规则集。如果存在一条至少有两个前提的规则 $r \in ER$，使得 sf_1 和 sf_2 可以分别实例化 r中的两个不同前提，那么我们称 sf_1 和 sf_2 是 ER 中的共前提。如果 sf_1 是 r的主前提，那么 sf_2 称为次要

共前提 sf_1 和 sf_1 的主要共前提 sf_2。例如，在系统TS_{sets}中，带符号公式 $-x \in y \cap (w \cup z)$ 是 $+x \in w \cup z$的主要共前提，而 $-x \in w$ 是 $+x \in w \cup z$的次要共前提。

现在，我们可以为一组理论特定规则 TS 定义一族切割规则，如下所示：

$$\frac{}{+\varphi \mid -\varphi}\ cut$$

Either $+\varphi$ or $-\varphi$ is a minor co-premise in TS of a node above

请注意，根据第一条理论特定分析性限制，每条理论特定规则都只允许有限个实例。因此，对于任意一组理论特定公式ER 以及任意一个 ER-语义树 T，T 中能够作为规则 r 在 ER 中的主前提实例的带符号公式集合也是有限的。由于切割公式必须是这组理论特定规则中的次要共前提，因此可容许切割实例的集合也是有限的。

在 TS_{sets} 图 3 的证明中，我们通过一次切割推导出节点 (5) 和 (8)。在理论特定规则的提取程序下，这个特定的切割实例可用于 TS_{sets}，因为 $+v \in x \cup w$符合 $-v \in (x \cup w) \cap (y \cup z)$的次要共前提条件。

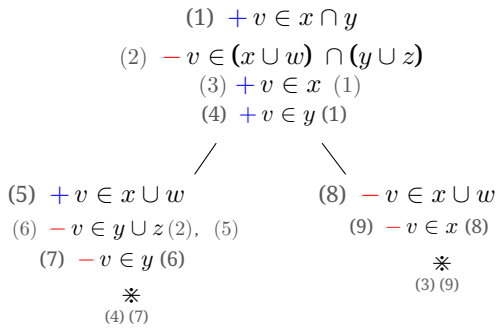


图 3： $\{+v \in x \cap y, -v \in (x \cup w) \cap (y \cup z)\}$ 的一个 TS_{sets}-证明。

5. 证明练习的证明复杂度

练习的复杂度分类应以解题所需的努力为依据。因此，若要声称两个证明练习具有可比的证明复杂度水平，就必须提供一种有原则的方法来确定证明它们的复杂程度。本节首先解释，为何第 4 节中介绍的理论特定证明，是直接将非形式证明翻译为形式证明的合适替代方案。随后，我们将为理论特定证明引入证明复杂度的形式化定义，从而能够确定两个带符号公式的可反驳集何时具有可比的证明复杂度水平。我们将利用这一证明复杂度概念，控制第 6 节所述程序生成的练习的难度等级。

5.1. 形式化非形式证明

高等数学的初学者通常接受通过非形式证明来证明定理的训练。他们学习如何运用定理的已知条件和目标

，借助以句子表达的证明策略，这些句子结合了自然语言和数学语言。随后，这些句子被编排成连续文本——即一份非形式证明——其结构虽然逻辑连贯，却缺乏明确的形式组织。事实上，正如 Lamport 在 [11], 中所指出的，遵循十九世纪数学中常见的叙述性散文风格，往往会掩盖证明的逻辑结构，并隐藏验证证明所必需的依赖关系。

如果我们想确定两个证明练习何时具有可比的证明复杂度水平，一个重要的问题是：我们如何设计一种方法，精确衡量这类非结构化对象的证明复杂度？接下来，我们将论证，第 4 节中介绍的理论特定证明可以视为对非形式证明的形式化。利用形式证明中固有的树结构，我们便可以定义一种计算证明复杂度的方法，详见第 5.2 小节。

学生通常在没有接受形式逻辑训练的情况下开始学习数学证明。因此，他们学着撰写的非形式证明，往往会将其所采用的证明策略背后的一些逻辑步骤隐去。这一观察在我们的提议中发挥着核心作用，也是我们采用理论特定证明作为非形式证明的形式对应物的依据。以图 4 所示的“元素论证”为例，它被称为“集合论中的基本证明技巧” [9]:

Element Argument: The Basic Method for Proving That One Set Is a Subset of Another

Let sets X and Y be given. To prove that $X \subseteq Y$,

1. **suppose** that x is a particular but arbitrarily chosen element of X,
2. **show** that x is an element of Y.

图 4: [9] 中对“元素论证”的描述。

这种证明策略通常在不明确说明背后原理的情况下使用：用逻辑术语来说，包含关系被定义为一个处于前束范式的量化公式，其矩阵是一个蕴含，连接的是涉及隶属关系的断言。

非形式证明中的逻辑结构往往隐而不显，另一个例子见图5所示的摘录，该摘录出自一个证明：两个连续整数的奇偶性相反。从形式角度看，从 $m = 2k$ 到 $m + 1 = 2k + 1$ 这一步依赖于同一者可替换性；更准确地说，它依赖于等号是一种同余关系，而加法（或者实际上是此处的后继）与之相容。然而，证明中并未明确说明这一步的依据。

非形式证明中常常省略逻辑步骤，并不意味着它们总是被隐藏起来。例如，请看图6所示的例子，其中证明了对于所有集合 A 和 B， $A \cap B \subseteq A$ 。请注意，交集定义中的合取由红色矩形标出的词语“and”体现。这反映了交集的通常定义，即“ $x \in X \cap Y$ 当且仅当 $x \in X$ 且 $x \in Y$ ”。然而，现在请考虑以下三个单向陈述所给出的交集的另一种定义：

1. 如果 $x \in X \cap Y$，那么 $x \in X$。

Theorem 4.4.2 The Parity Property

Any two consecutive integers have opposite parity.

Proof:

Suppose that two *[particular but arbitrarily chosen]* consecutive integers are given; call them m and $m+1$. *[We must show that one of m and $m+1$ is even and that the other is odd.]* By the parity property, either m is even or m is odd. *[We break the proof into two cases depending on whether m is even or odd.]*

Case 1 (m is even): In this case, $m = 2k$ for some integer k, and so $m+1 = 2k+1$, which is odd *[by definition of odd]*. Hence in this case, one of m and $m+1$ is even and the other is odd.

图5：关于 [9] 任意两个连续整数的奇偶性相反的证明片段。

2. 如果 $x \in X \cap Y$，那么 $x \in Y$。

3. 如果 $x \in X$，那么如果 $x \in Y$，那么 $x \in X \cap Y$。

在这个例子中，也可以直接根据$x \in A \cap B$ 得出 $x \in A$ ，而不必明确写出与合取相对应的“且”。

Example 6.2.1 Proof of a Subset Relation

Prove Theorem 6.2.1(1)(a): For all sets A and B, $A \cap B \subseteq A$.

Solution We start by giving a proof of the statement and then explain how you can obtain such a proof yourself.

Proof:

Suppose A and B are any sets and suppose x is any element of $A \cap B$.
Then $x \in A$ and $x \in B$ by definition of intersection. In particular, $x \in A$.
Thus $A \cap B \subseteq A$.

图6：改编自 [9]的证明：对于所有集合 A 和 B， $A \cap B \subseteq A$ 。

5.2. 证明复杂度的可比性

如果我们将理论特定证明视为非形式证明的形式化对应物，那么证明练习陈述的逻辑描述会是什么？假设 E是一个证明练习。我们可以说， E 已知条件中的公式构成集合$\Gamma :=\{\gamma_1,\ldots,\gamma_m\}$而 E目标中的公式构成集合 $\Delta :=\{\delta_1,\ldots,\delta_n\}$。那么，后承陈述 $\Gamma \vdash \Delta$ 以及由此得到的可反驳集 $\{+\gamma_1,\ldots,+\gamma_m,-\delta_1,\ .....,\ -\delta_n\}$可以作为 E 的逻辑描述。不过请注意，在逻辑上描述 E 的公式集合可能包含命题联结词，因此无法转换为STSNF中的等可满足集。此类练习不属于我们的方法的适用范围。在本小节中，为了能够说明两个证明练习何时具有可比的证明复杂度水平，我们提出了理论特定带符号公式的可反驳集之间证明复杂度可比性的概念。

设 ER 是定义理论 Th 的一组理论特定规则， SF_1和 SF_2 是 L 的两组可由 ER 反驳的理论特定带符号公式。直观地说，如果证明 SF_1 和 SF_2 所需的最小努力能够得到结构相同的 ER-证明，那么我们就希望将这两组公式视为具有可比的

证明复杂度层级。我们首先定义证明之间的同构概念。为此，我们借助表格证明所具有的树结构，以便利用树之间的同构概念。

对能够充分体现其演绎可比性的语义树证明而言，精确定义它们之间的同构不能仅依赖相关语义树的树结构。比较图 7 所示的语义树证明时，这一点便显而易见：二者都是使用规则集 TS_{sets} 构造的。尽管每个语义树都包含 6 个节点，但它们的构造方式略有不同：在 (a) 中，双前提规则被应用了一次，而在 (b) 中则没有使用此类规则。此外，语义树 (a) 是干净的，而语义树 (b) 不是。因此，我们在下文引入的演绎同构这一形式概念，是建立在以下两种互补的同构形式之上的，二者分别定义于公式层面和我们称为“论证树”的层面。

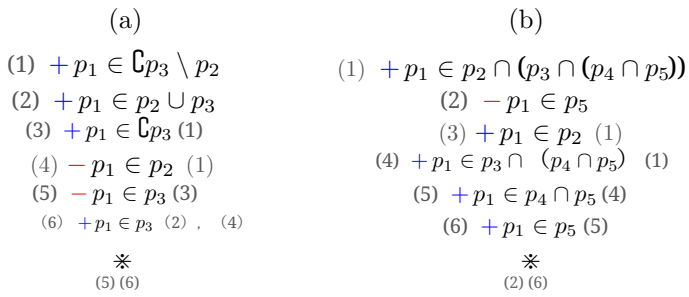


图 7: (a) 的表格证明 $\{+p_1 \in \complement p_3 \setminus p_2, +p_1 \in p_2 \cup p_3\}$ 以及 (b) $\{+p_1 \in p_2 \cap (p_3 \cap (p_4 \cap p_5))\ -p_1 \in p_5\}$。

给定一种语言 L，若 L 中的两个理论特定公式 φ_1 和 φ_2 的 φ_1 的抽象表示与 φ_2 的抽象表示句法同构，且 φ_1 的变元集合与 φ_2的变元集合相等，则称它们句法同构。我们将此定义扩展到带符号公式：若两个带符号公式 sf_1和 sf_2中的 $fmla(sf_1)$与 $fmla(sf_2)$句法同构，则称这两个带符号公式句法同构。

考虑以下三个集合论公式： $\varphi_1 := x \subseteq y \cap z$,$\varphi_2 := z \in x \cap y$ 和 $\varphi_3 := x \subseteq x \cup y$。公式 φ_1 和 φ_2 句法同构，但它们与 φ_3并不同构，因为变元 z同时出现在 φ_1 和 φ_2中，却没有出现在 φ_3中。如果我们不要求变元集合相等，那么一个理论特定公式可能会有无限多个与之句法同构的公式。因此，我们将无法像在第 6 节中那样，实现一个能够搜索与给定公式句法同构的所有公式的程序。

设 ER 是某个定义理论的一组理论特定规则。对于一个 ER-证明 P 的分支 θ，θ 的论证树 J 以一个内容为符号 $\divideontimes$ 的节点为根节点，而 J 中的其他节点都是 θ 的节点。此外，根节点的子节点是使分支 θ 在 P 中闭合的节点。对于 J 中的任意其他节点 i， J 中它的子节点是为将 i 加入 θ 提供依据的节点。给定一个包含分支 $\theta_1, \ldots, \theta_n$ 的 ER-证明 P，若 $J_1, \ldots, J_n$ 分别是 $\theta_1, \ldots, \theta_n$ 的论证树，我们也可以说 $J_1, \ldots, J_n$ 是 P 的论证树。在图8中，我们展示了一个关于 $\{+x \in y \cap z, -x \in y\}$ 的证明、一个关于 $\{+x \in y \setminus z, -x \in y\}$ 的证明，以及它们各自的论证树。

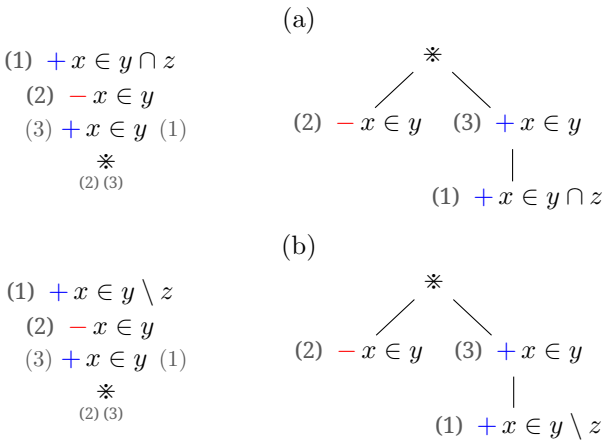


图8：(a) $\{+x \in y \cap z, -x \in y\}$ 的证明及其论证树，以及 (b) $\{+x \in y \setminus z, -x \in y\}$ 的证明及其论证树。

设 θ_R 和 θ_L 是理论特定语义树证明的两个分支。给定分别对应于 θ_R 的论证树 R 和对应于 θ_L 的论证树 L，如果 R 与 L 之间存在一种同构，且该同构保持其公式的句法同构关系，我们就称 R 与 L 是演绎同构的。我们将这一概念扩展到证明：若存在 P_1 与 P_2 的分支之间的一一对应，将长度相同且论证树演绎同构的分支对应起来，则称两个理论特定证明 P_1 与 P_2 是演绎同构的。在图8中，这两个证明是演绎同构的。

证明的演绎同构取决于其论证树的演绎同构，因为可能存在语义树同构但证明结构不同的情况。例如，可以通过应用规则 $\{+x \in y \setminus y\}$ 构造出一个语义树证明 $+\setminus \mathbf{E}_1$ 和 $+\setminus \mathbf{E}_2$。该语义树与图 8 中展示的语义树同构，但后者仅通过一次规则应用构造而成。

设 ER 是某个定义理论的一组理论专属扩展规则。ER-证明的分支 θ 的演绎规模 由其论证树的节点数给出，而 ER-证明的演绎规模 则由其各分支的规模之和给出。图 8 中证明的演绎规模为 4，图 3 中证明的演绎规模为 13。

设 ER 是定义理论 $\langle L,\ Ax,\ <\rangle$ 的理论特定扩展规则集合。给定 L 的理论特定带符号公式集合 SF，对于 SF 而言，一个干净的 ER-证明 P 若不存在其他针对 SF 的干净 ER-证明，其演绎规模小于 P 的演绎规模，则称其为极小证明。一个带符号公式集合的最小证明集合为确定其证明复杂度提供了基础。

现在，我们已经拥有了一些工具，可以将集合之间证明复杂度可比性的概念形式化。设 ER 是定义理论 $\langle L,\ Ax,\ <\rangle$ 的一组理论专属扩展规则。对于 L 的任意两组理论特定带符号公式 SF_1 和 SF_2，当满足以下条件时，我们称 SF_1 和 SF_2 具有可比的证明复杂度水平：(i) 存在一个针对 SF_1 的 ER-证明 P_1，该证明是最小的；并且存在一个针对 SF_2 的 ER-证明 P_2，该证明同样是最小的，且 P_1 与 P_2 演绎同构；(ii) SF_1 与 SF_2 之间存在双射对应，将带有相等符号的句法同构带符号公式彼此对应。在这种

情况下， SF_1 和 SF_2 也被称为证明同构。

请回想一下，我们使用带符号公式的集合来表示后承关系，而后承关系又对证明练习的表述作形式化描述。前一定义中的子句 (ii) 旨在保证具有可比的证明复杂度水平的证明练习，其已知项与目标的结构得以保留。

图8所示的证明是最简的，因为它们仅通过一次扩展规则的应用构造而成，并且对于这些带符号公式的集合，不存在不应用扩展规则的证明。此外，与之关联的论证树在演绎上同构。因此， $\{+\,x \in y \cap z, -\,x \in y\}$ 和$\{+\,x \in y \setminus z, -\,x \in y\}$ 具有可比的证明复杂度水平。

6. 具有可比证明复杂度水平的证明练习生成程序

请回想一下，我们用 STSNF 中的带符号公式集合对证明练习作形式化描述。给定一个定义理论 $Th := \langle L,\ Ax,\ < \rangle$，我们提出的用于自动生成具有可比证明复杂度水平的证明练习的程序，其输入为 L 的一组带符号公式（即该练习的逻辑描述），以及 Th 的一组理论特定规则。接着，我们将该程序分为两个主要步骤:

1. 搜索最小证明，为确定输入练习的证明复杂度提供依据。2. 搜索带符号公式的证明同构集合。每个带符号公式集合都表示证明复杂度级别与输入逻辑描述所对应的证明练习相当的练习。

值得一提的是，我们的方法生成的并非严格意义上的问题，而是讲师可以用来为学生设计问题的数学对象。这并不妨碍将这种方法视为一种 AQG 方法。设 c 为该工具的一个输出。我们可以假设，该工具实际生成的是“ c 的证明可以是什么？”这一格式的问题。此外，知识源可以视为我们用于提取证明的理论特定规则集合；而输出所需具备的属性，也就是其证明复杂度级别，则包含在输入练习中。

在接下来的小节中，我们将介绍生成程序的两个主要步骤。我们已开发出该程序的原型实现，可通过 https://github.com/joaomendesln/GePECC 获取。

6.1. 搜索极小证明

在给出程序之前，我们先引入一个额外的辅助定义。设ER 为一组理论特定规则， SF 为一组带符号公式。我们将集合 $T_n(ER,\ SF)$定义为：对于 SF，所有应用ER中的规则（包括针对 ER的切规则）且含有 n 次规则应用的 ER-语义树所构成的集合。由于 ER 中的规则和针对 ER 的切规则都只有有限个实例，因此集合 $T_n(ER,\ SF)$是有限的。

给定定义理论 $Th := \langle L,\ Ax,\ < \rangle$、一组针对 Th 的理论特定规则 ER，以及 L 的一组 ER-可反驳带符号公式 SF，该程序输出 SF 的所有极小 ER-证明。它首先构造集合 $T_1(ER,\ SF)$，接着构造集合 $T_2(ER,\ SF)$，依此类推。对于每个集合 $T_n(ER,\ SF)$，我们检查 $T_n(ER,\ SF)$ 中的每棵语义树是否为 SF 的 ER-证明。找到 SF 的第一个 ER-证明 P 时，将 P 的演绎规模设为上界，并将 P

加入极小证明的输出集 M。集合 $T_i(ER,\ SF)$ 的构造继续进行；每当找到 SF 的一个 ER-证明 S 时，就更新极小证明集。如果 S 的演绎规模等于当前上界，则将 S 加入 M。如果 S 的演绎规模小于当前上界，则清空 M，将 S 加入其中，并将新的上界设为 S 的演绎规模。构造集合 $T_{up}(ER,\ SF)$ 后，程序终止，此时 up 为当前上界。

关于程序的终止性，我们依次检查 $T_n(ER,\ SF)$ 集合中所有元素是否存在证明。由于这些集合始终是有限的，因此它们的构造会在有限时间内终止，所以这一步不会造成问题。此外，在搜索最小证明的过程中，该程序必定最终会设定一个上界，因为程序的前置条件是输入的带符号公式集合 SF 是 ER-可反驳的。

该程序正确性面临的一种威胁是，存在一个未被添加到输出集的极小证明；也就是说，对于带符号公式集合 SF 和理论特定规则集合 ER，集合 $T_n(ER,\ SF)$ 中存在一个大小为 m 的证明 P，其中$n > m$，且 m 是输出集中的证明大小。如果 P 的某些节点不属于其任何论证树，就可能出现这种情况。然而在这种情况下，证明 P 并不干净，因此根据定义，P 不可能是极小证明。

6.2. 搜索带符号公式的证明同构集合

给定从语言L的带符号公式集合 SF 中提取的极小 ER证明P，我们给出一个程序，用于搜索 L 的所有带符号公式集合，这些集合具有与P演绎同构的极小证明。为了举例说明一种简单但未必高效的方法，考虑猜想 $\{+\,x \in y \cap (w \cup z)\ -\,x \in (y \cap w)\ \cup z\}$ 以及图9中所述的针对该猜想的极小证明M。对于每一个集合$\{+\,x \square_1 y \circ_1 (\ w \circ_2 z)\ -\,x \square_2 (y \circ_1 w) \circ_2 z\}$，其中 $\square_1$ 和 $\square_2$ 在 $\{\in, \subseteq, \not\subset\}$ 的谓词符号中取值，且 $\circ_1$、$\circ_2$、$\circ_3$ 和 $\circ_4$ 在 $\{\cup, \cap, \setminus, \triangle, \times\}$的函数符号中取值，我们可以尝试构造一个与M演绎同构的证明。然而，检验集合 $SF_1 := \{+\,x \in y \triangle (w \cup z)\ -\,x \subseteq (y \cap w)\ \cup z\}$ 是否存在与M演绎同构的极小证明是没有意义的。首先，在以 $-\,x \subseteq (y \cap w)\ \cup z$为前提应用规则 $-\subseteq$E之后，斯科伦函数符号 f 将出现在证明中，而M中并不存在斯科伦函数符号。其次，所有涉及 $\triangle$ 的规则都是双前提规则。毕竟，对于这个小例子，将会有5625个待检验的情形，其中包括诸如 SF_1这样的带符号公式集合。

$$
\begin{array}{ll}
(1) & +\,x \in y \cap (w \cup z) \\
(2) & -\,x \in (y \cap w)\ \cup z \\
(3) & +\,x \in y\ (1)(4) \\
 & +\,x \in w \cup z\ (1)(5) \\
 & -\,x \in y \cap w\ (2)(6) \\
 & -\,x \in z\ (2)(7) \\
 & -\,x \in w\ (3),(5)(8) \\
 & +\,x \in w\ (4),(6)\text{✻}(7)\ (8)
\end{array}
$$

图9：一个极小证明 M对于 $\{+\,x \in y \cap (w \cup z)\ -\,x \in (y \cap w)\ \cup z\}$。

在介绍该程序之前，我们指出如何限制搜索空间

证明同构的候选带符号公式。非正式地说，给定针对语言 L上的一组带符号公式 SF 提取的极小 ER-证明P，我们将出现在 SF 的某个带符号公式中的符号生成由 L的其他符号构成的候选替换。P的结构与 ER 的规则约束这一过程，从而确定哪些替换是可容许的。我们在下文中形式化这一概念。

符号在公式中的出现 n s φ 是考虑其具体表示时， n在 s 中的第 n次出现；也就是说，它是对 φ的抽象表示进行中序遍历后得到的序列中的第 n项。在理论特定公式 φ中，谓词符号 p在 φ中至多出现一次。然而，函数符号则不一定如此。因此，我们引入以下定义。

考虑理论特定公式 φ的抽象表示时， φ 中的每个符号都可以用一个位置表示，该位置是由符号“·”连接起来的自然数组成的字符串。令 S 为 φ的抽象表示。因此， S 的根节点的位置用空字符串 ϵ表示。对于 S中的任何其他节点 s， s 的位置由 $i{\cdot}\pi$ 给出，其中 s是节点t 的第 i个子节点，而 t 的位置由 π给出。公式 φ 中函数符号 f 的第 o 次出现的位置记作 $pos(o, f, \varphi)$。例如，对于公式$\varphi_1 := x \in (y \cap w) \cap z$，有 $pos(1, \cap, \varphi_1) = 2$，以及 $pos(2, \cap, \varphi_1) = 1{\cdot}2$。

设 s_1 和 s_2 是语言 L中的两个符号。当存在规则 r_2 属于 ER，并满足以下条件时： (i) r_1 的前提与 r_2 的前提句法同构；(ii) r_1 的结论与 r_2 的结论同构；(iii) 符号 s_2出现在规则 r_2的一个前提 $prem_2$中，其中 $fmla(prem_1)$与 $fmla(prem_2)$同构，且对于 s_2 在 $fmla(prem_2)$中的某个出现 o_2 ，有 $pos(o_2, s_2, fmla(prem_2)) = pos(o_1, s_1, fmla(prem_1))$，那么符号 s_2 就是规则 $r_1 \in ER$ 的前提 $prem_1$ 中 s_1 的出现 o_1 的演绎匹配符号。如果 r_1的前提中 s_1 只出现一次，我们就简单地说， s_2 是相对于规则 r_1而言的 s_1 的演绎匹配符号。

在规则集 TS_{sets}中， $\cap$ 相对于规则 $+\cap\mathbf{E}$ 的演绎匹配符号集为 $\{\cap, \cup, \setminus\}$。但是，在同一规则集中， $\cap$ 相对于规则 $-\cap\mathbf{E}_1$ 的演绎匹配符号集为 $\{\cap, \cup, \setminus, \triangle\}$。

设 r是一个有 n个前提的理论特定规则，其前提为 $prem_1, \ldots, prem_n$。如果语义树 T中存在节点 $m_1, \ldots, m_n$，且它们分别是 $prem_1, \ldots, prem_n$的实例，则规则 r适用于该语义树。在这种情况下，我们称节点$m_1, \ldots, m_n$为 T中应用 r的依据，并称 m_i为此次在 T中应用 r 时前提 $prem_i$ 的依据节点，其中 $1 \leq i \leq n$。在证明 M 中，节点 (3) 和 (5) 作为依据，支持 $-\cap\mathbf{E}_1$, 的应用；节点 (1) 则作为依据，支持该证明中 $+\cap\mathbf{E}$ 的应用。

令 ER 为一组理论特定规则，T 为一个 ER语义树。现在，令 r为 ER 的一条规则，其一个前提是带符号公式 $prem$； n 为 T 的一个节点，满足 n 是 $prem$ 在 T 中应用 r 时的依据；并令 s 为一个同时出现在 $prem$ 和 n中的符号。如果 $pos(o_2$、 s、 $fmla(prem)) = pos(o_1$、 p_1、 $fmla(n))$，则称 s 在$prem$中的出现 o_2 是 s在 n中的出现 o_1 的依据匹配符号出现。

例如，在证明 M中， $\cap$在节点 (1) 中唯一一次出现的证成匹配符号，是它在前提 $+ x \in y \cap z$ 中的唯一一次出现。接着，在我们要介绍的程序中，我们可以尝试将 (1) 中的 $\cap$ 替换为集合 $\{\cap, \cup, \setminus\}$中的任意符号，因为该集合表示 $\cap$ 在 $+\cap\mathbf{E}$的前提中出现时对应的演绎匹配符号。不过，这仍然不够。例如，节点 (1) 中的 $\cup$ 没有匹配证成符号，因为 $\cup$ 甚至没有出现在 $+\cap\mathbf{E}$的前提中。对于这类出现，我们引入接下来的两个定义。

设在证明 P 中，m 是 n 的一个直接后裔节点，r 是应用于 P 的规则，并且 n 是此次应用 r时前提 $prem$ 的一个依据节点。若公式 $fmla(m)$中函数符号 f_1 的出现 o_1，与公式结论中的位置 $pos(o_1, f_1, fmla(m))$处的符号，以及 $fmla(prem)$中位置 $pos(o_2, f_2, fmla(n))$处的符号是同一个变元，则称该出现为函数 f_2 在 $fmla(n)$中出现 o_2 的直接后裔出现。函数符号 f 在 n 中的出现 o 的后裔出现集合，是证明 P 中f 的出现 o 的直接后裔出现的自反传递闭包。

在图10中，(3) 中以绿色突出显示的 $\cap$的出现 2，是 (1) 中以蓝色突出显示的 $\cap$ 的出现 3 的直接后裔。(1) 中 $\cap$的出现 1 没有直接后裔出现；(4) 中以红色突出显示的 $\cap$的出现也是如此。以蓝色突出显示的 $\cap$的出现，其后裔出现包括以蓝色、绿色和红色突出显示的 $\cap$的出现。现在，我们可以定义在此生成步骤中将搜索范围限制于其中的候选带符号公式集合。

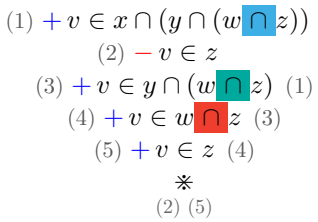


图10：一个极小证明 $\{+v \in x \cap (y \cap (w \cap z))\ -v \in z\}$。

设 SF 是 L 的带符号公式集合，ER 是扩展规则集合，T 是 SF 的一个 ER-证明，且 $sf \in SF$ 满足：对于 T 的 n 中某个节点，$sf = fmla(n)$，该节点是应用 ER 的规则 r 时的证成。若 p 是出现在 sf 中的谓词符号，则 p 在 sf 中的证明同构谓词符号 集合为以下二者之一：(i) 若 p' 是 T 中 p 的一个匹配证成，则该集合是相对于 r 而言，ER 中 p' 的演绎匹配符号集合；或 (ii) 若 T 中 p 没有匹配证成符号，则该集合是 L 的谓词符号集合。若 f 是出现在 sf 中的函数符号，则 T 中 sf 里的 f 的出现 o 的证明同构函数符号 集合为以下二者之一：(i) T 中 f 的后代节点出现的所有匹配证成符号的演绎匹配符号集合；或 (ii) 若 T 中 f 的后代节点出现均没有匹配证成符号，则该集合是 L 的函数符号集合。

设 SF 是一个带符号公式集，ER 是一个扩展规则集，P 是 SF 的一个 ER-证明。若 $SF' = SF$ 或 SF' 是通过将带符号公式 $sf \in SF$ 中至少一个谓词（或函数）符号 s 替换为 sf 中此类 s 的一个证明同构谓词（或函数）符号而得到的，则带符号公式集 SF' 是相对于 P、在 ER 中关于 SF 的带符号公式证明同构候选集。

考虑图9的证明中的集合 $\{+x \in y \cap (w \cup z)\ -x \in (y \cap w) \cup z\}$。我们计算与它们证明同构的候选带符号公式集合的大小。为此，我们将 $+x \in y \cap (w \cup z)$称为 sf_1，将 $-x \in (y \cap w) \cup z$称为 sf_2。$\cap$ 在 sf_1 中的出现以及 $\cup$ 在 sf_2 中的出现，可以分别替换为$\cap$、$\cup$ 或 $\setminus$。$\cap$ 在 sf_1 中的出现以及 $\cup$ 在 sf_2 中的出现，可以分别替换为 $\cap$、$\cup$、$\setminus$ 或 $\triangle$。因此，该集合的大小为 $3 \times 3 \times 4 \times 4 = 144$；在这个例子中，这使得需要测试的情况减少了近40倍。

利用上述定义，我们现在可以描述搜索带符号公式证明同构集合的程序。该程序的输入包括一个定义理论 $Th := \langle L,\ Ax,\ < \rangle$、一个针对 Th 的理论特定规则集合 ER，以及一个针对 SF的最小 ER-证明集合 M。对于 M中的每个证明P，以及每个集合 SF'（该集合相对于P是 SF 的证明同构候选集），我们都会检查是否可以为 SF' 构造一个与P演绎同构的 ER-证明；如果可以，则将 SF' 加入输出。

将该程序以带符号公式集合 $SF_2 := \{+x \in y \cap (w \cup z)\ -x \in (y \cap w) \cup z\}$、扩展规则集合 TS_{sets} 以及 ER的最小 SF_2 证明作为输入运行后，得到的带符号公式集合如下：

- $\{+x \in y \cap (w \cup z)\ -x \in (y \cap w) \cup z\}$
- $\{+x \in y \cap (w \triangle z)\ -x \in (y \cap w) \cup z\}$
- $\{+x \in y \setminus (w \setminus z)\ -x \in (y \setminus w) \cup z\}$
- $\{+x \in y \setminus (w \triangle z)\ -x \in (y \setminus w) \cup z\}$
- $\{+x \in y \setminus (w \setminus z)\ -x \in (y \triangle w)\ \cup z\}$
- $\{+x \in y \setminus (w \triangle z)\ -x \in (y \triangle w)\ \cup z\}$

列表中的第四组公式的一个极小证明见图 11。请注意，该证明与图9中的证明 M 演绎同构。实际上，这意味着证明练习“证明 $x \in y \cap (w \cup z)$ 推出 $x \in (y \cap w) \cup z$”和“证明 $x \in y \setminus (w \triangle z)$ 推出 $x \in (y \setminus w) \cup z$”具有相近的证明复杂度。

(1) $+x \in y \setminus (w \triangle z)$
(2) $-x \in (y \setminus w) \cup z$
(3) $+x \in y$ (1)(4)
$-x \in w \triangle z$ (1)(5)
$-x \in y \setminus w$ (2)(6)
$-x \in z$ (2)(7)
$+x \in w$ (3),(5)(8)
$-x \in w$ (4),(6)✻(7) (8)

图 11：关于 $\{+x \in y \setminus (w \triangle z)\ -x \in (y \setminus w) \cup z\}$。

7. 结论

本研究提出了一种方法，用于生成证明复杂度水平相当的证明练习。据我们所知，文献中对融入难度控制机制的自动题目生成工具的探讨仍不充分，尤其是在形式学科的背景下。我们的方法以证明复杂度这一概念为基础进行难度控制。为计算证明练习的证明复杂度，我们开发了特定理论的语义树。这些语义树的规则通过一系列范式变换从定义公理中提取。此外，为了形式化特定理论的

语义树中的切割概念，我们引入了一种替代性的解析性概念，专为不含逻辑符号的演绎系统量身设计。我们通过一个基于集合论片段的示例，说明了所提方法的适用性。

如第 4 节所述，由于施加的分析性限制，某些定义公理无法提取出理论特定规则。未来的工作中，我们计划研究如何放宽这些限制，以涵盖更广泛的定义公理，同时保持证明练习生成程序的终止性。第 5 节提到的另一项局限是：无法用 S TSNF 中的一个集合进行逻辑描述的证明练习不在我们的研究范围内。给定证明练习 E，其中集合 Γ 是其逻辑描述，而 Γ 无法转换为 STSNF 中的一个集合，一种解决方案是将 Γ 转换为一组 C 个带符号公式集合 Γ_1、…、 Γ_n，这些集合均属于 STSNF，并将 C 用作 E 的逻辑描述。这种解决方案需要对我们计划研究的方法进行一些调整。

从教学角度来看，在置换练习的参数或互换充当某些特定符号之论元的项时，练习的证明复杂度应保持不变。例如，设 φ^* 由φ 替换 $x \cap y$ 为 $y \cap x$而得到。如果 $\{-\varphi\}$ 可被反驳，那么 $\{-\varphi^*\}$ 也可被反驳，并且 $\{-\varphi\}$ 的任何极小证明都与 $\{-\varphi^*\}$的一个极小证明演绎同构。因此，应将练习 $\{-p_1 \cap p_2 \subseteq p_1 \setminus p_2\}$、$\{-p_2 \cap p_1 \subseteq p_1 \setminus p_2\}$和$\{-p_2 \cap p_1 \subseteq p_2 \setminus p_1\}$ 视为具有相当的证明复杂度。在本研究的后续工作中，我们计划形式化这些不变性，并将其纳入该方法，从而增强其生成潜力。

我们还计划在真实的教学场景中开展实验，以验证我们对证明复杂度的定义。具体而言，我们旨在探究学生是否会觉得证明中的某些结构特征更具挑战性。例如，设 E_1 和 E_2 是两个证明复杂度相近的练习。假设 E_1 存在一个完全由单前提规则组成的极小证明，而 E_2 的每个极小证明都至少涉及一条具有多前提的规则。尽管根据我们的框架衡量，它们的证明复杂度相近，但学生解答 E_2 可能会比解答 E_1感到更加困难。

参考文献

[1] Said Al Faraby、Adiwijaya Adiwijaya 和 Ade Romadhony。“面向教育用途的神经问题生成综述”。载于：《国际教育人工智能期刊》34.3（2024年9月），第1008–1045页。 ISSN: 1560- 4306. DOI: 10.1007/s40593-023-00374-x. URL: https://doi.org/10.1007/ s40593-023-00374-x.[2] Samah AlKhuzaey 等。“基于文本的问题难度预测：自动方法的系统综述”。载于：《国际教育人工智能期刊》34.3（2024年9月），第862–914页。 ISSN: 1560-4306. DOI: 10.1007/s40593-023-00362- 1. URL: https://doi.org/10.1007/s40593- 023-00362-1.[3] Tahani Alsubait。《基于本体的多项选择题生成》。博士论文。曼彻斯特大学，2015年。 URL: https://research.manchester. ac.uk/en/studentTheses/ontology- based-multiple- choice- question- generation.

[4] Luca Benedetto 等。“文本题目难度估计近期方法综述”。载于：ACM计算机调查 55.9（2023年1月）。 ISSN: 0360- 0300. DOI: 10.1145/3556538. URL: https://doi.org/10.1145/3556538.[5] Carlos Caleiro、Jo˜ao Marcos 和 Marco Volpe。“二值语义、广义组合性与有限值逻辑的解析型经典风格语义树”。载于：理论计算机科学 603（2015）。逻辑、语言、信息与计算，第84–110页。 ISSN: 0304-3975. DOI: 10.1016/j.tcs.2015. 07 . 016. URL: https :/ / www . sciencedirect . com / science / article / pii / S0304397515006374.[6] Marcello D’Agostino。“语义树比真值表更好吗？”载于：逻辑、语言与信息期刊 1.3（1992年9月），第235–252页。 ISSN: 1572-9583. DOI: 10 . 1007 / BF00156916. URL: https : / / doi . org / 10 . 1007 / BF00156916.[7] Marcello D’Agostino。《对若干命题演算系统复杂性的研究》。博士论文。牛津大学计算机实验室，1990年11月，第135页。 URL: https://www.cs.ox.ac.uk/publications/publication3869-abstract.html.[8] Marcello D’Agostino 和 Marco Mondadori。“驯服切规则：带解析切规则的经典反驳”。载于：《逻辑与计算期刊》4（1994年6月），第285–319页。 DOI: 10.1093/logcom/4.3.285.[9] Susanna S. Epp。《离散数学及其应用》。第4版。美国：Brooks/Cole出版公司，2010年。 ISBN: 0495391328.[10] Ghader Kurdi 等。“面向教育用途的自动问题生成系统综述”。载于：International Journal of Artificial Intelligence in Education 30.1（2020年3月），第121–204页。 ISSN: 1560-4306. DOI: 10.1007/s40593-019-00186-y. URL: https://doi.org/10.1007/s40593-019-00186- y.[11] Leslie Lamport。“如何撰写证明”。载于：《美国数学月刊》102.7（1995年），第600–608页。[12] Rohit Singh、Sumit Gulwani 和 Sriram Rajamani。“自动生成代数题”。载于：AAAI人工智能会议论文集 26.1（2021年9月），第1620–1628页。 DOI: 10.1609/aaai.v26i1.8341. URL: https://ojs.aaai.org/index.php/AAAI/article/view/8341.[13] Raymond M. Smullyan。《一阶逻辑》。纽约：施普林格，1968年。 DOI: https://doi.org/10.1007/978-3-642-86718-7.[14] Patrick Terrematte。“通过 IMS学习工具互操作性集成 TryLogic 交互式教程：构建基于证明与反驳策略的逻辑教学基础设施”。硕士论文。巴西北里奥格兰德州纳塔尔：北里奥格兰德联邦大学，2013年6月。 URL: https : / / repositorio . ufrn . br / handle/123456789/18685.[15] Patrick Terrematte、Fabr´ıcio Costa 和 Jo˜ao Marcos。“Logicamente：基于学习对象的逻辑虚拟学习环境”。载于：《逻辑教学工具》。Patrick Blackburn 等编。柏林、海德堡：施普林格柏林海德堡，2011年，第223–230页。 ISBN: 978-3-642-21350-2. DOI: 10.1007/978-3- 642-21350-2_27. URL: https://doi.org/10.1007/978-3-642-21350-2_27。

[16] Ke Wang 和 Zhendong Su。《维度引导的数学文字题合成》。载于：第二十五届国际联合人工智能会议论文集。IJCAI'16。美国纽约：AAAI出版社，2016，第2661–2668 页。 ISBN：9781577357704。 URL：https://dl.acm.org/doi/abs/10.5555/3060832.3060993。