2

Use of a proof assistant as a learning tool in an introductory logic course for computer science undergraduates

Holger Bock Axelsen

Department of Computer Science, University of Copenhagen

Introduction

The present report documents a teaching experiment undertaken in an introductory logics course for computer science undergraduates, where I introduced an interactive computer program, a so-called *proof assistant*, as a tool for learning.

As part of a larger revision effort to improve on low student retention and moderate pass rates in the course, I allowed the students to use the *ProofWeb*[1] proof assistant for their solutions of exercises in formal deduction proofs. In the exam, student performance in this area improved considerably compared to prior years, and also in comparison to students who did not use of the proof assistant. However, feedback on the student experience suggest that this improvement comes at the cost of a somewhat steep learning curve and increased workload. A few students even reported that they found the proof assistant to be a hindrance to understanding the formal systems it was intended help with. Still, most students ultimately found the tool useful and adopted its use. Even though it is difficult to separate this from the other revision instruments, the results suggest that a proof assistant is effective as a learning tool in this course, but that it needs to be supported by additional teaching and learning activities to improve efficiency.

[1] An online version of ProofWeb is available at http://prover.cs.ru.nl.

Background

Course information

"Logik i datalogi" (Engl. Logic in Computer Science, abbrv. LiCS) is an elective 7.5 ECTS advanced undergraduate course at the Department of Computer Science at the University of Copenhagen. LiCS is offered once a year in block 1 (early autumn), introduces the students to formal logics of various kinds, and develops this in directions generally interesting to computer scientists, i.e. decision procedures, algorithms, data structures, and model checking for the verification of software, cf. the course description[2]. The 60 students who enrolled in the course in 2013 were overwhelmingly 3rd year Bachelor's students with the department.

The students are required to submit homework sets for 6 weeks running, and 5 of these have to be approved (by teaching assistants) to qualify for the exam. The course has historically suffered from low retention and moderate pass rates: from 2008 through 2012, about 55% of the enrolled students have qualified for and attended the exam, and of these about 70% passed[3]. This gives an aggregate of less than 40% of the enrolled students who end up passing[4]. In course evaluations students consistently report that they find the academic level and work load of the course high.

I became course responsible in early 2012, and have taught the subsequent two editions of the course. In an internal evaluation of the 2012 course run I identified a number of problem points in the course structure that may have contributed to student dropout and exam failure rates. For the 2013 edition of the course I therefore instigated a number of changes to realign the course elements, in keeping with the principles of constructive alignment (Biggs & Tang 2011), while keeping the syllabus and intended learning outcomes essentially unchanged.

To keep this report reasonable in scope, I shall focus on one particular aspect of the revision effort, namely the use of the ProofWeb proof assistant as a learning tool. The most notable of the remaining revisions are structural: a different exam format (a 27h take-home assignment rather than a 4h sit-in exam), and the use of exercise classes (mostly lab work) to supplement the lecture-based teaching sessions.

[2] Available at http://kurser.ku.dk/course/ndab05005u/2013-2014.

[3] In 2008 the course adopted the structure kept up to and including the 2012 edition with respect to exam format and qualification requirements.

[4] Not including reexams, as this data was not readily available.

Problem Statement

A particular recurring problem not directly addressed by the mentioned structural changes is the abilities of the students in constructing *formal proofs*. A major part of the first half of the course is devoted to the *natural deduction* system for constructing such proofs. The students are expected to achieve proficiency in deriving natural deduction proofs, as partial demonstration of the following intended learning outcomes (ILOs) of the course (English translation mine):

- *Knowledge of:* the definition of logics in terms of syntax, semantics, and natural deduction systems, and of how one formally reasons about logical formulas.

- *Skills in:* deciding and proving formal properties of logical formulas (e.g. satisfiability, validity, implication and equivalence) both using semantic arguments, and by natural deduction.

Formal deductions play a key role in LiCS. The idea of a formal proof is the very first novel concept introduced to the students in the course, and much of the first half of the course directly builds on the students' understanding of natural deduction and how it is used to construct proofs.

Previous teacher course evaluations and student performance in previous exams suggest that such proficiency may serve a 'gate keeper' function in the course. A student that fails to demonstrate reasonable proficiency with natural deduction in the exam will likely also fail to demonstrate adequate achievement of the other course ILOs, and is thus likely to fail.

Thus, although formal proofs only make up a limited part of a typical LiCS exam set (about 10–20%), an experimental effort aimed at early widespread achievement of this learning goal appeared reasonable. The hoped-for effect is that achieving this proficiency early 'leaks' into achieving other ILOs, and that this in turn would improve retention and exam performance. For the 2013 edition of the course, I therefore chose to allow the students to use ProofWeb to solve exercises in constructing natural deduction proofs. The didactical underpinnings for this is explored below, but the informal reasoning is this: a proof assistant can error check a given formal proof reliably, instantly, and while the student builds the proof.

This leads to the following problem statement: *Is the ProofWeb proof assistant an effective and efficient tool for learning in LiCS?*

Theoretical considerations

Problem analysis.
Prior to the revision, the course supported achieving proficiency with formal proofs only to a weaker extent. The required training was 3 relegated almost exclusively to the weekly homework sets, i.e., to learning activities with assessment. This appears to be a failure point[5].

Although intended as formative, the homework sets in LiCS serve a summative assessment purpose also, in that qualifying for the exam requires 5 of 6 sets are approved. This kind of setup has been linked to students ignoring the formative aspects and focussing exclusively on the summative aspects (Gibbs & Simpson 2004). Although this may well be in effect in LiCS, it is likely not the only effect in play. In fact, it appears that for the low-achieving learners the ILO in question is probably not well served by the existing assessment format at all, in particular with respect to feedback[6].

To understand why, one first needs to appreciate the unforgiving nature of formal logic. One does not aim for merely ‘morally’ correct proofs, but absolutely correct ones: a formal logic proof is always unambiguously correct or not, and even subjectively tiny errors technically invalidate the entire proof. Now, for the average LiCS student this course is their first encounter with formal logic, and they are furthermore usually not proficient in the considerably more lax notion of an ordinary mathematical proof. This means that constructing a formal proof is (certainly initially) a *difficult* task for them, and that they have low confidence in their hand-generated solutions. Multiple revision cycles of hand-written proofs are usually necessary to get *everything* right, and the students are strongly reliant on feedback for this.

Formative feedback is an incredibly complex issue with much conflicting research. However, some trends can be identified at the task level, *cf.* the review article by (Shute 2008), which I have used as a framework (and useful reference list) for the theoretical analysis.

For the kind of task at hand, the existing literature, in guise of the handy guidelines in (Shute 2008, Tables 3-5), suggests that the feedback format

[5] One response to the problems with achieving proficiency with formal proofs has been to increase the number of deduction exercises in the homework sets, which has not been particularly effective.

[6] For this reason the revision effort also included the design of new teaching and learning activities (TLAs) to support this ILO, but describing them in detail is beyond the scope of this report.

employed in LiCS is flawed: For *difficult* tasks, the feedback should (at least initially) be *immediate*, as it should for *low-achieving learners* in general. But, the course structure does not allow for this: only one revision of a homework set is usually possible, and feedback is *delayed*, not immediate. Furthermore, feedback for these exercises is usually mildly elaborated error flagging (of *low-to-mid complexity*), but presented *in bulk* for all errors identified in a given proof, rather than in *manageable units*. An additional problem in this context is that much of such bulk feedback may even be irrelevant to obtaining a correct solution to a particular exercise, as it can pertain to a line of reasoning which will no longer be visited if an earlier error is corrected.

These properties of the formative feedback as hitherto offered may all impede learning, or serve to promote surface learning over deep learning.

How a proof assistant can help

Luckily, there is help to be had: formal systems are sufficiently rigid and mechanical that errors in formal proofs can be identified purely syntactically, something which computers are especially good at. The use of computers to assist with and verify mathematical proofs is an old idea, going at least as far back as the AUTOMATH programming language in the 1960's. Tools of this kind have long been used in the teaching of formal logic, and a bewildering array are available. These range from very minimalistic non-interactive proof checkers to elaborate highly interactive e-tutor systems. For a more elaborate explanation of which kind of tools are available, see (Huertas 2011).

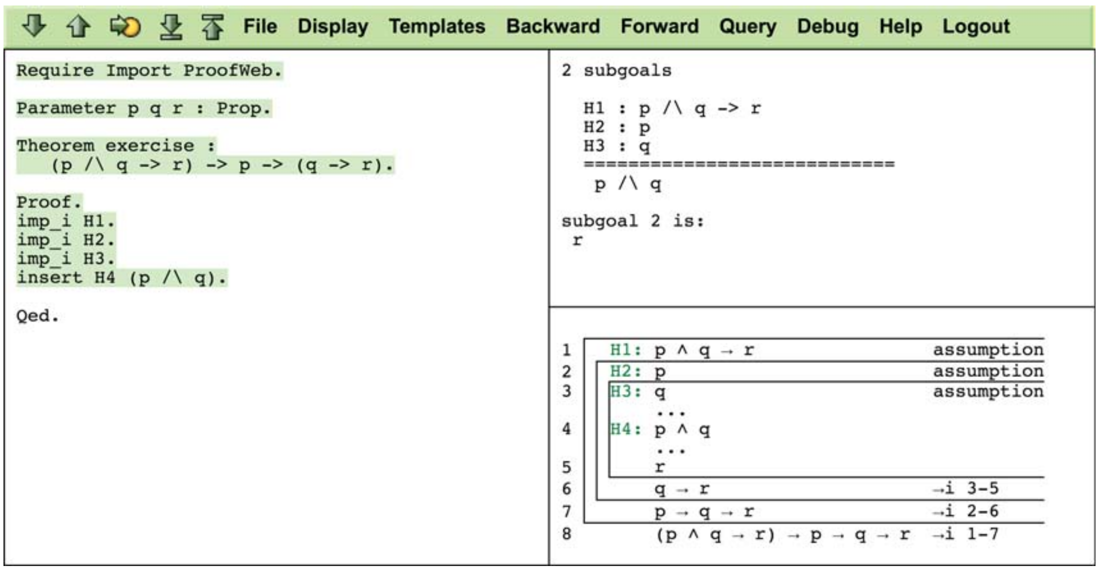
Among the diversity of tools are so-called *proof assistants*. A proof assistant is an interactive computer program for building and mechanically verifying mathematical proofs. In particular, proof assistants provide (corrective) feedback *while* constructing a proof, rather than 'just' checking an already completed proof. It does so (usually) via error messages explaining why a particular line in the proof is unacceptable, and refuses to progress further until this is remedied. While this could be interpreted as *feedback intrusion*, which can impede learning (Kluger & DeNisi 1996), this type of restrictive *answer-until-correct* tutor control has also been linked to more efficient learning (Corbett & Anderson 2001).

In particular, a proof assistant is able to provide *manageable units* of *immediate, corrective* formative feedback, which has been linked to enhanced learning (especially for low-achieving learners) in computer-based instruction, *cf*. the review by (Mason & Bruning 2001). Proof assistants also tick more of the right boxes from (Shute 2008, Tables 3-5): They provide *unbiased, objective* feedback with focus on the *task, not the learner*, and immediate feedback has been linked to *immediate gains* and *more efficient* learning, which is an intended effect. Finally, (Nipkow 2012) suggests that proof assistants provide 'gamification' of theorem proving, which should enhance student motivation.

The choice of weapon for LiCS was *ProofWeb*, developed at Radboud University in the Netherlands. This was a conscious choice to limit the *impeding* effects on learning that a proof assistant might have. For instance, there is the risk that students substitute learning the *tool* for the intended learning of the *logic*. This risk comes from the fact that the student interacts with the proof assistant via short lines of code (so-called *tactics*), and the proof assistant responds to these with either an error message or by updating the view of the proof state presented to the user. (Figure 2.1 shows the ProofWeb interface.) In contrast, a logic is 'just' a set of *rules*, out of which one can build proofs, and there may or may not be a close correspondence between interaction with the proof assistant and proofs in the logic. There is thus in general the possibility of attaining proficiency in one, without the other. However, ProofWeb was *explicitly* developed to support teaching the formal logics of the textbook used in LiCS, *cf.* (hendriks et al. 2010). In particular, the textbook proofs and ProofWeb's rendering of proofs are diagrammatically almost identical. Furthermore, there is a nearly one-to-one correspondence between ProofWeb tactics and the rules of the natural deduction system, in that each tactic explicitly specifies which rule and assumptions are used to justify each line in the diagrammatic proof.

A separate risk is that the inherent *answer-until-correct* format may lead the students to not only use the proof assistant for *scaffolding*, but actually *abuse the feedback* to brute-force their way to solutions, which would constitute surface learning (Aleven & Koedinger 2002). However, the facilitative aspect of ProofWeb's feedback does not appear strong enough for this (except in the case of extremely short proofs.) When accepting a tactic as correct, ProofWeb guarantees merely that this individual step corresponds to the legal use of a rule in the logic. However, there is no evaluation of, or feedback given on, whether this tactic is a viable way towards a complete proof: indeed, proof *strategy* requires, and has, its own additional TLAs in the course[7].

[7] ProofWeb use may help here as well, by pruning false strategies relying on erroneous rule use.



```
Require Import ProofWeb.

Parameter p q r : Prop.

Theorem exercise :
   (p /\ q -> r) -> p -> (q -> r).

Proof.
imp_i H1.
imp_i H2.
imp_i H3.
insert H4 (p /\ q).

Qed.
```

```
2 subgoals

  H1 : p /\ q -> r
  H2 : p
  H3 : q
  ============================
   p /\ q

subgoal 2 is:
 r
```

```
1    H1: p ∧ q → r                     assumption
2    H2: p                             assumption
3    H3: q                             assumption
         ...
4    H4: p ∧ q
         ...
5        r
6        q → r                         →i 3-5
7        p → q → r                     →i 2-6
8        (p ∧ q → r) → p → q → r       →i 1-7
```

Fig. 2.1. The ProofWeb interface. The left window is where the student writes tactics; the upper right window is a representation of the proof state in terms of which things have to be proven (the subgoals), and under which hypotheses; and the lower right window shows a rendering of the proof so far, in the same format as the course textbook.

The above risks are predicated on the students becoming proficient with the tool, but this in itself is almost certainly hard: using a proof assistant effectively constitutes having to learn a new programming language (and so does learning a logic.) Students will very likely initially struggle with *both* the logic itself *and* with expressing it in ProofWeb. Although the student' *background* is expected to be helpful, as computer science students at the department are exposed to a variety of widely different programming languages through their programme, the risk of *cognitive overload* appears high in this context, and should be addressed in the implementation.

The conclusion is that although using a proof assistant is expected to enhance learning, it is not just a substitute for human assessment, and must be supported by teaching and learning activities to offset the complication overhead. As a final remark, we note that the use of a proof assistant serves to align the course more with the surrounding educational programme, which emphasizes the use of computers and computational methods in the programme learning outcomes[8].

[8] See the study programme at http://www.science.ku.dk/ studerende/ studieordninger/bachelor/datalogi/Sto_datalogi_2009.pdf.

Methodology

Course modification

Incorporation of ProofWeb into the course was implemented as follows.

- *Exercise solutions:* From week 2 and onwards the students were given the option to submit either a hand-written proof or a proof made via ProofWeb (consisting of a proof script and a screenshot of the resulting proof) for their solutions to exercises demanding formal proofs. Note that ProofWeb use was not made mandatory. This policy was in effect both in the homework assignments and the exam.

- *Exercise classes:* 3 of the 4 hours of exercise classes in weeks 2 and 3 were used to introduce ProofWeb to the students and have them work with it in class, under supervision by the teachers and TAs.

- *Lectures:* In the first week of the course ProofWeb was *not* mentioned, to allow the student to familiarize themselves with the concepts of formal systems and natural deduction for propositional logic separately. In week 2, after introducing ProofWeb in the exercise classes, a follow-up 20 min. lecture on ProofWeb use was conducted. In week 3, in the lecture introducing the second major deduction system (predicate logic), the additional ProofWeb rules required for this were covered.

- *Course materials:* Supplementary notes on ProofWeb use were (somewhat hastily) produced by the teachers, following a number of students reporting that they found the official documentation confusing.

Data collection

To evaluate the effect and efficiency of using this proof assistant as a learning tool in LiCS, and of our teaching of it, I collected data from the following sources. Note that this data was collected with the intent of performing a soft analysis only, and that the implementation was not set up to facilitate statistical testing.

Exam solutions
The exam had 4 subquestions asking the students to do 6 formal deduction

proofs (approx. 15% weight of the total exam.) How many students reached the exam, and how many passed? What was the grade distribution? How did the students perform on the exam parts with formal deduction? How many students used ProofWeb, and did this correlate to performance in any way?

Questionnaire on ProofWeb
In the final (summary) lecture of the course students were asked to fill out a 20 min. questionnaire on their use and experiences with ProofWeb. This was done in class to maximize response. Students who did not attend this lecture were asked to fill out the questionnaire electronically. The questionnaire form (in Danish) is shown in Appendix A, and contains roughly four sections as follows.

- Demographics data, including prior exposure to logic and proof assistants. (Q1, Q28–Q30)
- Questions regarding the extent of the student's use of ProofWeb over the course period, self-evaluated proficiency, and current usage type. (Q2–Q5)
- Student experience measured by level of agreement with predefined statements on a scale from 1 to 5. Statements were formulated to provide insight into the students' *attitude* towards using ProofWeb, *practical experiences* with it, their *trust* in the system, whether they believe it *aided their understanding*, their experience of our *teaching* of the system, and quality of *documentation materials*.

  In the development of this part I attempted to emulate the 'Course Evaluation Questionnaire' (see (Wilson et al. 1997, Appendix, p. 53)) which has 8 a long history of development and validation (McInnis et al. 2001). Each statement is (hopefully) unambiguous and strong, and the statements are unsorted. This was done to limit the risk of misunderstanding, strengthen response interpretation, and to encourage focus and reflection on each individual statement from the students. (Q6–Q25)
- Student-formulated comments on the use of ProofWeb. By placing these questions *after* the multiple choice part above, I hoped to have forced them to already reflect on the many aspects involved in ProofWeb use. The late placement runs the risk of response fatigue, so to encourage response I asked for deliberately structured feedback (name 3 good

and 3 bad things, mark the 2 most important of these), as well as free-form comments. (Q26, Q27)

Course evaluation
Comments from the generic final course evaluation questionnaire. In addition to this, I had informal discussions with students runningly, although interviews were not formally conducted.

Results

Exam performance

42 out of 45 qualified students handed in solutions to the exam, which means that 75% of the 60 students enrolled at course beginning qualified for the exam. 80% of the qualified students passed, for an aggregate 60% pass rate for all enrolled students. With 3 abstentions, the pass rate rises to 86% for those who submitted a solution to the exam.

The students largely adopted the use of ProofWeb, with 76% using ProofWeb wholly or partially for their formal proofs in the exam solutions. 25 students used ProofWeb exclusively, 7 students used a mixture of both ProofWeb and handwritten proofs, and 10 students did not use ProofWeb at all and had only handwritten proofs. Figure 2.2 shows the grades (for the entire exam) distinguished by the type of solution to the formal deduction parts.

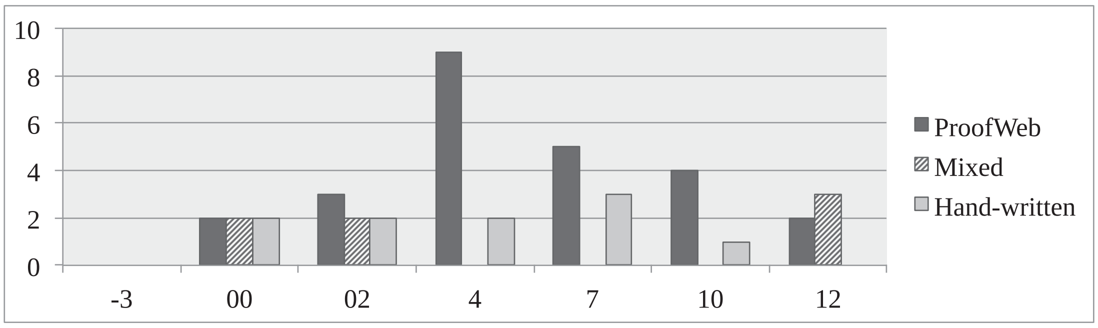


Fig. 2.2. Grades for the 2013 exam distinguished by ProofWeb use.

By their nature as verified proofs, submitting a ProofWeb script as a solution meant getting that question correct. (Of course, not all students submitted solutions to all exercises.) However, for those students who submitted at least some hand-written solutions performance was very diverse, the only constant being that no such student escaped making at least one formal error (however slight). In fact, the (subjective) performance on the formal proofs for these groups closely resemble the grade distribution: the students who did not use ProofWeb at all are fairly uniformly distributed (but taper off at the highest grades), and the mixed students are bimodal.

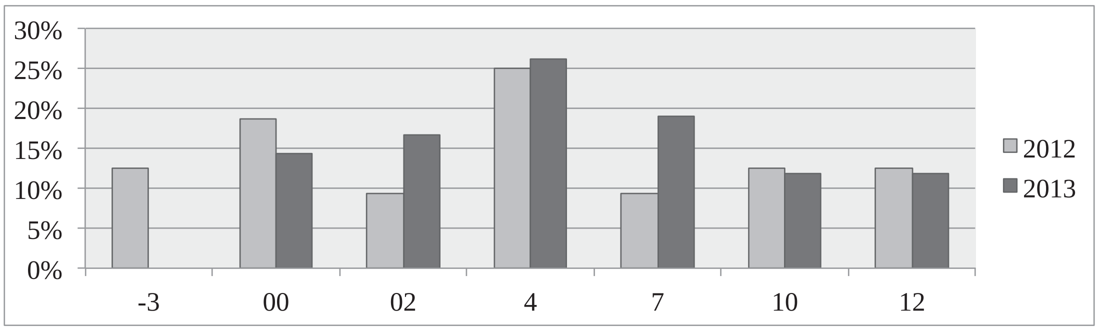


Fig. 2.3. Grade distributions for the 2012 and 2013 editions of the course

While the grade distribution in 2013 compared to 2012 version (see Figure 2.3) suggests that the course revision effort may have been successful in aiding a portion of the weaker students (who moved up the grade scale) the above suggests that adopting ProofWeb in this course run may not have had a strong impact on performance (although only one student managed to reach a high grade without using it.) However, another interpretation could be that a number of students use ProofWeb as scaffolding, and attain higher grades thereby, since these students generally performed better in the formal proofs than the students not using ProofWeb. Either way, the exam data does not appear strong enough for definitive conclusions regarding ProofWeb use. The only certainty is that both the student retention and pass rate rose markedly compared to previous years, and that performance in formal proofs improved considerably, supporting that ProofWeb was at least effective in supporting attainment of the ILOs in question.

For the failed students, we find one item of note: showing enough proficiency to demonstrate weak attainment of the ILOs in question does not entail passing the course. This is reasonable: that failure to achieve these ILOs implies failing the course does not entail that achieving these ILOs implies passing the course.

Questionnaires

I received 28 responses (17 on paper, 11 electronic) to the ProofWeb questionnaire.

The demographics held no surprises. The students were almost all 3rd year computer science undergraduates, with the exception of a few mathematics undergraduates. For all except a single student this was their first exposure to formal logic in a university course, and this single student was the only one to report prior experience with a proof assistant.

Slightly more interesting are the self-reported data on ProofWeb practice. The questionnaire data may all be slightly skewed towards the non-ProofWeb users, as these were represented in the questionnaire data to a greater extent than in the exam. As expected, current exclusive (or near-exclusive) ProofWeb use correlates with heavy use through the course, and with the level of self-reported proficiency. Interestingly, the *gamification* effect posited by (Nipkow 2012) does not appear not to hold true for LiCS: while students ended up largely adopting the system, there was no universal agreement (or even polarity of responses) as to whether it was fun, and only very few students ended up proving significantly more theorems in the system than required by the exercises and homework sets. This is counter to the expectation if gamification had been a significant effect.

Aggregate data for Q6–Q25 and Q26 is shown in Appendix B. The responses show a number of interesting points, not least that I made some design mistakes with the questionnaire: at least one student (and likely more) certainly swapped the polarity of the Likert scale in Q6–Q25. Additionally, by Q26 there is evidence of some response fatigue, in that many responses do not match the asked-for format. This is corroborated by the near complete lack of responses to the free-form comment asked for in Q27. Still, the student responses show the following trends.

- Very strong trust in the correctness of ProofWeb proofs (Q6), and only moderately strong trust in paper proofs (Q13).
- Very strong preference to working with the logics on paper before attempting them in ProofWeb (Q25). Moderately strong preference towards more teaching with ProofWeb, in particular exercise classes (Q11, Q12).
- A small preference to declaring ProofWeb use fun (Q10), but not enough to spark a general interest in proof assistants (Q14) or rework proofs for optimisation (Q16). A slightly stronger preference to perse-

vere when frustrated (Q9), although there was no strong trend of frustration (Q17).
- Weak documentation. Although students studied the material closely (Q21) they cannot rely on it for help (Q13).
- Bimodal response to verification of handwritten proofs (Q18), although the students do not believe that this often located errors (Q22).
- Strong preference for the use of a proof assistant in the course (Q23). This is particularly interesting since the students do not strongly believe it aids their understanding (Q8, Q24) or is faster to work with (Q20), even (mildly) disagreeing that it helps them understand the deduction rules (Q19).

These points were largely repeated in the responses to Q26, where the following trends can be seen:

- Strong appreciation of the (instant) verification and corrective aspects.
- Strong dissatisfaction with certain aspects of the ProofWeb interface, *in particular* the quality of the front-end and the error messages.
- Many students report that ProofWeb has a steep learning curve, and that the introduction to ProofWeb use was lacking.
- Appreciation of the systematic computer science approach to formal proofs offered by ProofWeb.
- Positive aspects were more frequently marked as important than *negative* aspects.

Finally, very few students remarked on ProofWeb in the generic course evaluation questionnaire, instead focussing on workload, in particular the the size of the homework assignments (which they, consistent with previous years, found too large and demanding.) Taken together, these responses suggest that students by and large accepted ProofWeb as an integral part of the course.

The general picture offered by the questionnaire responses corresponds to my impressions from informal discussions. Despite a number of misgivings, the students mostly appreciated the use of a proof assistant, and found it useful. However, workload remained high, suggesting that the efficiency bought by automating assessment of the proofs may have been (at least partially) subsumed by the added overhead it introduced. The interface problems and perceived low quality of the error messages are worrisome, as appropriate feedback was a central motivation for the introduction of a proof assistant in the first place. The reported steep learning curve is also alarming, as I aimed for early efficient gains by allowing a proof assistant.

Conclusion

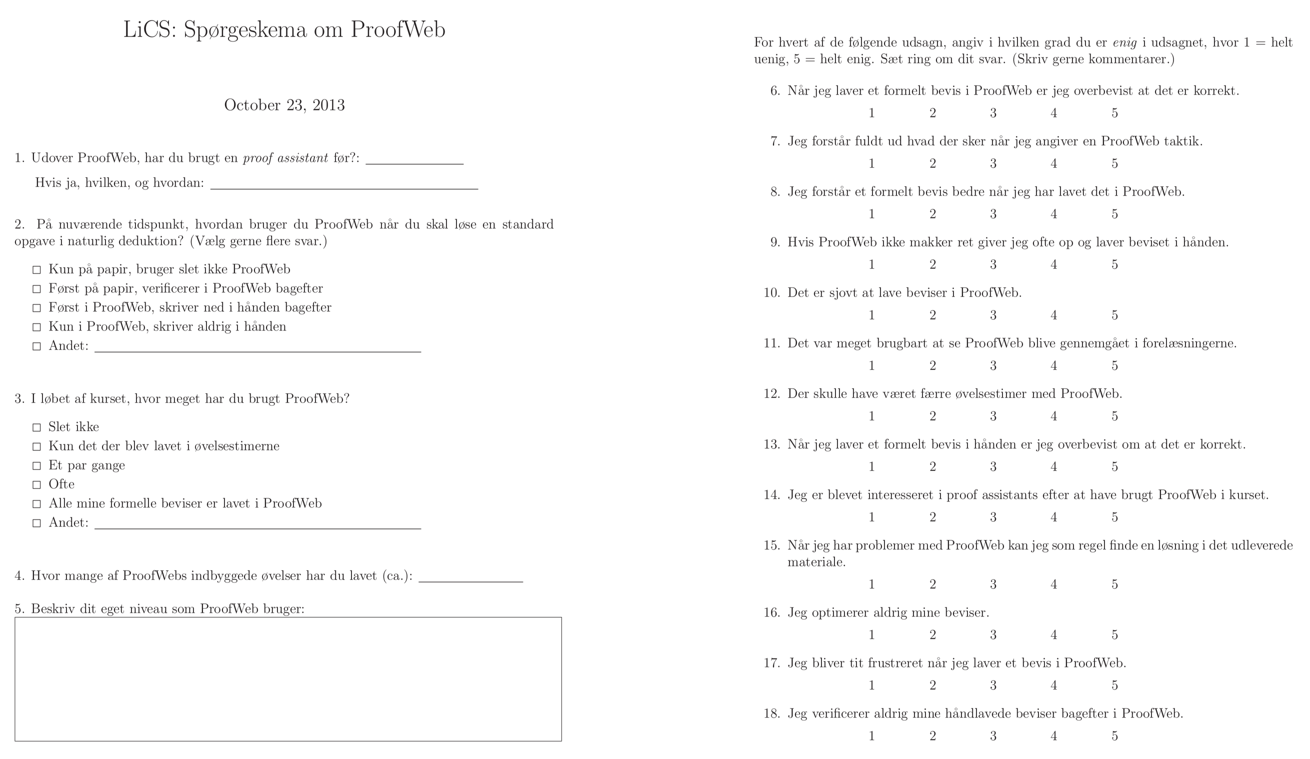
Student performance in the exam with respect to formal proofs was good, especially for students who adopted the use of ProofWeb. Combined with the other revisions, the course saw a significant rise in retention and pass rates compared to previous years. However, many students were frustrated with the ProofWeb interface and documentation, and would additionally have liked a better introduction to the tool. Still, its adoption was widespread, and the students did appreciate the strengths of such a system (if not necessarily this one in particular.)

Although frustration is to be expected when learning a new tool, I agree that the teaching implementation can probably be significantly improved. In particular, rather than relying mostly on exercise classes, more lecture time should be devoted to the system. One possibility is to integrate ProofWeb more tightly in the introductory lectures. However, given that the students widely appreciated having worked with propositional logic before the use of ProofWeb, and reported that this went too quickly with predicate logic, separation is advisable. On the other hand, because mastering the tool in itself was not the primary objective, teaching a more *restricted* use of the tool than done for this experiment is also worth considering (and might avoid having to pushing other lecture content out.) For instance, one might teach its use as a proof checker for existing (hand-written) proofs *only*, by teaching a particular fixed methodology for converting hand-written proofs to ProofWeb scripts, and leave more sophisticated use up to the individual student. Then again, many students reported that the approach to proof building offered by the *interaction* with ProofWeb was positive, and a number even professed greater proving abilities *inside* ProofWeb than outside. In either case, particular care should be taken to ensure that the students understand and can act on the feedback that ProofWeb provides.

In answer to the problem statement, I believe that, yes, ProofWeb is *effective* as a learning tool for the Logic in Computer Science course, but that the *efficiency* with the current teaching implementation is more questionable. The investment on part of the students was considerably higher than I estimated it would be, making it less valuable for my purposes than expected. In spite of this, I believe that a proof assistant is a useful and relevant addition to the course. ProofWeb may work well in this role, given adequate teaching support to improve efficiency, but this requires additional and/or redesigned teaching and learning activities. Additionally, even though the intended learning outcomes were kept unchanged through this experiment, proof assistant use can be incorporated into future intended learning outcomes directly, strengthening the alignment between the intended learning outcomes of the course and those of the surrounding study programme.

A Questionnaire

The following 2 pages shows the paper version of the ProofWeb questionnaire. An electronic version was created as an electronic survey on Absalon (the digital course platform used by the University of Copenhagen.)



LiCS: Spørgeskema om ProofWeb

October 23, 2013

1. Udover ProofWeb, har du brugt en *proof assistant* før?: __________

   Hvis ja, hvilken, og hvordan: ____________________

2. På nuværende tidspunkt, hvordan bruger du ProofWeb når du skal løse en standard opgave i naturlig deduktion? (Vælg gerne flere svar.)

   ☐ Kun på papir, bruger slet ikke ProofWeb
   ☐ Først på papir, verificerer i ProofWeb bagefter
   ☐ Først i ProofWeb, skriver ned i hånden bagefter
   ☐ Kun i ProofWeb, skriver aldrig i hånden
   ☐ Andet: ____________________

3. I løbet af kurset, hvor meget har du brugt ProofWeb?

   ☐ Slet ikke
   ☐ Kun det der blev lavet i øvelsestimerne
   ☐ Et par gange
   ☐ Ofte
   ☐ Alle mine formelle beviser er lavet i ProofWeb
   ☐ Andet: ____________________

4. Hvor mange af ProofWebs indbyggede øvelser har du lavet (ca.): __________

5. Beskriv dit eget niveau som ProofWeb bruger:

For hvert af de følgende udsagn, angiv i hvilken grad du er *enig* i udsagnet, hvor 1 = helt uenig, 5 = helt enig. Sæt ring om dit svar. (Skriv gerne kommentarer.)

6. Når jeg laver et formelt bevis i ProofWeb er jeg overbevist at det er korrekt.
   1 2 3 4 5
7. Jeg forstår fuldt ud hvad der sker når jeg angiver en ProofWeb taktik.
   1 2 3 4 5
8. Jeg forstår et formelt bevis bedre når jeg har lavet det i ProofWeb.
   1 2 3 4 5
9. Hvis ProofWeb ikke makker ret giver jeg ofte op og laver beviset i hånden.
   1 2 3 4 5
10. Det er sjovt at lave beviser i ProofWeb.
   1 2 3 4 5
11. Det var meget brugbart at se ProofWeb blive gennemgået i forelæsningerne.
   1 2 3 4 5
12. Der skulle have været færre øvelsestimer med ProofWeb.
   1 2 3 4 5
13. Når jeg laver et formelt bevis i hånden er jeg overbevist om at det er korrekt.
   1 2 3 4 5
14. Jeg er blevet interesseret i proof assistants efter at have brugt ProofWeb i kurset.
   1 2 3 4 5
15. Når jeg har problemer med ProofWeb kan jeg som regel finde en løsning i det udleverede materiale.
   1 2 3 4 5
16. Jeg optimerer aldrig mine beviser.
   1 2 3 4 5
17. Jeg bliver tit frustreret når jeg laver et bevis i ProofWeb.
   1 2 3 4 5
18. Jeg verificerer aldrig mine håndlavede beviser bagefter i ProofWeb.
   1 2 3 4 5

19. ProofWeb har hjulpet mig med at forstå hvordan deduktionreglerne virker.

    1 2 3 4 5

20. Det tager meget kortere tid at lave et bevis i ProofWeb end i hånden.

    1 2 3 4 5

21. Jeg har ikke studeret det udleverede materiale om ProofWeb særligt grundigt.

    1 2 3 4 5

22. Jeg finder ofte fejl i mine håndlavede beviser hvis jeg checker dem i ProofWeb.

    1 2 3 4 5

23. Jeg ville foretrække at der slet ikke var brugt en proof assistant i kurset.

    1 2 3 4 5

24. ProofWeb hjælper mig med at forstå hvis min bevis-idé er forkert.

    1 2 3 4 5

25. Det var rart at have arbejdet med deduktionsreglerne på papir før de blev introduceret i ProofWeb.

    1 2 3 4 5

26. Beskriv 3 dårlige, og 3 gode ting ved brugen af ProofWeb i kurset.
Af dine 6 punkter markér de 2 vigtigste med $*$:

Dårligt:	Godt:

27. Yderligere kommetarer til brugen af ProofWeb i kurset:

28. Du er:
- ☐ bachelorstuderende i: ____________________

  på __________ år.
- ☐ kandidatstuderende i: ____________________,

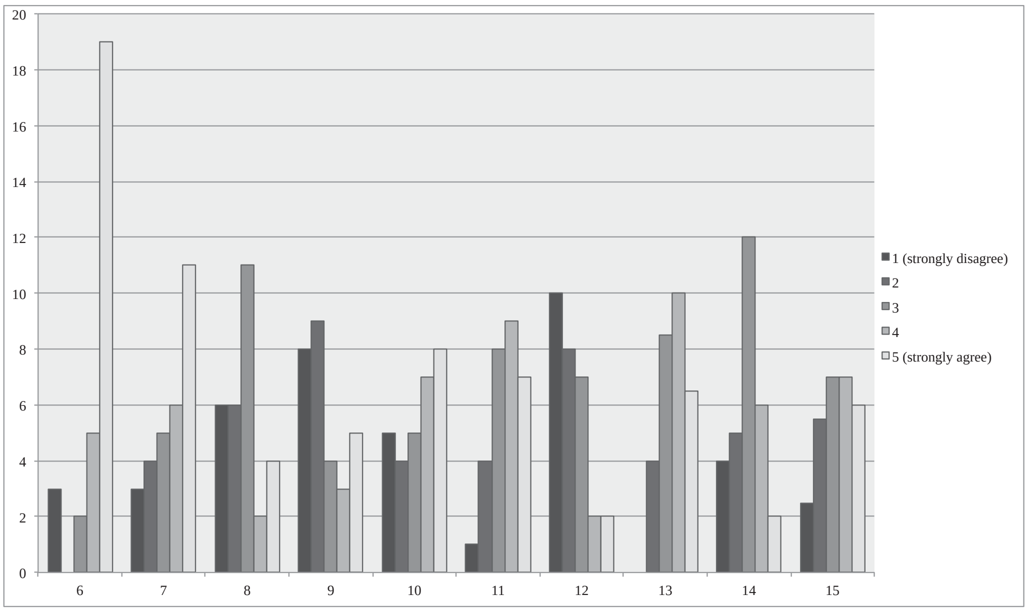
  og din bachelorgrad er i: ____________________

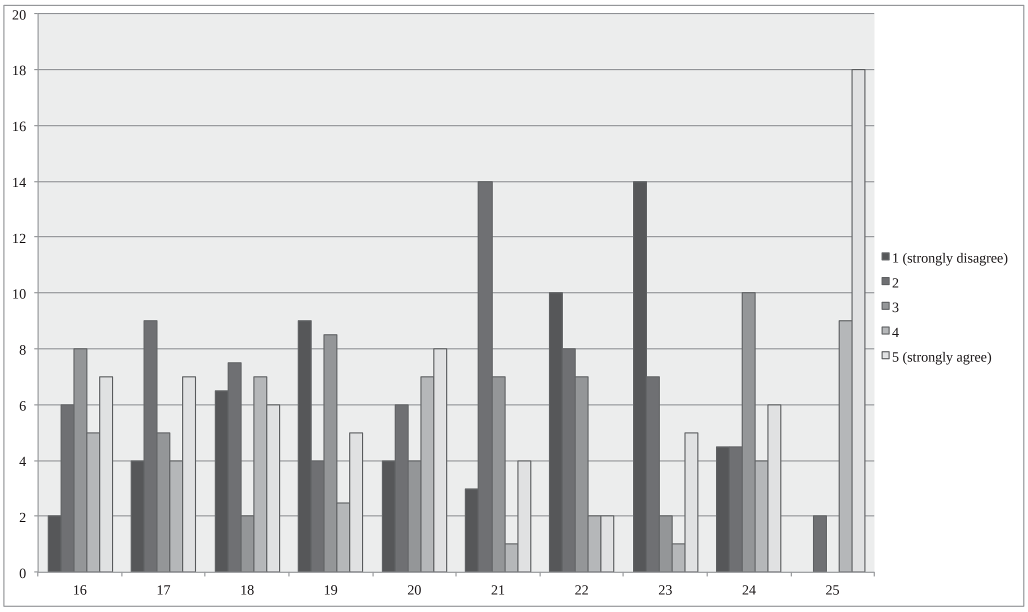
29. Har du haft et kursus i logik før?
- ☐ Ja. Hvilket, hvorhenne: ____________________
- ☐ Nej.

30. Hvilke kurser om *programmering* har du haft?

B Selected questionnaire data

Aggregate responses to questions Q6 through Q25.





The following page shows the responses to Q26 (in Danish.)

Q26 (negative)	Q26 (positive)
intet, proofweb er godt.	*1. subgoals. *2. backwad pænere grafisk
1. Giver ikke så god vejledning hvis man bruger en taktik forkert 2. I Chrome kan linjerne "hoppe" lidt rundt 3. Hmm...	*1. Hurtigere end i hånden *2. Sikker på det er rigtigt 3. Hjælper med den formelle opskrivning af regler
1. Merkelig oppførsel i PWs text-editor 2. Noen (få) bugs	*1. Lettere å strukturere bevisene 2. Bruker kortere tid *3. Bekreftelse om korrekt bevis
*1. Der bør være en bedre måde, at sætte beviserne i rapporterne end screenshots.	1. Uoverskuelige beviser bliver lettere at lave. 2. Det gør det lidt sjovere at lave beviser.
1. Virker umiddelbart indviklet *2. Dårlig dokumentation *3. For stor arbejdsbelastning, når man "bare" skal lære det sideløbende med ugeafleveringer, øvelser og læsning.	*1. Når man har lært det er det godt arbejdsværktøj,. *2. Giver systematisk tilgang til bevisførsel. 3. Printer et godt formateret bevis til slut.
1. Svært at komme i gang. 2. Lidt forvirret symboler taktics sammenlignet med bogen.	1. Hurtig 2. Hjælpsom
1. Syntax var nogle gange irriterende fx. manglende parenteser. *2. Fejlmeddelelserne var ikke særlig overbvisende 3. Fejl i prædikat-logik.	*1. Det har gjort kurset mere sjovt, samtidig med at løse deduktionsbeviser. *2. Hurtig og verificerende. 3. Det har gjort kurset mere praktisk og computerfagligt. 4. Indbyggede eksempler var gode, det hjalp mig til at forstå hvordan man gjorde det (brugte ProofWeb)
*1. ProofWeb er ikke særligt intuitivt, heller ikke selvom man har bred erfaring med programmering. 2. Dokumentation er ringe. 3. Editoren er dårlig og kræver at man bruger Firefox.	*1. Man kan være sikker på at beviset er korrekt. Så man ryger ikke til genaflevering på trivielle ting. 2. ProofWeb er hurtigere end at lave sit bevis manuelt. 3. Godt at der blev brugt meget tid på at forklare hvordan man bruger PW. Dokumentationen er ikke tilstrækkelig.
*1. Dårlig platform som giver mange frustrationer 2.Tilbagemeldingerne er ofte intetsigende 3.Interfacet er ekstremt dårligt og svært at arbejde med 4. Når man laver en fejl mister man overbliksvinduets indhold. 5. Mange lange linier er svære at se og overskue.	1. Let at se mål/delmål og se hvornår man er færdig *2. Masser små opgaver at lave og se på
*1. Dårlig dok. *2. Opdeling af vinduerne i browseren (ville gerne kunne tilpasse frames) 3. Online ved dårligt net	*1. Skudsikkert når PW har sagt OK. *2. Viser når man laver "ulovlige" ting.
1. Introduktionen var lidt rodet som gjorde læringskurven lidt stejl.	*1. Har gjort det nemmere at lave beviser
1. Læringskurve (men ikke svært)	*1. Verificering *2. Hastighed 3. Kunne arbejde på samme beviser på forskellige pc'er
1. Det er buggy og selvom man installerer den "rigtige" browser er de samme bugs der *2. Indlæringskurven er for stejl - kurset er hårdt nok i forvejen *3. Det er forvirrende, man skal løse ting baglæns	? (Hvis man kan finde ud af det kan man tjekke korrekthed)
1. Det kan være svært at forstå fejlmeldinger i ProofWeb. 2. Nogle gange ved jeg hvordan problemet skal løses, men kan have svært ved at få sat det op, da ProofWeb kræver en bestemt rækkefølge for at man har de korrekt mål og antagelser for at kunne bruge en taktik.	1. Når man får noget træning og har set opgaver og fejlmeldinger kan man ofte gætte hvad problemet er 2. Det er nemmere end når jeg skriver i LaTeX
1. Forvirring ved forlæns og baglæns taktik 2. Mere besværligt end at skriv i hånden 3. Grimt interface og dårlige brugeregenskaber	1. Sikkerhed for ingen syntaxfejl (måske) 2. Datalogisk indgang til deduktion. 3. Hjælper folk til nemmere at dumpe kurset
1. Svært at vende sig til "backwards tactics" 2. Fejlmeddelelser er ikke altid tilstrækkeligt oplysende.	1. Forhøjer arbejdsglæden ved at løse beviser 2. Hjælper til at forsikre én om, at man har forstået reglerne korrekt. 3. Mulighed for at arbejde sig beglæns igennem et bevis.
1. Det er ikke brugervenligt *2. Det er ikke intuitivt. 3. Editoren virker dårligt.	1. Hvis man får et rigtigt svar er beviset korrekt. 2. Hvis man kan gennemskue det er det sikker godt. 3. Hvis det gav et bedre førstehåndsindtryk ville man nok ikke være så kritisk ved det.
*1. Det virker ikke i andet end Firefox. 2. Notationen giver ikke altid mening. 3. Alting skal gøres med INSERT BLAH	*1. Man kan blive ved med at prøve til det lykkes. 2. Når det lykkes er man sikker på, at det er rigtig. 3. Det aflaster instruktorerne = hurtigere feedback
1. Dokumentationen er elending, man sidder og gætter sig frem ved trial-and-error. 2. Formatet og understøttelsen er ringe (kun én browser virkede det korrekt i), og ved tabt internet forbindelse, eller ingen internet forbindelse (f. eks. i toget på vej hjem fra eller til instituttet) kan løsningen ikke benyttes — hvilket er meget skidt, fordi det er dér jeg laver mange af mine lektier og øvelser. Løsningen skal helst være mulig at benytte offline også. 3. Man brugte mere (spild-)tid på at lære, at få ProofWeb til at makke ret, end på at løse opgaverne, hvilket faktisk havde en negativ effekt på både forståelsen af hvad man lavede, samt demotiverede en fra at fortsætte med at lave lektier — resulterede ofte i at tage en pause af ren frustration over at det ikke virkede.	1. Jeg synes godt om idéen bag det, men ProofWeb lever ikke op til den nødvendige brugervenlighed (mht. dokumentation, syntaks, etc.) 2. Det var fedt at få noget visuel feedback på, hvad man lavede (herunder bevis-boksene nederst til højre). 3. Det var reassuring at få at vide, at hvis ProofWeb sagde god for et bevis, så var det også korrekt — bare ærgeligt at jeg ikke nåede at få lært at bruge det ordenligt.
*1. Besværligt at lære/vende sig til (mest forvirring mellem forwards/backwards)	*1.Bekræftelse på validitet af beviser
1. Front-enden til ProofWeb er elendig: grim, bugget og kun brugbar i Firefox. Generelt et lavt brugervenligheds niveau. 1½. Fejlbeskederne er næsten ubrugelige. Tit ofte kryptiske, andre gange forkerte. Det er meget svært at vide om man har lavet en syntax eller sematisk fejl, og/eller en taktisk fejl. 2. En god dokumentation samt eksempler er ikke eksisterende. Selv den udleverede hjælp er ikke så udtømmende som ønskeligt er. 3. Introduktionen til ProofWeb var meget forvirrende. Fair nok - man lærer programmering bedst ved selv at lege med det, men en langsom og præcis introduktion til konceptet havde hjulpet.	1. Mulighed for at verificere ens løsninger 2. At det ikke er nødvendigt at bruge lang tid på at sætte et pænt bevis op i LaTeX, men i stedet bare kunne kopiere proofweb-taktikker. 3. Mulighed for at "brute-force" et bevis: forsøge sig med kvalificierede gæt på taktiskemuligheder, uden helt at vide hvor det fører hen. Dette kan tage uendeligt lang tid i hånden.
1. Introduktionen til ProofWeb gik alt for hurtigt, og mange fangede det ikke første gang. 2. Interfacet opførte sig underligt til tider. Det var irriterende at skulle skifte browser (Fra Chrome til Firefox) for at funktionaliteten var tilstedeværende. *3. Den officielle dokumentation var ikke særlig god.	1. Smart at kunne tjekke om ens bevisstrategi var lovlig.* 2. Man sparede meget tid ved ikke selv at skulle skrive sit bevis ind i TeX, eller andre skriveprogrammel. 3. At underviserne var hurtige til selv at udgive dokumentation og guides til brugen af ProofWeb. De fleste på kurset ville måske have været mere positive over for brugen af ProofWeb, hvis disse guides var tilgængelige, så snart ProofWeb kom ind i undervisningen.
Proofweb er ret buggy Det er ret svært at komme i gang med, fordi måden det bliver gjort på ikke nødvendigvis giver mening	Det gjorde det mere sikkert at ens beviser var korrekte* Det gjorde det hurtigere at løse opgaver Det var nemmere at strukturere ens beviser helt korrekt*
1. De fitch style proofboxes man lavede kunne vokse vildt store, og det var ikke altid der stod det helt rigtige i beskrivelsen af hvilken deduktion der var brugt.	1. Sikkerheden i at beviser er korrekte * 2. Nemheden i at tilrette eventuelle fejl *
- Lidt besværligt at bruge (kender ikke alternativerne) -* Blev brugt for tidligt til predicate logic (blev ikke selv god nok til papir)	+* Digitalt arbejde med beviser + Verificering af beviser + Introduktion til proof assistance
* Det tager tid at lære og var svært indtil man kom ind i tankegangen	- * Det er rart at få tjekket sine svar. - Det går hurtigt, når man har lært det - Muligheden for at aflevere proofWeb udskrift i stedet for fx latex - jeg har brugt lang tid på at skrive ind i latex...
*Jeg gider ikke webinterface. Hvad med Proof General i stedet eller som alternativ?	*Det hjælper sikkert. Jeg tog mig aldrig sammen til at bruge det.

All contributions to this volume can be found at:

http://www.ind.ku.dk/publikationer/up_projekter/2014-7/

The bibliography can be found at:

http://www.ind.ku.dk/publikationer/up_projekter/kapitler/2014_vol7_nr1-2_bibliography.pdf/