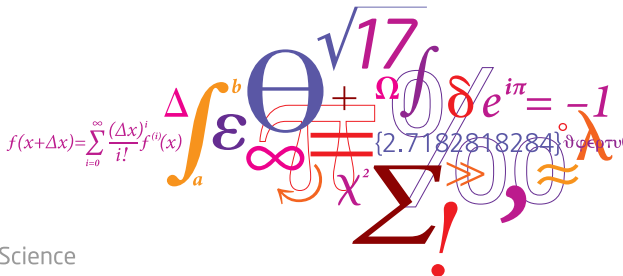


Natural Deduction Assistant (NaDeA)

ThEdu'18 — 18 July 2018

Jørgen Villadsen, Andreas Halkjær From and Anders Schlichtkrull



Motivation

- NaDeA allows students to test their natural deduction skills interactively.
- Open source software, runs in a standard browser and is available at <https://nadea.compute.dtu.dk/>.
- Has been used for teaching first-order logic to hundreds of computer science bachelor students since 2015 [1, 2].
- Integrated log-in system for students to upload proofs and TAs to assess them.
- Upon completion of a proof the student obtains a formal proof in Isabelle/HOL [3].
- Formalization of the proof system extends work by Stefan Berghofer [4] and Melvin Fitting [5].

Terms

- Variables, x, y, z, \dots
- Functions, f, g, h, \dots with list of terms.

Terms

- Variables, x, y, z, \dots
- Functions, f, g, h, \dots with list of terms.

Formulas

- Falsity, \perp .
- Predicates, p, q, r, \dots with list of terms.
- Conjunction, $A \wedge B$. Disjunction, $A \vee B$. Implication, $A \rightarrow B$.
- Universal quantification, $\forall x.A$. Existential quantification, $\exists x.A$.

Terms

- Variables, x, y, z, \dots
- Functions, f, g, h, \dots with list of terms.

Formulas

- Falsity, \perp .
- Predicates, p, q, r, \dots with list of terms.
- Conjunction, $A \wedge B$. Disjunction, $A \vee B$. Implication, $A \rightarrow B$.
- Universal quantification, $\forall x.A$. Existential quantification, $\exists x.A$.

Abbreviations

- $\top \equiv \perp \rightarrow \perp$
- $\neg A \equiv A \rightarrow \perp$

Natural Deduction Assistant

What is NaDeA?



Natural Deduction Assistant

1 [] ▢

[Load](#) [Code](#) [Help](#)

[ProofJudge](#)

1/1 [Stop](#) [Undo](#) [1](#)

Natural Deduction Assistant

What is NaDeA?



Natural Deduction Assistant

Load
Code
Help
ProofJudge
27/27 Stop Undo

Cancel code
Definition of natural deduction proof system
Definition of first-order logic syntax and semantics
Notes on Isabelle formalization code
Show Isabelle theory file
✕

The natural deduction proof system is defined by the provability predicate 'OK' using the auxiliary predicate 'news' (new identifier in formulas) and the auxiliary function 'sub' (substitution for variable in formula).

$\frac{\text{member } p \ z}{\text{OK } p \ z} \text{Assume}$	$\frac{\text{OK Falsity } ((\text{Imp } p \ \text{Falsity}) \ # \ z)}{\text{OK } p \ z} \text{Boole}$	$\frac{\text{OK } (\text{Imp } p \ q) \ z}{\text{OK } q \ z} \text{Imp_E}$	$\frac{\text{OK } q \ (p \ # \ z)}{\text{OK } (\text{Imp } p \ q) \ z} \text{Imp_I}$	Operator # is between the head and the tail of a list.
$\frac{\text{OK } (\text{Dis } p \ q) \ z \quad \text{OK } r \ (p \ # \ z) \quad \text{OK } r \ (q \ # \ z)}{\text{OK } r \ z} \text{Dis_E}$	$\frac{\text{OK } p \ z}{\text{OK } (\text{Dis } p \ q) \ z} \text{Dis_I1}$	$\frac{\text{OK } q \ z}{\text{OK } (\text{Dis } p \ q) \ z} \text{Dis_I2}$	$\frac{\text{OK } (\text{Con } p \ q) \ z}{\text{OK } p \ z} \text{Con_E1}$	$\frac{\text{OK } (\text{Con } p \ q) \ z}{\text{OK } q \ z} \text{Con_E2}$
$\frac{\text{OK } (\text{Exi } p) \ z \quad \text{OK } q \ ((\text{sub } 0 \ (\text{Fun } c \ []) \ p) \ # \ z) \quad \text{news } c \ (p \ # \ q \ # \ z)}{\text{OK } q \ z} \text{Exi_E}$	$\frac{\text{OK } (\text{sub } 0 \ t \ p) \ z}{\text{OK } (\text{Exi } p) \ z} \text{Exi_I}$	$\frac{\text{OK } (\text{Uni } p) \ z}{\text{OK } (\text{sub } 0 \ t \ p) \ z} \text{Uni_E}$	$\frac{\text{OK } (\text{sub } 0 \ (\text{Fun } c \ []) \ p) \ z \quad \text{news } c \ (p \ # \ z)}{\text{OK } (\text{Uni } p) \ z} \text{Uni_I}$	

```

member p [] = False
member p (q # z) = (if p = q then True else member p z)

new_term c (Var n) = True
new_term c (Fun f) = (if f = c then False else new_list c l)

new_list c [] = True
new_list c (t # l) = (if new_term c t then new_list c l else False)

new c Falsity = True
new c (Pre f) = new_list c l
new c (Imp p q) = (if new c p then new c q else False)
new c (Dis p q) = (if new c p then new c q else False)
new c (Con p q) = (if new c p then new c q else False)
new c (Exi p) = new c p
new c (Uni p) = new c p

news c [] = True
news c (p # z) = (if new c p then news c z else False)

inc_term (Var n) = Var (n + 1)
inc_term (Fun f) = Fun f (inc_list l)

inc_list [] = []
inc_list (t # l) = inc_term t # inc_list l

sub_term v s (Var n) = (if n < v then Var n else if n = v then s else Var (n - 1))
sub_term v s (Fun f) = Fun f (sub_list v s l)

sub_list v s [] = []
sub_list v s (t # l) = sub_term v s t # sub_list v s l

sub v s Falsity = Falsity
sub v s (Pre f) = Pre f (sub_list v s l)
sub v s (Imp p q) = Imp (sub v s p) (sub v s q)
sub v s (Dis p q) = Dis (sub v s p) (sub v s q)
sub v s (Con p q) = Con (sub v s p) (sub v s q)
sub v s (Exi p) = Exi (sub (v + 1) (inc_term s) p)
sub v s (Uni p) = Uni (sub (v + 1) (inc_term s) p)

```

4 DTU Compute

Natural Deduction Assistant (NaDeA) 18 July 2018

Natural Deduction Assistant

What is NaDeA?



Natural Deduction Assistant [Load] [Code] [Help] [ProofJudge] 27/27 [Stop] [Undo] [Refresh]

[Cancel code] [Definition of natural deduction proof system] [Definition of first-order logic syntax and semantics] [Notes on Isabelle formalization code] [Show Isabelle theory file] X

The natural deduction proof system assumes the following definition of first-order logic syntax and semantics:

Syntax
A function/predicate identifier is a list of characters which can be written "...".

identifier := [char, ..., char]
term := Var nat | Fun identifier [term, ..., term]
formula := Falsity | Pre identifier [term, ..., term] | Imp formula formula | Dis formula formula | Con formula formula | Exi formula | Uni formula

The quantifiers use de Bruijn indices (natural numbers) and truth, negation and bimplication are abbreviations.

Semantics
The domain of quantification is implicit in the environment 'e' for variables and in the function semantics 'f' and predicate semantics 'g' of arbitrary arity.

semantics_term e f (Var n) = e n	semantics e f g Falsity = False
semantics_term e f (Fun i l) = f i (semantics_list e f l)	semantics e f g (Pre i l) = g i (semantics_list e f l)
semantics_list e f [] = []	semantics e f g (Imp p q) = (if semantics e f g p then semantics e f g q else True)
semantics_list e f (t # l) = semantics_term e f t # semantics_list e f l	semantics e f g (Dis p q) = (if semantics e f g p then True else semantics e f g q)

Operator # is between the head and the tail of a list.
Operator % is for lambda abstraction, operator ! is for universal quantification and operator ? is for existential quantification.

All meta-variables are implicitly universally quantified in the following derived rule connecting the provability predicate 'OK' and the semantics:

$$\frac{\text{OK } p []}{\text{semantics } e \text{ f } g \text{ p}} \text{Soundness}$$

The computer-checked soundness proof is provided in the Isabelle theory file here: <https://github.com/logic-tools/nadea>

Natural Deduction Assistant

What is NaDeA?



Natural Deduction Assistant

Load Code Help ProofJudge 27/27 Stop Undo

Cancel help Welcome Tutorial Exercises Hint 0 Hint 1 Hint 2 Hint 3 Hint 4 Hint 5 Hint 6 Hint 7 Hint 8 Hint 9 Publications About NaDeA 0.9.9 X

A test resumes from the last proof state but a hint replays from the first proof state (click Undo to show the proof states).

Test #	Hint
0	$A \rightarrow A$
1	$\perp \rightarrow \perp$
2	$(A \rightarrow B) \rightarrow A \rightarrow B$
3	$A \wedge (A \rightarrow B) \rightarrow B$
4	$A(c) \wedge (A(c) \rightarrow (\forall x.A(x))) \rightarrow (\forall x.A(x))$
5	$(\forall x.A(x)) \rightarrow A(c)$
6	$A(c) \rightarrow (\exists x.A(x))$
7	$A \rightarrow B \rightarrow A$
8	$(A \rightarrow B \rightarrow C) \rightarrow (A \rightarrow B) \rightarrow A \rightarrow C$
9	$A \rightarrow (A \rightarrow \perp) \rightarrow \perp$
10	$((A \rightarrow \perp) \rightarrow \perp) \rightarrow A$
11	$A \wedge B \rightarrow C \rightarrow A \wedge C$
12	$(A \rightarrow \perp) \vee (B \rightarrow \perp) \rightarrow A \wedge B \rightarrow \perp$
13	$(\forall x.\forall y.A(x, y)) \rightarrow (\forall x.A(x, x))$
14	$(\forall x.A(x)) \rightarrow (\exists x.A(x))$
15	$\exists x.A(x) \rightarrow (\forall y.A(y))$
16	$(\forall x.(r(x) \rightarrow \perp) \rightarrow r(f(x))) \rightarrow (\exists x.r(x) \wedge r(f(f(x))))$

Exercises are available as Hint 0-9 in this Help window and sample proofs are available as Test 0-9 in the Load window.

Tabs in the browser are useful to switch between the load window, the code window, the help windows and the main proof window.

NaDeA starts with a hint if the hash mark # and the hint number is added to the address in the browser address line.

Natural Deduction Assistant

What is NaDeA?



Natural Deduction Assistant

What is NaDeA?



Natural Deduction Assistant Load Code Help Logged in... 27/27 Stop Undo Logout

Logged in as HHstudent (DTU 02156 2017)

Assignment 4	Goal	Selected solution
Exercise 1	$A \wedge B \rightarrow B$	Ex1 ▾
Exercise 2	$A(c, c) \rightarrow (\exists x \exists y A(x, y))$	Ex2 ▾
Exercise 3	$(\forall x A(x)) \vee (\forall x B(x)) \rightarrow (\forall x A(x) \vee B(x))$	- ▾
Exercise 4	$A \vee (A \rightarrow \perp)$	- ▾
Exercise 5	$(A \rightarrow B) \vee (B \rightarrow A)$	- ▾

Draft	Goal	Number of \Rightarrow	Proof rules used
Ex1	$A \wedge B \rightarrow B$	0	2 Load draft -
Ex2	$A(c, c) \rightarrow (\exists x \exists y A(x, y))$	1	0 Load draft -

Create draft of current proof state:

Draft name:

Goal: $A \wedge B \rightarrow C \rightarrow A \wedge C$

Number of \Rightarrow : 0

Proof rules used: 4

Save draft

Natural Deduction Assistant

Constructing a Formula



1 OK \square []

Natural Deduction Assistant

Constructing a Formula

1 OK []

Formulas: x

Falsity

Predicate

Implication

Disjunction

Conjunction

Existential Quantifier

Universal Quantifier

Natural Deduction Assistant

Constructing a Formula



1 OK (Imp α α) []

Natural Deduction Assistant
Constructing a Formula



1 OK (Imp α (Pre α α)) []

Natural Deduction Assistant

Constructing a Formula



1 OK (Imp \square (Pre \square \square)) []

New ID: x

A ▼

Done

Natural Deduction Assistant

Constructing a Formula



1 OK (Imp \square (Pre "B" \square)) \square) \square

New list of terms: X

Done	Add term	Remove last term
------	----------	------------------

1 OK (Imp (Con α α) (Pre "B" [])) []

1 OK (Imp (Con α (Imp α α)) (Pre "B" [])) []

Constructing a Formula



1 OK (Imp (Con (Pre "A" []) (Imp (Pre "A" []) (Pre "B" [])))
 (Pre "B" []))[]

1 ✘ [] $A \wedge (A \rightarrow B) \rightarrow B$

1 □ [] $A \wedge (A \rightarrow B) \rightarrow B$

Natural Deduction Assistant

Sample Proof



1 α $\boxed{\perp} \quad [A \wedge (A \rightarrow B) \rightarrow B]$

Boole	x
Imp_E	
Imp_I	
Dis_E	
Con_E1	
Con_E2	
Exi_E	

Sample Proof

1 Imp_I [] $A \wedge (A \rightarrow B) \rightarrow B$

2 \square [$A \wedge (A \rightarrow B)$] B

Sample Proof

1 Imp_I [] $A \wedge (A \rightarrow B) \rightarrow B$

2 \square [$A \wedge (A \rightarrow B)$] B

Boole	x
Imp_E	
Dis_E	
Con_E1	
Con_E2	
Exi_E	

Sample Proof



- 1 Imp_I [] $A \wedge (A \rightarrow B) \rightarrow B$
- 2 Imp_E [$A \wedge (A \rightarrow B)$] B
- 3 [$A \wedge (A \rightarrow B)$] $\square \rightarrow B$
- 4 [$A \wedge (A \rightarrow B)$] \square

Sample Proof



- 1 Imp_I [] $A \wedge (A \rightarrow B) \rightarrow B$
- 2 Imp_E [$A \wedge (A \rightarrow B)$] B
- 3 α [$A \wedge (A \rightarrow B)$] $A \rightarrow B$
- 4 α [$A \wedge (A \rightarrow B)$] A

Sample Proof

- 1 Imp_I [] $A \wedge (A \rightarrow B) \rightarrow B$
- 2 Imp_E [$A \wedge (A \rightarrow B)$] B
- 3 Con_E2 [$A \wedge (A \rightarrow B)$] $A \rightarrow B$
- 4 [$A \wedge (A \rightarrow B)$] $\neg \wedge (A \rightarrow B)$
- 5 \square [$A \wedge (A \rightarrow B)$] A

Sample Proof



- 1 Imp_I [] $A \wedge (A \rightarrow B) \rightarrow B$
- 2 Imp_E [$A \wedge (A \rightarrow B)$] B
- 3 Con_E2 [$A \wedge (A \rightarrow B)$] $A \rightarrow B$
- 4 Assume [$A \wedge (A \rightarrow B)$] $A \wedge (A \rightarrow B)$
- 5 \square [$A \wedge (A \rightarrow B)$] A

Sample Proof

- 1 Imp_I [] $A \wedge (A \rightarrow B) \rightarrow B$
- 2 Imp_E [$A \wedge (A \rightarrow B)$] B
- 3 Con_E2 [$A \wedge (A \rightarrow B)$] $A \rightarrow B$
- 4 Assume [$A \wedge (A \rightarrow B)$] $A \wedge (A \rightarrow B)$
- 5 Con_E1 [$A \wedge (A \rightarrow B)$] A
- 6 [$A \wedge (A \rightarrow B)$] $A \wedge \square$

Sample Proof

- 1 Imp_I [] $A \wedge (A \rightarrow B) \rightarrow B$
- 2 Imp_E [$A \wedge (A \rightarrow B)$] B
- 3 Con_E2 [$A \wedge (A \rightarrow B)$] $A \rightarrow B$
- 4 Assume [$A \wedge (A \rightarrow B)$] $A \wedge (A \rightarrow B)$
- 5 Con_E1 [$A \wedge (A \rightarrow B)$] A
- 6 Assume [$A \wedge (A \rightarrow B)$] $A \wedge (A \rightarrow B)$






- + The proof system formally proved sound and complete.
- + Structured environment, focus on the proof development process.
- + Only allows input of well-formed formulas and application of applicable rules.

- 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.

NaDeA has been used for teaching first-order logic to hundreds of computer science bachelor students.

NaDeA has recently been used 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>).

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

-  JØRGEN VILLADSEN, ALEXANDER BIRCH JENSEN AND ANDERS SCHLICHTKRULL, *NaDeA: A Natural Deduction Assistant with a Formalization in Isabelle*, **IFCoLog Journal of Logics and their Applications**, vol. 4 (2017), no. 1, pp. 55–82.
-  JØRGEN VILLADSEN, ANDREAS HALKJÆR FROM AND ANDERS SCHLICHTKRULL, *Natural Deduction and the Isabelle Proof Assistant*, Proceedings 6th International Workshop on **Theorem proving components for Educational software** (Gothenburg, Sweden), (Pedro Quaresma and Walther Neuper, editors), vol. 267, Electronic Proceedings in Theoretical Computer Science, Open Publishing Association, 2018, pp. 140–155. <http://eptcs.org/paper.cgi?ThEdu17.9>
-  TOBIAS NIPKOW, LAWRENCE C. PAULSON AND MARKUS WENZEL, *Isabelle/HOL — A Proof Assistant for Higher-Order Logic*, vol. 2283, Lecture Notes in Computer Science, Springer, 2002.
-  STEFAN BERGHOFER, *First-Order Logic According to Fitting*, **Archive of Formal Proofs**, August 2007. <http://isa-afp.org/entries/FOL-Fitting.html>
-  MELVIN FITTING, *First-Order Logic and Automated Theorem Proving, Second Edition*, Graduate Texts in Computer Science, Springer, 1996.