STEPHAN SCHMITT

An Efficient Refiner for First-order Intuitionistic Logic

— JProver —
<table>
<thead>
<tr>
<th>Major Modifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>not extensible without</td>
</tr>
<tr>
<td>additional components needed</td>
</tr>
<tr>
<td>proves complicated formulæ as well</td>
</tr>
<tr>
<td>Connection prover (JProver)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Larger Fragments of $I^\perp$</th>
</tr>
</thead>
<tbody>
<tr>
<td>easy extensible to</td>
</tr>
<tr>
<td>obvious, &quot;build-in&quot;</td>
</tr>
<tr>
<td>very restricted</td>
</tr>
<tr>
<td>efficiency: $\times$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tactic-based vs. &quot;connection&quot;-based theorem proving:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fully automated proof search:</td>
</tr>
<tr>
<td>Goal: Increase proof automation during the interactive prove process in NuPRL</td>
</tr>
</tbody>
</table>

**Motivation: Automated Theorem Proving**
Connection to NuPRL-5

Concept: Add JProver as a new refiner to the NuPRL-5 architecture:
ARCHITECTURE OF THE JProver REFINER

J-formula / J-sequent: MetaPRL term, interface between NuPRL and OCaml

Connection prover: Proof search strategy based on the extension procedure

Reconstruction component: Search-free proof reconstruction procedure into LJ or LJ_{mc}
(Schmitt & Kreitz 1995 – 2000)

LJ sequent proof: Sequent proof in Gentzen’s single conclusioned sequent calculus LJ, i.e., the first-order fragment of ITT
Example: $A \wedge B \not\iff A \not\wedge B$.

Non-permutability of sequent rules: Efficient computation of a global substitution $\sigma$.

Connection calculi: Proof search = building chains of connectives (axioms).

Segment calculus: Proof search guided by connectives; successive decomposition.

CONNECTION-BASED THEOREM PROVING
Proof Reconstruction: Example

Basic Idea: Traverse reduction ordering \( \prec \) to construct a sequent proof in \( \Gamma_{mc} \)

Constraints

Acknowledgments

Substitution constraints + Connectives + Connection - Formula tree - \( \prec \)

Diagram:

\[ \frac{\Gamma \vdash A \land \Gamma \vdash B \iff \Gamma \vdash A \land B \rightarrow A, \Gamma \vdash B \land B \equiv A, \Gamma \vdash B, \Gamma \vdash \neg B, \Gamma \vdash \neg A}{\therefore \Gamma \vdash \neg A, \Gamma \vdash \neg B, \Gamma \vdash \neg B, \Gamma \vdash B} \]
Proof Reconstruction II: \( \land \)-split
Refiner: Not yet stand-alone, embedded into MetaPROLOG to use term operations.

- Search-free proof reconstruction relies on complete redundancy detection in \( \mathcal{O}. \)
- Switching between \( \mathcal{L}\text{mc} \) (multiply conclusioned), or \( \mathcal{L} \) (single conclusioned).

Reconstruction component: Constructs sequent proofs in \( \mathcal{L}\text{k} \) (classical logic).

Proof search for propositional logic (intuitionistic or classical).

Input: Formula as MetaPROLOG term.

Realized components:

Demonstration I: Actual State of JProver
(Provability needs interaction with explicit hypotheses numbers)

\[
(\forall A \lor \exists B) \land (\exists A \lor \exists B) \land (\forall A \lor \exists B) \land (\forall A \lor 0B) \land 0A \iff \\
(0B \land (0A \land 0B)) \lor (\exists A \land (\exists A \land 1B) \iff 0B) \lor \\
(\exists B \land (\exists A \land 2B) \iff 1B) \lor (3B \land (\exists A \land 3B) \iff 2B) \lor \forall A \star
\]

Prover does not terminate (solvable picking the right implication on the left)

\[
S \lor S \lor \downarrow \leq \iff ((A \iff T) \lor (T \iff \downarrow)) \iff (\downarrow \iff (A \iff T) \iff) \lor S
\]

Prover can't handle (two interactions, picking \( \downarrow \) instead of \( A \) instead of \( B \))

(Prover (same as prover with stronger chaining and less control) vs. (Prover (straightforward steps, goes into disjunctions as well)

(Prover (this magic as well)

\[
\forall \iff \forall A \land \forall A
\]

Compared Prover to Variants of tactic-based Simple Prover:

DEMONSTRATION II: EXAMPLES
Requires complete redundancy deletion in $\otimes$

Selection strategy:

$\Gamma, \neg \alpha \vdash \beta \iff \neg \alpha \vdash \beta$

Dynamic completion of reduction ordering $\otimes$ during proof reconstruction necessary

Switching between $\text{L}^\text{uc}$ and $\text{L}^\text{uc}$
Suggestions Further Requirements?

- The whole sequent serves as input for JProver.
- User marks relevant (first-order) sequent formulas for JProver.
- Define interface to NuPRL-5; start JProver directly from NuPRL-5.
- Separate JProver from MetaPRL modules; keep only required term operations.

To do:

Extend prover / reconstruction components to quantifiers