Proof Methods

Makarius Wenzel
TU München

August 2009

Structured proof texts

Structured proofs:

from $facts_1$ **have** $props$ **using** $facts_2$
proof $(initial\text{-}method)$
  $body$
qed $(terminal\text{-}method)$

Abbreviations:

by m_1 m_2	$\equiv$	**proof** m_1 **qed** m_2
..	$\equiv$	**by** $rule$ $succeed$
.	$\equiv$	**by** $this$ $succeed$
then	$\equiv$	**from** $this$
with $facts$	$\equiv$	**from** $facts$ **and** $this$

Unstructured proof scripts

Unstructured proofs:

have $props$
  apply $method_1$
  apply $method_2$
  apply $method_3$
  apply $method_4$
  done

ML tactics:

have $props$
  by $(tactic\ my\text{-}tactic)$

Examples

See `Slides1/Ex1.thy`

Structured proof state

Isar proof state:

- proof context: $Proof.context$
- chained facts: $thm\ list$
- primitive goal state: thm

  $\vdash\ subgoals \Longrightarrow main\text{-}goal$

Interactive ML access:

```
Proof.get_goal (Toplevel.proof_of (Isar.state ())) :
  Proof.context * (thm list * thm)

Isar.goal () : thm
```

Simple methods

Common case:

- Facts: inserted into goal state
  (emulating tactical encoding of local facts)
- Goal addressing: either all goals or head goal
- Plain arguments (context, additional theorems)

Note: Isar methods are supposed to make progress
(might require `CHANGED` tactical internally)

See §6.3.5 in `isar-ref` manual

See `Slides1/Ex2.thy`

More method categories

1. structured method with cases, e.g. $induct$
2. structured method: strong emphasis on facts, e.g. $rule$
3. simple method (see above)
4. tactic emulation, e.g. $rule\text{-}tac$
   - naming convention $foo\text{-}tac$
   - numeric goal addressing
   - explicit references to internal goal state (invisible from text!)