RESEARCH INSTITUTE FOR
SYMBOLIC COMPUTATION

JOHANNES KEPLER
UNIVERSITY LINZ

# Automated Theorem Proving in the Classroom

W. Windsteiger

August 2021

**RISC Report Series No. 21-15**

ISSN: 2791-4267 (online)

Available at `https://doi.org/10.35011/risc.21-15`

**JOHANNES KEPLER**
**UNIVERSITY LINZ**
Altenberger Str. 69
4040 Linz, Austria
www.jku.at
DVR 0093696

# Automated Theorem Proving in the Classroom

Wolfgang Windsteiger

Research Institute for Symbolic Computation (RISC)
Johannes Kepler University Linz (JKU)
4232 Hagenberg, Austria
Wolfgang.Windsteiger@risc.jku.at

**Abstract.** We report on several scenarios of using automated theorem proving software in university education. In particular, we focus on using the Theorema system in a software-enhanced logic-course for students in computer science or artificial intelligence. The purpose of using logic-software in our teaching is *not* to teach students the proper use of a particular piece of software. In contrast, we try to *employ* certain software in order to spark students' motivation and to support their understanding of logic principles they are supposed to understand after having passed the course. In a sense, we try to let the software act as a logic-tutor, the software is not an additional subject we teach.

## 1 Introduction

In the course of the development of a new curriculum for a bachelor study in computer science at JKU Linz several years ago we had the opportunity to design a new logic-course [1,6]. Logic should play a more prominent role and will be taught in the first semester for approximately 400 beginner students of computer science or, since 2019, the then newly introduced bachelor studies in artificial intelligence. The novel aspects that we wanted to add to a classical course in logic were

- that students should see *logic in action* through the incorporation of logic software and
- that certain *modern topics* (such as e.g. automated or interactive theorem proving and satisfiability modulo theories (SMT)) should be presented to the students early in their studies instead of somewhere towards the end in some special lecture.

The course is designed consisting of four more or less independent modules taught by four different lecturers. In each module the lecturer uses the software s*he prefers. In this report we concentrate on module three devoted to *reasoning in first-order predicate logic* and supported by the Theorema system [5,7].

It is important to clarify the content and the didactical goals of the course-module on proving in first-order logic in order to be able to judge the appropriateness of software support in that lecture. Propositional logic and reasoning therein as well as the syntax and semantics of first-order predicate logic can be assumed to be familiar to the students from previous modules. The ultimate goal of the proving-module is to teach students how to do mathematical proofs "by hand", because we consider it as an important capability for computer scientists to being able to reason and argue in a logically sound way, e.g. when talking about correctness of computer programs.

Unlike in mathematics, where proving is taught to students traditionally through a multitude of proof examples exhibited during their studies, we follow the approach of *explicitly teaching the method of proving* via formal proofs. On the one hand, we present a small set of proof rules and, on the other hand, we consider a proof to be a finite tree, whose vertices are *proof situations* and each edge represents *a transition*, i.e. a logical step, from one situation to another justified by one of the above rules. Instead of one of the known classical deduction calculi like sequence calculus, Gentzen calculus, or natural deduction calculus, we use our own set of reasoning rules, which is not necessarily the smallest possible set that allows a complete reasoning procedure for first-order logic but which aims at generating proofs like one would do them by hand. Moreover, we present the activity of proving as a *search procedure* aiming at finding a successful proof tree, and we explain the transition from a formal proof tree into a traditional mathematical proof.

In Section 2 we give a brief overview on the philosophy and the main features of the Theorema system. Section 3 describes the didactical setting how we incorporated Theorema into the teaching of logic, and, finally, Section 4 summarizes some of the insights we gained from this didactical experiment.

## 2 The Theorema System

### 2.1 Theorema: A Brief Overview

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/his preferred flavor of doing math.

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. [3,2,4]. Since Mathematica 7 released in 2008, the Mathematica Notebook FrontEnd supports dynamic objects, which allow 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 [5]. Theorema 2.0 is open-source and available through GitHub.

### 2.2 Proving with Theorema 2.0

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 various mathematical activities, like doing proofs or computations. Mathematical content can be mixed with text and graphics and 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. When one wants to interact with the math, one switches to the Theorema commander in order to initiate some action, whose result will then be documented back in the notebook. Hence, Theorema 2.0 can also be seen as *semi-automated mathematical document creator*, which creates parts of a mathematical document, e.g. proofs, in a fully automated way. We want to emphasize, however, that Theorema is *not just an automated prover*, although in this presentation we *concentrate on proving with Theorema.*

In order to generate an automated proof of some statement, the user would first type the statement into a formula-cell in the notebook, usually inside a named Theorem-, Lemma-, or Proposition-environment. All environments may carry names as well as each formula inside an environment can carry a separate label as name. Then one would switch to the Theorema commander and choose the Prove-activity, which guides 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* that shows an outline of each open notebook displaying only formal mathematical 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 whole document. 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 as a tool-tip when hovering the mouse over the label. Each formula is accompanied with a checkbox that, when checked, includes the corresponding formula into the knowledge base.

The next step in the PROVE-activity 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 a priority to each rule. In the final step a summary of all chosen settings is presented and the prove-task can be submitted. Usually, it is a good strategy to first run the prover with default settings and a limited search depth and search time (both configurable in the prover setup). In case the proof would not succeed, the failing proof can be inspected and checked whether certain 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. When the prover stops it writes an answer back into the notebook right below the environment, from which the proof has been initiated, indicating success or failure, a version number of the proof, and a link to the automatically generated proof. When clicking the link, a nicely formatted version of the proof explained in natural language is displayed in a separate window, which also offers several options to simplify the proof.

During proof generation and when a proof window is open, the Theorema commander shows a *tree visualization* of the proof search. In a successful proof all nodes belonging to a successful branch are colored green, nodes in failing branches are red, and pending nodes are blue. Pending nodes are proof situations that can still be handled by one of the available rules. If such nodes are present in a proof this is an indication that the proof search did not complete, either due to reaching the search time limit or through manual interruption by the user. Simplification of a successful proof essentially removes all failing branches and pending nodes resulting in an all-green proof tree that in fact corresponds to a formal proof tree as taught in our logic-course. Click-navigation connects the Theorema commander and the proof window, i.e. when clicking a cell in the proof window, an indicator mark highlights the corresponding node in the tree view, whereas clicking a node in the tree leads to the corresponding textual description of the respective proof step in the proof display. Moreover, when hovering the mouse over a node in the tree, the name of the rule applied in that node is displayed as a tool-tip.

## 3 The Design of the Logic Course

### 3.1 The Structure of the Course

The logic-course is composed of *mandatory* and *elective components* that contribute in different ways to the final grade for the whole course. The lecture (L) is accompanied by an exercise class (E), where students need to solve problems on their own. The predicate logic module consists of *three units* and the grading of the module is based mainly on the mandatory *minitests* (M) after each unit. For each unit there is a voluntary *bonus exercise* (B) and for the whole module there is a voluntary *lab exercise*, through which students can enhance a *bad result in* *a minitest* or a *bad overall result.* Of course, students can volunteer to do bonus and lab exercises regardless of whether their achievements in the minitests require them to do so. Through publication dates and deadlines for the respective items we enforce a nested sequence of lecture/exercise—minitest/bonus for each unit as depicted in Fig. 1. This means that in one week we present the theory unit in the lecture and do concrete examples for that unit in the exercise class, in the following week there is the minitest for that unit immediately followed by the bonus exercise and, in parallel, the presentation plus the exercise class for the next unit.

|  | Week 1 | Week 2 | Week 3 | Week 4 | Week 5 |
|---|---|---|---|---|---|
| Unit 1 | L1/E1 | M1/B1 |  |  | L |
| Unit 2 |  | L2/E2 | M2/B2 |  | A |
| Unit 3 |  |  | L3/E3 | M3/B3 | B |

**Fig. 1.** Nested Module Schedule

### 3.2 The Use of Theorema in the Frame of the Logic Course

The features of the Theorema system described in Section 2 make it an ideal companion for teaching mathematical proving. However, since logic software should not be the main subject of the course, we decided to move all software-related tasks into the voluntary bonus and lab exercises. Other reasons for shifting software aspects into voluntary course tracks include the fact

- that some of the software has more the character of a prototypical research software,
- that availability and reliability may depend to some extent on operating system requirements,
- that, as in the case of Theorema with Mathematica, some software requires other proprietary software that is not available legally free of charge, and
- that we have no capacity for in-depth tutorials on how to use the different pieces of software.

In each bonus exercise students need to generate one or more proofs from the previous exercise sheet with the help of the Theorema system, i.e. they use Theorema to generate an automated proof that they previously tried (or succeeded) to do by hand during an earlier exercise. Since Theorema shows the students both a natural language presentation of the proof as well as a graphical visualization of the logical structure in form of the proof tree, we consider the Theorema system as a proof tutor. Students can effortlessly investigate how different settings of the prover influence the final proof, e.g. what difference it makes whether they allow the prover to apply certain rules or not and what effect it has whether they apply specific rules earlier than others. Through the proof simplification feature they can observe how the entire proof search including failing attempts and unused steps finally translates into a proof tree that corresponds to a successful proof that can then be presented using natural language including all the usual mathematical phrases as known from their mathematics courses.

The names the Theorema proof tree shows when hovering the mouse over the node in the tree correspond to the names of the proof rules introduced during the lecture. Again, this should contribute to the students' understanding of how to generate correct and complete formal proofs. Over the many years of teaching proving to students of various branches, the main difficulties we observed were the following:

- At the beginning of a proof, students often have no idea how to start.
- They are uncertain, whether particular steps are allowed or not.
- They are uncertain, what step to do next.
- When the proof is actually finished, they are uncertain whether something else still needs to be shown.

The Theorema system just shows them in practice what we try to advertise for their daily business to remedy the above difficulties:

- First write down all formulas in exact syntax and be careful to use the correct syntactical structure.
- Try to do a formal proof, simplify it, and present it in natural language.
- When unsure how to proceed or unsure whether a certain step is allowed or not, concentrate on the syntactical structure of the formulas and carefully check, which rules can be applied and which not.
- Finally, just watch out to close all branches of the tree through application of an appropriate rule.

We try to let students experience this procedure in the frame of the lab exercise. When they do the lab exercise most of them have done already the three bonus exercises with Theorema, so they are familiar with generating computer proofs using the system. While in the bonus exercises the requirement was always to submit a computer-generated proof, the lab exercise requires them to submit a hand-written proof. To be precise, the task is to produce a mathematical proof of some simple statement that they know from one of their mathematics courses in the style how they do proofs *in these courses*. In order to come up with a proof, the advice is to first let Theorema do the proof, then study this proof and try to understand each single step, and finally write up their own version of the proof in their own words.

### 3.3 Students' Difficulties when Using Theorema

There are two main issues when working with Theorema, namely *input syntax for mathematical expressions* and *prover configuration*. We now briefly describe how we try to circumvent them in our use of Theorema in the logic-course.

Although Theorema supports *user-friendly two-dimensional input syntax* as used in mathematics textbooks, it turns out in practice that people—and this is not limited to students only—have severe difficulties to enter formulas exhibiting an appropriate syntactical structure using the keyboard. This is due to the fact that not enough emphasis is put on the tree structure of a formula when teaching the language of mathematics. Most people still regard a formula as basically a "line of (special) characters", inter-mixed with sub- and superscripts and other special constructions such as fractions etc. Theorema offers *input palettes*, through which the most common formula structures can be entered by mouse-click. Provided one proceeds along the tree-structure of the formula, it is guaranteed that the resulting formula has the right structure because every input button adds invisible parentheses that fix the syntactic structure regardless of any operator precedences that might be defined in the background. Nevertheless, we hide the potential difficulties regarding formula input from the students by providing Mathematica notebooks containing all formulas needed in their exercises. It should be noted, however, that if we had more time available in the course, it could be worthwhile to use Theorema palette input when teaching the syntax of predicate logic in order to let students gain practice with the tree structure of mathematical expressions in the language of predicate logic.

Concerning *prover configuration*, Theorema allows the user to turn every proof rule *on* or *off* and for every rule to set the *priority*, which influences the sequence in which rules are applied during the proof search. Of course, Theorema has defined default settings that turn out to generate reasonable proofs in many cases. Turning off certain rules can lead to "unprovable theorems", i.e. statements that *are* true but cannot be proved with Theorema, whereas turning on additional rules will not prevent Theorema from giving a proof provided it does give one without these rules. What can happen is that it takes longer to find the proof because additional rules might distract the proof search, or that Theorema generates a different proof when a new rule guides the proof search into a new direction. If this new direction still gives success then it may hide another successful proof, because the proof search stops with the first success. The same is true for priorities in the configuration, i.e. assigning different priorities may lead to longer search or to different proofs. Moreover, changing the setup may require an adaption of the search depth limit in order to still find the desired proofs. Playing with these features is an interesting endeavor for advanced students and professionals, for beginner students it might lead to frustration if the success rate is too low. Therefore, we provide hints in the notebooks for those cases where the default settings need some adjustment or we provide estimated computing times for those cases where students should be prepared to wait a little while until the successful proof appears on the screen.

## 4 Observations, Results, and Lessons Learned

From the arrangement of units in the module as shown in Fig. 1 we see the crucial role of bonus 1 and 2, because they are enforced to be done *before* minitest 2 and 3, respectively. From a didactical point of view, this gives us the possibility to design their content in such a way that students benefit from the bonus for the *next minitest*. Our teaching hypothesis is that students improve their own proving skills by working with the Theorema system in the ways described in Section 3. Ideally, this improvement should then show an impact in the consecutive minitests. Any effect from doing the lab exercise cannot be measured through the minitests due to the sequential setup. However, the content of the lab exercise was targeted towards mathematics-style proving, so doing this exercise probably contributes to a better overall performance in mathematics.

All data on which the following observations are based are given in Appendix A.

### 4.1 Impact on Logic Minitests

We compare *average points* in the minitests among groups of students, namely

**Group All:** all students that attended the minitest.
**Group Bonus 1:** those students who did bonus exercise 1 successfully.
**Group Bonus 1+2:** those students who did *both* bonus 1 *and* 2 successfully.
**Group no Bonus:** those students who did *neither* bonus 1 *nor* 2 successfully.

In our assessments below, if we claim *statistical significance* of some average being higher or lower than some other then this is based on a *one-sided T-Test* (assuming different variances) comparing the means of the respective samples. When we list a $p$-value, then this value is the maximum probability that the averages are in fact equal although we claim them being unequal. If we omit the $p$-value then it is less than 0.05, i.e. all our claims below are statistically safe to more than 95%. It is important to recall that statistical tests do not reveal any causalities. This has to be emphasized in particular in our scenario where the groups are not assigned randomly but students actively join a group or not. Now, if group A performs better than group B, this can be because students are better *because* of being in group A or the reason can be that the *better students* (more talented, more interested, or more motivated) *choose* group A[1].

**Table 1.** Results of minitest 2 (max. 5 points) with $p$-values for equal means. Values in parentheses show the size of the groups and column ∅ contains the average scores in the group samples (based on Fig. 4).

| | ∅ | All | Bonus 1 |
|---|---|---|---|
| All (307) | 3.28 | — | — |
| Bonus 1 (139) | 3.62 | 0.002 | — |
| no Bonus (168) | 3.00 | 0.006 | $1.21 \times 10^{-6}$ |

[1] A superior setup would be if we divide the entire class into a group that does the bonus exercises and compare them to the rest that does not do the bonus exercises, but this is not feasible in our logic-course because of the voluntary character of the software-related parts.

Summarizing the results of minitest 2 shown in Table 1, we can say that those who successfully did bonus 1 perform *significantly better* with an average score of 3.62 than the overall average with 3.28, whereas those who did not participate or did not succeed in bonus 1 perform with 3.00 *significantly worse* than both other groups. Since the groups are well populated we think that these results are not random. Table 2 shows the corresponding results for minitest 3, where the differences are by far less accentuated. No group significantly over- or underperforms compared to the overall average of 3.34, only the average score 3.47 of those who succeeded in both bonus exercises is significantly higher ($p = 0.04$) than the 3.26 of those who did not participate or did not succeed in both bonus exercises. Closest to statistical significance are the $3.42 > 3.26$ of Group Bonus 1 vs. Group no Bonus ($p = 0.08$) and the $3.47 > 3.34$ of Group Bonus 1+2 vs. Group All ($p = 0.1$).

**Table 2.** Results of minitest 3 (max. 5 points) with $p$-values for equal means. Values in parentheses show the size of the groups and column ∅ contains the average scores in the group samples (based on Fig. 5).

| | ∅ | All | Bonus 1 | Bonus 1+2 |
|---|---|---|---|---|
| All (286) | 3.34 | — | — | — |
| Bonus 1 (135) | 3.42 | 0.20 | — | — |
| Bonus 1+2 (104) | 3.47 | 0.10 | 0.33 | — |
| no Bonus (141) | 3.26 | 0.22 | 0.08 | **0.04** |

### 4.2 Impact on Mathematics Skills in General

We were also interested whether the tutoring by the Theorema system might have a positive effect on the mathematical capabilities *in general*, not just on formal proving as taught in the logic-course. We therefore inspected how students performed in the final exam in the math course “Discrete Structures”, because, at least to some extent, the final exam consists of some mathematical argumentation about certain properties in discrete mathematics. In this category we compare the overall average (13.56, sample size 166) against those who successfully participated in all three bonus exercises (14.73/26), those who did not participate or did not succeed in any bonus exercise (13.19/88), those who succeeded ($\geqslant 3$ out of five points) in the lab exercise (13.70/10), and finally those who succeeded in the lab exercise and in all three bonus exercises (15.00/7). The only significant relations among those are that those with the bonus exercises are better than those without and also better than average. One would expect that those with bonus *and* lab would also outperform those that do worse than the “bonus only”-students. Statistics does not confirm that due to the small sample size in former group with only 7 students, 6 of which score 16 and the last one spoils the result with a score of 9. Without the spoiler this group would score an average of 16 and be significantly better than *all others*, see Fig. 6 for details.

### 4.3 Self-Assessment of Students when Working with Theorema

Finally, we also report about some self-assessment from students' side. As a prerequisite to being able to submit a bonus exercise, students had to answer at most two standard questions about their experiences with Theorema depending on whether they were successful in generating an automated proof with Theorema (Group A) or not (Group B). The possible answers are shown in Fig. 2 and 3. For bonus 1–3 there were 274, 251, and 180 self-assessments, respectively, and the ratio A:B was a constant 2:1. Again, with these high numbers we consider the results to be non-random.

1. I did not try or was not able to do the examples by hand, but now I think would be able to do them.
2. I did not try or was not able to do the examples by hand. I think I would still not be able to do such proofs.
3. I had no problems doing the proofs by hand. However, they are different from the Theorema proofs and I'm confused now whether my proofs are wrong.
4. I had no problems doing the proofs by hand. However, they are slightly different from the Theorema proofs because Theorema uses certain rules that I did not know. Still, I think my proofs are fine.
5. I had no problems doing the proofs by hand. However, they are slightly different from the Theorema proofs and in the future I would do my proofs differently.
6. I had no problems doing the proofs by hand. After doing the proofs with Theorema I realized that at least one of my original proofs was wrong.
7. I had a hard time doing the proofs by hand. However, I think when doing the next proof by hand, it will be equally difficult, doing the proof with Theorema did not help me for improving my own skills.
8. I had a hard time doing the proofs by hand. After doing the proof with Theorema I understand much better how all of this works. I feel that my own skills improved by using Theorema.
9. I don't see any connection between the examples from the exercises and the Bonus Exercise with Theorema

**Fig. 2.** Possible answers for Group A.

First we analyze Group A, see Fig. 9. In bonus 1 and 2 as well as when we take all three bonus exercises together there is a rather significant gap between the top-four answers (4–1–5–8), which sum up to ~75%, and the trailing five with ~25%. In bonus 3 the top-four are still the same and make up 72%, but now the top-two are 1 and 8 and make up 42%, which had severe problems with proving by hand but consider the Theorema-tutoring to be of help. Still, we want to mention that answer 7 ("proving stays equally difficult even after Theorema-tutoring") almost made it into the top-four in bonus 3, which can probably be

10. I did not try or was not able to do these examples by hand. I wanted to see how Theorema does the proofs, but I failed to produce a compete proof.
11. I did not try or was not able to do these examples by hand. Theorema is much too complicated for me to use it for such exercises.
12. I had no problems doing the proofs by hand. Unfortunately, I failed to produce a complete proof with Theorema. It would have been interesting to compare.
13. I had no problems doing the proofs by hand. I'm not interested how an automated proof looks, I have done them by hand anyway.
14. I had a hard time doing the proofs by hand. Unfortunately, I failed to produce a complete proof with Theorema. It would have been interesting to compare.
15. I had a hard time doing the proofs by hand. I'm not interested how an automated proof looks, I have done them by hand anyway.
16. I don't see any connection between the examples from the exercises and the Bonus Exercise with Theorema.

**Fig. 3.** Possible answers for Group B.

explained through the fact that exercise and bonus 3 were the most difficult with quantifier proving, proper use of definitions, and induction in the natural numbers, the latter not being supported by Theorema. The most stunning result is, however, Group A.8 ("had a hard time doing proofs by hand but feel they had improved through Theorema-tutoring"), which showed one of the weakest performances in minitest 2 with an average of 2.82 points (rank 14) and improved to rank 4 in minitest 3 with an average of 3.47 and being second-biggest group in bonus 3. Also interesting, but less easy to explain is Group A.9, i.e. those who succeeded with Theorema but do not see any connection to doing their proofs by hand. They improve from rank 8 in minitest 2 with an average of 3.00 to rank 1 in minitest 3 with an average of 3.86, where in almost half of the comparisons to other groups the better average is even statistically significant.

The picture is far less accentuated in Group B, where answer 13 is the most popular in bonus 1, bonus 3, and also over all three exercises with ~25%, see Fig. 10. Answer 12 starts in second place with ~23% in bonus 1, is then by far the most popular choice in bonus 2 with ~35% and falls down to sixth place in bonus 3, only trailed by answer 15. Those that do not see any connection between the exercises and the bonus are fortunately always the smallest portion, both in Group A and B, except for bonus 3 in Group B. An interesting development is displayed by Group B.14, which performs second-weakest with an average of 2.51 in minitest 2 and improves to second-best with an average of 3.58 in minitest 3, so maybe just spending the time with Theorema can spark the interest and improve performance even if the proof with Theorema fails.

Analyzing Groups A and B together, see Fig. 9 and 10, we see that the top-four of Group A stay top-four in the same order overall (4–1–5–8) followed by the top-two of Group B (12 and 13). It should be mentioned, however, that those, who were not able to do proofs by hand and believed they could do them after Theorema-tutoring (Group A.1, rank 2) could not justify their bold claim in the minitests, where they only rank 10th by their average scores. See Fig. 7–14 for the detailed data.

## 5 Conclusion

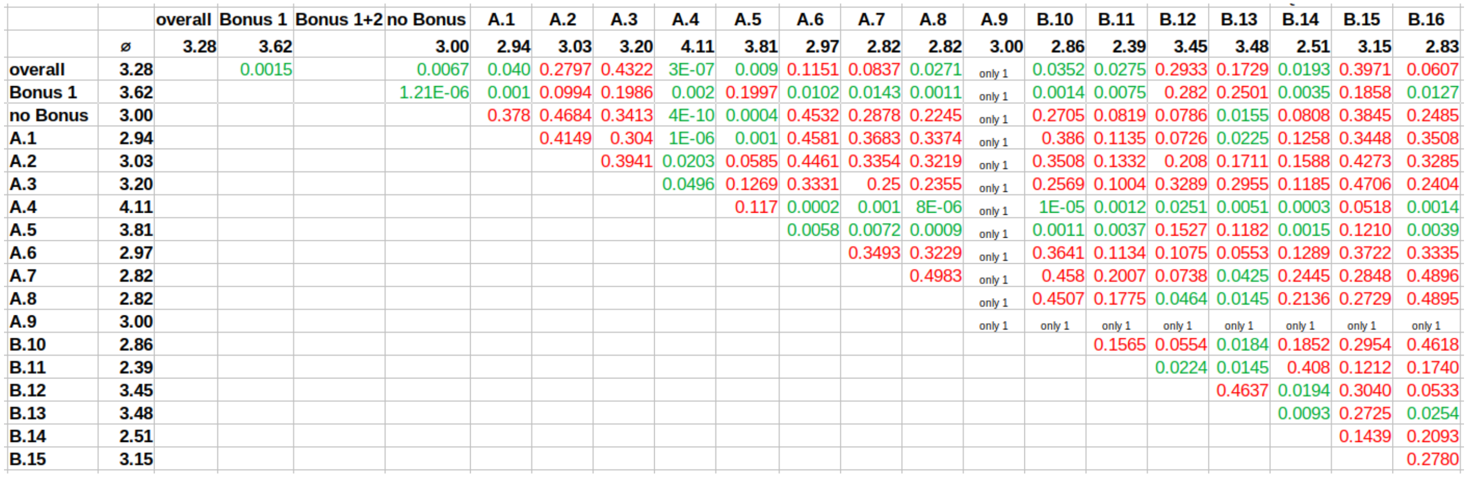
We report on a classroom experiment using the automated theorem proving software Theorema in the teaching of logic. We describe how software is applied to aid the learning process of students, how tutoring-by-software correlates with students' performance in exams, and we report on how students experienced their being tutored by software. Some interresting connections between classroom-use of software and students' own proving capabilities were observed, most notably that those who had a hard time doing proofs by hand in the frame of the exercises and claimed an improvement of their understanding through being tutored by software showed a significant improvement from one exam to the next.

## References

1. A. Biere, W. Schreiner, M. Seidl, and W. Windsteiger. Logic for Computer Science, 2020. Course in the first year in the bachelor program for computer science at Johannes Kepler University Linz (JKU), taught since 2013.
2. B. Buchberger. Theorema: A Proving System Based on Mathematica. *The Mathematica Journal*, 8(2):247–252, 2001.
3. 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.
4. 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.
5. B. Buchberger, T. Jebelean, T. Kutsia, A. Maletzky, and W. Windsteiger. Theorema 2.0: Computer-Assisted Natural-Style Mathematics. *JFR*, 9(1):149–185, 2016.
6. D. M. Cerna, M. Seidl, W. Schreiner, W. Windsteiger, and A. Biere. Computational Logic in the First Semester of Computer Science: An Experience Report. In Springer, editor, *CSEDU 2020*, pages 1–8, 2020.
7. W. Windsteiger. Theorema 2.0: A Brief Tutorial. In T. Jebelean and D. Zaharie, editors, *Proceedings of SYNASC 2017*, IEEE Explore, pages 1–3, 2017.

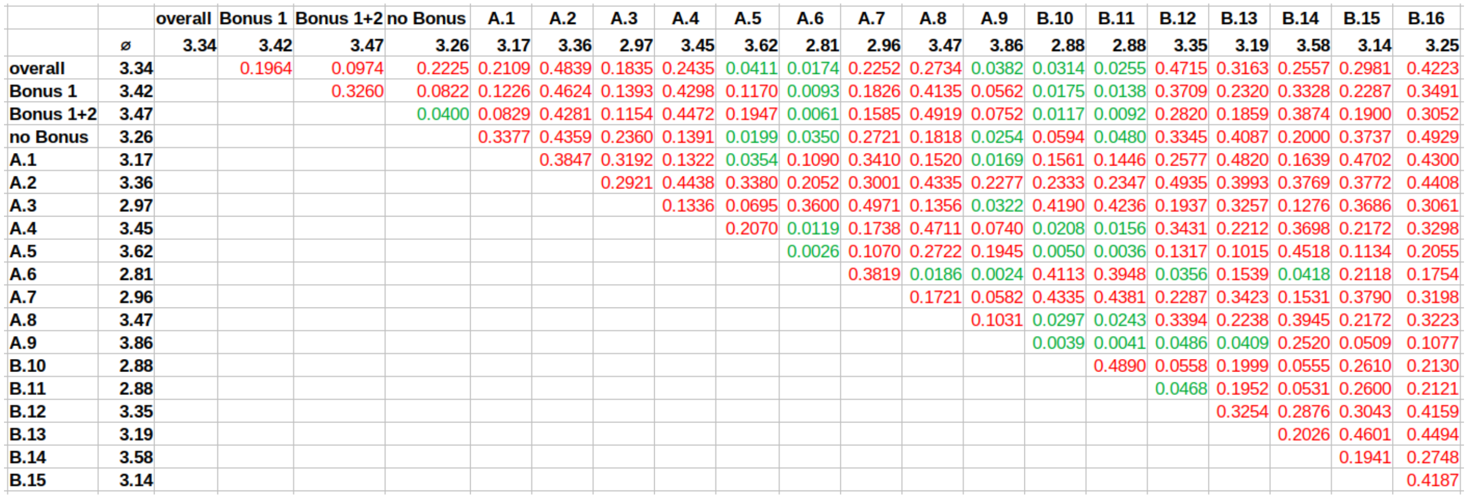
## A Appendix

We include some of the raw data tables and graphical illustrations, on which the results and conclusions in Section 4 are based.



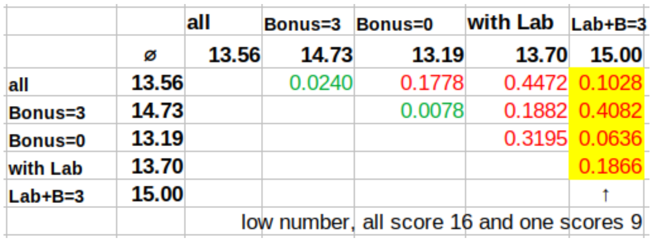
| | | overall | Bonus 1 | Bonus 1+2 | no Bonus | A.1 | A.2 | A.3 | A.4 | A.5 | A.6 | A.7 | A.8 | A.9 | B.10 | B.11 | B.12 | B.13 | B.14 | B.15 | B.16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | ø | 3.28 | 3.62 | | 3.00 | 2.94 | 3.03 | 3.20 | 4.11 | 3.81 | 2.97 | 2.82 | 2.82 | 3.00 | 2.86 | 2.39 | 3.45 | 3.48 | 2.51 | 3.15 | 2.83 |
| overall | 3.28 | | 0.0015 | | 0.0067 | 0.040 | 0.2797 | 0.4322 | 3E-07 | 0.009 | 0.1151 | 0.0837 | 0.0271 | only 1 | 0.0352 | 0.0275 | 0.2933 | 0.1729 | 0.0193 | 0.3971 | 0.0607 |
| Bonus 1 | 3.62 | | | | 1.21E-06 | 0.001 | 0.0994 | 0.1986 | 0.002 | 0.1997 | 0.0102 | 0.0143 | 0.0011 | only 1 | 0.0014 | 0.0075 | 0.282 | 0.2501 | 0.0035 | 0.1858 | 0.0127 |
| no Bonus | 3.00 | | | | | 0.378 | 0.4684 | 0.3413 | 4E-10 | 0.0004 | 0.4532 | 0.2878 | 0.2245 | only 1 | 0.2705 | 0.0819 | 0.0786 | 0.0155 | 0.0808 | 0.3845 | 0.2485 |
| A.1 | 2.94 | | | | | | 0.4149 | 0.304 | 1E-06 | 0.001 | 0.4581 | 0.3683 | 0.3374 | only 1 | 0.386 | 0.1135 | 0.0726 | 0.0225 | 0.1258 | 0.3448 | 0.3508 |
| A.2 | 3.03 | | | | | | | 0.3941 | 0.0203 | 0.0585 | 0.4461 | 0.3354 | 0.3219 | only 1 | 0.3508 | 0.1332 | 0.208 | 0.1711 | 0.1588 | 0.4273 | 0.3285 |
| A.3 | 3.20 | | | | | | | | 0.0496 | 0.1269 | 0.3331 | 0.25 | 0.2355 | only 1 | 0.2569 | 0.1004 | 0.3289 | 0.2955 | 0.1185 | 0.4706 | 0.2404 |
| A.4 | 4.11 | | | | | | | | | 0.117 | 0.0002 | 0.001 | 8E-06 | only 1 | 1E-05 | 0.0012 | 0.0251 | 0.0051 | 0.0003 | 0.0518 | 0.0014 |
| A.5 | 3.81 | | | | | | | | | | 0.0058 | 0.0072 | 0.0009 | only 1 | 0.0011 | 0.0037 | 0.1527 | 0.1182 | 0.0015 | 0.1210 | 0.0039 |
| A.6 | 2.97 | | | | | | | | | | | 0.3493 | 0.3229 | only 1 | 0.3641 | 0.1134 | 0.1075 | 0.0553 | 0.1289 | 0.3722 | 0.3335 |
| A.7 | 2.82 | | | | | | | | | | | | 0.4983 | only 1 | 0.458 | 0.2007 | 0.0738 | 0.0425 | 0.2445 | 0.2848 | 0.4896 |
| A.8 | 2.82 | | | | | | | | | | | | | only 1 | 0.4507 | 0.1775 | 0.0464 | 0.0145 | 0.2136 | 0.2729 | 0.4895 |
| A.9 | 3.00 | | | | | | | | | | | | | only 1 | only 1 | only 1 | only 1 | only 1 | only 1 | only 1 | only 1 |
| B.10 | 2.86 | | | | | | | | | | | | | | | 0.1565 | 0.0554 | 0.0184 | 0.1852 | 0.2954 | 0.4618 |
| B.11 | 2.39 | | | | | | | | | | | | | | | | 0.0224 | 0.0145 | 0.408 | 0.1212 | 0.1740 |
| B.12 | 3.45 | | | | | | | | | | | | | | | | | 0.4637 | 0.0194 | 0.3040 | 0.0533 |
| B.13 | 3.48 | | | | | | | | | | | | | | | | | | 0.0093 | 0.2725 | 0.0254 |
| B.14 | 2.51 | | | | | | | | | | | | | | | | | | | 0.1439 | 0.2093 |
| B.15 | 3.15 | | | | | | | | | | | | | | | | | | | | 0.2780 |

**Fig. 4.** Group comparison for minitest 2. Average score for each group in row/column ∅, A.1–B.16 correspond to the self-assessment groups described in Section 4.3. Values in the matrix are *p*-values of a one-sided T-Test comparing the mean values of the samples corresponding to the respective row and column. The *p*-value gives the maximum probability that the mean values are actually equal. $p \leqslant 0.05$ is displayed green whereas $p > 0.05$ is displayed red, i.e. green means that the respective mean values really differ with a probability of at least 95%. The T-test does not assume identical variances in the samples. Group A.9 is not compared against the others because the sample consists of only one value.



| | | overall | Bonus 1 | Bonus 1+2 | no Bonus | A.1 | A.2 | A.3 | A.4 | A.5 | A.6 | A.7 | A.8 | A.9 | B.10 | B.11 | B.12 | B.13 | B.14 | B.15 | B.16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | ø | 3.34 | 3.42 | 3.47 | 3.26 | 3.17 | 3.36 | 2.97 | 3.45 | 3.62 | 2.81 | 2.96 | 3.47 | 3.86 | 2.88 | 2.88 | 3.35 | 3.19 | 3.58 | 3.14 | 3.25 |
| overall | 3.34 | | 0.1964 | 0.0974 | 0.2225 | 0.2109 | 0.4839 | 0.1835 | 0.2435 | 0.0411 | 0.0174 | 0.2252 | 0.2734 | 0.0382 | 0.0314 | 0.0255 | 0.4715 | 0.3163 | 0.2557 | 0.2981 | 0.4223 |
| Bonus 1 | 3.42 | | | 0.3260 | 0.0822 | 0.1226 | 0.4624 | 0.1393 | 0.4298 | 0.1170 | 0.0093 | 0.1826 | 0.4135 | 0.0562 | 0.0175 | 0.0138 | 0.3709 | 0.2320 | 0.3328 | 0.2287 | 0.3491 |
| Bonus 1+2 | 3.47 | | | | 0.0400 | 0.0829 | 0.4281 | 0.1154 | 0.4472 | 0.1947 | 0.0061 | 0.1585 | 0.4919 | 0.0752 | 0.0117 | 0.0092 | 0.2820 | 0.1859 | 0.3874 | 0.1900 | 0.3052 |
| no Bonus | 3.26 | | | | | 0.3377 | 0.4359 | 0.2360 | 0.1391 | 0.0199 | 0.0350 | 0.2721 | 0.1818 | 0.0254 | 0.0594 | 0.0480 | 0.3345 | 0.4087 | 0.2000 | 0.3737 | 0.4929 |
| A.1 | 3.17 | | | | | | 0.3847 | 0.3192 | 0.1322 | 0.0354 | 0.1090 | 0.3410 | 0.1520 | 0.0169 | 0.1561 | 0.1446 | 0.2577 | 0.4820 | 0.1639 | 0.4702 | 0.4300 |
| A.2 | 3.36 | | | | | | | 0.2921 | 0.4438 | 0.3380 | 0.2052 | 0.3001 | 0.4335 | 0.2277 | 0.2333 | 0.2347 | 0.4935 | 0.3993 | 0.3769 | 0.3772 | 0.4408 |
| A.3 | 2.97 | | | | | | | | 0.1336 | 0.0695 | 0.3600 | 0.4971 | 0.1356 | 0.0322 | 0.4190 | 0.4236 | 0.1937 | 0.3257 | 0.1276 | 0.3686 | 0.3061 |
| A.4 | 3.45 | | | | | | | | | 0.2070 | 0.0119 | 0.1738 | 0.4711 | 0.0740 | 0.0208 | 0.0156 | 0.3431 | 0.2212 | 0.3698 | 0.2172 | 0.3298 |
| A.5 | 3.62 | | | | | | | | | | 0.0026 | 0.1070 | 0.2722 | 0.1945 | 0.0050 | 0.0036 | 0.1317 | 0.1015 | 0.4518 | 0.1134 | 0.2055 |
| A.6 | 2.81 | | | | | | | | | | | 0.3819 | 0.0186 | 0.0024 | 0.4113 | 0.3948 | 0.0356 | 0.1539 | 0.0418 | 0.2118 | 0.1754 |
| A.7 | 2.96 | | | | | | | | | | | | 0.1721 | 0.0582 | 0.4335 | 0.4381 | 0.2287 | 0.3423 | 0.1531 | 0.3790 | 0.3198 |
| A.8 | 3.47 | | | | | | | | | | | | | 0.1031 | 0.0297 | 0.0243 | 0.3394 | 0.2238 | 0.3945 | 0.2172 | 0.3223 |
| A.9 | 3.86 | | | | | | | | | | | | | | 0.0039 | 0.0041 | 0.0486 | 0.0409 | 0.2520 | 0.0509 | 0.1077 |
| B.10 | 2.88 | | | | | | | | | | | | | | | 0.4890 | 0.0558 | 0.1999 | 0.0555 | 0.2610 | 0.2130 |
| B.11 | 2.88 | | | | | | | | | | | | | | | | 0.0468 | 0.1952 | 0.0531 | 0.2600 | 0.2121 |
| B.12 | 3.35 | | | | | | | | | | | | | | | | | 0.3254 | 0.2876 | 0.3043 | 0.4159 |
| B.13 | 3.19 | | | | | | | | | | | | | | | | | | 0.2026 | 0.4601 | 0.4494 |
| B.14 | 3.58 | | | | | | | | | | | | | | | | | | | 0.1941 | 0.2748 |
| B.15 | 3.14 | | | | | | | | | | | | | | | | | | | | 0.4187 |

**Fig. 5.** Group comparison for minitest 3. Average score for each group in row/column ∅, A.1–B.16 correspond to the self-assessment groups described in Section 4.3. Values in the matrix are $p$-values of a one-sided T-Test comparing the mean values of the samples corresponding to the respective row and column. The $p$-value gives the maximum probability that the mean values are actually equal. $p \leqslant 0.05$ is displayed green whereas $p > 0.05$ is displayed red, i.e. green means that the respective mean values really differ with a probability of at least 95%. The T-test does not assume identical variances in the samples.



| | | all | Bonus=3 | Bonus=0 | with Lab | Lab+B=3 |
|---|---|---|---|---|---|---|
| | **ø** | **13.56** | **14.73** | **13.19** | **13.70** | **15.00** |
| **all** | **13.56** | | 0.0240 | 0.1778 | 0.4472 | 0.1028 |
| **Bonus=3** | **14.73** | | | 0.0078 | 0.1882 | 0.4082 |
| **Bonus=0** | **13.19** | | | | 0.3195 | 0.0636 |
| **with Lab** | **13.70** | | | | | 0.1866 |
| **Lab+B=3** | **15.00** | | | | | ↑ |
| | | low number, all score 16 and one scores 9 | | | | |

**Fig. 6.** Group comparison for exam “Discrete Structures”. Average score for each group in row/column ∅. Values in the matrix are $p$-values of a one-sided T-Test comparing the mean values of the samples corresponding to the respective row and column. The $p$-value gives the maximum probability that the mean values are actually equal. $p \leqslant 0.05$ is displayed green whereas $p > 0.05$ is displayed red, i.e. green means that the respective mean values really differ with a probability of at least 95%. The T-test does not assume identical variances in the samples.

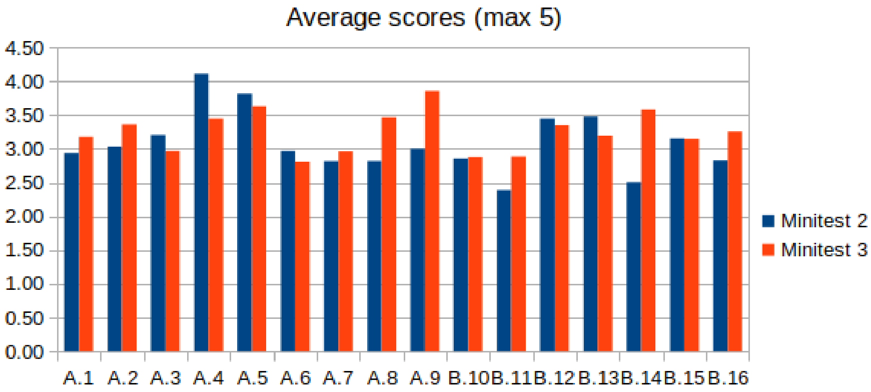
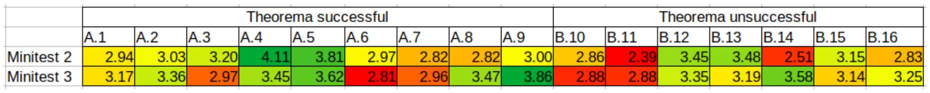
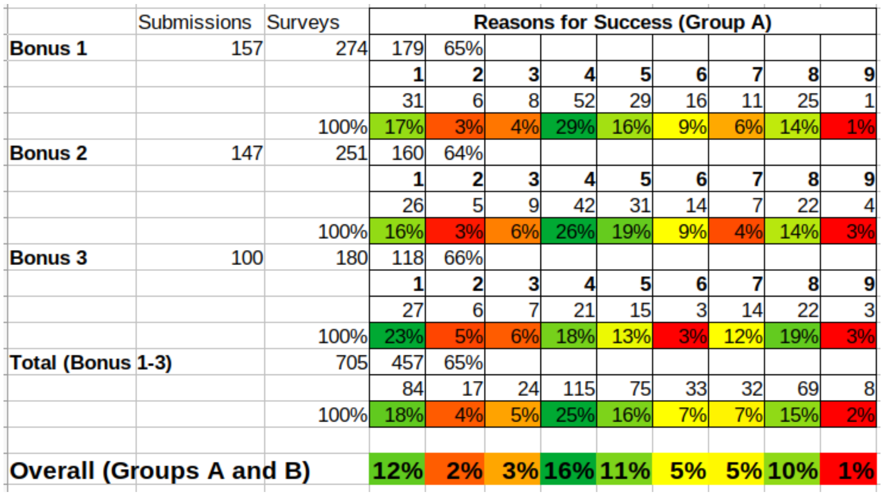


**Fig. 7.** Performance in minitests 2 and 3 (average scores by groups). A.1–B.16 correspond to the self-assessment groups described in Section 4.3. Easy determination of groups with big gain or big loss.



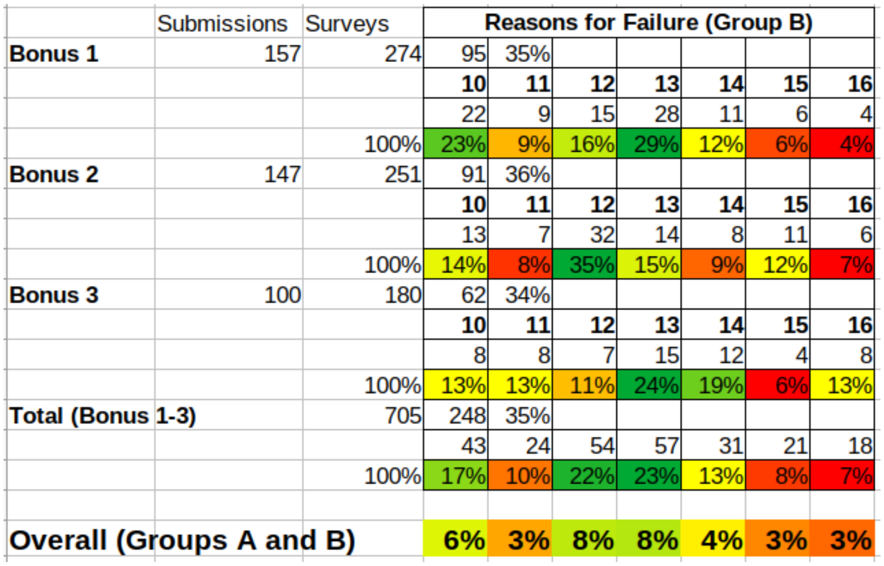
| | Theorema successful | | | | | | | | | Theorema unsuccessful | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | A.1 | A.2 | A.3 | A.4 | A.5 | A.6 | A.7 | A.8 | A.9 | B.10 | B.11 | B.12 | B.13 | B.14 | B.15 | B.16 |
| Minitest 2 | 2.94 | 3.03 | 3.20 | 4.11 | 3.81 | 2.97 | 2.82 | 2.82 | 3.00 | 2.86 | 2.39 | 3.45 | 3.48 | 2.51 | 3.15 | 2.83 |
| Minitest 3 | 3.17 | 3.36 | 2.97 | 3.45 | 3.62 | 2.81 | 2.96 | 3.47 | 3.86 | 2.88 | 2.88 | 3.35 | 3.19 | 3.58 | 3.14 | 3.25 |

**Fig. 8.** Performance in minitests 2 and 3 (average scores by groups). A.1–B.16 correspond to the self-assessment groups described in Section 4.3. Color scales indicate rank (green=rank 1 over yellow to red=rank 16).



| | Submissions | Surveys | Reasons for Success (Group A) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Bonus 1 | 157 | 274 | 179 | 65% | | | | | | | |
| | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| | | | 31 | 6 | 8 | 52 | 29 | 16 | 11 | 25 | 1 |
| | | 100% | 17% | 3% | 4% | 29% | 16% | 9% | 6% | 14% | 1% |
| Bonus 2 | 147 | 251 | 160 | 64% | | | | | | | |
| | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| | | | 26 | 5 | 9 | 42 | 31 | 14 | 7 | 22 | 4 |
| | | 100% | 16% | 3% | 6% | 26% | 19% | 9% | 4% | 14% | 3% |
| Bonus 3 | 100 | 180 | 118 | 66% | | | | | | | |
| | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| | | | 27 | 6 | 7 | 21 | 15 | 3 | 14 | 22 | 3 |
| | | 100% | 23% | 5% | 6% | 18% | 13% | 3% | 12% | 19% | 3% |
| Total (Bonus 1-3) | | 705 | 457 | 65% | | | | | | | |
| | | | 84 | 17 | 24 | 115 | 75 | 33 | 32 | 69 | 8 |
| | | 100% | 18% | 4% | 5% | 25% | 16% | 7% | 7% | 15% | 2% |
| | | | | | | | | | | | |
| Overall (Groups A and B) | | | 12% | 2% | 3% | 16% | 11% | 5% | 5% | 10% | 1% |

**Fig. 9.** Development of the group sizes (absolute and relative) in self-assessment from bonus 1 to bonus 3 in Group A. Color scales indicate rank (green=biggest group over yellow to red=smallest group). Scales span over Group A, except in row "Overall", where scale spans over Groups A and B.



| | Submissions | Surveys | Reasons for Failure (Group B) | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Bonus 1 | 157 | 274 | 95 | 35% | | | | | |
| | | | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| | | | 22 | 9 | 15 | 28 | 11 | 6 | 4 |
| | | 100% | 23% | 9% | 16% | 29% | 12% | 6% | 4% |
| Bonus 2 | 147 | 251 | 91 | 36% | | | | | |
| | | | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| | | | 13 | 7 | 32 | 14 | 8 | 11 | 6 |
| | | 100% | 14% | 8% | 35% | 15% | 9% | 12% | 7% |
| Bonus 3 | 100 | 180 | 62 | 34% | | | | | |
| | | | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| | | | 8 | 8 | 7 | 15 | 12 | 4 | 8 |
| | | 100% | 13% | 13% | 11% | 24% | 19% | 6% | 13% |
| Total (Bonus 1-3) | | 705 | 248 | 35% | | | | | |
| | | | 43 | 24 | 54 | 57 | 31 | 21 | 18 |
| | | 100% | 17% | 10% | 22% | 23% | 13% | 8% | 7% |
| | | | | | | | | | |
| Overall (Groups A and B) | | | 6% | 3% | 8% | 8% | 4% | 3% | 3% |

**Fig. 10.** Development of the group sizes (absolute and relative) in self-assessment from bonus 1 to bonus 3 in Group B. Color scales indicate rank (green=biggest group over yellow to red=smallest group). Scales span over Group B, except in row "Overall", where scale spans over Groups A and B.

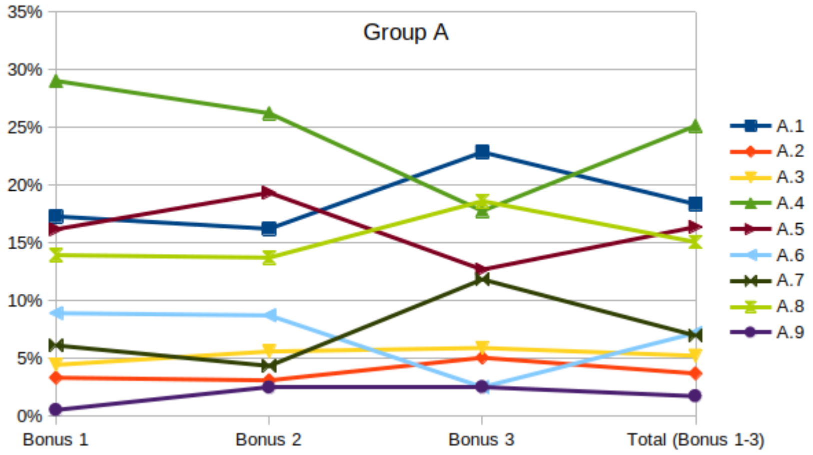


**Fig. 11.** Development of the group sizes (relative) in self-assessment from bonus 1 to bonus 3 in Group A. Easy determination of groups with big gain or big loss. A.1–A.9 correspond to the self-assessment groups described in Section 4.3.

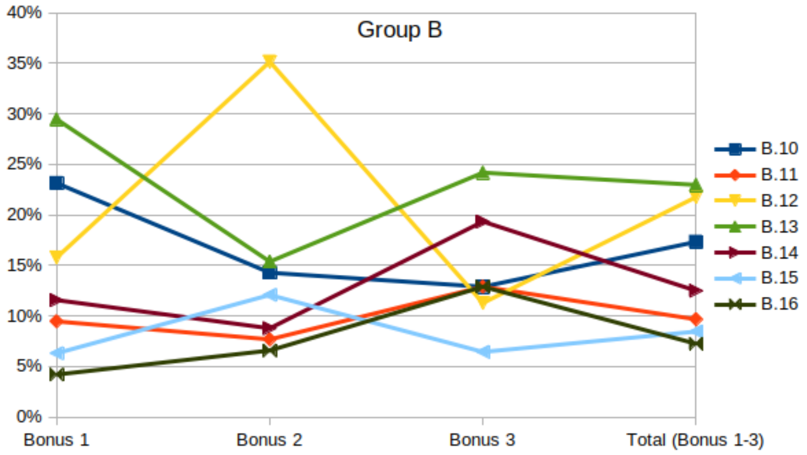


**Fig. 12.** Development of the group sizes (relative) in self-assessment from bonus 1 to bonus 3 in Group B. Easy determination of groups with big gain or big loss. B.10–B.16 correspond to the self-assessment groups described in Section 4.3.

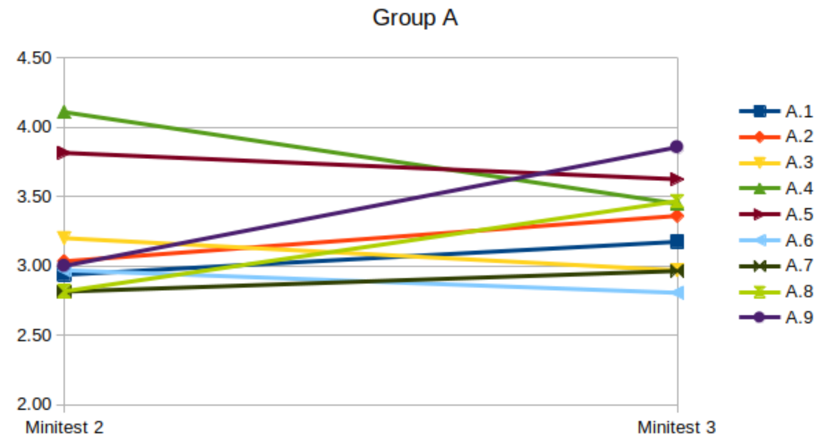


**Fig. 13.** Development of the minitest performance in self-assessment Group A. Easy determination of groups with big gain or big loss. A.1–A.9 correspond to the self-assessment groups described in Section 4.3.

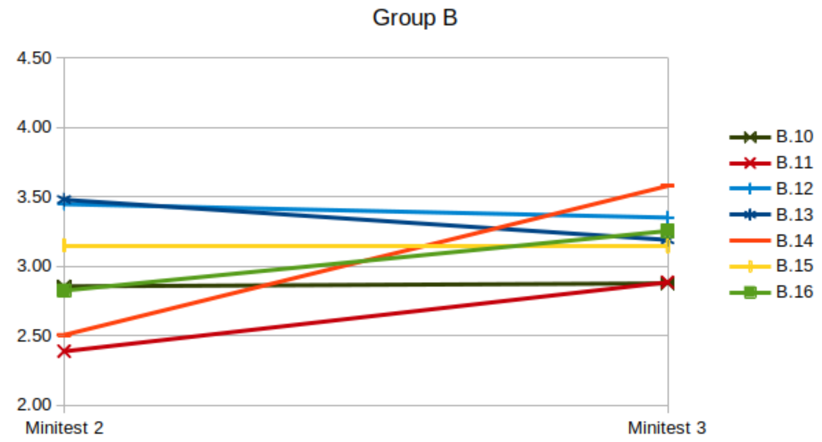


**Fig. 14.** Development of the minitest performance in self-assessment Group B. Easy determination of groups with big gain or big loss. B.10–B.16 correspond to the self-assessment groups described in Section 4.3.