D-06

論理学の形式証明に対する学習支援システムの試作と評価

Prototyping and Evaluation of Learning Support System for Formal Proof

宮澤 清介†　岡野 浩三†　楠本 真二†
Kiyoyuki Miyazawa　Kozo Okano　Shinji Kusumoto

あらまし　ソフトウェア設計開発においてフォーマルアプローチが近年注目を浴びている．この技術の基本は数理論理学であり，とりわけ形式証明の概念を理解することは重要である．形式証明の学習は形式体系の重要な概念であるが，厳密な操作を要求されるため初学者に対する敷居は高いと思われる．そこで，学部生の論理学の授業において形式証明の学習を支援するツールを提案している．本稿ではツールを用いた評価実験について述べる．ツールの設計と実装にあたり，通常の学習者がとる手書きによる問題演習の欠点を改善する目標を定めた．ツールを使用しない被験者にはその問題演習の欠点が現れたが，ツールを使用した場合ではそれが改善された．

1．まえがき

フォーマルアプローチに基づいた情報システム開発はミッションクリティカルなシステム開発などを対象として，MDA(Model Driven Architecture)のコア技術として利用されている．フォーマルアプローチ手法として大きく，Coq 1），Agda, Isabelle/HOL などを用いた対話型定理証明とモデル検査手法 2）の 2 つのアプローチがある．前者は論理学の形式的証明が必須であり，後者は論理式の充足不能性判定の高速化が重要な技術となっている．

一方，文献 3）で“mathematical reasoning is intrinsic to both traditional engineering and software engineeri-ng, [……] Software engineers usually use discrete mathematics and logic in a declarative mode for specifying and verifying system behaviors and for analyzing system features.” とあるように，ソフトウェア工学において数理論理学の理解は重要であり，その学習を支援し，フォーマルアプローチの基本技術の習得を促進させることで，ソフトウェアの生産性を高めることができる．

その他，数理論理学は Computing Curricula2001 4）で CS 分野のコアカリキュラムの一部として設定されている．日本でも情報処理学界の標準カリキュラム J07 で CS 分野の基礎科目として位置づけられている．本研究で扱う教育支援システムは公理に基づいた形式証明の教育を支援するものである．

関連研究としては以下が挙げられる．スタンフォード大学の Jon Barwise らは学生のモデル理解を促進させるため，チューリング機械をグラフィカルに学習するシステムである Turing’s World や，数理論理学の意味論をグラフィカルに学習するシステムである Tarski’s World 5）などを開発した 6）．

Tarski’s World は 3D 空間に大きさや形状を自由に変更できるブロックを配置することにより，その空間に対して記述された論理式の真偽を解答する問題や，論理式を用いてその空間の意味を記述する問題など，様々な形式の問題を出題できる．また Gentzen 流の自然演繹 7）の学習支援システムとして，MacLogic というツールがある．

著者らは文献 9)で論理学の初学者を対象に Hilbert の公理系を用いた形式証明の学習を支援するツールを試作し，簡単な評価実験を行った．本ツールは論理学の初学者を対象にしているため，対話型定理証明器の直接の使用は考えていない．逆に論理学の初学者を対象にした機能構成を考えた．また，関連研究で紹介したツールは意味理解に重点をおいているものが多いが，本ツールは形式証明に重点をおいている．また，授業では Hilbert の公理系を中心に扱っており，異なる公理系を演習で用いるのは上級学習者にとって有益であるが，初学者にかえって混乱を招く点で不適切と考えた．

本稿では文献 9) の評価実験の追加実験を行ったことについて報告する．評価実験を実際の授業で行ったため，より大規模なデータを収集することができた．その実験の結果を受けて新たな機能を実装した．さらに，実験方法やアンケート項目などを見直し，再度簡単な評価実験を行った．

以降 2 節では形式的証明について簡単に述べ，3 節では本研究で試作したツールについて触れる．4 節ではツールの評価実験について述べ，5 節で考察を，最後に 6 節でまとめる．

2．形式証明

公理とは，その他の命題を矛盾なく導出するための前提である．そして推論規則とは，証明済みの論理式から新たな論理式を導出するための規則である．例えば，前件肯定（modus ponens）と呼ばれる推論規則は，$P \rightarrow Q$ と Pから Qを導くことができるという規則である(Pと Qは任意の論理式)．公理と推論規則をまとめて公理系と呼ぶ．

公理から推論規則を用いて定理を証明していく過程を形式的証明と呼ぶ．図 1 に$\vdash X \rightarrow X$ という定理を導く形式的証明の例を挙げる．

3. 一階述語論理の形式証明学習支援システム

本研究で作成したシステムである“Learning Assist System for Proof”（以降 LASP）について述べる．

3.1　システム概要

本研究で試作した学習支援システム LASP は，大阪大学基礎工学部情報科学研究科の数理論理学の授業である，

†大阪大学 大学院情報科学研究科
Graduate School of Information Science and Technology, OsakaUniversity

公理
(1) $\vdash (P \rightarrow (Q \rightarrow P))$
(2) $\vdash ((P \rightarrow (Q \rightarrow R)) \rightarrow ((P \rightarrow Q) \rightarrow (P \rightarrow R)))$
(3) $\vdash ((\neg P \rightarrow \neg Q) \rightarrow ((\neg P \rightarrow Q) \rightarrow P))$

推論規則
(1) P と$P \rightarrow Q$ からQ を得る.

証明系列
$\vdash ((X \rightarrow ((X \rightarrow X) \rightarrow X)) \rightarrow$
$((X \rightarrow (X \rightarrow X)) \rightarrow (X \rightarrow X)))$
（公理2 に$P = X, Q = (X \rightarrow X), R = X$ を代入） (1)
$\vdash (X \rightarrow ((X \rightarrow X) \rightarrow X))$
（公理1 に$P = X, Q = (X \rightarrow X)$ を代入） (2)
$\vdash (X \rightarrow (X \rightarrow X))$
（公理1 に$P = X, Q = X$ を代入） (3)
$\vdash ((X \rightarrow (X \rightarrow X)) \rightarrow (X \rightarrow X))$
（証明系列(1),(2)と推論規則1 より） (4)
$\vdash (X \rightarrow X)$
（証明系列(3),(4)と推論規則1 より） (5)

図 1. 証明系列例
Fig. 1 A proof example

「情報論理学」で使用されることを目的としている．「情報論理学」では，Hilbert の公理系 8）に基づく形式的証明を扱っており，1 節で挙げた Tarski's World は意味論を扱っている点で，MacLogic は Gentzen 流の自然演繹を扱っている点で，提案するツールとは異なっている．

手書きによる証明の問題演習の欠点として，以下のようなことが挙げられる．

（1）長い論理式を手書きにすると，構文対応や変数名の書き損じを起こす

（2）証明系列の作成にあたり，コピー&ペーストに相当する操作をすることが頻繁に起こる

したがって，以上の点を改善するため，本ツールでは以下のような目標を定めた．

（1）論理式の入力を最小限にとどめる入力インタフェースを提供

（2）コピー&ペーストに相当する操作を要求しない

以下の節で，それらの具体的な機能やインタフェースについて述べる．

3.2 問題データ読み込み

本機能の目的は，問題を解答するにあたって，不要な手間を削減することにある．

本機能は，ユーザが問題データを記述したファイルを選択すると，その問題における公理，推論規則，仮定，推論すべき定理のデータをシステムに入力する．

本機能で使用する問題データは xml 形式で記述する．xml のタグは表 1 のように定義した．また，xml 形式で作成した問題データの例を図 2 で示す．

表 1 XML タグの定義
Table1 Definition of the XML tag

タグ名	意味
root	ルート要素
axioms	axiom 要素の集合
axiom	公理
inference_rules	inference_rule 要素の集合
inference_rule	推論規則
proof	1 つ以上の question 要素と 0 個以上の hypothesis 要素からなる集合
question	証明すべき論理式
hypothesis	仮定
answer-list	解答情報．以下 answer タグの登場順で，証明の順を定義する．
answer	解答となる論理式
substitute_axiom	解答で利用する公理番号
substitute_exp	公理のどの変数に，どの論理式を代入したか
inference_num	利用する推論規則番号
inference_proof	推論規則を適用する証明系列の番号
deduction_proof	演繹定理を適用する証明系列の番号
deduction_exp	演繹定理を適用した仮定
annotation	論理式ごとの解説
comment	全体的な解説

3.3 代入支援

LASP の代入支援機能について述べる．

本機能の目的は，代入ミスによる証明時間の浪費を削減することにある．なぜなら，代入ミスは本来の証明における学習とは無関係だからである．

例を挙げて説明する．以下の公理(1)の P, Q, R に

$$(P \rightarrow (Q \rightarrow R)) \rightarrow ((P \rightarrow Q) \rightarrow (P \rightarrow R)) \quad (1)$$

それぞれ $X \rightarrow Y, Z, ((Y \rightarrow Z) \rightarrow X)$ を代入すると，

```
<?xml version="1.0" encoding="Shift_JIS" ?>
<root>
   <axioms>
      <axiom> P -> (Q -> P) </axiom>
      <axiom> (P -> (Q -> R) ) -> ((P -> Q) -> (P -> R)) </axiom>
      <axiom> (P -> Q) -> ((P -> not Q) -> not P) </axiom>
      <axiom> P -> P </axiom>
   </axioms>
   <inference_rules>
      <inference_rule> P . P -> Q :: Q </inference_rule>
   </inference_rules>
   <proof>
      <question>A -> (not not A) </question>
   </proof>
   <answer-list>
      <answer  exp="A"  reason="hypothesis" >
         <annotation/>
      </answer>
      <answer  exp=" (not A -> A) -> (( not A -> not A) -> not not A) " reason="substitute" >
         <substitute_axiom   num="3"/>
         <substitute_exp      in="P" out="not A"/>
         <substitute_exp      in="Q" out="A"   />
         <annotation/>
      </answer>
                                  (中略)
      <answer exp=" not  not A"  reason="inference" >
         <inference_num     num="1" />
         <inference_proof   num="5" />
         <inference_proof   num="6" />
         <annotation/>
      </answer>
      <answer exp=" A -> not not A " reason="deduction" >
         <deduction_proof  num="7" />
         <deduction_exp    exp="A" />
         <annotation/>
      </answer>
      <comment   str="これは解説です。"/>
   </answer-list>
</root>
```

図 2 xml ファイルの例
Fig. 2 An example of the xml file

$$((X \rightarrow Y) \rightarrow (Z \rightarrow ((Y \rightarrow Z) \rightarrow X))) \rightarrow (((X \rightarrow Y) \rightarrow Z) \rightarrow ((X \rightarrow Y) \rightarrow ((Y \rightarrow Z) \rightarrow X))) \quad (2)$$

論理式(2)のように複雑な論理式になる．もし代入ミスをしていた場合，どの部分を間違えたのかを探していると，非常に手間がかかってしまう．前述の通り，このプロセスは命題証明における学習とは無関係である．また，手書きで論理式(2)のような長くて複雑な式を書くのは非常に時間と手間がかかる．自動で代入を行うことで，無駄な時間と手間を削減する効果も期待できる．

本機能の概要を述べる．本機能は，ユーザが任意の公理を選択し，その公理の書く命題変数を書く論理式に置換して，新しい論理式を生成する機能である．

例えば，ユーザが(3)で表す公理を選択し，

$$(P \rightarrow (Q \rightarrow P)) \quad (3)$$

この公理 3 の P に(X→Y)を，Q に Z を代入する，ということを指定すると，論理式(4)を自動で生成する．

$$((X \rightarrow Y) \rightarrow (Z \rightarrow (X \rightarrow Y))) \quad (4)$$

また，一階述語論理における変数への代入に関しては，代入対象の変数が自由変数であるか，そして代入する項が元の論理式の変数に対して自由かどうかを考慮しなければならない．例えば以下の論理式(5)を考える．

$$\forall x \exists y f(y,z) \quad (5)$$

論理式(5)の y は$\exists y$ に束縛されているため代入することはできない．また z に x または y を含んだ項を代入すると，その結果が量化記号に束縛されてしまう．例えば z に $g(x)$ を代入すると，論理式(6)になる．

$$\forall x \exists y f(y, g(x)) \quad (6)$$

論理式(6)中の変数 x は$\forall$x に束縛されているため，論理式(5)とは異なる意味になってしまっている．本ツールでは，各変数に代入禁止変数リストを設けることによって，これらの不正な代入が行われそうになると，例外処理を発生するようにした．

実際のツール上における，ユーザへの機能の提供方法について述べる．公理は，図 3 の上のように，テーブルを用いて管理する．

ユーザはテーブルから，代入を行いたい公理をクリックして選択する．選択状態にした後，「代入」ボタンを

クリックする．すると，図 3 の中にあるような代入パネルが開くので，各命題変数に，代入したい論理式を入力し，完了ボタンを押す．

完了ボタンを押すと証明系列のテーブルの最後に，代入した結果が追加される（図 3）．

3.4 推論支援

証明系列に推論規則を適用する機能について述べる．本機能の目的は，手書きで証明をする上で，頻繁に起こりうるコピー＆ペーストに相当する操作を削減することにある．なぜなら，推論規則や演繹定理の概念を理解している者が手書きで演習するにあたって，コピー＆ペーストとの操作はただ時間を浪費するだけだからである．また、推論の自動化による証明の効率化のため，推論ミスを無くすために，推論規則を適用しようとしている論理式が，本当に適用できる形かをチェックし，正しい場合

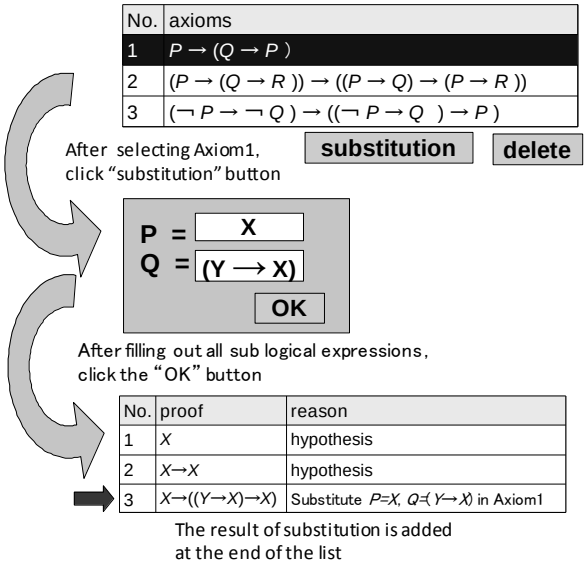

図 3 代入の流れ
Fig.3 term substitution flow

にのみ推論結果と，生成理由を出力する．生成理由とは，生成された論理式が，どの論理式と推論規則から生成されたのかを示すものである．また，推論を自動で行うことによって，冗長で複雑な論理式と生成理由を書く手間が省ける．

本機能の概要について述べる．本機能ではユーザが，適用したい推論規則と，その推論規則に適合する証明済みの論理式を選択すると，規則に対応する新しい論理式を自動で生成する．また，自動で生成した推論結果に，その推論理由を付加することで，手書きで証明をする際の生成理由を付加する手間も削減する．

実際のツール上における，ユーザへの機能の提供方法について述べる．推論規則と証明系列は，それぞれテーブルで管理している．ユーザはまず，適用したい推論規則を，推論規則のテーブルからクリックして選択する．次に，選択した推論規則に適合する証明系列をクリックして選択し，導出ボタンを押すと，正しく証明系列を選択できていた場合に対し，推論規則に基づいて，新たな論理式を生成する．システム上での実行は，図 4 のようになる．

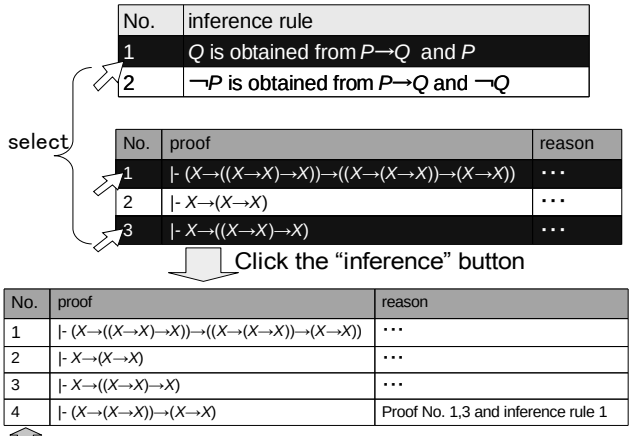

図 4　推論の流れ

Fig. 4 Flow of constructing a proof with the system

3.5　演繹定理

演繹定理は，形式証明において必要不可欠な定理である．ユーザが実行する過程は図 5 のようになる．

3.6　ヒント機能

本機能は証明に不慣れな初学者に対し，3段階の難易度に分けてヒントを提供する機能である．4節で述べる実験1から，「ステップ数が多い証明を自力で解くのは難しい」という意見があったため，新たに本機能を作成した．ヒントはユーザに与える情報量順に，以下のヒントを用意した．

・穴埋め形式のヒントを表示
・マイルストーン形式のヒントを表示
・次のヒントのみ表示

穴埋め形式のヒントは表 2 のようなもので与え，問題データを入力したときにランダムに生成する．マイルスト

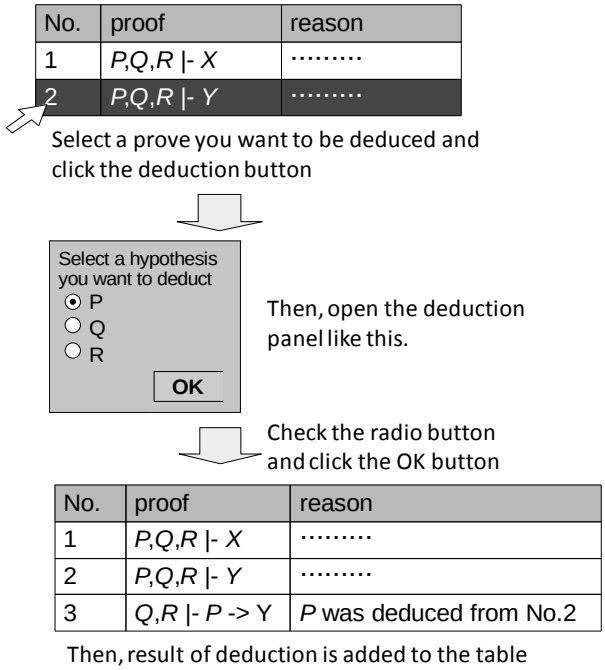

図 5　演繹定理の適用

Fig. 5 Applying Deductive theorem with the system

ーン形式のヒントは表 3 のようなヒントを与え，推論規則を用いて導く式のみを表示する．次のヒントのみ与える機能は，マイルストーン形式のヒントのうち，まだ証明できていない最初の式を表示する．

表 2　穴埋め形式のヒント
Table 2 A hint of fill-in-the-blanks

1		仮定
2		公理?に代入
3		公理 1 に P=A, Q=?を代入
4	(¬A -> A)	推論規則?を適用
5	((¬A -> ¬A) -> ¬¬A)	推論規則?を適用
6		公理 4 に P=?を代入
7		推論規則?を適用
8	(A -> ¬¬A)	演繹定理：証明系列?から?を演繹

表 3　マイルストーン形式のヒント
Table 3 A hint of milestone

1		
2		
3		
4	(¬A -> A)	証明系列 1,3 に規則規則 1
5	((¬A -> ¬A) -> ¬¬A)	証明系列 2,4 に規則規則 1
6		
7	¬¬A	証明系列 5,6 に規則規則 1
8		

3.7 グラフ表示

本機能を用いることでネストの複雑な論理式を図 6 のように木構造で表すことができる．ノードをクリックすることで，部分木を展開，収納することができる．

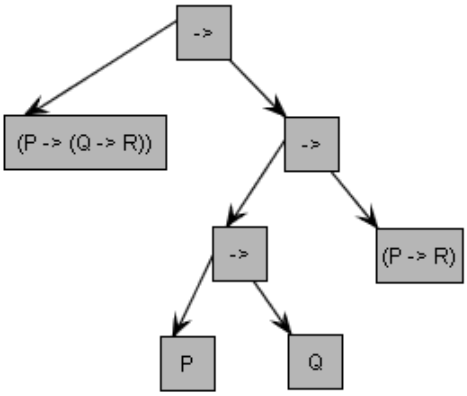

図 6 論理式のグラフ表示
Fig. 6 Graphical representation of logical expression

3.8 LaTeX 清書

本機能の目的は，レポートや答案を作成する手間を省き，証明結果を目視で用意に確認可能にすることである．3.3 節でも述べたが，証明問題を解く上で，証明系列を生成することは，非常に文章量が多く手間がかかる．3.3 節や 3.4 節で導入した学習支援機能を利用して証明した結果を，自動でファイルに出力でき，目視で容易に確認できることは，非常に学習に効果があると考える．

3.7 LASP の開発

LASP の外観を図 7 に示す．

図 7 LASP
Fig. 7 Overview of LASP

LASP の開発には Java を用いた．総クラス数は 17，規模は約 6000 行になった．

4. 評価

本節では評価実験について述べる．実験は，学部生の授業の演習形式で行ったもの（以下実験 1）と，その結果を受けて実験方法やアンケート項目を改善して小規模に行ったもの（以下実験 2）の 2 点について述べる．

4.1 目的

本実験の目的は，第一に，現状における学習支援システムの機能がユーザにとってどの程度有用であるかを計測すること．第二に，本システムをさらに有用なものにするためには，どのような機能を実装するべきかという意見を集めるためである．

4.2 環境

ここでは，本実験における被験者について記述する．実験 1 の被験者は，大阪大学基礎工学部情報科学科の学生約 50 名である．実験 2 の被験者は，著者の所属研究科の学生 16 名であり，その内訳は，情報科学研究科博士後期課程 3 年 1 名，博士前期課程 2 年 8 名，1 年 3 名，情報科学科 4 年 4 名である．

4.3 評価項目

実験 1 では，解析対象を主に LASP のユーザビリティと証明の効率化の程度とした．ユーザビリティの評価として，ユーザにとって LASP をどの程度抵抗なく受け入れられるのか，また，よりユーザビリティを高めるにはどのようなインタフェースが必要とされているのかを調査するために，以下のようなアンケート項目を作成した．

- Q1 代入支援機能はどの程度使いやすいか
- Q2 推論支援機能はどの程度使いやすいか
- Q3 本ツール全体としての使いやすさはどの程度か
- Q4 本ツールを使用することで証明は効率化されたと思うか
- Q5 本ツールを使用することで論理学の証明の学習はやりやすくなったか
- Q6 本ツールを使用することで，学習効果はあったと思うか

回答は 5 段階評価で，数字が大きいほど評価が高い．また，自由記述欄を設けることで，実験方法やツールに関する意見を収集した．証明の効率化の程度を計測する指標として，演習問題の解答時間を計測した．

実験 2 では，解析対象を主に証明の効率化の程度に絞った．実験 1 では演習問題の解答時間の差による評価を行ったが，これはユーザの主観を含んでいない．本実験では，問題解答時間の計測によって証明の効率化の程度を客観的に計測することに加え，ユーザの視点からどの程度証明の労力が軽減されているのかを計測することを目的とした．よって，アンケート項目は以下のものを作成した．

- Q1 代入機能はどの程度証明の効率化に貢献しているか
- Q2 推論機能はどの程度証明の効率化に貢献しているか
- Q3 本ツールを使用することで証明は効率化されたと思うか
- Q4 本ツールを使用することで演習の効率が上がると思うか

Q5 本ツールを使用することで，学習意欲は向上すると思うか
Q6 本ツールは使いやすいか

回答は実験 1 と同様に 5 段階評価である．また，自由記述欄として，ツールに足りないと思う点とよかった点の 2 点を問う項目を追加した．

4.4 方法

実験 1 の評価実験の手順は，以下の通りである．

（ 1 ）学生を 6 グループに分割
（ 2 ）手書きでの演習
（ 3 ）2 週間後にツールを用いた演習
(i) ツールマニュアル配布
(ii) サンプル問題 1 問を 20 分間かけて解答させてツールに慣れさせる
(iii) 演習問題を解答させる
（ 4 ）アンケート回答

被験者が実験中に学習してしまい，後に解く問題ほど解答時間が早くなってしまうことを防ぐため，まず学生を 6 グループに分割し，グループごとに解く問題の順番を割り当てた．1 問あたりの制限時間は 15 分とし，解答時間の計測は以下のように行った．

手書き：被験者全員が同時に演習に取り掛かり，被験者が証明を完了すると同時に，スクリーンに映したストップウォッチの時間を記入してもらうことで測定．
LASP：ツールにタイマーを組み込み，演習問題データを入力したときから解答と同じ論理式が証明系列に現れるまでの時間を測定．

授業の演習形式での実験だったので，一度に長時間の演習時間を設けることができず，手書きとツールで 2 回に分けて実験を行った．

一方，実験 2 は研究科内で行ったものであるため，1 回の実験で手書きとツールの両方で問題を解いてもらう時間を設けることができた．手順は以下の通りである．

（ 1 ）学生を 16 グループに分割
（ 2 ）ツールマニュアルと論理学のテキストを配布
（ 3 ）サンプル問題 4 問を 1 時間かけて解答させ，ツールに慣れさせる
（ 4 ）手書きとツールで交互に解答
（ 5 ）アンケートに回答

実験 1 では，ツールに慣れさせる時間が不十分だったため，実験 2 では 1 時間かけて被験者にツールに慣れさせることで，操作に慣れないために解答時間が遅くなってしまうことを防いだ．また，16 人の被験者それぞれに対し，LASP を使って解答する問題番号や解答する問題の順番を別々にして割り当てた．解答時間の計測方法は実験 1 と同様である．

4.5 結果

4.5.1 アンケート結果

実験1で得られた，アンケート項目ごとの平均点の結果を表4で示す．本稿では，アンケートの回答は間隔尺度として扱う．またアンケートの自由記述欄から以下のような意見が得られた．

・フォントの色分けなどで，論理式のネストを明確にできれば良い
・マウスカーソルを毎回合わせるのが面倒である
・マウス移動が大きい
・ウインドウのサイズを自由に調節したい
・公理，推論，証明系列などのパネルの配置が悪い
・ショートカットキーがあればよい
・ヒント機能があればよい
・解答を表示する機能が欲しい
・時間をかければ慣れることができそうである

表 4 実験 1 のアンケート結果
Table 4 A result of the Questionnaire of experiment 1

アンケート項目	平均点
Q1	3.70
Q2	3.28
Q3	2.84
Q4	2.88
Q5	3.28
Q6	2.84

次に実験 2 で得られた結果を表 5 で示す．

表 5 実験 2 のアンケート結果
Table 5 A result of the Questionnaire of experiment 2

アンケート項目	平均点
Q1	4.69
Q2	4.56
Q3	4.13
Q4	3.63
Q5	3.75
Q6	3.63

また，アンケートの自由記述欄から，ツールの不足点として以下のような意見が得られた．

・正解を導き出せたとき，それを示す機能が欲しい
・∀を「forall」を書かなければならないなど、複雑な式を書きにくい
・証明系列の途中の式を削除できない
・証明系列の途中に式を挿入したい
・証明の方針を考えるため，メモ機能が欲しい
・演習問題をユーザ自身が作れるようにしたい
・ショートカットキー機能がない
・画面レイアウトの工夫
・式がネストすると見づらい
・ヒントの内容を工夫すべき

ツールの良かった点として以下として，以下の意見が得られた．

・手書きでの労力が軽減できる
・ケアレスミスを削減できる
・考えうる式を多く試せるので解を導きやすい
・（証明を効率化する機能のため）すぐに自分の

考えを試せる
・代入に使う労力や時間を軽減できるため，
試行錯誤する時間が増えた

4.5.2 解答時間の測定結果

手書きで証明した場合と，LASP を利用して証明した場合の証明完了までの時間を測定した結果を表に示す．表の解答時間は制限時間内に解けなかった場合と，不正解の場合の解答時間を除外して平均値を計算した．

まず実験 1 の解答者数に対する正答者数（正答者数/全解答者数）を表 6 で示し，正答者の平均解答時間の結果を表 7 で示す．実験 1 では問題 3 に出題ミスがあったため，問題 3 のデータはここには掲載していない．

表 6 実験 1 の正答者数
Table 6 The number of subjects who answer correctly in experiment 1

	手書き	LASP
問題 1	5/23	7/23
問題 2	6/24	7/23
問題 4	0/27	1/22

表 7 実験 1 の正答者の平均解答時間
Table 7 An average time of answer in experiment 1

	手書き	LASP
問題 1	9 分 13 秒	10 分 4 秒
問題 2	6 分 56 秒	8 分 15 秒
問題 4	解答者無し	8 分 18 秒

次に実験 2 での解答者数に対する正答者数を表 8 に示し，正答者の平均解答時間の結果を表 9 で示す．

表 8 実験 2 の正答者数
Table 8 The number of subjects who answer correctly in experiment 2

	手書き	LASP
問題 1	5/8	2/8
問題 2	4/8	4/8
問題 3	4/8	2/8
問題 4	2/8	2/8

表 9 実験 2 の正答者の平均解答時間
Table 9 An average time of answer in experiment 2

	手書き	LASP
問題 1	10 分 35 秒	5 分 57 秒
問題 2	7 分 19 秒	6 分 04 秒
問題 3	8 分 48 秒	5 分 54 秒
問題 4	12 分 56 秒	5 分 19 秒

5. 考察

まず実験 1 で評価の対象とした，ユーザビリティに関して考察する．表 4 の Q1, Q2, Q3 に注目すると，ユーザは LASP のユーザビリティに関しては満足していないことがわかる．それは実験 2 でも表 5 の Q6 で示されており，ユーザにストレスを感じさせずに利用してもらうためには，ユーザインタフェースを改善する必要があると考えられる．そのためには，アンケートの自由記述欄で得られた意見を取り入れていく予定である．まずショートカットキーの実装や，正答を導いたときにそれを示す機能は実装が容易であり，かつ多くの被験者が必要としていた機能であるのでそれを実装し，その後，パネルの配置の再考などを行う．

次に，実験 2 で評価の対象とした，ユーザの視点での証明の効率化に関して考察する．表 5 の Q1, Q2, Q3 に注目すると，代入や推論機能は十分に証明の効率化に貢献しており，本ツールを用いることで証明が効率化できたと考える被験者が多かったことがわかる．また，実験 2 の LASP の良かった点を問う自由記述からも「手書きでの労力が軽減できる」と同等の意見を 16 人中 12 人から得ることができ，LASP が目的としている証明を効率化することは十分に果たされていることがわかる．Q1 や Q2 と比べて Q3 がやや平均点が下がっているのは，実験 1 の結果からわかったように，LASP 全体のユーザビリティが被験者の満足のいくものでないためであると考えられる．したがって，ショートカットキーの実装やパネルの配置の再考などを行い，より効率的に証明が行えるように LASP を改善したいと考えている．

そして，証明の効率化の程度を，データを用いて客観的に計測した結果について考察する．表 7，表 9 で得られた結果から実験 1 ではすべての演習問題に対し，やや LASP の方が遅くなったが，逆に実験 2 ではすべての演習問題に対して LASP の方が早いという結果が出ている．この結果は実験 1 では LASP に慣れる時間が 20 分だったのに対し，実験 2 では 1 時間だったからであると考えられる．よって，LASP の操作に慣れていれば，問題演習は効率的に行えるということが推測できる．実験 2 はサンプル数が少ないので，この推測が正しいことを導くためには，より大規模な実験を行う必要がある．

表 6 と表 8 を見ると，手書きで解答した場合と LASP で解答した場合で正答者の数はあまり変化がないことがわかる．この結果から，本稿の実験方法では，証明を効率化した効果が問題解答の可否にまで影響しなかったのではないかと考えられる．

また，実験 2 の手書きでの解答を検査していると，1 件の推論規則適用間違い，2 件の代入間違いが見られた．LASP を用いると，推論規則を適用できない場合にエラーメッセージを提示することができるため，正しく推論規則を適用する方法を学習することができる．同様に，代入間違いも完全に無くすことができるので，証明について効率的に学習することに関して，LASP は有用であると考えられる．

6. あとがき

本稿では，手書きによる問題演習の欠点を改善するために，問題データ読み込み，代入支援，推論支援などの機能を実装した形式証明の学習支援システムを試作した．

そして 2 回に分けて評価実験を行い，本稿で試作したツールは，ユーザインタフェースに改善の余地があるものの，証明の効率化を図る上で有用だということがわかった．

今後の課題として，ユーザインタフェースを改良し，LASP に慣れる時間を多めにとった上で，学部の授業に適用することなどが挙げられる．

謝辞 本研究は一部，科学研究費補助金基盤 C(21500036)と文部科学省「次世代 IT 基盤構築のための研究開発」（研究開発領域名：ソフトウェア構築状況の可視化技術の開発普及）の助成を得た．

参考文献

1) Bertot, Y. and Castéran, P.: Interactive theorem proving and program development: Coq'Art: the calculus of inductive constructions, Texts in Theoretical Computer Science. An EATCS Series, Springer-Verlag (2004).
2) Baier, C. and Katoen, J.-P.: Principles of model checking, MIT Press (2008).
3) Henderson, P.: Mathematical reasoning in software engineering education, Communications of the ACM, Vol. 46, pp. 45–50 (2003).
4) Engel, G. and Roberts, E.: Computing Curricula 2001 Computer Science, IEEE Press (2001).
5) Dave Barker-Plummer, J. B. and in collaboration with Albert Liu, J. E.: Tarski's World: Revised and Expanded, Center for the Study of Language and Inf (2007).
6) Barwise, J. and Etchemendy, J.: Computers, visualization, and the nature of reasoning, In: Bynum, T.W. and Moor, J.H. (eds) The digital phoenix: How computers are changing philosophy, pp. 93–116 (1998) London: Blackwell.
7) Gentzen, G.: Investigations into logical deduction, The Collected Papers of Gerhard Gentzen, In: M. E. Szabo (eds) pp. 68–131 North-Holland Publishing (1964).
8) Hilbert, D.: The foundations of geometry, K. Paul, Trench, Trübner & co., ltd. (1902).
9) 宮澤清介, 岡野浩三, 楠本真二.: フォーマルアプローチの基本技術習得のための学習支援システムの試作, ソフトウェアエンジニアリング最前線 2009,pp. 69-74, (2009).