Natural Deduction Assistant (NaDeA)

Jørgen Villadsen Andreas Halkjær From

Anders Schlichtkrull

DTU Compute - Department of Applied Mathematics and Computer Science,
Technical University of Denmark, Richard Petersens Plads, Building 324, DK-2800 Kongens Lyngby, Denmark

We present the Natural Deduction Assistant (NaDeA) and discuss its advantages and disadvantages as a tool for teaching logic. NaDeA is available online and is based on a formalization of natural deduction in the Isabelle proof assistant. We first provide concise formulations of the main formalization results. We then elaborate on the prerequisites for NaDeA, in particular we describe a formalization in Isabelle of "Hilbert's Axioms" that we use as a starting point in our bachelor course on mathematical logic. We discuss a recent evaluation of NaDeA and also give an overview of the exercises in NaDeA.

1 Introduction

The Natural Deduction Assistant (NaDeA) [7, 8] runs in a standard browser:

`https://nadea.compute.dtu.dk/`

NaDeA is written in the TypeScript programming language and is open source (MIT License):

`https://github.com/logic-tools/nadea`

NaDeA is based on a formalization of natural deduction in the Isabelle proof assistant and the overall aim of the present paper is to discuss its advantages and disadvantages as a tool for teaching logic.

Our formalization in the Isabelle proof assistant [6] of the syntax, semantics and the inductive definition of the natural deduction proof system extends work by Berghofer [1] but with a much more detailed soundness proof that can be examined and tested by the students. The corresponding completeness proof is also available but it is of course quite demanding for a student to understand. NaDeA can be used with or without installing Isabelle and it is not necessary that the students have any knowledge about proof assistants [3].

The present paper extends our previous publications about NaDeA [7, 8]. In section 2 we provide more concise formulations of the main formalization results and in sections 3 and 4 we describe a sample proof in NaDeA and list selected features for students. In section 5 we elaborate on the prerequisites for NaDeA and we discuss a recent evaluation of NaDeA in section 6. In section 7 we give an overview of the exercises in NaDeA and finally we conclude in section 8.

Pedro Quaresma, Walther Neuper (Eds.): Theorem proving components for Educational software (ThEdu'18)
EPTCS 290, 2019, pp. 14–29, doi:10.4204/EPTCS.290.2

```
section ‹Main Result› — ‹NaDeA is sound and complete›

abbreviation ‹valid p ≡ ∀(e :: nat ⇒ nat) f g. semantics e f g p›

theorem main: ‹valid p ⟷ OK p []›
proof
 assume ‹valid p›
 with completeness show ‹OK p []›
  using denumerable_bij by blast
next
 assume ‹OK p []›
 with soundness show ‹valid p›
  by (intro allI)
qed

corollary ‹valid p ⟶ semantics e f g p›
proof
 assume ‹valid p›
 then have ‹OK p []›
  unfolding main .
 with soundness show ‹semantics e f g p› .
qed

theorem any_unis: ‹OK (put_unis k p) [] ⟹ OK (put_unis m p) []›
 using main ex_closure put_unis_collapse remove_unis_sentence valid_put_unis by metis

corollary ‹OK p [] ⟹ OK (put_unis m p) []› ‹OK (put_unis m p) [] ⟹ OK p []›
 using any_unis put_unis.simps(1) by metis+
```

Figure 1: Main Formalization Results

2 Main Formalization Results

NaDeA is based on a formalization of natural deduction in the Isabelle proof assistant. Figure 1 shows the main results which are more concise formulations of the formalization results discussed in our previous publications [7, 8].

We define the validity of a formula as the formula evaluating to true in all variable denotations (e), function denotations (f) and predicate denotations (g) with the natural numbers as universe. This is different from the usual notion of validity which considers all universes — not only that of the natural numbers. We can, however, prove in Isabelle that our notion of validity implies the truth of a formula in any variable denotation, function denotation and predicate denotation — and thus the formula must indeed be valid with respect to the usual notion of validity.

We prove that the valid formulas are exactly the same as those which can be proved using the prov-

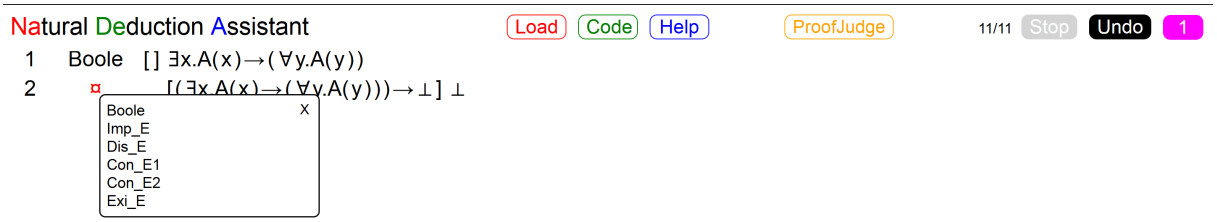

Figure 2: A Sample Proof – Start

ability predicate OK, hence the natural deduction proof system is sound and complete (theorem main in Figure 1). We prefer the word OK to longer words like "provable" or symbols like $\vdash$ mainly because it is easier to pronounce in class. The complete Isabelle theory file with proofs of soundness and completeness is checked by Isabelle/HOL in a few seconds. And from this theory loaded into Isabelle's front-end a mouse-click, for instance on OK, leads to the respective definition — an invitation to investigate the underlying mechanized mathematics deeper and deeper. The more than 4000 lines are available here:

`https://nadea.compute.dtu.dk/Natural_Deduction_Assistant.thy`

The theorem any-unis is a result of having completeness for open formulas and states that given a proof with some number k of outer universal quantifiers, a proof with m quantifiers, either fewer or more, can be derived. The theorem any-unis follows from main as well as the lemmas ex_closure, put_unis_collapse, remove_unis_sentence and valid_put_unis. When a formula has been proved in NaDeA a small Isabelle theory file is generated that verifies the validity of the original formula as well as the validity of all versions of it with some number of outer universal quantifiers omitted. The theorem any-unis is used by this generated Isabelle theory file.

3 A Sample Proof

We consider the following formula and its online proof:

$$\exists x.A(x) \to (\forall x.A(x))$$

The proof in NaDeA is obtained by clicking Cancel help, Load, Test 9 and Load shown proof. The formula is the so-called drinker paradox:

> There is someone in the pub such that, if that person is drinking, then everyone in the pub is drinking.

It was popularized by Raymond Smullyan:

`https://en.wikipedia.org/wiki/Drinker_paradox`

Figure 2 shows the start of the proof — more or less — where a natural deduction rule is to be chosen in the proof step 2. This state can be obtained from the previous one by clicking Undo repeatedly – one can always click on Undo to go back all the way to the very first proof step 1.

Figure 3 shows the finished proof which can be reached again by first clicking Stop and then Undo repeatedly.

A formula [] p in line 1 corresponds to the expression OK p [] in the previous section and the names of the natural deduction rules are the same as used in the formalization in Isabelle.

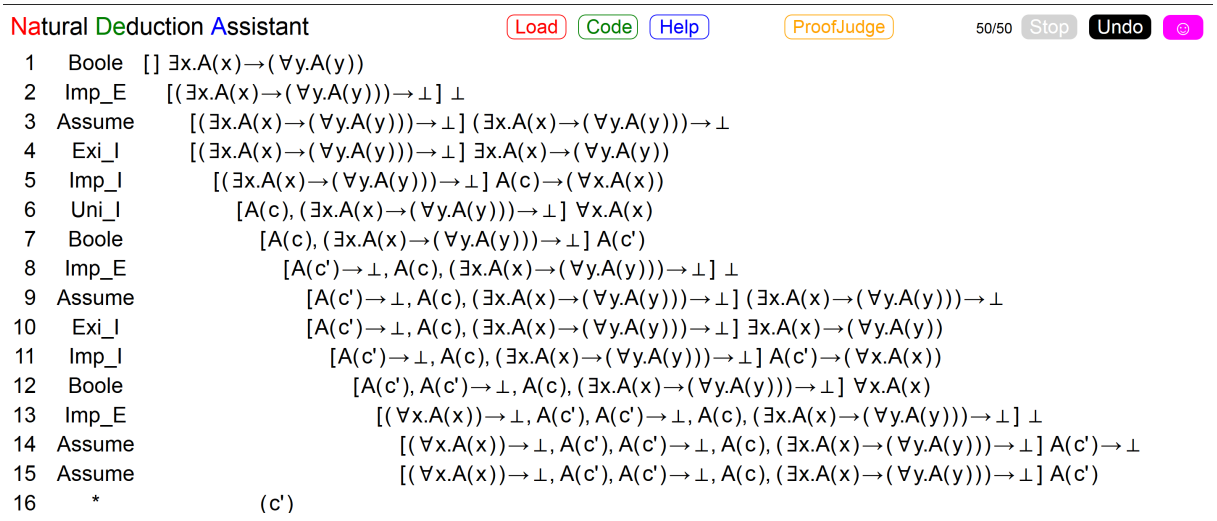

Figure 3: A Sample Proof – Finish

4 Selected Features for Students

We briefly describe a number of NaDeA features for students:

- Figure 4 shows the Welcome window. The Help button brings up the help window with this welcome information and a number of so-called hints.
- Figure 5 shows the Tutorial window. It contains a getting started guide as well as a list of the natural deduction primitives.
- Figure 6 shows the Exercises window. Solutions to all exercises are provided and can be revealed step-by-step with hints.
- Figure 7 shows the special NaDeA, soundness and completeness, window. The so-called verification button allows the user to verify any finished proof in Isabelle.
- Figure 8 shows the major Isabelle Code window – entitled: Definition of natural deduction proof system – with the formalization in Isabelle.
- Figure 9 shows the minor Isabelle Code window – entitled: Definition of first-order logic syntax and semantics – with the formalization in Isabelle.

There are several other NaDeA features for students – for example, the ProofJudge system in NaDeA can manage student assignments in courses with teaching assistants.

NaDeA uses the automation of our verified declarative prover tool [4] to give students feedback on the provability of their goals and subgoals. Overall both that prover and NaDeA are related to the IsaFoL project which unites researchers in formalizing logic in Isabelle:

`https://bitbucket.org/isafol`

Among the formalizations in the project are SAT-solving, first-order resolution, a paraconsistent logic, sequent calculi and more.

5 Student Prerequisites for NaDeA

We use NaDeA in a bachelor course with the following prerequisites:

- Discrete mathematics (basic set theory)
- Algorithms and data structures (searching and sorting)
- Programming in a functional programming language

We use 5 lectures (each 45 minutes) plus exercise sessions with teaching assistants:

1. Warm-up — Truth Tables & Isabelle Introduction
2. Axiomatics — Propositional Logic
3. Natural Deduction — Mainly Propositional Logic
4. Natural Deduction — First-Order Logic
5. Cool-down — Summary & Higher-Order Logic Introduction

For the axiomatics we use a formalization in Isabelle of "Hilbert Axioms" based on David Hilbert's *Die Grundlagen der Mathematik* 1928 and used in Alonzo Church's *Introduction to Mathematical Logic* 1956, cf. [2] page 163 (the system P1 also borrows from Gottlob Frege 1879, John von Neumann 1927 and in particular Mordchaj Wajsberg 1939).

It is a good exercise for the students to enter and verify the axioms in Isabelle:

```
theorem
  ‹A ⟶ B ⟶ A›
  ‹(A ⟶ B ⟶ C) ⟶ (A ⟶ B) ⟶ A ⟶ C›
  ‹(A ⟶ C) ⟶ (B ⟶ C) ⟶ A ∨ B ⟶ C›
  ‹A ⟶ A ∨ B›
  ‹B ⟶ A ∨ B›
  ‹A ∧ B ⟶ A›
  ‹A ∧ B ⟶ B›
  ‹A ⟶ B ⟶ A ∧ B›
  ‹((A ⟶ False) ⟶ False) ⟶ A›
  by simp_all
```

Most students find it straightforward to enter the formulas in Isabelle. In the above Isabelle proof we have used the Isabelle's simplifier to prove the 9 axioms (simp_all) but in general a proof method like Isabelle's classical tableau prover (blast) is required (for example, the simplifier does not succeed for $(A \to B \to \mathit{False}) \to B \to A \to \mathit{False}$). Our reason for using the simplifier is that it is sufficient for the proofs to follow.

The following formalization of the axiomatics is based on "Propositional Proof Systems" by Julius Michaelis and Tobias Nipkow (Archive of Formal Proofs 2017) [5].

First the syntax and the semantics is defined and a small lemma is proved ($A \rightarrow A$):

```
datatype form =
  Falsity | Pro string | Imp form form | Dis form form | Con form form


primrec semantics :: ‹(string ⇒ bool) ⇒ form ⇒ bool› where
  ‹semantics _ Falsity = False› |
  ‹semantics i (Pro s) = i s› |
  ‹semantics i (Imp p q) = (if semantics i p then semantics i q else True)› |
  ‹semantics i (Dis p q) = (if semantics i p then True else semantics i q)› |
  ‹semantics i (Con p q) = (if semantics i p then semantics i q else False)›


lemma ‹semantics i (Imp p p)›
  by simp
```

The datatype form defines the set of formulas. The primitive recursive function semantics takes an interpretation and a formula. We have chosen to use if-then-else expressions for the binary operators like in NaDeA. We then define the provability predicate OK and easily prove soundness of the axiomatics (including the *modus ponens* rule).

```
inductive OK :: ‹form ⇒ bool› where
  ‹OK (Imp p (Imp q p))› |
  ‹OK (Imp (Imp p (Imp q r)) (Imp (Imp p q) (Imp p r)))› |
  ‹OK (Imp (Imp p r) (Imp (Imp q r) (Imp (Dis p q) r)))› |
  ‹OK (Imp p (Dis p q))› |
  ‹OK (Imp q (Dis p q))› |
  ‹OK (Imp (Con p q) p)› |
  ‹OK (Imp (Con p q) q)› |
  ‹OK (Imp p (Imp q (Con p q)))› |
  ‹OK (Imp (Imp (Imp p Falsity) Falsity) p)› |
  ‹OK p ⟹ OK (Imp p q) ⟹ OK q›


theorem soundness: ‹OK p ⟹ semantics i p›
  by (induct rule: OK.induct) simp_all
```

It is again a good exercise for the students to enter the axioms, now for the provability predicate OK as shown above, just by looking at the normal syntax (and see how Isabelle highlights all mistakes in the process):

```
A ⟶ B ⟶ A

(A ⟶ B ⟶ C) ⟶ (A ⟶ B) ⟶ A ⟶ C

(A ⟶ C) ⟶ (B ⟶ C) ⟶ A ∨ B ⟶ C

A ⟶ A ∨ B

B ⟶ A ∨ B

A ∧ B ⟶ A

A ∧ B ⟶ B

A ⟶ B ⟶ A ∧ B

((A ⟶ False) ⟶ False) ⟶ A
```

We find that it is a good preparation to work in Isabelle with the much simpler axiomatics for propositional logic before considering the natural deduction proof system for first-order logic in NaDeA.

Of course it is not easy to carry out proofs in the axiomatics. Even the proof of $A \to A$ requires 5 lines:

1. $\vdash (A \to (A \to A) \to A) \to (A \to A \to A) \to A \to A$ Axiom 2
2. $\vdash A \to (A \to A) \to A$ Axiom 1
3. $\vdash (A \to A \to A) \to A \to A$ MP 1, 2
4. $\vdash A \to A \to A$ Axiom 1
5. $\vdash A \to A$ MP 3, 4

This leads naturally to the notion of completeness:

```
theorem main: ‹(∀i. semantics i p) ⟷ OK p›
```

A formal proof in Isabelle of the soundness and completeness theorems as given by main above is available for the interested students (about 1000 lines including other results).

We intend in the future also to let the students work on a formalization in Isabelle of a sequent calculus for propositional logic since this proof system can be taken as a simple automatic prover in contrast with the above axiomatics.

6 Student Evaluations of NaDeA

The course evaluation spans all 13 weeks of the course, not just the couple of weeks spent on NaDeA. We deem the overall evaluation relevant anyhow, and note that people tend to agree that the teaching material and course is good, and that they learn something from it.

Course evaluation results for 02156 Logical Systems and Logic Programming 2018 (Fall)

`https://www.compute.dtu.dk/english/research/research-sections/algolog`

In total 17 out of 48 registered students answered the anonymous online form (35%):

1	I think I am learning a lot in this course	3.7
2	I think the teaching method encourages my active participation	3.5
3	I think the teaching material is good	3.6
4	I think that throughout the course, the teacher has clearly communicated to me where I stand academically	3.7
5	I think the teacher creates good continuity between the different teaching activities	3.7
6	5 points is equivalent to 9 hours/week — I think my performance during the course is	3.2
7	I think the course descriptions prerequisites are	2.9
8	In general, I think this is a good course	3.7

The scale is “strongly disagree” (1) to “strongly agree” (5) except for 6 and 7 where it is “much more” (1) to “much less” (5) and “too high” (1) to “too low” (5), respectively.

To complement the course evaluation we asked students about feedback specifically on NaDeA. We handed out to each of the 27 students present in classroom a paper sheet with the following text:

Questionnaire — Natural Deduction Assistant (NaDeA)

Advantages

- The proof system is formally proved sound and complete.
- The structured environment makes one focus on the proof development process.
- It forces one to input well-formed formulas and use applicable rules only.

Please add at least one new item:

Disadvantages

- Mouse-clicking can be tedious.
- It can be difficult to know whether the shortest proof has been achieved.
- For smaller proofs one can make more or less progress by clicking blindly and not understanding what is happening.

Please add at least one new item:

---

12 students returned the paper sheet in box (some students informed us that they answered based on discussions in a small ad hoc group).

One student wanted the possibility to change parts of a formula without having to undo all the way back to the step where it was introduced and rebuilding the remaining proof. This would require detecting how much of the proof is still correct, i.e. which applications of rules should be invalidated and determining what to do with those. While cumbersome at times, the current requirement of undoing is a much simpler solution, both to implement and for students to understand. One student appreciated that you can always undo to all previous states, which is contrary to many applications, even text editors, where performing an action after undoing means that the previous undos cannot be redone.

A related requested feature was the trimming of finished proofs to remove all undo steps, such that the proofs could be undone and redone without detours. We have an external AWK script that does this, which is also useful to prepare proofs for presentation purposes, and integrating the feature into the system could make sense.

Another request was better scaling of the interface with the window width, and we miss this ourselves as well. Especially for presentation purposes when the projector is narrower than the common laptop screen. In the classroom where everyone uses laptops, it has not appeared to be a problem.

Finally someone requested a dark theme, that is, light-colored text on a dark background, while others praised the interface and especially the font.

The biggest gripe observed in the classroom was the need to click many times to input formulas. As mentioned this problem was known to us already. Another problem was confusion around the use of functions with no arguments as constants, but this problem is not specific to NaDeA.

Although NaDeA has a number of help/tutorial/exercises pages, we intend to provide more in-depth explanations in future versions.

7 Exercises in NaDeA

In the classroom we ask the students to prove the following formulas (in addition to the "standard" example $A \to A$ used in the online tutorial):

$\bot \to \bot$	Test 1
$\bot \to A$	Hint 1
$(A \to B) \to A \to B$	Test 2
$A \to (A \to B) \to B$	Hint 2
$A \wedge (A \to B) \to B$	Test 3
$A(c) \wedge (A(c) \to \forall x A(x)) \to \forall x A(x)$	Hint 3
$\forall x A(x) \to A(c)$	Test 4
$A(c) \to \exists x A(x)$	Hint 4
$A \to B \to A$	Test 5
$(A \to B \to C) \to (A \to B) \to A \to C$	Hint 5
$A \to (A \to \bot) \to \bot$	Test 6
$((A \to \bot) \to \bot) \to A$	Hint 6
$(A \wedge B) \to C \to (A \wedge C)$	Test 7
$((A \to \bot) \vee (B \to \bot)) \to (A \wedge B) \to \bot$	Hint 7
$\forall x \forall y A(x,y) \to \forall x A(x,x)$	Test 8
$\forall x A(x) \to \exists x A(x)$	Hint 8
$\exists x (A(x) \to \forall x A(x))$	Test 9
$\forall x (\neg r(x) \to r(f(x))) \to \exists x (r(x) \wedge r(f(f(x))))$	Hint 9 — Optional.

Solutions are available online in NaDeA either as so-called hints in the Help window or as so-called tests in the Load window in NaDeA. A test resumes from the last proof state but a hint replays from the first proof state (one can click Undo to show the proof states).

However, we do not provide the solution to the final formula, in order to challenge the best students (we have a solution with 42 lines in NaDeA). The formula is discussed on page 128 of the Handbook of Tableau Methods (Kluwer Academic Publishers 1999):

> *If every person that is not rich has a rich father, then some rich person must have a rich grandfather.*

The formalization uses r for *rich* and f for *father*.

In the course the students must individually hand in 4 assignments with several quesions about logical systems and logic programming. With respect to NaDeA the students are asked to prove the following 5 formulas:

1. $A \wedge B \to B$
2. $A(c,c) \to \exists x \exists y A(x,y)$
3. $(\forall x A(x) \vee \forall x B(x)) \to \forall x (A(x) \vee B(x))$
4. $A \vee (A \to \bot)$
5. $(A \to B) \vee (B \to C)$

Solutions to the assignments are not provided to the students. Instead we provide detailed individual feedback. Both the assignment with the NaDeA questions and the final written course exam are pending.

In case a student gets stuck with the last formula in the assignment, $(A \to B) \vee (B \to C)$, some assistance can be provided as Example 1 and Example 2.

Example 1

```
1    Dis_E   [ ] (A→B)∨(B→A)
2      ¤       [ ] A∨(A→⊥)
3      ¤       [A] (A→B)∨(B→A)
4      ¤       [A→⊥] (A→B)∨(B→A)
```

Example 2

```
 1    Boole    [ ] (A→B)∨(B→A)
 2    Imp_E      [(A→B)∨(B→A)→⊥] ⊥
 3    Assume       [(A→B)∨(B→A)→⊥] (A→B)∨(B→A)→⊥
 4    Dis_I1       [(A→B)∨(B→A)→⊥] (A→B)∨(B→A)
 5    Imp_I          [(A→B)∨(B→A)→⊥] A→B
 6    Boole            [A, (A→B)∨(B→A)→⊥] B
 7    Imp_E              [B→⊥, A, (A→B)∨(B→A)→⊥] ⊥
 8    Assume               [B→⊥, A, (A→B)∨(B→A)→⊥] (A→B)∨(B→A)→⊥
 9    Dis_I2               [B→⊥, A, (A→B)∨(B→A)→⊥] (A→B)∨(B→A)
10    Imp_I                  [B→⊥, A, (A→B)∨(B→A)→⊥] B→A
11    Assume                   [B, B→⊥, A, (A→B)∨(B→A)→⊥] A
```

We note that although the solution for $(A \to B) \vee (B \to C)$ is similar to Example 1 as well as Example 2, at least a non-trivial observation is necessary in order to complete the proof in NaDeA.

8 Conclusion

NaDeA has been used for teaching first-order logic to hundreds of computer science bachelor students. We have discussed its advantages and disadvantages as a tool for teaching logic. In general NaDeA has been a success in our bachelor course but we intend to provide more in-depth explanations in future versions of NaDeA. However, in our experience it is more or less mandatory with a number of exercise sessions with skilled teaching assistants, say, one teaching assistant for a class of 40 bachelor students. Advanced master or PhD students usually manage without help from teaching assistants. For example, NaDeA has recently been used without a teaching assistant by a class of mainly PhD students at the 29th European Summer School in Logic, Language, and Information (ESSLLI), University of Toulouse, France, 17-28 July 2017:

`https://www.irit.fr/esslli2017/courses/24.html`

Proof assistants such as Isabelle allow for many kinds of reasoning that go beyond natural deduction and their interfaces, of course, account for that. In NaDeA, on the other hand, there are no distractions – all buttons and texts in NaDeA have to do with natural deduction. The structured environment provided by NaDeA, based on clicking buttons instead of textual input, makes it possible for students to focus on the proof development process. Since NaDeA only allows input of well-formed formulas and application of applicable rules, the student does not have to worry about neither syntax nor well-formedness errors possible for instance in Isabelle. Conversely, experienced users may feel slightly inhibited by the system as textually inputting a formula is often faster than using the mouse. Furthermore for very simple proofs, students may be able to find them by clicking blindly and without understanding what they are doing, since only the applicable rules are shown. This is not a problem for larger proofs.

As future work we consider developing more teaching materials for NaDeA and making further evaluations of NaDeA as a tool for teaching logic.

Acknowledgements

We thank Alexander Birch Jensen for collaboration on the initial development of NaDeA and we thank John Bruntse Larsen and Stefan Berghofer for fruitful discussions. We also thank the anonymous reviewers for their comments.

References

[1] Stefan Berghofer (2007): *First-Order Logic According to Fitting*. *Archive of Formal Proofs*. `http://isa-afp.org/entries/FOL-Fitting.shtml`, Formal proof development.

[2] A. Church (1956): *Introduction to Mathematical Logic*. Princeton University Press.

[3] H. Geuvers (2009): *Proof Assistants: History, Ideas and Future*. *Sadhana* 34(1), pp. 3–25, doi:10.1007/s12046-009-0001-5.

[4] Alexander Birch Jensen, John Bruntse Larsen, Anders Schlichtkrull & Jørgen Villadsen (2018): *Programming and verifying a declarative first-order prover in Isabelle/HOL*. *AI Communications* 31(3), pp. 281–299, doi:10.3233/AIC-180764.

[5] Julius Michaelis & Tobias Nipkow (2017): *Propositional Proof Systems*. *Archive of Formal Proofs*. `http://isa-afp.org/entries/Propositional_Proof_Systems.shtml`, Formal proof development.

[6] Tobias Nipkow, Lawrence C. Paulson & Markus Wenzel (2002): *Isabelle/HOL - A Proof Assistant for Higher-Order Logic*. *Lecture Notes in Computer Science* 2283, Springer, doi:10.1007/3-540-45949-9.

[7] Jørgen Villadsen, Andreas Halkjær From & Anders Schlichtkrull (2017): *Natural Deduction and the Isabelle Proof Assistant*. In Pedro Quaresma & Walther Neuper, editors: *Proceedings 6th International Workshop on Theorem proving components for Educational Software (ThEdu)*, *EPTCS* 267, pp. 140–155, doi:10.4204/EPTCS.267.9.

[8] Jørgen Villadsen, Alexander Birch Jensen & Anders Schlichtkrull (2017): *NaDeA: A Natural Deduction Assistant with a Formalization in Isabelle*. *IFCoLog Journal of Logics and their Applications* 4(1), pp. 55–82.

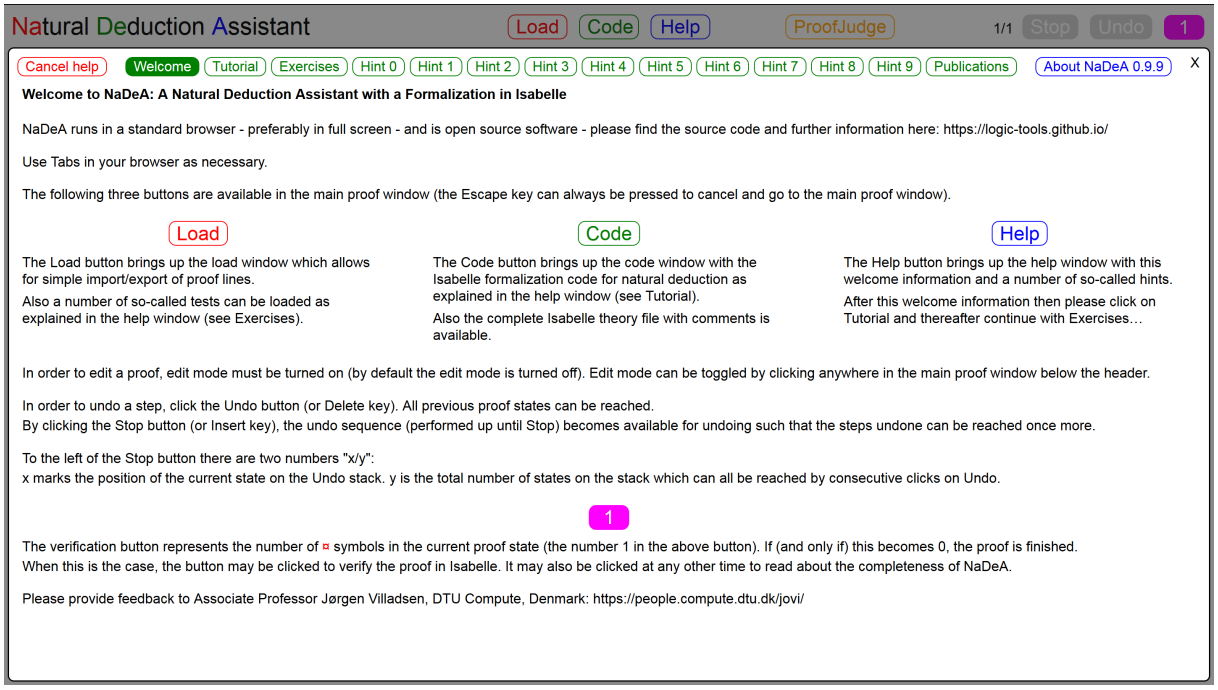

Figure 4: Welcome

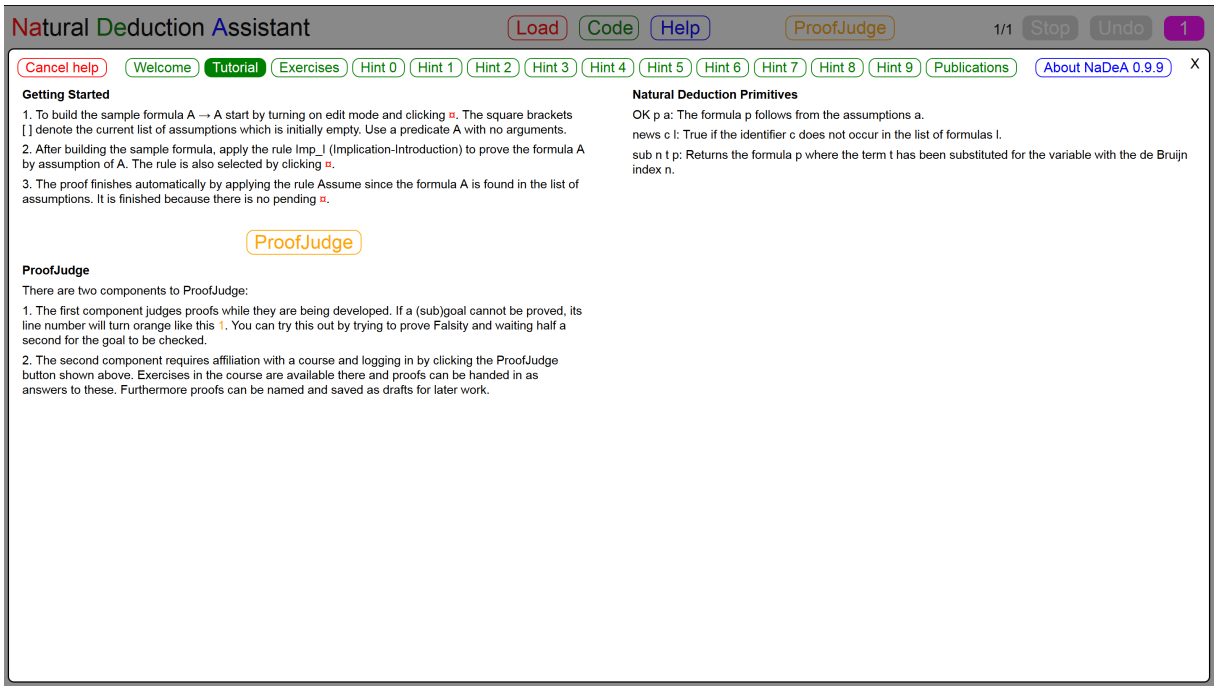

Figure 5: Tutorial

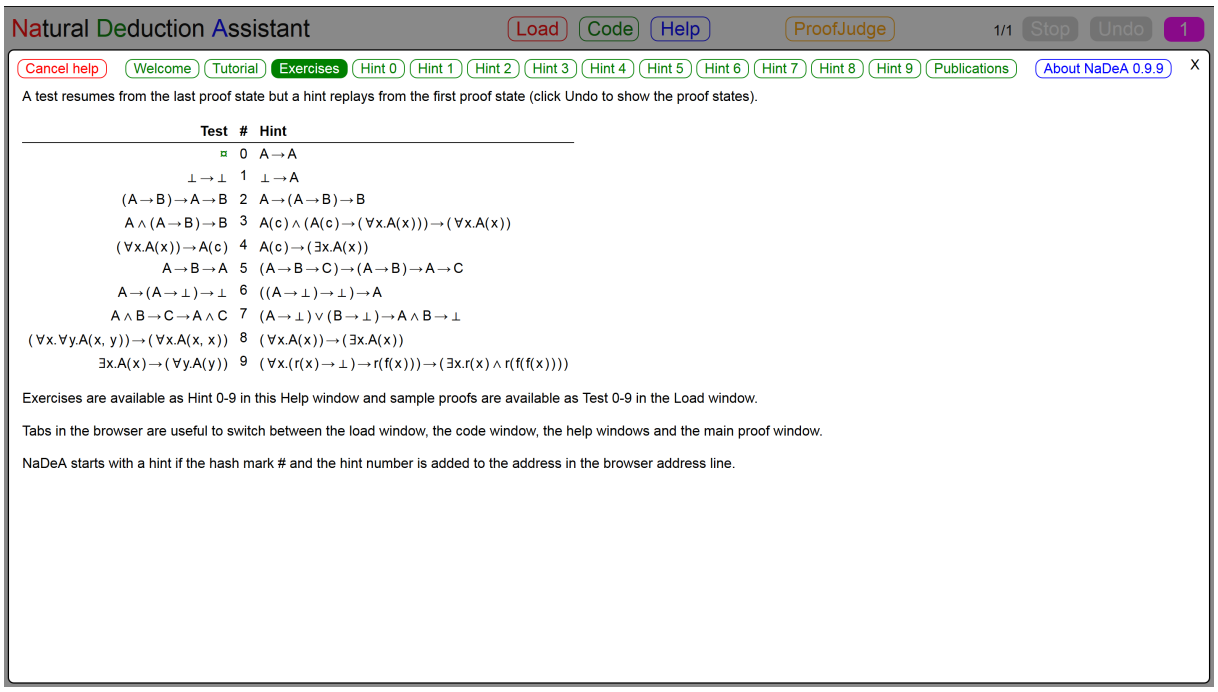

Figure 6: Exercises

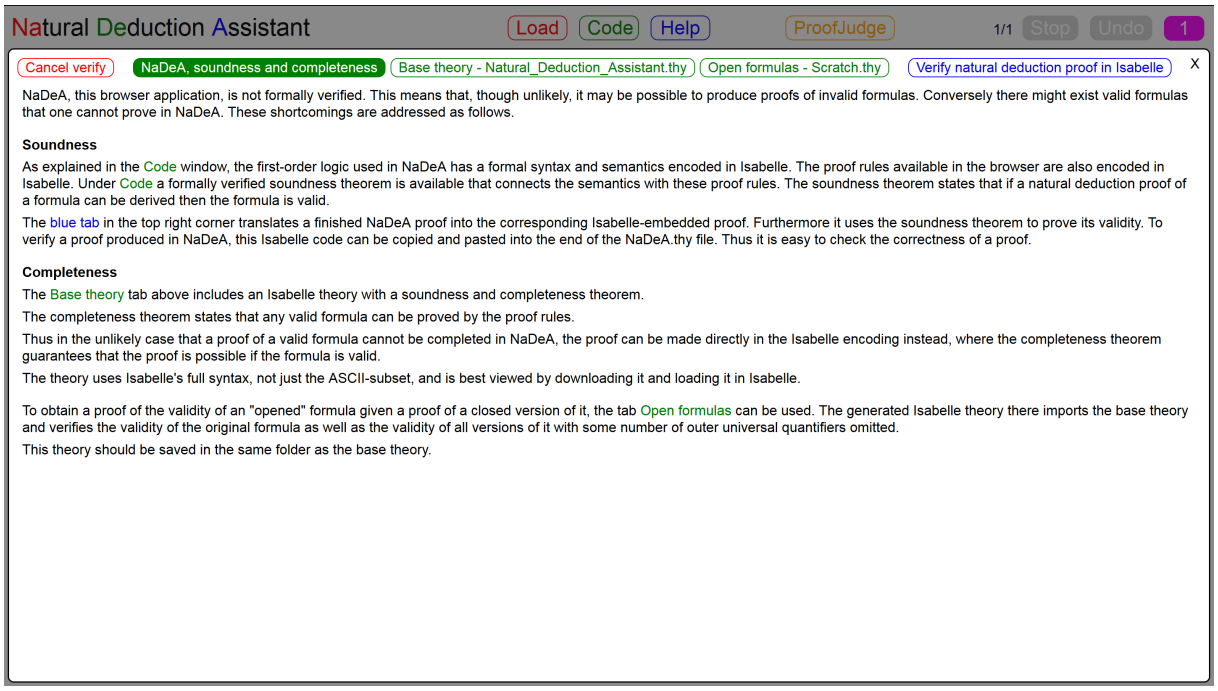

Figure 7: NaDeA, soundness and completeness

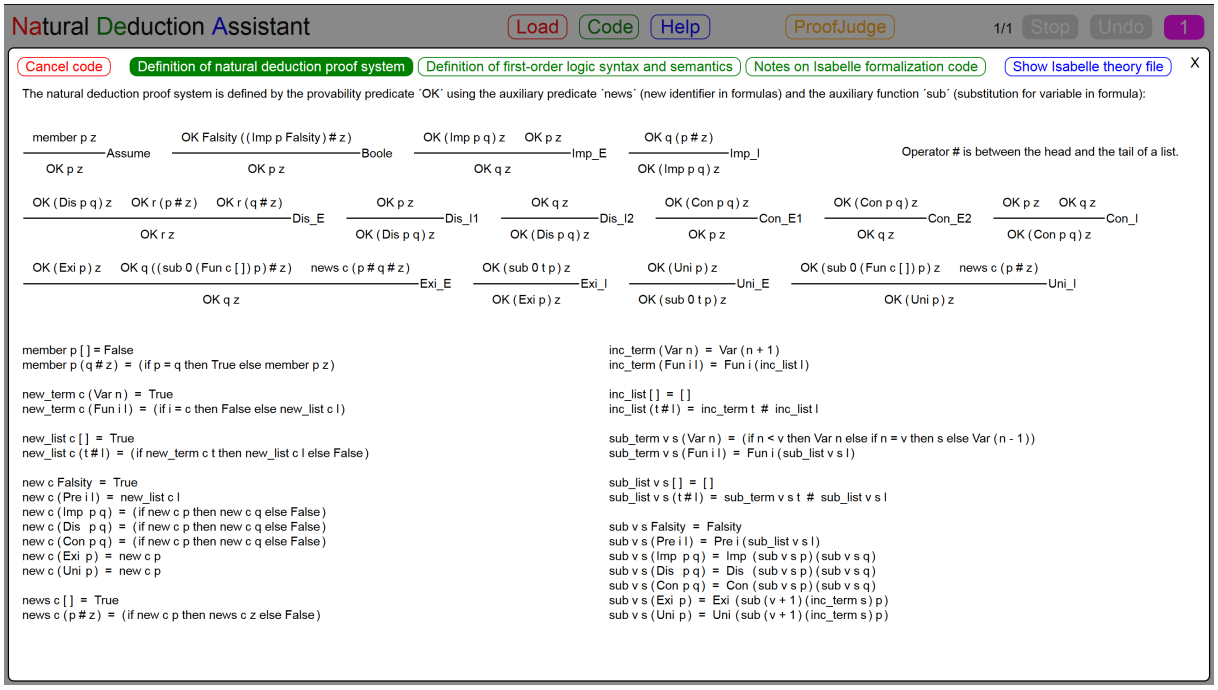

Figure 8: Definition of natural deduction proof system

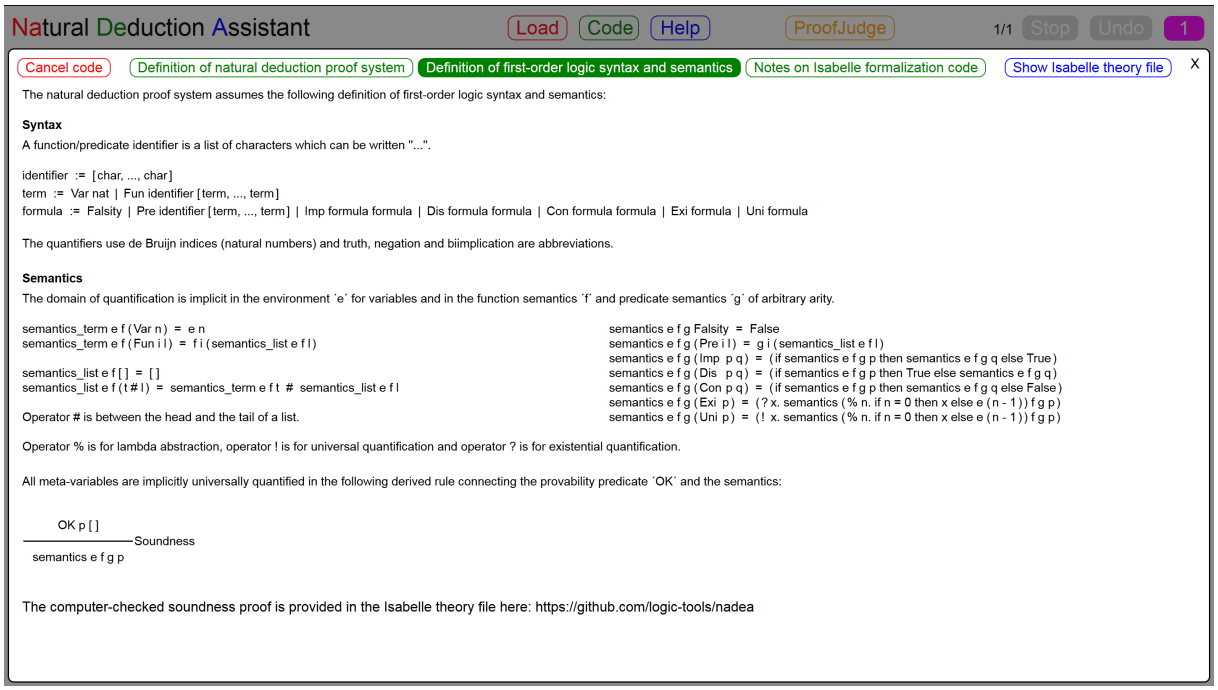

Figure 9: Definition of first-order logic syntax and semantics