Theorema 2.0: A Brief Tutorial

(Invited Tutorial)

Wolfgang Windsteiger
RISC, JKU Linz
Linz, Austria
Email: Wolfgang.Windsteiger@risc.jku.at

***Keywords*-Theorema; Mathematical Assistant System**

I. What is this all About?

The Theorema system aims to be a computer assistant for the working mathematician. Support should be given throughout all phases of mathematical activity, from introducing new mathematical concepts by definitions or axioms, through first (computational) experiments, the formulation of theorems, their justification by an exact proof, the application of a theorem as an algorithm, to the dissemination of the results in form of a mathematical publication, the build up of bigger libraries of certified mathematical content and the like. One focus lies on the natural style of system input (in form of definitions, theorems, algorithms, etc.), system output (mainly in form of mathematical proofs) and user interaction. When using the Theorema system, a user should not have to follow a certain style of mathematics enforced by the system (e.g. basing all of mathematics on set theory or certain variants of type theory), rather should the system support the user in her preferred flavor of doing math. The new implementation of the system, which we refer to as Theorema 2.0, is open-source and available through GitHub.

II. Working with Theorema 2.0

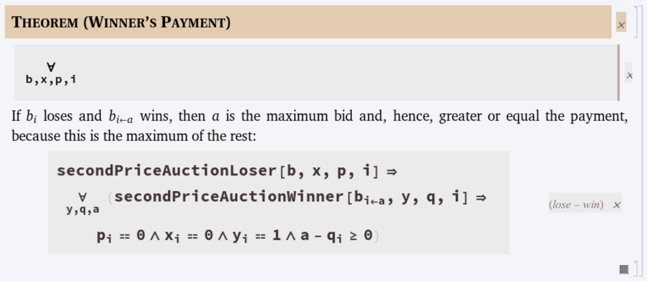
The development of the Theorema System has been initiated by Bruno Buchberger in the beginning of the 1990's. He implemented a first version of the Theorema language and some first automated provers in Mathematica. Over the years, the Theorema group at RISC extended the system remarkably in various directions. Special reasoning methods for elementary analysis, set theory, induction over various domains, geometry, boundary value problems, and many more have been invented and implemented in the frame of Theorema, see e.g. [2], [1], [3]. Since version 7 released in 2008, the Mathematica Notebook FrontEnd supports dynamic objects, which allows to implement modern interactive user-interfaces within the Mathematica environment. This was the starting point for Theorema 2.0, a re-design and a complete re-implementation of the Theorema system with special emphasis on intuitive click-based user interaction, see [4].

Theorema 2.0 is implemented as a Mathematica add-on package, and when loaded into Mathematica it presents to the user its main interface component, the *Theorema Commander*. The Theorema Commander is where the user gets support in mathematical activities she wants to undertake, like doing proofs or computations. The math she is working on is written in Theorema notebooks, which are just Mathematica notebooks using a special stylesheet in order to support special behavior of certain Theorema-specific items, like e.g. definitions or theorems. When working with Theorema, one creates a mathematical document using all the capabilities of the Mathematica notebook interface. Then, when one wants to act on the math, one switches to the Theorema commander in order to initiate some action, whose result will then be documented in the notebook. Hence, Theorema 2.0 can also be seen as *semi-automated mathematical document creator*, which creates parts of a mathematical document in a fully automated way, like e.g. proofs or results of computations including their trace.

III. An Example

We demonstrate some of the features of Theorema 2.0 in an example taken from the formalization of auction theory presented in [5]. We investigate second-price auctions with n participants. An auction is described by n-tuples b, x, and p, where $b_i, p_i \in \mathbb{R}_0^+$ refer to participant i's bid and payment, respectively, and $x_i = 1$ if participant i wins and $x_i = 0$ otherwise. We want to prove a theorem that captures the situation when i loses with b, x, and p and wins with $b_i = a$ (and all other bids in b unmodified), y and q. Figure 1 shows how this theorem can be expressed in the Theorema language and how a theorem is written in a Theorema notebook. Note that in the formulation of the theorem we refer to predicates 'secondPriceAuctionLoser' and 'secondPriceAuctionWinner', which are defined elsewhere in the notebook. In order to initiate an automated proof of this theorem, we would switch to the Theorema commander and choose the Prove-activity, which guides us through the process of setting up the prover. The proof goal is defined by simply selecting the notebook cell containing the goal formula.

Next is the setup of the knowledge base being available in the proof, which is achieved through the *knowledge browser* as illustrated in Figure 2. The knowledge browser shows an outline of each notebook displaying only formal mathemat-



Theorem (Winner's Payment)

$\forall_{b,x,p,i}$

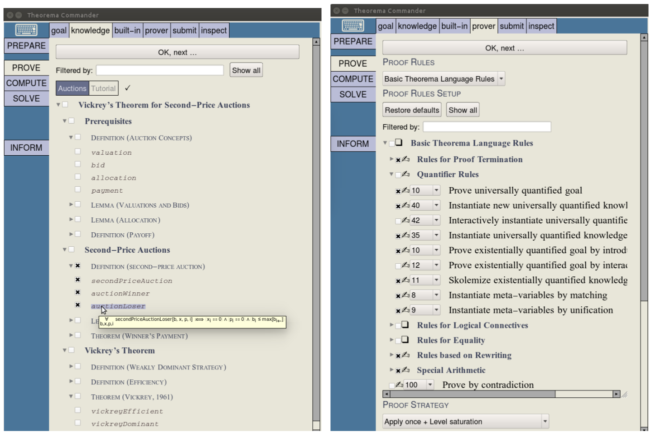
If b_i loses and $b_{i\leftarrow a}$ wins, then a is the maximum bid and, hence, greater or equal the payment, because this is the maximum of the rest:

$$\text{secondPriceAuctionLoser}[b, x, p, i] \Rightarrow \forall_{y,q,a} \left(\text{secondPriceAuctionWinner}[b_{i\leftarrow a}, y, q, i] \Rightarrow p_i == 0 \wedge x_i == 0 \wedge y_i == 1 \wedge a - q_i \geq 0\right)$$

(lose – win)

Figure 1. The theorem written in Theorema language.

ical content, like definitions and theorems, preserving the sectional structure of the notebook while hiding all informal parts, such as text and graphics. Sectional groupings can be collapsed in order to gain overview over the sectional structure. For the formal entities, the commander does not display the formulas in detail, it rather shows the formulas' labels only and presents the entire formula only as a tooltip when hovering the mouse moves over the label[1]. Each formula is accompanied with a checkbox that, when checked, add the corresponding formula to the knowledge base. Thus, in the example displayed, the knowledge base consists of three definitions labeled 'secondPriceAuction', 'auctionWinner', and 'auctionLoser', respectively.



Filtered by:

Auctions | Tutorial ✓

- ▼ □ Vickrey's Theorem for Second–Price Auctions
  - ▼ □ Prerequisites
    - ▼ □ Definition (Auction Concepts)
      - □ valuation
      - □ bid
      - □ allocation
      - □ payment
    - ▶ □ Lemma (Valuations and Bids)
    - ▶ □ Lemma (Allocation)
    - ▶ □ Definition (Payoff)
  - ▼ □ Second–Price Auctions
    - ▼ ☒ Definition (second–price auction)
      - ☒ secondPriceAuction
      - ☒ auctionWinner
      - ☒ auctionLoser
    - ▶ □ Le
    - ▶ □ Theorem (Winner's Payment)
  - ▼ □ Vickrey's Theorem
    - ▶ □ Definition (Weakly Dominant Strategy)
    - ▶ □ Definition (Efficiency)
    - ▼ □ Theorem (Vickrey, 1961)
      - □ vickreyEfficient
      - □ vickreyDominant

$\forall_{b,x,p,i}$ secondPriceAuctionLoser[b, x, p, i] :⟺ $x_i == 0 \wedge p_i == 0 \wedge b_i \leq \max[b_{i\leftarrow}]$

Proof Rules

Basic Theorema Language Rules ▾

Proof Rules Setup

Filtered by:

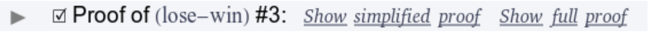
- ▼ □ Basic Theorema Language Rules
  - ▶ ☒ Rules for Proof Termination
  - ▼ □ Quantifier Rules
    - ☒ 10 Prove universally quantified goal
    - ☒ 40 Instantiate new universally quantified knowl
    - □ 42 Interactively instantiate universally quantifie
    - ☒ 35 Instantiate universally quantified knowledge
    - ☒ 10 Prove existentially quantified goal by introd
    - □ 12 Prove existentially quantified goal by intera
    - ☒ 11 Skolemize existentially quantified knowledg
    - ☒ 8 Instantiate meta–variables by matching
    - ☒ 9 Instantiate meta–variables by unification
  - ▶ □ Rules for Logical Connectives
  - ▶ □ Rules for Equality
  - ▶ ☒ Rules based on Rewriting
  - ▶ ☒ Special Arithmetic
  - □ 100 Prove by contradiction

Proof Strategy

Apply once + Level saturation ▾

Figure 2. The Theorema Commander: knowledge browser (left) and prover configuration (right).

Next is the setup of the prover. A prover in Theorema 2.0 consists of individual inference rules that are applied by a proof search engine in a certain order. In the Theorema commander we are able to activate or deactivate every single inference rule, and we can influence the order, in which they are applied, by assigning priorities to each rule. Usually, it is a good strategy to run the prover with default settings and a limited search depth and search time (both configurable in the prover setup). In case the proof does not succeed, one can inspect the failing proof and check whether some settings might be changed in order to prevent the prover from running into an undesired path. Otherwise, search depth and search time can be increased in order to allow the prover to terminate successfully. In any case, when the prover stops it writes an answer into the notebook as shown in Figure 3.



▶ ☑ Proof of (lose–win) #3: Show simplified proof Show full proof

Figure 3. A reference to the Theorema proof in the notebook.

The checked box indicates a successful proof, otherwise a question mark '?' would indicate failure. Since different settings may lead to different proofs we also see a version number for this proof, in the example it is proof version number 3. Finally, we have links to the automatically generated proofs, in the example we have a simplified version showing only the branches and steps that are necessary for the successful proof and the full version displaying all the branches explored during proof search.

When clicking the link to the simplified proof, a nicely formatted version of the proof is displayed in a separate window as shown in Figure 4. Theorema explains every logical step in the proof in natural language. The language for the proof is configurable, the example shows the default case English. On the top border of the proof window we see the proof simplification bar, an area where the dislayed proof can be simplified. Theorema 2.0 currently offers three aspects of proof simplification:

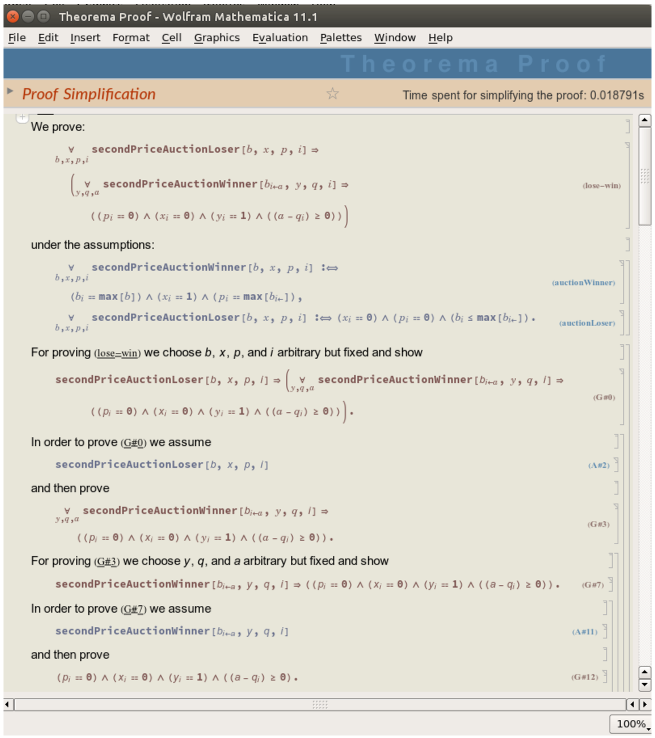
1) elimination of failing and pending branches in the proof tree generated during proof search,
2) elimination of proof steps that do not contribute to the successful proof, and
3) elimination of formulas in the proof that do not contribute to the successful proof.

In the example in Figure 4 we show a simplified proof with all aspects activated. We see that formula 'secondPriceAuction' does not occur in the proof's assumptions although it was specified to be in the knowledge base by the user explicitly. Since definition 'secondPriceAuction' is never used in the proof it is eliminated from the original knowledge base automatically during proof simplification.

IV. Conclusion

In this tutorial we present the current version of Theorema 2.0. The main focus of the system lies on natural style communication with the user in both input and output. Theorema notebooks give a possibility to write mathematical documents containing mathematical formulas in human-readable style but still in a machine-processable form. Furthermore, we demonstrate how a user interacts with the system in order to obtain an automated proof of a given statement, and how the system answers with a human-comprehensible proof written in human style. These

[1] The screenshot in Figure 2 is not readable in all details, it should only give an idea how the knowledge browser appears to the user.


Theorema Proof - Wolfram Mathematica 11.1
File Edit Insert Format Cell Graphics Evaluation Palettes Window Help
Theorema Proof
Proof Simplification
100%

Time spent for simplifying the proof: 0.018791s

We prove:

$$\underset{b,x,p,i}{\forall} \mathtt{secondPriceAuctionLoser}[b, x, p, i] \Rightarrow \left(\underset{y,q,a}{\forall} \mathtt{secondPriceAuctionWinner}[b_{i\leftarrow a}, y, q, i] \Rightarrow ((p_i == 0) \wedge (x_i == 0) \wedge (y_i == 1) \wedge ((a - q_i) \geq 0))\right) \quad \text{(lose-win)}$$

under the assumptions:

$$\underset{b,x,p,i}{\forall} \mathtt{secondPriceAuctionWinner}[b, x, p, i] :\Leftrightarrow (b_i == \max[b]) \wedge (x_i == 1) \wedge (p_i == \max[b_{i\leftarrow}]), \quad \text{(auctionWinner)}$$

$$\underset{b,x,p,i}{\forall} \mathtt{secondPriceAuctionLoser}[b, x, p, i] :\Leftrightarrow (x_i == 0) \wedge (p_i == 0) \wedge (b_i \leq \max[b_{i\leftarrow}]). \quad \text{(auctionLoser)}$$

For proving (lose=win) we choose *b*, *x*, *p*, and *i* arbitrary but fixed and show

$$\mathtt{secondPriceAuctionLoser}[b, x, p, i] \Rightarrow \left(\underset{y,q,a}{\forall} \mathtt{secondPriceAuctionWinner}[b_{i\leftarrow a}, y, q, i] \Rightarrow ((p_i == 0) \wedge (x_i == 0) \wedge (y_i == 1) \wedge ((a - q_i) \geq 0))\right). \quad \text{(G\#0)}$$

In order to prove (G#0) we assume

$$\mathtt{secondPriceAuctionLoser}[b, x, p, i] \quad \text{(A\#2)}$$

and then prove

$$\underset{y,q,a}{\forall} \mathtt{secondPriceAuctionWinner}[b_{i\leftarrow a}, y, q, i] \Rightarrow ((p_i == 0) \wedge (x_i == 0) \wedge (y_i == 1) \wedge ((a - q_i) \geq 0)). \quad \text{(G\#3)}$$

For proving (G#3) we choose *y*, *q*, and *a* arbitrary but fixed and show

$$\mathtt{secondPriceAuctionWinner}[b_{i\leftarrow a}, y, q, i] \Rightarrow ((p_i == 0) \wedge (x_i == 0) \wedge (y_i == 1) \wedge ((a - q_i) \geq 0)). \quad \text{(G\#7)}$$

In order to prove (G#7) we assume

$$\mathtt{secondPriceAuctionWinner}[b_{i\leftarrow a}, y, q, i] \quad \text{(A\#11)}$$

and then prove

$$(p_i == 0) \wedge (x_i == 0) \wedge (y_i == 1) \wedge ((a - q_i) \geq 0). \quad \text{(G\#12)}$$

Figure 4. The nicely formatted Theorema proof.

features make Theorema 2.0 a perfect tool for supporting the teaching of mathematics, in particular the teaching of rigorous proving, and a nice tool for building up formalized mathematical theories.

References

[1] B. Buchberger. Theorema: A Proving System Based on Mathematica. *The Mathematica Journal*, 8(2):247–252, 2001.

[2] B. Buchberger, A. Craciun, T. Jebelean, L. Kovacs, T. Kutsia, K. Nakagawa, F. Piroi, N. Popov, J. Robu, M. Rosenkranz, and W. Windsteiger. Theorema: Towards Computer-Aided Mathematical Theory Exploration. *Journal of Applied Logic*, 4(4):470–504, 2006.

[3] B. Buchberger, C. Dupre, T. Jebelean, F. Kriftner, K. Nakagawa, D. Vasaru, and W. Windsteiger. The Theorema Project: A Progress Report. In M. Kerber and M. Kohlhase, editors, *Symbolic Computation and Automated Reasoning (Proceedings of CALCULEMUS 2000, Symposium on the Integration of Symbolic Computation and Mechanized Reasoning)*, pages 98–113. St. Andrews, Scotland, Copyright: A.K. Peters, Natick, Massachusetts, 6-7 August 2000.

[4] Bruno Buchberger, Tudor Jebelean, Temur Kutsia, Alexander Maletzky, and Wolfgang Windsteiger. Theorema 2.0: Computer-Assisted Natural-Style Mathematics. *JFR*, 9(1):149–185, 2016.

[5] A. Maletzky and W. Windsteiger. The Formalization of Vickrey Auctions: A Comparison of Two Approaches in Isabelle and Theorema. In H. Geuvers, M. England, O. Hasan, F. Rabe, and O. Teschke, editors, *Intelligent Computer Mathematics: 10th International Conference, CICM 2017, Edinburgh, UK, July 17-21*, volume 10383 of *Lecture Notes in Computer Science*, pages 25–39. Springer, 2017. doi 10.1007/978-3-319-62075-6_3.