Introduction to Interactive Theorem Provers

- Code: MTAT.05.126; 3 ECTS
- E-mail: *kalmer.apinis@{ut.ee | gmail.com}*
- Labs: On Mondays, 14.15 - 16.00, room 512

Grading:

- Standard scale: 51% .. 60% $\longrightarrow$ E, etc.
- Solve all homework proofs! (no points)
- Final exam: 5 proofs (80%), 4 short questions (20%).

- Or some larger proof of your own choosing.

- Or generate exercises for the next year.

Motivation

Machine checked formal proofs:

- Makes you explicitly state assumptions and goals.
- The reader does not read the proofs, but examines the assumptions and the proved goal.

Motivation

Machine checked formal proofs:

- Makes you explicitly state assumptions and goals.
- The reader does not read the proofs, but examines the assumptions and the proved goal.

Leaning only on computer generated proofs does not give you *intuition* about the matter. And intuition is required if automation fails.

Motivation

Machine checked formal proofs:

- Makes you explicitly state assumptions and goals.
- The reader does not read the proofs, but examines the assumptions and the proved goal.

Leaning only on computer generated proofs does not give you *intuition* about the matter. And intuition is required if automation fails.

And intuition is useful for formalizing goals.

Motivation

Machine checked formal proofs:

- Makes you explicitly state assumptions and goals.
- The reader does not read the proofs, but examines the assumptions and the proved goal.

Leaning only on computer generated proofs does not give you *intuition* about the matter. And intuition is required if automation fails.

And intuition is useful for formalizing goals.

$\Longrightarrow$ Interactive Theorem Provers

Interactive Theorem Provers

In this course we concentrate on

practical skills

not

"theory" or "understanding"

i.e. the functionalist approach

Course Plan:

- Learn Coq rules for: implication, conjunction, disjunction, not, True, False, iff
- Exercise classical logic
- Rules for: forall, exists, equalities/inequalities, natural numbers.
- Proofs by induction for propositions using addition, multiplication, subtraction, $\leq$.
- Declarative proofs.*
- Proofs about lists and sets.
- Managing assumptions using modules. *

*) not in the final

Next steps after the course

- "Certified Programming with Dependent Types", Adam Chlipala, 2013, `http://adam.chlipala.net/cpdt/`
- Other generic theorem provers, e.g., Isabelle
- Specialized tools: F*, EasyCrypt