

数学教育における証明学習のための論理構造図式化 GUI アプリケーション

田中 孝治* 宮崎 佳典**

Development of GUI Application Diagrammatizing Logical Structure for the Study of Mathematical Proofs

Koji TANAKA* Yoshinori MIYAZAKI**

Abstract: In the study of mathematics, learning process to reach a conclusion from a given set of inferences is important. Focusing on the study of inference process of mathematical proofs, the authors are working on the development of a GUI (Graphical User Interface) application to facilitate understandings of the derivation process. In general, mathematical proofs are described in natural language one-dimensionally. On the other hand, mathematical proofs have more complex logical structures, such as deriving a consequence from multiple premises, making it difficult for learners to understand the whole proof in natural language. Therefore, in this research, an application has been developed to support learners' understandings by presenting flow chart-looking diagrams utilizing directed graphs. In the previous research, the usefulness using this diagram was shown, and its fundamental parts of GUI were proposed and evaluated. At present, we have aimed to improve the application by making it possible to create diagrams by more intuitive manipulation. In this presentation, what has been modified so far will be reported, along with the result of assessment and future improvement.

Keywords: e-Learning, Math education, Logical structure, GUI application

1. はじめに

学校数学における証明の学習に困難を抱える学習者は多い。証明を記述する問題の正答率は低く、無回答率も多いことは長年の課題とされてきた^[1]。学習者が自力で証明を記述するためには、証明中の命題同士の関係、すなわち証明の論理構造を理解する能力が必要である。そこで、先行研究では、証明中の論理構造を明確にし、初学者の理解を補助する研究がなされてきた。

一般に、数学証明は自然言語で一次的に記述される。しかし、証明はしばしば、2つの前提から1つの帰結が推論される場合などがあり、その構造は単純ではない。先行研究では、初学者がこの複雑な論理構造を理解するための補助として、証明の構造を図式化するための体系が提案された^[2]。この体系において、証明は有向グラフ様の図を用いて表現される。また、図の構造は、シーケント計算における証明図をその基礎としている。この体系を用いて証明を図式化することの有効性はすでに示されており、本研究ではこの証明の図式化を行うための教師向け Web アプリケーション開発が進められている。

先行研究^[3]では、直感的に図示を行うためのインタフェースとして、ドラッグ&ドロップ操作を中心とする GUI による作図インタフェースが提案され、基本的な部分の実装及び評価が行われた。

しかし、先行研究で提案された GUI においては、特定の手順を踏まなければ作図できない論理構造や、そもそも作図できない論理構造がいくつか存在していたほか、命題中に含まれる数式を記述する上で、記述できない数式構造が存在するなどの仕様上、実装上の問題点が存在した。本研究では、これらの問題を解決するために、仕様の変更及び再実装を行った。その後、ユーザビリティテストを実施し、アプリケーションの評価ならびに改善点の抽出を行い、それらの一部をアプリケーションへ反映した。さらに、比較実験を行うことで先行研究の GUI を改善できたかどうかを評価した。

本稿の構成を説明する。まず、2節で、本研究に関連する先行研究について説明する。次に、3節では、先行研究で提案された証明図式化体系について説明する。その後、4節で、先行研究^[3]で提案された GUI による作図アプリケーションの仕様について説明し、5節でその問題点とそれに対して本研究で行った変更内容について述べる。6節では GUI の変更を評価するための比較実験について説明し、7節では本研究のまとめと今後の展望について述べる。

2. 先行研究

数学証明の推論過程や根拠に関するものなど、本研究に関連する研究についていくつか挙げる。また、先行研究^{[2][3]}と本研究との関係についても述べる。

辻山^[4]は証明の意味や、仮定から結論へ至るプロ

* 静岡大学 情報学部情報科学科

Faculty of Informatics, Shizuoka University

** 静岡大学大学院 情報学領域

Graduate School of Informatics, Shizuoka University

セスの全体把握に基づく証明の構想を学習することの意義、方法について研究している。そこでは、学習指導方針として、証明の構想過程を明らかにする視点について考察、議論している。また、岩坂^[6]は議論しながら証明学習を行う過程を Toulmin^[6]のモデルを用いて説明することで、数学的主張を正当化させる過程における「根拠」の重要性について述べた。これらの研究から、証明の学習指導においては、推論の過程や、それに紐づく根拠を学習させることが重要であると分かる。

また、一方で、小松^[7]は証明の考え方として具象物を用いて推論を行う action proof の有用性について、子どもの表象作用の発達に関するブルーナーの研究^[8]を用いて説明し、具象物を用いて推論過程を学習させることの有効性を示した。Aczel^[9]は証明を構造化、視覚化するツールを用いることで、論理の組み立てを補助し、学習者の理解を促進することができることを示した。これらの研究からは、証明の論理構造を具体的なイメージとして視覚化する指導方法には一定の意義があることが分かる。

渡部^[2]は証明の構造を図式化するための体系を提案した。その後、青木^[10]は実験によって、渡部^[2]が提案した図式化体系の有用性を示した後、教師がその体系に従う図を授業で利用できるように、作図を支援するアプリケーションを開発した。また森田^[3]は、作図アプリケーションの利便性を高めるために、作図用の GUI を提案、アプリケーションとして実装した。本研究は森田^[3]が開発したアプリケーションをさらに改良し、より直感的な GUI を目指すものである。

3. 証明図式体系の概要

本節では、先行研究で提案された証明の構造を図式化する体系について述べる。渡部^[2]が提案したこの体系において、証明は上から下への命題の列として表現され、命題同士は線分で結ばれる。この線分は推論を意味し、線分には、推論の根拠をラベル付けすることができる。推論の上側の命題を前提、下側の命題を帰結と呼ぶ。それぞれの命題は矩形に囲まれて表現される。

この体系において、それぞれの命題は前提となる命題がすべて真であるときに、帰結の命題が真となる関係にある。これにより、複数の前提から1つの帰結を得るような推論や、1つの前提から複数の帰結を得るような推論を表現することもできる。例えばある命題 A,B,C について「A かつ B ならば C である」という推論は図 1 左図のように表現することができる。また、同様に「A ならば B と C が成り立つ」という推論は図 1 右図のように表現することができる。

青木^[10]は渡部^[2]が提案した推論に対する根拠のラベル付けを拡張し、命題に対する補足情報のラ

ベル付けとして体系化した。このラベル付けを「コメント」と呼ぶ。

ここまで述べた方法を用いて「自然数 a, b, c について $a^2 + b^2 = c^2$ が成り立つ時、 a, b, c のうち少なくとも1つは偶数である」という証明を図示すると図 2 のようになる。図 2 中の下部にある吹き出しは、推論の根拠をラベル付けするコメントである。

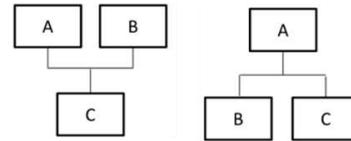


図 1 左：「A かつ B ならば C である」
右：「A ならば B と C が成り立つ」

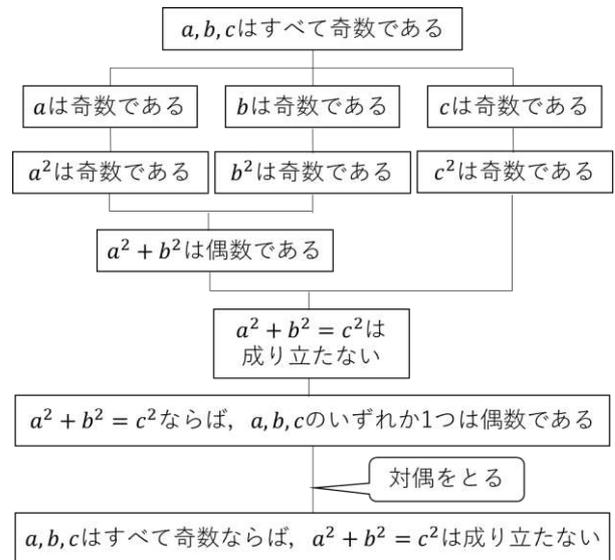


図 2 証明を図示した例

4. GUI による作図アプリケーション^[3]

本節では、森田^[3]が提案した GUI による作図アプリケーションの仕様について述べる。まず、GUI で用いられる用語について説明する。

- ・ キャンバス：作図を行うための画面上の区画
- ・ 命題矩形：各命題の内容を記述する矩形
- ・ 結合：2 命題が論理構造を持つ状態
- ・ ジョイント：結合する際のインタフェースとしての役割を持つ命題矩形付属の部品
- ・ 推論線分：結合された命題矩形同士を結ぶ線分
- ・ 直列に結合：ある 2 つの命題が前提と帰結の関係になるように結合すること
- ・ 並列に結合：複数の前提から 1 つの帰結を導く結合または、1 つの前提から複数の帰結を導く結合を作成すること
- ・ コメント矩形：前節で説明した「コメント」を書き込むための矩形

図 3 に実際の作図画面と用語との対応を示す。

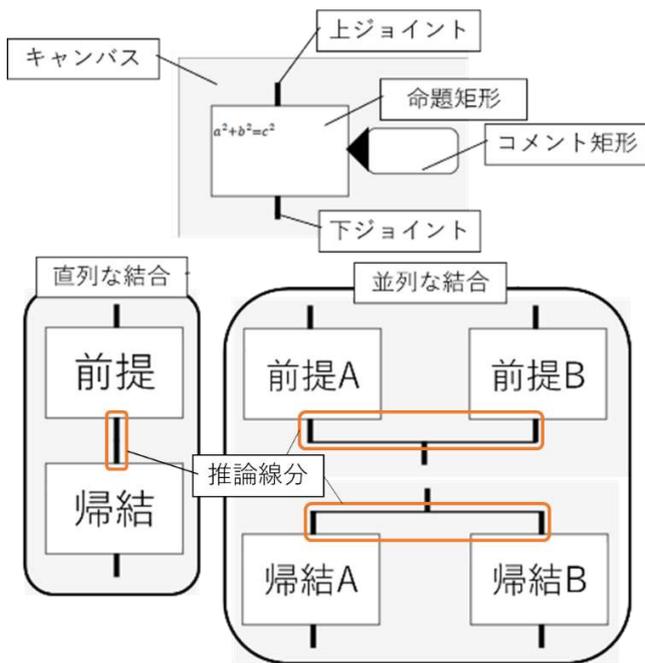


図 3 作図画面と用語の対応

以下より、森田ら^[9]が提案した作図の方法を説明する。

4.1 命題の生成

キャンバスをダブルクリックすることで命題矩形を生成することができる。なお、この操作は右クリックによってコンテキストメニューを表示し、「命題生成」の項目を選択することによって行うこともできる。命題矩形をダブルクリックすることでテキストエリアが出現し、テキスト及び数式を記述することができる。数式の記法については後述する。

4.2 命題の結合

結合を作るための操作には大きく 2 種類がある。1 点目は命題矩形をドラッグ&ドロップして移動する方法、2 点目は命題矩形のジョイントをドラッグ&ドロップし、ジョイントから線を伸ばす方法である。1 点目の操作では、命題矩形をドラッグ操作によって移動し、前提となる命題の命題矩形に付している下ジョイントと帰結となる命題の命題矩形に付している上ジョイントを重ねることで、それらの命題を直列に結合することができる。また、命題矩形同士を重ねると、コンテキストメニューが表示され、適当な項目を選択することによって、並列に結合することができる。また、結合済みの命題の間にある推論線分に対して命題矩形の端を重ねることで、対象の命題の間に命題を挿入することも可能である。ただし、実装上において、コンテキストメニューを用いた並列な結合作成機能の正しい動作は 2 個の命

題を並列に結合する場合しか保証されていない。例えば、並列に結合したことで複数の命題矩形が重なって表示される場合がある。

2 点目の結合方法については、ジョイントに対してドラッグ操作をすることで、線を伸ばすことができ(図 4 中央)、特定のジョイントに重ねた状態でドロップすることによって、その部分を結合することができる。この操作による結合は特定の状況下で行えない。図 4 の例は、「AかつBよってC」という部分証明と「DかつEよってF」という部分証明を結合し、「CとFからさらに 1 つの帰結を導く」論理構造を作成する例である。この例における「特定の状況」とは、「CとFが 2 つの命題から得られた帰結である」という状況である。この条件が満たされていない命題同士は、この操作で結合することはできない。

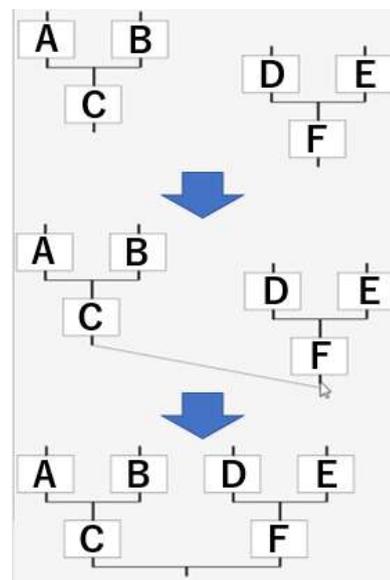


図 4 ジョイントを用いた結合作成の例

4.3 命題の削除

命題矩形を右クリックし、コンテキストメニューの「削除」を選択することによって命題を削除することができる。

4.4 コメントの挿入・削除

命題矩形または推論線分に対し、右クリックによってコンテキストメニューを表示させ、「コメント」を選択することによって、コメント入力用テキストエリアが出現し、そこに、テキスト及び数式を入力することによって、コメントを作成することができる。コメントは、何も操作をしていないときは、図 5 左図のように縮小された状態になっており、その部分をマウスオーバーすることによって、図 5 右図のようにその内容を見ることができる。コメントの削除は、コメントがついている命題矩形や推論線分を再度右クリックし、コンテキストメニューの「コ

メント削除」を選択することで実行できる。

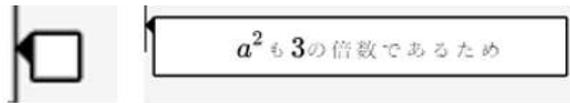


図 5 コメント

4.5 数式入力の様

数式は、該当する区間の始端と終端に「\$\$」をつけて記述することによって、記述することができる。数式入力は、以下の形式で記述する。なお、各括弧記号 ([]) の区間のものは「存在しない場合もある」ことを意味する。

$\backslash\text{keyword}\{\text{contents1}\}\{\{\text{contents2}\}\{\text{contents3}\}$

keyword と contents の一覧を以下の表 1 に示す。contents には別の数式構造を入れることができ、構造を入れ子にすることができる。これを利用することで、繁分数なども記述することができる。ただし、「 \leq 」や「 \int 」のような表 1 中がない記号は、数式に 2 バイト文字や記号を入力することで対処しなければならない。

表 1 数式構造一覧^[3]

構造名 (keyword)	数式 の 例	Contents		
		1	2	3
下付き文字 (sub)	1	下付き 文字		
上付き文字 (sup)	2	上付き 文字		
上下付き文字 (subsup)	∞_0	下付き 文字	上付き 文字	
下文字 (under)	$\lim_{n \rightarrow \infty}$	基底	上部の 文字	
上文字 (over)	\bar{a}	基底	下部の 文字	
上下文字 (underover)	$\sum_{i=0}^n$	基底	下部の 文字	上部の 文字
分数 (frac)	$\frac{1}{2}$	分子	分母	
平方根 (sqrt)	\sqrt{x}	根号の 内部		
冪根 (root)	$\sqrt[3]{a}$	根号の 内部	根指数	

この記法において、 $\frac{1}{\sqrt{2}}$ と a^2 はそれぞれ次のように記述される。

$\frac{1}{\sqrt{2}}$: $\backslash\text{frac}\{1\}\{\backslash\text{sqrt}\{2\}\}$
 a^2 : $a\backslash\text{sup}\{2\}$

5. 作図アプリケーションの様・実装の変更

前節で示した従来の作図アプリケーションの様及び実装には以下のような問題点が存在する。

- ① ジョイントを用いた結合が特定の状況下でしか実行できないほか、代替手段もないため、ある論理構造を作成するためには特定の手順を踏むことが必須である
- ② 「特定の状況」による限定、並列な結合時における命題矩形同士の干渉が考慮されていないなどの理由から、作図できない論理構造がある
- ③ 「 \leq 」や「 \int 」のような表 1 中がない記号は、数式に 2 バイト文字や記号を入力することで対処しなければならない。また、行列のような表 1 中がない構造を伴うものについては、その構造が正しく記述できない
- ④ 挿入機能を動作させるための操作範囲が狭い
- ⑤ 推論の削除ができない
- ⑥ 命題矩形上においてダブルクリックが有効な範囲が狭く、命題を編集しにくい

本研究では、これらの問題を解決するために、仕様・実装の変更を行ったのち、ユーザビリティを検証するために、ユーザビリティテストを実施した。その後、テストで抽出された要改善点について、さらなる仕様・実装の変更を行った。以降では、森田ら^[6]の GUI を旧 GUI、本研究で仕様・実装の変更をした GUI を新 GUI とする。

本節では、旧 GUI から新 GUI に至るまでの過程及び、その変更内容について説明する。まず、5.1 節では、従来の作図アプリケーションの問題点①～⑥を解決するために、本研究で行った仕様・実装の変更について説明する。その後、5.2 節で、ユーザビリティテスト要領・結果について説明し、最後に 5.3 節では、テストで抽出された要改善点を解決するために行った仕様・実装の変更について説明する。

5.1 旧 GUI の問題点を解決するために行った変更

5.1.1 結合作成仕様の変更点

結合作成においては、前述の①、②の 2 点が問題であった。そのため、今回はこれらの問題を解消するために、以下のような仕様変更を行った。

- ① 結合方法の仕様を変更し、ジョイントを用いた結合及び命題矩形を用いた並列な結合を制限なく利用可能にする
- ② 作図機能の実装を変更し、正しい論理構造を描画できるようにする

以下、それぞれの変更点について詳説する。まず①の変更においては以下の表 2 に示すような動作が実現されるようにした。

表 2 本研究で策定した結合生成の仕様

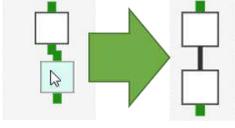
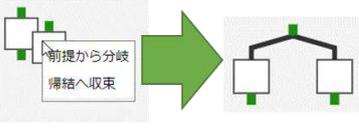
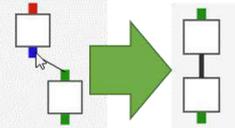
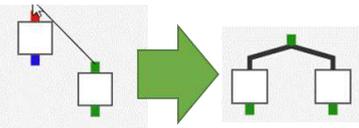
結合の 操作分類 対象	直列な結合	並列な結合
命題 矩形	 <p>上下ジョイントを重ねた状態でドロップ</p>	 <p>選択したメニュー項目(ここでは「前提から分岐」)に従う結合</p>
ジョ イ ン ト	 <p>ドロップ操作で直列に結合される場合、ジョイントを青く表示</p>	 <p>ドロップ操作で並列に結合される場合、ジョイントを赤く表示</p>

表 2 について、従来の作図アプリケーションからの変更点を述べる。まず、ジョイントを用いた結合について、表 2 の「ジョイント」行で示しているように、状況によらず、直列または並列な結合を追加できるようにした。これは命題矩形に対する操作についても同様で、状況によることなく並列に結合するような仕様にした。この仕様変更は、GUI の内的一貫性^[11]を考慮した結果、操作とその時に行われる結合の仕方を 1 対 1 で対応させる目的で行った。

次に、②の変更点について述べる。旧 GUI において、結合作成は、1 対 1 の命題同士の結合を作成する動作と図 4 に示したような特殊な結合作成をそれぞれ個別に定義していた。また、結合によってできたグラフ構造を保持する機構もなかったため、図の構造を次の動作に反映させることができていなかった。そのため、今回はこれら 2 点の問題を解決するために、描画時に行われる命題矩形及び推論線分の座標計算を `dagre-d3`^[12]ライブラリを用いて行うように改良を行った。同ライブラリは、独自のデータ構造に即してノードとエッジの情報を定義することにより、自動的に適当な有向グラフの `svg` 画像を生成することができるものであり、結果、命題矩形同士の重なり等、実装上の問題点をなくすことができた。

5.1.2 数式入力仕様の変更点

本節冒頭の③でも述べたように、数式入力においては、表 1 に示した範囲内での数式入力しかできないため、行列のような構造を伴う数式を正しく記述できないことが問題であった。そのため、今回はこの問題を解決する方法として `MathJax`^[13]の導入を行った。`MathJax` は `MathML`、`LaTeX`、`ASCIIMathML` で記述された数式を Web 上で表示するためのライブラリである。`MathJax` は `Tex` と同等の数式記述能力を有しており、本アプリケーシ

ョンで要求される数式記述に十分耐えうると考えられたため、採用した。本アプリケーションでは、`MathJax` の `svg` 出力機能を利用し、命題やコメントに `LaTeX` 形式で入力された数式を `svg` テキストとして表示するよう実装した。これにより、行列や積分、極限などの構造を伴う予約語を含む様々な数式の入力に対応することができるようになった。

5.1.3 ユーザビリティ改善のための変更点

本節冒頭の④～⑥で示したユーザビリティ上の問題点について、それぞれ以下のような仕様変更を行った。

- ④ 命題同士を重ねた時のコンテキストメニューに挿入機能を追加し、利用しやすくする
- ⑤ 命題の削除と同様な手順で推論を削除できるようにする
- ⑥ 命題矩形全体がダブルクリックの有効な範囲になるように実装を変更する

④の変更点については、動作させるための操作範囲を拡大してしまうと、他の動作をさせるための操作範囲と干渉してしまうことが考えられたため、今回は命題矩形同士を重ねたときの動作の選択肢として「前提との間に挿入」、「帰結との間に挿入」の 2 種を対象とする有向グラフ構造に従って、コンテキストメニューに適切に追加することで、挿入機能を動作させるための操作範囲が狭いという問題への対処とした。

5.2 ユーザビリティテスト

5.1 節に示した機能拡張、実装の変更を行ったことによるユーザビリティを検証するとともに、要改善点を抽出するためにユーザビリティテストを実施し、その結果について言語プロトコル分析を行った。

ここでは、その結果を示すとともに、考察を加える。

5.2.1 言語プロトコル分析

言語プロトコル分析とは、人の発話行動によって得られた自然言語データを対象とするもので、人が問題解決を行うプロセスにおいてどのようなことを考え、行動しているのかを分析するための手法である。今回は、発話思考法と呼ばれる手法によって、発話行動を収集し、分析を行った。発話思考法とは、実験協力者に実験タスクを実行してもらいながら、タスク実行時に考えていることをその都度発話によって報告してもらう手法である。

言語プロトコル分析は一般に、データ採取、書き起こし、分析の3段階のプロセスによって行われる。データ採取は発話行動を収集するプロセス、書き起こしは発話とその時の行動を対にして記述し、整理するプロセス、分析は書き起こしたデータを用いてその時の認知過程を明らかにするプロセスである。今回の実験では、平木^[14]の手法に倣い、書き起こしたデータのうち、トラブルが発生していたと考えら

れる発話行動に着目、要改善点の抽出を行った。

5.2.2 ユーザビリティテストの概要

今回、以下の表3に示す要領に従ってユーザビリティテストを実施した。図6の図式化された証明はタスク1において複製の対象とした証明である。

表3 ユーザビリティテスト実施要領

実験協力者	某大学の大学生、大学院生8名 ・学部3年生2名、学部4年生3名、修士1年生1名、修士2年生2名 ・女性3名、男性5名
テスト環境	デスクトップPC(Diginos Monarch XT)上でGoogle Chromeブラウザを用いて起動
実施時期	2017/7/31~2017/8/8
内容	・下記2つのタスクを実施 [タスク1]アプリケーションを用いて図式化された証明を複製する [タスク2]自然言語で記述された証明に相当する図をアプリケーションで作図する

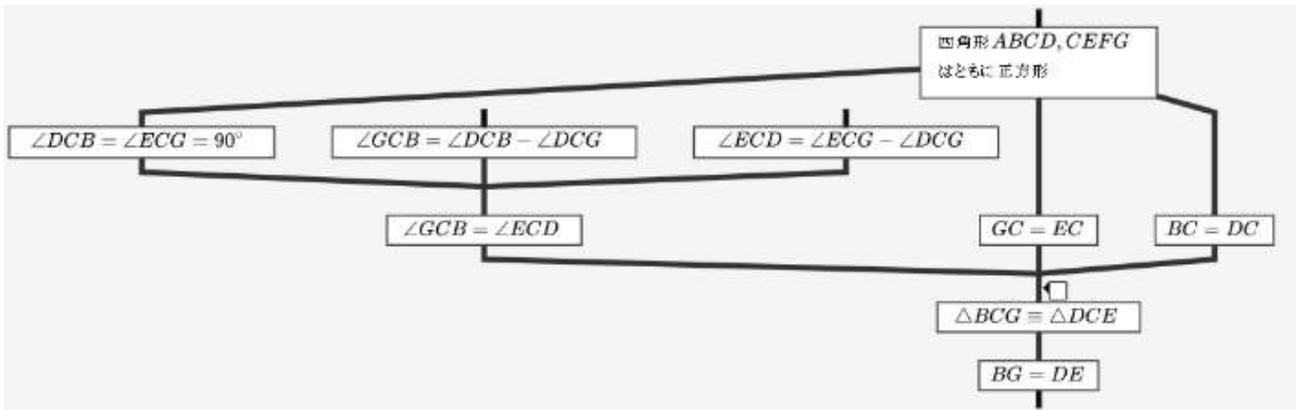


図6 テストタスクで用いた証明

それぞれのタスクを実施した意図を述べる。タスク1は本研究で仕様・実装を変更したアプリケーションを用いてユーザーが証明を図式化ができるかどうかを検証するとともに、ユーザーがGUIに慣れることができるかどうかを検証することが目的である。タスク2は実際の利用方法を履行できるかの検証を目的として実施した。テスト時は、発話の録音と、作業画面のキャプチャを行うとともに、実験者がテストの様子を観察した。また、テスト中に発生した問題への対処は、付属の取扱説明書を参照するなどし、基本的に実験協力者が自力解決するよう指示した。ただし、どうしても問題解決できない場合には実験者へ質問することを許可した。

5.2.3 テスト結果分析要領

ここでは、タスク1をある実験協力者が実施した時に得られた実際のプロトコルデータを例示し、本

ユーザビリティテストで行われた言語プロトコル分析による要改善点抽出の要領を示す。

表4は、ある実験協力者がタスク1を実行している様子から得られたプロトコルデータについて、トラブルが発生していたと推定される箇所を抽出したものである。「時間」列は実験開始からの経過時間を「分：秒」形式で表している。表中の「発話」列において「※」として表記されている箇所は発話があることは確認できたが、その内容が聞き取れなかった箇所を表し、「Q:」として表記されている箇所は、実験協力者からの質問があった箇所を表す。また、この「Q:」と同じ行の「行動」列の「A:」以降は実験者からの回答の概要を端的にまとめたものである。なお、表に「:」のみの列がある場合は、実験中においてトラブルの見られなかった区間を表している。

表 4 テストタスク 1 に関するプロトコルデータ (一部抜粋)

時間	発話	行動
11:05	tri,angle,うーん,ん?いいの か,BCG,と D,triangle,DCE,で合同?合同, えー,あれ?で, equ...	テキスト入力(途中で一回, カーソル位置変更をしている)
12:16	※, えっ, で, BG=DE	「△BCG...」を画面左側に移動 →命題生成
12:37	はみだしてる	「△BCG...」を画面左側に移
12:49	あれ, ええっ...どこ	右, 下へスクロール→左上周 辺まで戻す
13:11	ええっ, BG,BG=DE. は いつ, ええっ,	命題生成→テキスト入力→右 へスクロール
13:37	消えたんだけど, えっどこ いつ...下?	下へスクロール→左上周辺ま で戻す→別な命題の移動
15:04	えー, これどうしたらいいの	結合済みの「DCB=ECG...」 と「GCB=ECB...」と 「ECD=ECG...」を推論の部
15:15	これ何よこれ. これどうした らいい. なにこれ, えっ, 何 これどうしたらいい.	残った推論と接点のうち, 接 点や推論の端をクリック
15:23	Q: これって聞いていいの?	A:内部的に結合が外れていな いことの説明を実施
17:23	EC,あこれか, ええ全然でな い. だ. よいしょ	「四角形ABCD...」の下ジョ イント線を「DCB=ECG...」 に伸ばし, 結合.
17:40	でー, 線, 線でない, GC= 線, 線でない	「BC=DC」と「GC=EC」の 上ジョイントから線を伸ばそ うとする
17:58	線, はあ	取説図10を確認
18:13	ここを, 持って, 移動する... GC=EC...よいしょ	「四角形ABCD...」の下ジョ イント線を「GC=EC」に伸 ばし, 結合.
18:30	と, ここ...ええっ, はっ, えーでないんだけど.	「四角形ABCD...」につな がる推論から線を伸ばそうと する動き→取説確認→命題矩形 自体を動かす→戻す→再度推 論から線を伸ばそうとする動
18:55	何これどうしたらいいの. あ, 出た...えっ, くっつか ない.	「BC=DC」の上ジョイント 線を「四角形ABCD...」につ ながる推論に対してD&D
19:14	どこにくっつけばくっつく の?くっつかない(笑い)... こっち?えいっ. ええっ	↑を数回→「四角形ABCD ...」につながる別の推論に対 するD&D
19:27	並列?前...	取説図11~13あたりを確認
20:20	あ, 出た...ああ	「四角形ABCD...」の辺の複 数箇所から線を伸ばそうとす る動き→「四角形ABCD...」 の下ジョイント線を 「BC=DC」に伸ばし, 結合

まず, 11:05~13:37 の区間のトラブルについて述べる. この実験協力者は, 図 6 の複製を行う際, 最初に図に含まれる命題をすべて作成するような行動をしていた. 表 4 の 11:05 の発話及び行動より, このタイミングで実験協力者は「 $\triangle BCG \equiv \triangle DCE$ 」という命題の作成を行っている. その後, これの命題矩形を画面左側に移動する行動がみられたが, これは, 命題矩形の自動整列によって, キャンバスの右端に寄っていた命題矩形を手動で戻しているものとみられる. 特に 12:37 の「はみだしてる」というような発話でもその意図を見ることができる. また,

12:49~13:37 の発話及び行動からわかるように, 「 $\triangle BCG \equiv \triangle DCE$ 」の次に生成した命題矩形は, 完全に画面外に出てしまっており, 実験協力者は画面をスクロールして探したのちに, 諦めて別の作業に取り掛かっている.

次に, 15:04~15:23 の区間のトラブルについて述べる. 15:04 において, 結合済み命題群から命題単体を切り離す動作をしながら, その挙動に困惑する様子が見られた. その後, 15:15 では, その場に残った推論線分をドラッグ&ドロップ操作によって移動しようとする様子が見られる. その後はこの事象について実験者に質問することで解決をした. このトラブルは, 結合済みの命題群から単体の命題矩形を切り離して移動することができる仕様を使用者が理解しにくいことを示唆するものであるといえる.

次に, 17:23~20:20 の区間のトラブルについて述べる. この区間においては, ジョイントから線を伸ばす結合方法を使おうとしており, 17:23~17:58 の発話及び行動からは, ジョイントから線を伸ばすこと自体に成功していないことが読み取れる. また, 18:30~19:14 では, 推論線分の端をジョイント同様に扱おうとする動きがみられており, 推論線分とジョイントの区別ができていないことがわかる. この実験協力者は, 最終的に, 命題矩形の辺から, 偶然的にジョイント結合用の線を出すことに成功している. 以上より, このプロトコルデータから得られた実験時のトラブルは以下の 3 点である. なお, 各トラブルの後ろに付している括弧書きの数字は本ユーザビリティテスト全体を通して, それぞれに類するトラブルが見られた回数を示している.

- ・ 命題及び論理構造の自動描画により, 生成した命題矩形がキャンバスからはみ出てしまう (4)
- ・ 結合済みの命題群から 1 つの命題矩形を切り離せる仕様がユーザーにとって理解しづらい (5)
- ・ 推論線分と区別できないなど, ジョイントに対する操作による結合作成がわかりにくい (5)

以上のようにして, 各タスクについてのプロトコルデータからそのときに発生しているトラブルを取得することで, GUI の要改善点を抽出した.

5.2.4 テストの全体評ならび抽出できた要改善点

ここではユーザビリティテスト全体を俯瞰して考えられる GUI の評価と, 前節で示した分析方法によって得られた GUI の要改善点について述べる.

まず, 全体的な評価のための事実として, すべての実験協力者が両タスクで正しい論理構造を表す図を完成させることができたということが挙げられる. このことから, 本研究で仕様・実装を変更した GUI は証明図式化アプリケーションとしての機能を果たすことができるということがいえる. また,

実験時の様子から、GUIに苦戦している実験協力者とそうでない実験協力者はおおよそ半々であるように見受けられた。これに関係する事実としてテストタスク履行にかかった時間が挙げられる。タスク1においては、最も短時間でタスクを終了できた協力者がタスク履行にかけた時間は8分であったのに対し、最も長時間であった協力者は36分をタスク履行にかけていた。また、タスク2については最短時間が10分、最長時間が29分であった。これらの事実からは、GUIに対する慣れには個人差があることと、それを加味した上でもなおGUI自体が原因となっているトラブルがあるということが言えるだろう。

次に言語プロトコル分析によって得られたユーザビリティ上の要改善点のうち、多くの実験協力者に見られたトラブルの要因となっていたものについて以下に列挙する。

- ・ ジョイントに対する操作による結合作成において、適切なドラッグ開始位置、ドロップ位置が分からない
- ・ コンテキストメニューを用いた結合作成において、メニューの項目と作成できる結合の対応関係が分からない
- ・ 新しく生成した命題矩形がキャンバスからはみ出してしまふなど、命題及び論理構造の自動描画機能の実装方法に問題がある
- ・ 結合済み命題群の命題矩形に対するドラッグ操作によって、その命題矩形が見た目上切り離されるが、結合は維持されているという仕様がわかりにくい

これらの要改善点のほかにも、数式入力時におけるスペルチェック機能の不備などが問題点としてみられた。Nielsen^[15]によると、5人のユーザーを対象とし、あらゆる機能について事細かにユーザビリティテストを実施することによって、ユーザビリティ上の問題点の85%を抽出できるといわれている。故に、今回のユーザビリティテストにおいて想定されるアプリケーションの使用方法を履行させるテストを一通り実施することにより、GUIの問題点のある程度抽出することができたといえるだろう。

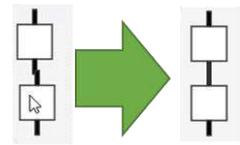
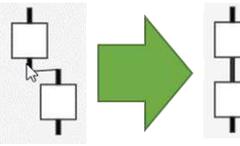
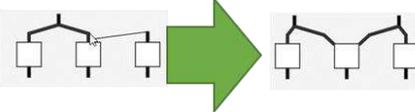
5.3 ユーザビリティテスト後の仕様・実装変更

前節で示したユーザビリティテストで多くの実験協力者に見られたトラブル要因を解決するため、仕様・実装の変更を行った。以下より、それぞれの変更について説明する。

5.3.1 結合作成仕様の変更

結合作成において、操作と動作の対応付けが分かりにくいという問題を解決するために、5.1節で提示した結合作成仕様を再度変更し、コンテキストメニューを用いた結合作成でも並列な結合の構造を追加する動作と並列に結合されているところへ命題を追加する動作を使い分けられるようにするとともに、ジョイントを用いた結合作成のルールを作図画面上でもわかりやすくした。以下の表5に再度変更を行った結合作成仕様を示す。この仕様変更においてはGUIの内的一貫性^[11]を考慮し、命題矩形、ジョイントのいずれに対する操作を用いても同一の論理構造を作成できるようにした。

表5 再定義した結合作成仕様

結合の 操作分類 対象	直列な結合	並列な結合
命題 矩形	 <p>上下ジョイントを重ねた状態でドロップ</p>	 <p>選択したメニュー項目(ここでは「併結を並列に結合」)に従い、併結(前提)を追加</p>
ジョ イ ン ト	 <p>上→下または下→上ジョイントに対する操作</p>	 <p>上→上または下→下ジョイントに対する操作で並列な結合を追加</p>

ジョイントを用いた結合作成のルールを作図画面上でもわかりやすくするため、表 5 の「ジョイント」行に示しているように、ジョイントに対するドラッグ操作時、直列な結合が作成できるドロップ位置のジョイントを青く、並列な結合が作成できるドロップ位置のジョイントを赤く表示するようにした。さらに、コンテキストメニューの項目と作成できる結合の対応関係をわかりやすくするため、命題矩形同士を重ねてドロップした際に表示されるメニューの項目上にマウスオーバーした際に、項目に対応する結合内容が GIF 画像によってプレビュー表示されるようにした。これらは、色差や動きなど、変化をつけることによる認知性向上^[11]を狙った仕様変更である。

5.3.2 自動描画機能実装の変更

自動描画機能においてはいくつかの実装上の問題点がユーザビリティテストからわかったが、特にユーザーにとってのストレスとなっていた部分として、5.2.3 節でも挙げられた「作成した命題がキャンバスからはみ出してしまふ」がある。この問題を解決するために、命題や論理構造を作成した際に、行われる自動描画結果に対して、各命題矩形や結合済みの命題群の位置をもとあった位置に戻す処理を行わせるような実装を行ったが、結合済みの命題群について、svg における矩形と線分の描画方法の違いが処理しきれず、実装することができなかった。しかしながら、「作成した命題がキャンバスからはみ出してしまふ」という問題については、結合されていない単一の命題矩形について、前述の処理を実装することによって解決することができたといえる。

5.3.3 結合済み命題群を移動する機能の実装

ユーザビリティテスト以前の GUI においては、並列な結合作成について、命題矩形またはジョイントに対する操作のいずれかによってしか作成することができない結合があったが、5.3.1 節で示した変更によって、そのような結合仕様はなくなった。これにより、結合済みの命題群から 1 つの命題矩形を切り離して移動する必要性がなくなったため、結合済みの命題群に対するドラッグ操作によって、結合済みの命題群全体がキャンバス上を移動するように仕様を変更した。これにより、結合済みの命題群から 1 つの命題を切り離して移動できる仕様のわかりにくさに関する問題点を解決することができたと推測される。

6. 比較実験

旧 GUI と新 GUI とを比較し、GUI のユーザビリティを改善できているかどうかを検証するために比較実験を実施した。本節では、実験の内容及び結果について説明する。

6.1 実験実施要領

今回、以下の表 6 に示す要領に従って比較実験を実施した。

表 6 比較実験実施要領

実験協力者	理工学分野を専攻する大学生、大学院生20名 ・学部2年生6名、学部3年生3名、学部4年生7名、 修士1年生2名、修士2年生2名 ・女性4名、男性16名
テスト環境	実験協力者各人のPCにてFirefoxブラウザを用いて起動
実施時期	2017/12/19-2018/1/11
内容	・自然言語で記述された証明に相当する図をアプリケーションで作図するタスクを2題出題 ・それぞれを旧GUIと新GUIを用いて作図

実験実施の手順を説明する。実験のはじめ、各実験協力者に証明図式化体系と GUI の操作方法について説明し、その後、操作練習を兼ねて図式化された簡単な証明の複製をしてもらった。その後、各実験協力者には計 4 つのタスクを課した。その内訳を表 7 に示す。表 7 の各列はそれぞれのタスクの番号、行はそれぞれのタスクの条件を表す。例えば、タスク I は問題 1 を新 GUI で作図するタスクである。タスクの実施順はすべての実験協力者に共通して、タスク I から IV の番号順に実施した。各実験タスクは 12 分の制限時間を設けて実施した。この 12 分という制限時間設定については、比較実験の本実験を行う前に、3 名の大学生を対象にして行ったプレ実験において、新 GUI での作図にかかった最長の時間が 12 分程度であったほか、旧 GUI での作図を観察した際、3 名とも操作で詰まってしまい、一向に作図できる気配が見られなかったため、新 GUI での作図の最長時間を基準とした 12 分を設定した。

表 7 比較実験のタスク実施順序

タスク番号 タスク条件	I	II	III	IV
使用したGUI	新GUI	旧GUI	旧GUI	新GUI
問題番号	問題1	問題1	問題2	問題2

実験では、作図にかかった時間並びに作図結果の図（または制限時間を過ぎた時点での図）を取得した。この作図にかかった時間とは実験協力者が作図を開始してから、作図をし終えたと自己判断し、実験者に対してその旨を報告するまでの時間である。また、実験後、各実験協力者には両 GUI の操作性、視認性等に関する 5 件法及び自由記述によるアンケートに回答してもらった。5 件法によるアンケートは全項目について 5（とても良い）、4（良い）、3（普通）、2（悪い）、1（とても悪い）という尺度によるものである。表 8 は 5 件法によるアンケートでの質問項目である。

表 8 5件法によるアンケートの項目

設問番号	設問内容
1	作図時の操作性について
2	GUIのわかりやすさについて
3	思い通りの図が作図できたか
4	テキスト・数式入力機能の使いやすさ
5	完成図の見やすさ・論理構造のわかりやすさ
6	テキスト・数式のみやすさ

6.2 実験結果

実験結果について、表 9 に制限時間以内に作図終了を報告した人数を問題ごと、GUI ごとに示す。表 9 における括弧内の数字は制限時間以内に作図終了を報告した人数のうち、作図結果が正しい図であった人数を示したものである。ここでの「正しい図」の基準は、「作図結果を 2 節で述べた証明図式化体系のルールに従って自然言語の証明として読み替えたときに、正しい証明になること」とした。これについて、問題 1 に対する実際の作図結果を用いて例示する。図 7 は問題 1 に対する正しい作図結果である。対して、図 8 の作図結果を自然言語の証明として読み替えると、途中で「 $a^2 + b^2$ は偶数ならば c^2 は奇数である」という推論が生じることになるが、それ以前にこれらの変数 a, b, c についての関係を示す式はないため、この推論は間違った推論であり、図 8 の作図結果は間違った図であるとわかる。また、図 9 においては推論線分がそもそも正しく描画できておらず、命題同士の結合ができていない箇所があるため、証明図式化体系のルールから逸脱した図であることが分かる。表 9 の括弧内の数字は、制限時間以内に作図終了を報告した人数のうち、図 8、図 9 のように作図結果が間違っていた人数を除いた人数である。

表 9 各タスクで制限時間以内に作図できた人数

問題 使用 番号 した GUI	問題1	問題2
新GUI	20(19)名	18(17)名
旧GUI	3(1)名	2(0)名

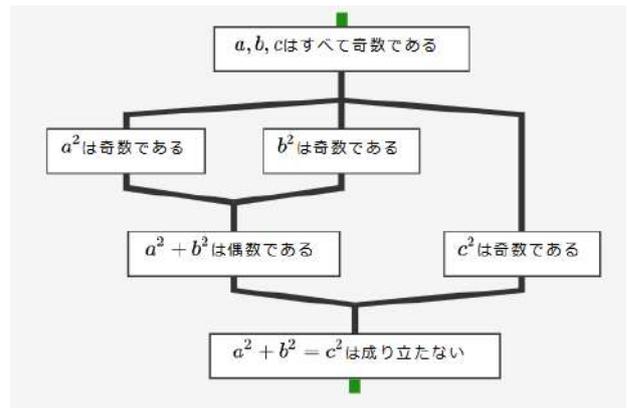


図 7 正しい作図結果の例

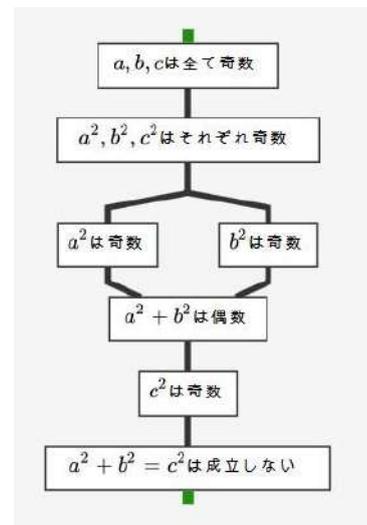


図 8 間違った作図結果の例

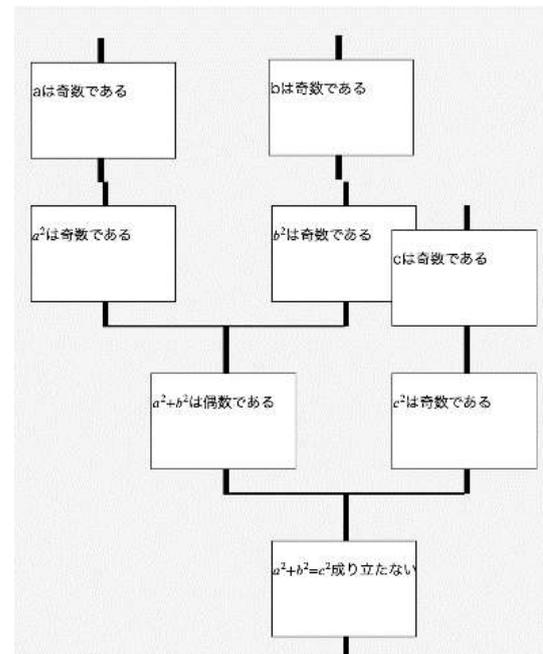


図 9 証明図式化体系のルールから逸脱した作図結果の例

また、5 件法によるアンケートの設問ごとの各評価値について、回答した人数の割合を旧 GUI について表 10 に、新 GUI について表 11 に示す。表 10 及び表 11 の各セルの背景色は、人数の割合が大きいものほど赤く、小さいものほど白く示している。また、表 10、表 11 における設問番号は表 8 で示したものに对应している。

表 10 5 件法によるアンケート結果(旧 GUI)

設問番号	1	2	3	4	5	6
5(とても良い)	0.00	0.00	0.00	0.00	0.00	0.05
4(良い)	0.00	0.10	0.00	0.10	0.10	0.20
3(普通)	0.05	0.30	0.00	0.25	0.25	0.25
2(悪い)	0.20	0.35	0.15	0.35	0.25	0.35
1(とても悪い)	0.75	0.25	0.85	0.30	0.40	0.15

表 11 5 件法によるアンケート結果(新 GUI)

設問番号	1	2	3	4	5	6
5(とても良い)	0.40	0.25	0.45	0.40	0.35	0.30
4(良い)	0.60	0.65	0.40	0.50	0.50	0.60
3(普通)	0.00	0.10	0.15	0.10	0.15	0.10
2(悪い)	0.00	0.00	0.00	0.00	0.00	0.00
1(とても悪い)	0.00	0.00	0.00	0.00	0.00	0.00

また、自由記述においては、「新 GUI のほうが論理結合を行いやすい」など、5 節で述べた改善内容に関連する記述が複数みられたほか、いくつかの改善点の提案も見られた。

6.3 考察

実験結果について考察する。まず表 9 について、新 GUI では制限時間以内にほとんどの実験協力者が作図を終了することができたのに対し、旧 GUI では制限時間以内に作図を終えることができた実験協力者はほとんどいなかったことがわかる。これについて、旧 GUI の実験で得た作図結果の図を参照すると、5 節で示した「作図手順の制限」や「特定状況でしか動作しない機能」といった問題によって、作図が阻害されていたことが見て取れる。また、アンケートの自由記述でもそれらの内容に関する回答が複数見られたことから、本研究で行った GUI の仕様・実装の変更によって、証明図式化を素早く、正確に行えるようになったということがわかる。

また、5 件法によるアンケートの全項目について、表 10 からは旧 GUI への評価で 2 (悪い) 以下を回答した実験協力者の割合が 5 割以上であったことがわかり、表 11 からは新 GUI への評価で、4 (良い) 以上を回答した実験協力者の割合が 8 割以上であったことがわかる。以上より、本研究で行った GUI の

仕様・実装の変更によって、GUI の操作性・視認性が大幅に向上したことがわかる。

以上の事柄より、本研究で行った GUI の仕様・実装の変更によって、旧 GUI の問題点を解決し、証明図式化支援 GUI アプリケーションのユーザビリティを向上させることができたといえる。

7. おわりに

本研究では、先行研究^[1]で提案・実装された証明図式化支援 GUI アプリケーションの問題点を分析、改善のための仕様・実装の変更を行った。その後、ユーザビリティテストを実施することによって、さらなる問題点を抽出し、改善のために再度の仕様・実装の変更を行った。さらに、これまでに行ってきた変更によって実際に先行研究で開発された GUI アプリケーションを改善することができているかを検証するために、比較実験を実施した。

比較実験の結果から、本研究で行った仕様・実装の変更によって、先行研究で開発された GUI アプリケーションにあった問題点が解決され、ユーザビリティを改善することができたといえる。

今後の課題としては、まず、本研究で行った実験により得られた意見をもとにした GUI の改善を行った後、作図機能本来の対象者である教師を対象とした使用実験を行うことで、教師用アプリケーションとしての有用性についてより詳しく検討する必要があると考えられる。また、数式入力を拡張し、より正しい形の数式を提示できるようになったことが、学習者に与える影響についても検証を行う必要があるといえるだろう。

また、将来的な機能拡充として、以下のようなことが考えられる。まず、学習者の証明に対する理解を促進させるための提案・実装として、自然言語で記述された証明との対応関係を表示するインタフェースの仕様を策定し実装することが考えられる。次に、数式検索^[16]や式変形支援^[2]などの数学教育を支援する他のアプリケーションとの連携を行い、命題の検索や式変形過程の入力を補助することなどが考えられる。また、学習者が試行錯誤しながら作図を進めることができるような機能を策定・実装することができれば、学習者が自らの学習のために利用することも可能なアプリケーションとすることができるだろう。

参考文献

- [1] 国立教育政策研究所, “平成 29 年度 全国学力・学習状況調査 報告書” (2017).
- [2] 渡部 孝幸, 宮崎 佳典, 林 佳樹, “導出規則に着目した証明視覚化・式変形支援システムの提案”, 京都大学数理解析研究所講究録, 第 1865 巻, pp137-145 (2013).
- [3] 森田 雄介, 宮崎 佳典, 渡部 孝幸, “数学教育

における証明学習のための論理構造提示 Web アプリケーション”, 情報処理学会第 78 回全国大会, pp845-846 (2016).

- [4] 辻山 洋介, “学校数学における証明の構想の過程: argumentation を視点として”, 筑波大学教育学系論集, Vol.35, pp41-53 (2011).
- [5] 岩坂 美鈴, “中学生の証明の学習過程としての argumentation を分析するための基礎的研究”, 上越数学教育研究, 第 25 号, pp.63-76 (2010).
- [6] Toulmin S.E., “The uses of argument”, Cambridge university press (2003).
- [7] 小松 孝太郎, “学校数学における action proof の意義”, 学校教育学研究紀要 1, pp69-85 (2008).
- [8] Bruner J.S., 岡本 夏木 (訳), “認識能力の成長<上>”, 明治図書 (1968).
- [9] J.Aczel, P.Fung, R.Bornat, M.Oliver, T.O’Shea, B.Sufrin “Software that Assists Learning within a Complex Abstract Domain: the Use of Constraint and Consequentiality as Learning Mechanisms”, British Journal of Educational Technology, Vol.34, No.5, pp625-638 (2003).
- [10] 青木 杏奈, 渡部 孝幸, 宮崎 佳典, “数学教育における証明学習のための図示化を用いた学習支援ツール”, 情報処理学会第 77 回全国大会, pp977-978 (2015).
- [11] Johnson Jeff, 武舎 広幸 (訳), 武舎 るみ (訳), “UI デザインの心理学: わかりやすさ・使いやすさの法則”, インプレス (2015).
- [12] cpettitt, “dagre-d3”, <https://github.com/cpettitt/dagre-d3> (2018 年 3 月 17 日参照).
- [13] The MathJax Consortium, “MathJax”, <https://www.mathjax.org/> (2018 年 3 月 17 日参照).
- [14] 平木 勉, “定性的分析手法 (プロトコル分析) を活用した授業分析”, 第 18 回工業技術教育全国研究大会 第 4 分科会 (2008).
- [15] Nielsen Jakob, “Why you only need to test with 5 users”, Nielsen Norman Group, <http://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/> (2018 年 3 月 17 日参照).
- [16] 渡部 孝幸, 宮崎 佳典, “正規表現を用いた数式検索手法の提案”, 情報処理学会論文誌, Vol.56, No.5, pp1417-1427 (2015).

[著者紹介]

田中 孝治 (非会員)



2016 年 4 月静岡大学情報学部情報科学科編入学. 2016 年 10 月より e-Learning システムの研究, 開発に従事. GUI アプリケーション開発, 教育工学に関心を持つ.

宮崎 佳典 (正会員)



1998 年 3 月筑波大学工学研究科単位取得満期退学. 静岡産業大学情報学部助教授, 静岡大学情報学部准教授を経て, 2017 年 4 月より静岡大学大学院情報学領域教授・静岡大学創造科学技術大学院教授 (兼任). 1999 年 7~8 月米国 UCLA 客員研究員. 2013 年 4~9 月 米国ノースカロライナ州立大学客員研究員. 博士 (工学). 日本 e-Learning 学会, 情報処理学会, 日本応用数理学会, 教育システム情報学会, 外国語教育メディア学会, 情報知識学会, 日本ソーシアルデータサイエンス学会各会員として, e-Learning, 英語&数学教育ツール開発, 数値解析周辺の研究に従事.