Implementing CERES: tools for proof analysis

Daniel Weller

Theory and Logic Group
Computer Science Department

Technische Universität Wien, Austria

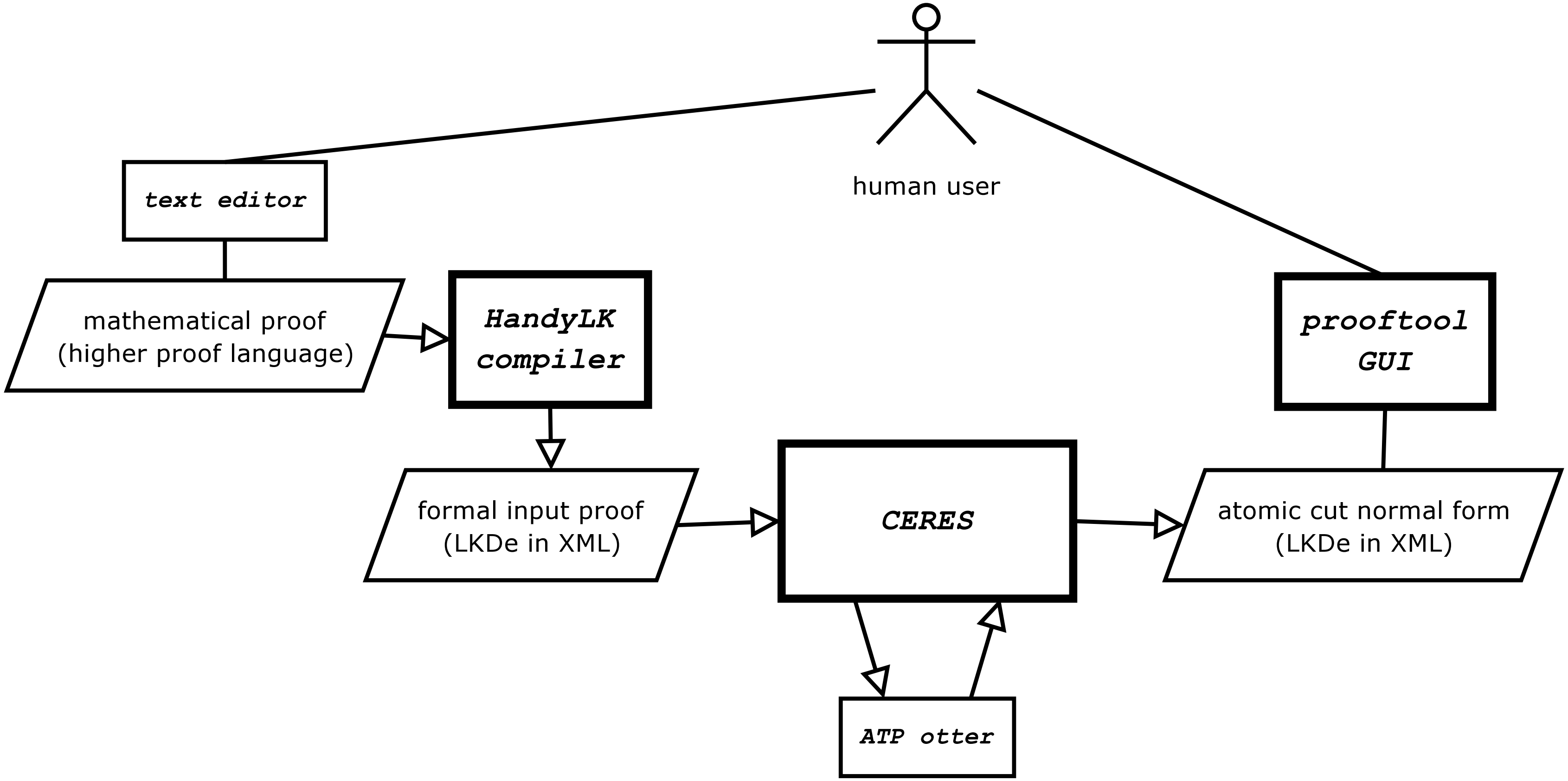
Outline

- Motivation
- Overview of the architecture
- Writing proofs
- Viewing proofs
- Transforming proofs

Motivation

- Goal: analyze mathematical proofs using cut-elimination.
- Obstacles:
  - Proof formalization.
  - Analysis of the cut-free proof by a human.

Architecture



Writing proofs

- Writing **LK** proofs directly is tedious.
- Why?

Writing proofs (cont'd)

- Most rule applications in a **LK** proof duplicate redundant information.

$$\frac{P(0) \supset P(s(0)), P(0), \forall x(P(x) \supset P(s(x))) \vdash P(s^2(0))}{\forall x(P(x) \supset P(s(x))), P(0), \forall x(P(x) \supset P(s(x))) \vdash P(s^2(0))}\ \forall : l$$

Writing proofs (cont'd)

- Most rule applications in a **LK** proof duplicate redundant information.
- Propositional parts of proofs can be computed automatically.

$$
\dfrac{
  \dfrac{
    \dfrac{
      \dfrac{
        \dfrac{P(a) \vdash P(a)}{\vdash \neg P(a), P(a)}\;\neg : r
      }{\vdash \neg P(a) \lor Q(a), P(a)}\;\lor : r_1
      \qquad
      \dfrac{Q(a) \vdash Q(a)}{Q(a) \vdash \neg P(a) \lor Q(a)}\;\lor : r_2
    }{P(a) \supset Q(a) \vdash \neg P(a) \lor Q(a), \neg P(a) \lor Q(a)}\;\supset : l
  }{P(a) \supset Q(a) \vdash \neg P(a) \lor Q(a)}\;c : r
}{\forall x P(x) \supset Q(x) \vdash \neg P(a) \lor Q(a)}\;\forall : l
$$

Handy LK

- Handy LK (HLK) allows comfortable writing of **LK** proofs.
- Supports ASCII and UNICODE input.
- HLK compiler outputs **LKDe** proofs to XML or LaTeX.

HLK features

- Usage of definitions.

HLK Source

```
define predicate DIVIDES by ex r l * r = k;
```

$$\frac{\exists r\; t_1 * r = t_2, \Gamma \vdash \Delta}{DIVIDES(t_1, t_2), \Gamma \vdash \Delta}\; d:l$$

HLK features

- Usage of definitions.
- Type checking.

HLK Source

```
define constant 0, 1 of type nat;
define infix function + of type nat,nat to nat
```

HLK features

- Usage of definitions.
- Type checking.
- Automatic generation of structural rules.

HLK features

- Usage of definitions.
- Type checking.
- Automatic generation of structural rules.
- Automatic completion of propositional parts of proofs.

HLK features

- Usage of definitions.
- Type checking.
- Automatic generation of structural rules.
- Automatic completion of propositional parts of proofs.
- ... and much more.

HLK features

- Usage of definitions.
- Type checking.
- Automatic generation of structural rules.
- Automatic completion of propositional parts of proofs.
- ... and much more.

Demonstration.

ProofTool

- Motivation: non-trivial formal proofs are quite large.
- We need a tool to
  - Zoom in/out on parts of a proof.

ProofTool

- Motivation: non-trivial formal proofs are quite large.
- We need a tool to
    - Zoom in/out on parts of a proof.
    - Merge/split parts of a proof.

ProofTool

- Motivation: non-trivial formal proofs are quite large.
- We need a tool to
  - Zoom in/out on parts of a proof.
  - Merge/split parts of a proof.
  - Replace certain formula occurences by other syntax.

ProofTool

- Motivation: non-trivial formal proofs are quite large.
- We need a tool to
  - Zoom in/out on parts of a proof.
  - Merge/split parts of a proof.
  - Replace certain formula occurences by other syntax.
- Supports
  - XML input.
  - XML and LaTeX output.

CERES

- Implements algorithms for
  - Validation of **LKDe** proofs.

CERES

- Implements algorithms for
  - Validation of **LKDe** proofs.
  - Skolemization of **LKDe** proofs.

CERES

- Implements algorithms for
  - Validation of **LKDe** proofs.
  - Skolemization of **LKDe** proofs.
  - Cut-elimination via the CERES method.

CERES

- Implements algorithms for
  - Validation of **LKDe** proofs.
  - Skolemization of **LKDe** proofs.
  - Cut-elimination via the CERES method.
  - Extraction of Herbrand sequents.

CERES demonstration

- Proof under consideration: the tape proof
- Taken from C. Urban ’00

„On a tape with infinitely many cells, all of which are labelled by either 0 or 1, there are two cells labelled by the same number.“

XML format

- Specified via a DTD.
- Flexible, different calculi possible.
- XML document = collection of proofs in a theory.
- May contain
  - Proofs (of course).
  - Axioms.
  - Lists of sequents.

Conclusion

- 3 tools:
  - HLK for writing proofs.
  - ProofTool for viewing proofs.
  - CERES for performing cut-elimination.
- 1 XML format.
- Implemented in ANSI C++.