Distance Teaching of Mathematics Using *Theorema*

Bruno Buchberger, Tudor Jebelean

RISC (Research Institute for Symbolic Computation)
A-4232 Castle of Hagenberg, Austria
last-name@RISC.Uni-Linz.ac.at

Abstract

The *Theorema* system integrates computing and automated reasoning in an on-line tutoring system which uses natural language presentation and deduction in natural style. The system has an easy-to-use interface accessible both locally and over the internet (see http://www.risc.uni-linz.ac.at/projects/Theorema). We report on the various possibilities of using *Theorema* for tele-teaching, as well as on a small-scale experiment demonstrating its usefullness for training graduate students in predicate-logic proving.

Introduction

The Theorema project aims to close the gap between computing and proving by adding proving support to an existing computer algebra system (see [Buchb:96]). The distinctive features of Theorema are:
- integration of an existing rewrite rule language (Mathematica) into our logic;
- imitation of human proof style;
- combination of functor style programming with proving;
- integration of proving into math textbooks generation and interactive math teaching;
- multi-style proving triggered by special context.

One of the most important applications of the *Theorema* system is distance education over the computer network, either locally (LAN / WAN campus network) or globally (internet). For the use over the internet we developed a special interface connecting the system to an internet server, as well as a downscaled version of the system which allows input/output of mathematical formulae in/from an internet browser.

The main objectives of the formal training in our system are:
- Language training: learn and train the use of mathematical language for concise and exact expressing of models.
- Formal models: learn and train how to build mathematical models for concrete real problems (defining concepts, defining properties, defining and exploring problems).
- Conjecture and prove: learn and train the formulation of conjectures about the properties of mathematical objects, prove and disprove conjectures.

Software systems for distance education generally fall into two cathegories:

White-Box Systems are collections of sample-formulae and/or programs presented in source-code in a certain programming language or computer-algebra system, sometimes accompanied by black-box implementations of some additional functions needed by these examples. The formulae / algorithms complement a textbook by illustrating various notions from it, and are intented to be executed interactively by the student. The disadvantage of these type of systems is that the student has to become familiar with the programming language (computer algebra system) which is used, which usually requires a significant effort.

Black-Box Systems are collections of executable programs/routines which implement various algorithms and data structures corresponding to the notions presented in the textbook, and which can be executed using certain menus, input forms (for specifying parameters), etc., producing then certain results presented as numbers, tables, graphics, animations, etc. The disadvantage of these systems is that only a very narrow class of problems can be implemented by each program / routine, hence illustrating a textbook can become a quite huge task..

The Theorema system overcomes these shortcomings by implementing the sytanx and the semantics of a language which is almost identical to the natural mathematical language traditionally used in textbooks of mathematics, computer science, and engineering. Altohugh the system is implemented on top of the computer algebra system *Mathematica* 3.0 [Wolfram:96], the user does not need to learn the programming language of this system, but only some simple syntactical constructs similar to the ones used in the mathematical textbooks. The system is presented to the user in the form of an intelligent environment for specifying knowledge and defining algorithms using higher order predicate logic, for experimenting with them, for proving in various mathematical domains, and for developing mathematical texts in natural language. For a more detailed description of the components, as well as of the theoretical and algorithmical background of the project, the reader is refered to [Buchb-et-al:97] and [Buchb-Jebe:98].

1. Training Content

The aim of the system is to assist students in learning to use and to manipulate formal notions, using the language of mathematics. This training has several dimensions:

- training the language of mathematics;
- building new notions;
- organizing and using mathematical knowledge;
- training in proving theorems.

Training the Language of Mathematics

The Theorema system allows to construct arbitrarily complex formulae using the customary symbols of mathematics, including logical symbols and quantifiers. Moreover, these formulae are “executed” – that is evaluated – by the system whenever this is possible (e.g. in case of quantifiers over finite domains). Therefore, the student gets a “hands-on” experience of working with mathematical sentences, he can improve his understanding of the notions by seing them at work “live”.

The system provides the syntax and the semantics (functionality) of all the usual logical symbols, as well as of the basic mathematical domains: numbers (natural, integer, rational), tuples (lists) and sets. Using these, in fact, one is able to construct any mathematical domain. Fig. 1. presents some formulae in the Theorema syntax and the result of their evaluation.

Building New Concepts

Building new concepts in the Theorema system is done exactly as in the usual activity of mathematicians (scientists): one can define new functions over already existing domains using the facility for function definition, and one can define new mathematical domains using the more evolved facility of “functors”, which constructs new domains starting from known ones. All these new functions and domains can be immediately evaluated in the system, because they are implemented as “live” objects which can be executed. Fig. 2. presents some definitions and their use.

$$\underset{i=0,\ldots,10}{\exists} i \leq 0$$

True

$$\sum_{i=1,\ldots,3} i$$

6

$$\left\langle \text{is-prime}[i] \;\middle|\; {}_{i=1,\ldots,10} \right\rangle$$

⟨True, True, True, False, True, False, True, False, False, False⟩

$$\left\{ i \;\middle|\; {}_{i=1,\ldots,20} \; \text{is-prime}[i] \right\}$$

{2, 3, 5, 7, 11, 13, 17, 19}

Fig. 1. Computing with formulae

Organizig and Using Mathematical Knowledge

In any texbook of mathematics (computer science, engineering), the knowledge is organized following some more or less precised rules – by grouping the text into sections and paragraphs, by giving names / numbers to mathematical sentences, etc. The Theorema system allows a very precise organization of the knowledge into a hierarchy of theories containing various definitions, propositions, lemmata, theorems, etc. For this we provide specific simple constructs (as “Definition”, “Proposition”, “Theory”) which connect symbolic names to the various mathematical assertions. The use of the respective assertions is then allowed only through these symbolic names, such that there is absolutely clear at each moment which knowledge is used for a certain purpose. In the example presented in Fig. 3, one defines some simple axioms for the addition over natural numbers, and then one uses these knowledge both for computation and for proving a new property.

A very important construct is the **functor**, which allows the construction of new domains starting from old ones. In Fig. 4 we show how one defines the cartesian product of an arbitrary domain with itself, and then how this definition can be used in order to compute over pairs of natural numbers.

Proof Training

Proving is one of the most important skills of a mathematician. Theorema offers extensive support for proof training, because it implements various proof techniques (e.g. predicate logic proving, induction proving, etc.) over various mathematical domains (numbers, lists, sets, etc.). The proofs are performed in natural style [Buchb-Licht:81], that is they resemble very much to proofs constructed by human mathematicians. Fig. 5 presents the proof by induction created by the command **Prove[…]** from Fig. 3.

any[n] :

  factorial[1] := 1

  factorial[n] := n factorial[n − 1]

factorial[6]

720

any[X, Y] :

$$X \oplus Y := \left\langle (X_i + Y_i) \;\Big|_{i=1,\ldots,|X|} \right\rangle$$

$\langle 1, 2, 3\rangle \oplus \langle 2, 3, 4\rangle$

$\langle 3, 5, 7\rangle$

any[A, B]|:

$$A \ominus B := \left\{ x \;\Big|_{x\in A}\; (x \notin B) \right\}$$

$\{2, 3, 1, 3\} \ominus \{3, 5\}$

$\{1, 2\}$

Fig. 2. Function definition and use.

$$\textbf{Definition}\Big[\text{"addition"}, \text{any}[m, n], \begin{array}{ll} m + 0 = m & \text{"+: base case"} \\ m + n^{+} = (m + n)^{+} & \text{"+: ind case"} \end{array}\Big]$$

$$\text{Compute}[0^{+++} + 0^{++}, \text{KB} \to \text{Definition}[\text{"addition"}]]$$

$$((((0^{+})^{+})^{+})^{+})^{+}$$

$$\textbf{Proposition}[\text{"zero from left"}, \text{any}[n], \; 0 + n = n \quad \text{"ZL"}]$$

$$\begin{array}{l} \textbf{Prove}[\text{Proposition}[\text{"zero from left"}], \\ \quad \text{KB} \to \text{Definition}[\text{"addition"}], \\ \quad \text{Prover} \to \text{InductionProver}\;] \end{array}$$

Fig. 3. Organizing and using mathematical knowledge.

$$\begin{array}{l} \textbf{Definition}\Big[\text{"CartesianProduct"}, \text{any}[X, Y], \\ \quad \text{CartesianProduct}[X, Y] = \text{Functor}\Big[CP, \text{any}[a, b, c, d], \\ \quad \dfrac{\mathbb{S} = \langle + : CP \times CP \to CP,\; < \;: CP \times CP \to \mathbb{T}\rangle}{\begin{array}{ll} \underset{CP}{\in}[a] \Leftrightarrow \Big((|a| = 2) \bigwedge \underset{X}{\in}[a_1] \bigwedge \underset{Y}{\in}[a_2]\Big) & \text{"isElem"} \\ \langle a, b\rangle \underset{CP}{+} \langle c, d\rangle = \Big\langle a \underset{X}{+} c, b \underset{Y}{+} d\Big\rangle & \text{"+"} \\ \langle a, b\rangle \underset{CP}{<} \langle c, d\rangle \Leftrightarrow \Big(a \underset{X}{<} c \bigwedge b \underset{Y}{<} d\Big) & \text{"<"} \end{array}} \Big]\Big] \end{array}$$

$$\begin{array}{l} \textbf{Definition}[\text{"N2"}, \\ \quad N2 = \text{CartesianProduct}[\mathbb{N}, \mathbb{N}]] \end{array}$$

$$\begin{array}{l} \textbf{Proposition}\Big[\text{"associativity"}, \text{any}[x, y, z], \\ \quad x \underset{\mathbb{N}}{+} \Big(y \underset{\mathbb{N}}{+} z\Big) = \Big(x \underset{\mathbb{N}}{+} y\Big) \underset{\mathbb{N}}{+} z \quad \text{"a"}\Big] \end{array}$$

$$\begin{array}{l} \textbf{Theory}\Big[\text{"CP of } \mathbb{N}\text{"}, \\ \quad \textbf{Definition}[\text{"N2"}] \\ \quad \textbf{Proposition}[\text{"associativity"}]\Big] \end{array}$$

$$\text{Compute}\Big[\langle a0, a1\rangle \underset{N2}{+} \Big(\langle b0, b1\rangle \underset{N2}{+} \langle c0, c1\rangle\Big), \text{KB} \to \textbf{Theory}[\text{"CP of } \mathbb{N}\text{"}]\Big]$$

$$\Big\langle \Big(a0 \underset{\mathbb{N}}{+} b0\Big) \underset{\mathbb{N}}{+} c0, \Big(a1 \underset{\mathbb{N}}{+} b1\Big) \underset{\mathbb{N}}{+} c1 \Big\rangle$$

Fig. 4. Functor definition.

Prove:

(ZL) $\underset{n}{\forall}(0 + n = n)$,

under the assumptions:

(+: base case) $\underset{m}{\forall}(m + 0 = m)$,

(+: ind case) $\underset{m,n}{\forall}(m + n^{+} = (m + n)^{+})$.

We prove (ZL) by induction on n.

Induction Base: [this cell is closed]

Induction Step:

Induction Hypothesis:

(ZL.IH) $0 + n_1 = n_1$

Induction Conclusion:

(ZL.IS) $0 + n_1{}^{+} = n_1{}^{+}$.

A proof by simplification of (ZL.IS) works.

Simplification of the lhs term:

$0 + n_1{}^{+} =$ by (+: ind case)

$(0 + n_1)^{+} =$ by (ZL.IH)

$n_1{}^{+}$ ⌋

Fig. 5. A proof by induction.

2. Modalities for Distance Learning

Depending on the software available to the user, and on its place w.r.t. the system, there are several possibilities of using the Theorema system.

Campus Access

The most intensive interaction can be realised when the system is used in the same LAN / WAN (Local / Wide Area Network), which is the case in the university campus. In this situation the students receives access to a certain directory which contains the [compiled version of the] Theorema system and the start-up examples. (A practical experiment under these conditions is presented below.) The students access this directory using the usual X-windows interface under Unix and a campus-wide licence of Mathematica 3.0, thus they can inspect the examples and run them using the system, they can also modify and re-run the examples at will. The three important ways of working are:

- Passive inspection: The student examines the examples already prepared by the teacher (definitions of notions, lists of properties, computation samples, proofs of theorems).

- Interactive use: The students modifies the given examples and constructs new ones, and calls the Therema system for performing computations and for producing proofs of new theorems.

- Active use: During the computation and the proof process, the student can interact with the system giving hints on how the computation / proof should proceed. This is done using special windows (notebooks) which display additional information in real-time and which allow various control functions through a menu-driven interface.

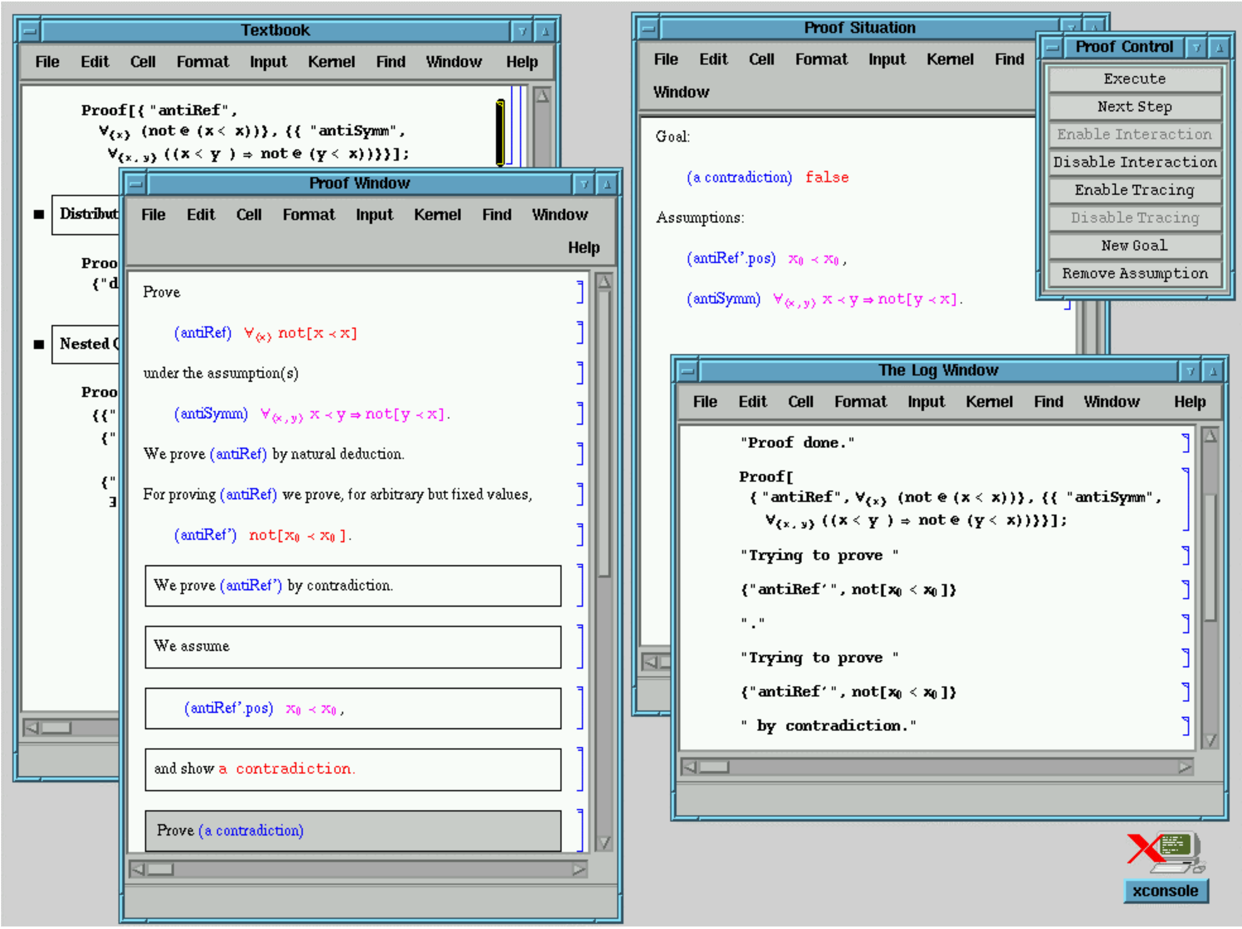

Fig. 6. The windows of Theorema during an active session.

Figure 6 presents a typical screen when using the predicate-logic prover in active mode, with several Mathematica notebooks:

- "Proof Control" displays various buttons for controlling the proofs and the level of interaction.
- "Textbook" contains the theorems to prove. By selecting a command of type "Proof[...]" and then pressing "Execute" from the proof control window, the proof is initiated.
- "Proof Window" contains the actual text of the proof. In interactive mode, the partial proof obtained at the current stage is displayed with gray cells for the still unsolved subproofs and with framed cells for the most recent proof step which was executed. By pressing "Next Step" in "Proof Control" a further proof step will be executed.
- "Proof Situation" shows the current goal (formula to be proven) and the current assumptions.
- "Log Window" shows a trace of the proof steps and other operations performed.

Internet access

The Theorema system is also accessible over the world-wide internet, starting at a demo entry point in the Theorema home-page (http://www.theorema.org). Again there are three main possibilities of use:

- Passive inspection: The prepared examples are presented in HTML format – hence they can be used whithout any additional software. The open / close facility of Mathematica notebooks is simulated by links and additional pages, and the mathematical formulae which cannot be rendered in HTML format are presented as GIF pictures, which alters to a certain extent the quality of the presentation.

However, the original notebooks can also be inspected, by using the free-of-charge MathReader from Mathematica – which can be coupled as a plug-in to any internet browser.

- Interactive use: The student employs a down-sized version of the Theorema system, which allows input / output of mathematical formulae in the Theorema syntax (of course, the use of Mathematica 3.0 is also necessary). This interface communicates with a small internet client via MathLink, which then communicates via internet with the Theorema server installed at RISC. The user can input requests and can obtain results from the Theorema system in form of notebooks or notebook cells.

- Active use: A full internet version of the system is not possible to implement, mainly due to the relative slowness of the internet connections over large distances. However, some of the functionality of the interactive version can be achieved by implementing certain control functions in the down-sized interface version of the system.

Classroom Experiment

During the academic year 1997/98 we performed a distance teaching experiment with 15 new PhD students at our institute, in the frame of a special lecture on scientific training. After one semester of training, which included also proof training, the students have been assigned a written test containing 7 proof problems from predicate logic, ranging from low difficulty to medium difficulty, and then they were introduced to the *Theorema* system and they were given access to a specific directory containing the software and several proof examples. The students had the possibility of using the system over the LAN of the institute, in all the three modes (passive, interactive and active), over the second semster of study. (A final questionnaire revealed that the students used the system in average 6 hours). Furthermore, during the semester three lectures were assigned to using the system together in the classroom and discussions about the proof problems and about the performance of the system. At the end of the semester a final test of 7 problems was given (3 from the old ones and 4 new), with higher difficulty than the first one. Both the first and the final test were graded by assigning scores from 0 to 4 to each problem-solution. Additionally, the students filled –in a questionnaire concerning their experience with the system.

The tests showed a significant improvement in the performance of the students for constructing natural proofs in predicate logic: the average student score raised from 37.5% to 62.5%.

Acknowledgements
Partially supported by the Austrian Science Fundation - project FO-1302.

References

[Buchb-Licht:81] Buchberger, B., & Lichtenberger, F. (1981). *Mathematics for Computer Science I - The Method of Mathematics* (in German). Second Edition. Springer Verlag.

[Buchb:96] Buchberger, B. (1996). Proving, Solving, Computing. A Language Environment Based on Mathematica. Proceedings of the Multiparadigm Logic Programming Conference, Bonn, Sept 1996, Springer Vienna – New York.

[Buchb-et-al:97] Buchberger, B., Jebelean, T., Kriftner, F., Marin, M., Tomuta, E. ,& Vasaru, D.(1997). A Survey of the Theorema Project. *Proceedings of ISSAC'97, W. Kuechlin (ed),* ACM Press, 384-391.

[Buchb-Jebe:98] Buchberger, B., Jebelean, T. (eds). (1998). *Proceedings of the Second International Theorema Workshop, June 1997,* RISC Report 98-10.

[Wolfram:96] Wolfram, S. (1996). *The Mathematica Book.* Wolfram Media and Cambridge University Press.