The Isar Proof Language in 2016

Makarius Wenzel
http://sketis.net

February 2016

Introduction

Isabelle: framework of domain-specific formal languages

Logic:

Isabelle/Pure: Logical framework and bootstrap environment

Isabelle/HOL: Theories and tools for applications

Programming:

Isabelle/ML: Tool implementation (Poly/ML)

Isabelle/Scala: System integration (JVM)

Proof:

Isabelle/Isar: Intelligible semi-automated reasoning

Document language: LaTeX type-setting of proof text

Document language

Structure markup

- section headings (6 levels like in HTML):
  chapter, **section**, **subsection**, . . . , **subparagraph**
- text blocks: **text**
- implicit lists (cf. Markdown): itemize, enumerate, description
- uncontrolled LaTeX macros

Antiquotations

full form: `@{`$name$ $[options]$ $arguments$ $\ldots$`}`

short form:

1. cartouche argument: `\<^`$name$`>`$\langle argument\rangle$
2. no argument: `\<^`$name$`>`
3. standard name: $\langle argument\rangle$

Isar Proof Language

Structured proofs

Natural Deduction: (Gentzen, 1935)

$$\frac{A \longrightarrow B \quad A}{B} \qquad \frac{\begin{array}{c}[A] \\ \vdots \\ B\end{array}}{A \longrightarrow B}$$

Isabelle/Pure rules: (Paulson 1989)

$$(A \longrightarrow B) \Longrightarrow A \Longrightarrow B \qquad (A \Longrightarrow B) \Longrightarrow A \longrightarrow B$$

Isabelle/Isar proofs: (Wenzel 1999)

assume $A \longrightarrow B$
also have A $\langle proof \rangle$
finally have B **.**

have $A \longrightarrow B$
proof
  assume A
  then show B $\langle proof \rangle$
qed

Structured rule statements (conclusions)

Horn-clause post-fix notation: "Eigen-context"

- premises: **have** B **if** A_1 **and** A_2 $\ldots$
  (default fact name: $that$)
- parameters: **have** B **for** a_1 **and** a_2 $\ldots$

Examples:

- conjunction introduction:
  have $A \wedge B$ **if** A **and** B
- existential introduction:
  have $\exists\, x.\ B\ x$ **if** $B\ a$ **for** a
- disjunction elimination:
  from $\langle A \vee B \rangle$ **have** C **if** $A \Longrightarrow C$ **and** $B \Longrightarrow C$ **for** C
- existential elimination:
  from $\langle \exists\, x.\ B\ x \rangle$ **have** C **if** $\bigwedge x.\ B\ x \Longrightarrow C$ **for** C

Weak premises

- strong premises: **have** B **if** A_1 **and** A_2 $\ldots$
- weak premises: **have** B **when** A_1 **and** A_2 $\ldots$

Example: suffices-to-show

have C
proof $-$
  show C **when** A **and** B
    using *that* **by** $(rule\ \langle A \Longrightarrow B \Longrightarrow C\rangle)$
  show A **sorry**
  show B **sorry**
qed

Note:

- **show** $A_1 \Longrightarrow A_2 \ldots \Longrightarrow B$ becomes free for re-interpretation:
  $\longrightarrow$ strong premises

Structured assumptions

- **assume** B **if** A_1 **and** A_2 $\ldots$ **for** $a_1\ a_2 \ldots$
  - corresponds to $\bigwedge a_1\ a_2 \ldots.\ A_1 \Longrightarrow A_2 \ldots \Longrightarrow B$
  - vacuous quantifiers are omitted

Future potential for this notation:

- locale expressions (**fixes–assumes**)
- long theorem statements (**fixes–assumes–shows**/**obtains**)
- inductive definitions, e.g.

  inductive_set $star :: ('a \times 'a)\ set \Rightarrow ('a \times 'a)\ set\ \ (\_\star\ [100]\ 100)$
  for $R :: ('a \times 'a)\ set$
  where
  $base$: $(x,\ x) \in R\star$ **for** x
  $|$ $step$: $(x,\ z) \in R\star$ **if** $(x,\ y) \in R$ **and** $(y,\ z) \in R\star$ **for** $x\ y\ z$

Elimination statements

$$
\begin{array}{l}
\textbf{consider } \overline{x} \textbf{ where } \overline{A}\ \overline{x} \ \mid\ \overline{y} \textbf{ where } \overline{B}\ \overline{y} \ \mid\ \ldots \ \equiv \\
\quad \textbf{have } \mathit{thesis} \\
\quad\quad \textbf{if } \bigwedge\overline{x}.\ \overline{A}\ \overline{x} \Longrightarrow \mathit{thesis} \\
\quad\quad \textbf{and } \bigwedge\overline{y}.\ \overline{B}\ \overline{y} \Longrightarrow \mathit{thesis} \\
\quad\quad \textbf{for } \mathit{thesis}
\end{array}
$$

Examples:

- existential elimination:
  from $\langle\exists\, x.\ B\ x\rangle$ **consider** x **where** $B\ x$
- conjunction elimination:
  from $\langle A \wedge B\rangle$ **consider** A **and** B
- disjunction elimination:
  from $\langle A \vee B\rangle$ **consider** $A \mid B$

Elimination and cases

- method "$cases$" detects its rule from chained facts
- command "**case**" allows name and attribute specification

Example:

consider $A \mid B \mid C$ $\langle proof \rangle$
then have $something$
proof $cases$
  case $prems$: 1
  show $?thesis$ **using** $prems$ $\langle proof \rangle$
next
  case $prems$: 2
  show $?thesis$ **using** $prems$ $\langle proof \rangle$
next
  case $prems$: 3
  show $?thesis$ **using** $prems$ $\langle proof \rangle$
qed

Obtain

$$\begin{array}{l} \textbf{obtain } \overline{x} \textbf{ where } \overline{A\ \overline{x}} \ \ \langle proof \rangle \equiv \\ \quad \textbf{consider } \overline{x} \textbf{ where } \overline{A\ \overline{x}} \ \ \langle proof \rangle \\ \quad \textbf{fix } \overline{x} \textbf{ assume}^* \ \overline{A\ \overline{x}} \end{array}$$

- meaning is unchanged, but definition simplified
- **is** patterns are supported (with λ-lifting over the parameters)

Block structure of proofs

Nesting levels:

$+$ goal statement (**have**, **show** etc.)
$=$ backwards refinement (**using**, **apply**, **supply** etc.)
$+$ **proof**
$+$ $\{$
$-$ $\}$
$-^2$ **qed**

Some consequences:

- cases in proof methods no longer special (regular context update)
- Eisbach: $match$ method can use generic context for bookkeeping
- PIDE: clarified text folding (indentation still missing)

Structured backwards refinement

$\langle goal \rangle$
subgoal premises $prems$ **for** $x_1\ x_2\ \ldots$
$\langle proof \rangle$

Example: structured proof scripts

$\langle goal \rangle$
subgoal by $method_1$
subgoal by $method_2$
done

$\langle goal \rangle$
subgoal premises $prems$ **for** $x\ y$
using $prems$ $\langle proof_1 \rangle$
subgoal premises $prems$ **for** $u\ v\ w$
using $prems$ $\langle proof_2 \rangle$
done

Eisbach: high-level proof procedures (D. Matichuk et al)

Proof method definitions:

- abstraction over terms and facts:
  method m **for** x y **uses** a b $= method\_body[m,\ x,\ y,\ a,\ b]$
- abstraction over facts, with declaration in the context:
  method m **declares** a $= method\_body[m]$
- abstraction over other methods:
  method m **methods** $m_1\ m_2 = method\_body[m,\ m_1,\ m_2]$

Method *match*:

- goal introspection with pattern matching
- subgoal focus (similar to **subgoal** command)
- control of backtracking

Conclusion

TODO

- proper HTML document output
- interactive document preparation

- re-unify *where*, *of* in Eisbach vs. Pure

- re-unify *atomize* and *atomize_elim* as *compact*
- make *compact* the default of automated methods

- re-unify of *induct* / *induction*, *coinduct* / *coinduction*
- eliminate *induct_tac*, *case_tac*, *rule_tac* eventually

- de-emphasize redundant **hence**, **thus**

Conclusion

The more it advances, . . .

. . . the less it is finished!