Some aspects of Isabelle/Isar

Makarius Wenzel
Univ. Paris-Sud, LRI

May 2011

The Isabelle/Pure framework

Pure syntax and primitive rules (λ-HOL)

$\alpha \Rightarrow \beta$	function type (terms depending on terms)
$\bigwedge :: (\alpha \Rightarrow prop) \Rightarrow prop$	universal quantifier (proofs depending on terms)
$\Longrightarrow :: prop \Rightarrow prop \Rightarrow prop$	implication (proofs depending on proofs)

$$\frac{\begin{array}{c}[x :: \alpha] \\ \vdots \\ b(x) :: \beta\end{array}}{(\lambda x :: \alpha.\ b(x)) :: \alpha \Rightarrow \beta}\,(\Rightarrow I) \qquad \frac{b :: \alpha \Rightarrow \beta \quad a :: \alpha}{(b\ a) :: \beta}\,(\Rightarrow E)$$

$$\frac{\begin{array}{c}[x :: \alpha] \\ \vdots \\ p(x) : B(x)\end{array}}{(\boldsymbol{\lambda} x :: \alpha.\ p(x)) : (\bigwedge x :: \alpha.\ B(x))}\,(\bigwedge I) \qquad \frac{p : (\bigwedge x :: \alpha.\ B(x)) \quad a :: \alpha}{(p\ a) : B(a)}\,(\bigwedge E)$$

$$\frac{\begin{array}{c}[p : A] \\ \vdots \\ q : B\end{array}}{(\boldsymbol{\lambda} p : A.\ q) : (A \Longrightarrow B)}\,(\Longrightarrow I) \qquad \frac{p : A \Longrightarrow B \quad q : A}{(p\ q) : B}\,(\Longrightarrow E)$$

Pure rules (standard version)

Note:

- propositions simply typed: omit types for x and judgement $t :: \tau$
- proofs formally irrelevant: omit proof terms p

$$\frac{\begin{array}{c}[x] \\ \vdots \\ B(x)\end{array}}{\bigwedge x.\ B(x)} \qquad \frac{\bigwedge x.\ B(x)}{B(a)}$$

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

Pure equality

$\equiv :: \alpha \Rightarrow \alpha \Rightarrow prop$

Axioms for $t \equiv u$: α, β, η, refl, subst, ext, iff

Unification: solving equations modulo $\alpha\beta\eta$

- Huet: full higher-order unification (infinitary enumeration!)
- Miller: higher-order patterns (unique result)

Note: no built-in computation!

Representing Natural Deduction rules

Examples:

$$\frac{P \quad Q}{P \wedge Q} \qquad\qquad \bigwedge P\ Q.\ P \Longrightarrow Q \Longrightarrow P \wedge Q$$

$$\frac{\begin{array}{c}[P]\\ \vdots \\ Q\end{array}}{P \longrightarrow Q} \qquad\qquad \bigwedge P\ Q.\ (P \Longrightarrow Q) \Longrightarrow P \longrightarrow Q$$

$$\frac{P\ 0 \quad \begin{array}{c}[n][P\ n]\\ \vdots \\ P\ (Suc\ n)\end{array}}{P\ n} \qquad\qquad \bigwedge P\ n.\ P\ 0 \Longrightarrow (\bigwedge n.\ P\ n \Longrightarrow P\ (Suc\ n)) \Longrightarrow P\ n$$

Representing goals

Protective marker:

$\# :: prop \Rightarrow prop$
$\# \equiv \lambda A :: prop.\ A$

Initialization:

$$\frac{}{C \Longrightarrow \#C}\,(init)$$

General situation: subgoals imply main goal

$$B_1 \Longrightarrow \ldots \Longrightarrow B_n \Longrightarrow \#C$$

Finalization:

$$\frac{\#C}{C}\,(\mathit{finish})$$

Hereditary Harrop Formulas

Define the following sets:

$\boldsymbol{x}$	variables
$\boldsymbol{A}$	atomic formulae (without $\Longrightarrow/\bigwedge$)
$\bigwedge \boldsymbol{x}^*.\ \boldsymbol{A}^* \Longrightarrow \boldsymbol{A}$	Horn Clauses
$\boldsymbol{H} \stackrel{\text{def}}{=} \bigwedge \boldsymbol{x}^*.\ \boldsymbol{H}^* \Longrightarrow \boldsymbol{A}$	Hereditary Harrop Formulas (HHF)

Conventions for results:

- outermost quantification $\bigwedge x.\ B\ x$ is rephrased via schematic variables $B\ ?x$
- equivalence $(A \Longrightarrow (\bigwedge x.\ B\ x)) \equiv (\bigwedge x.\ A \Longrightarrow B\ x)$ produces canonical HHF

Rule composition (back-chaining)

$$\frac{\overline{A} \Longrightarrow B \quad B' \Longrightarrow C \quad B\,\theta = B'\theta}{\overline{A}\,\theta \Longrightarrow C\,\theta}\,(compose)$$

$$\frac{\overline{A} \Longrightarrow B}{(\overline{H} \Longrightarrow \overline{A}) \Longrightarrow (\overline{H} \Longrightarrow B)}\,(\Longrightarrow\text{-}lift)$$

$$\frac{\overline{A}\ \overline{a} \Longrightarrow B\ \overline{a}}{(\bigwedge\overline{x}.\ \overline{A}\ (\overline{a}\ \overline{x})) \Longrightarrow (\bigwedge\overline{x}.\ B\ (\overline{a}\ \overline{x}))}\,(\bigwedge\text{-}lift)$$

General higher-order resolution

$$\frac{\begin{array}{rl} \mathit{rule}\colon & \overline{A}\ \overline{a} \Longrightarrow B\ \overline{a} \\ \mathit{goal}\colon & (\bigwedge \overline{x}.\ \overline{H}\ \overline{x} \Longrightarrow B'\ \overline{x}) \Longrightarrow C \\ \mathit{goal\ unifier}\colon & (\lambda \overline{x}.\ B\ (\overline{a}\ \overline{x}))\,\theta = B'\theta \end{array}}{(\bigwedge \overline{x}.\ \overline{H}\ \overline{x} \Longrightarrow \overline{A}\ (\overline{a}\ \overline{x}))\,\theta \Longrightarrow C\,\theta}\ (\mathit{resolution})$$

$$\frac{\begin{array}{rl} \mathit{goal}\colon & (\bigwedge \overline{x}.\ \overline{H}\ \overline{x} \Longrightarrow A\ \overline{x}) \Longrightarrow C \\ \mathit{assm\ unifier}\colon & A\,\theta = H_i\,\theta\ \ \text{(for some } H_i\text{)} \end{array}}{C\,\theta}\ (\mathit{assumption})$$

Both inferences are omnipresent in Isabelle/Isar:

- $\mathit{resolution}$: e.g. OF attribute, rule method, **also** command
- $\mathit{assumption}$: e.g. $\mathit{assumption}$ method, implicit proof ending

Example: tactic proof

lemma $A \wedge B \longrightarrow B \wedge A$
  apply $(rule\ impI)$
  apply $(rule\ conjI)$
  apply $(rule\ conjunct2)$ — schematic state!
  apply $assumption$
  apply $(rule\ conjunct1)$ — schematic state!
  apply $assumption$
  done

lemma $A \wedge B \longrightarrow B \wedge A$
  apply $(rule\ impI)$
  apply $(rule\ conjI)$
  apply $(erule\ conjunct2)$
  apply $(erule\ conjunct1)$
  done

Notions of proof

Informal proof: mathematical vernacular

[Davey and Priestley: Introduction to Lattices and Order, Cambridge 1990, pages 93–94]

The Knaster-Tarski Fixpoint Theorem. Let L be a complete lattice and $f\colon L \to L$ an order-preserving map. Then $\bigsqcap\{x \in L \mid f(x) \leq x\}$ is a fixpoint of f.

Proof. Let $H = \{x \in L \mid f(x) \leq x\}$ and $a = \bigsqcap H$. For all $x \in H$ we have $a \leq x$, so $f(a) \leq f(x) \leq x$. Thus $f(a)$ is a lower bound of H, whence $f(a) \leq a$. We now use this inequality to prove the reverse one (!) and thereby complete the proof that a is a fixpoint. Since f is order-preserving, $f(f(a)) \leq f(a)$. This says $f(a) \in H$, so $a \leq f(a)$.

Formal proof (1): lambda term

$$
\begin{array}{l}
\mathit{Knaster\_Tarski} \equiv \\
\quad \boldsymbol{\lambda}(H\colon \_)\ \mathit{Ha}\colon \_. \\
\quad\ \ \mathit{order\_trans\_rules\_24} \cdot \_ \cdot \_ \bullet (\mathit{thm} \bullet H) \bullet \\
\quad\ \ \ (\mathit{complete\_lattice\_class.Inf\_greatest} \cdot \_ \cdot \_ \bullet H \bullet \\
\quad\ \ \ \ \ (\boldsymbol{\lambda} x\ \mathit{Hb}\colon \_. \\
\quad\ \ \ \ \ \ \ \mathit{order\_trans\_rules\_7} \cdot \bigsqcap\{x.\ ?f\,x \le x\} \cdot x \cdot ?f \cdot \_ \bullet (\mathit{thm} \bullet H) \bullet (\mathit{thm} \bullet H) \bullet \\
\quad\ \ \ \ \ \ \ \ (\mathit{complete\_lattice\_class.Inf\_lower} \cdot \_ \cdot \_ \bullet H \bullet \mathit{Hb}) \bullet \\
\quad\ \ \ \ \ \ \ \ (\mathit{iffD1} \cdot \_ \cdot \_ \bullet (\mathit{mem\_Collect\_eq} \cdot x \cdot (\lambda x.\ ?f\,x \le x) \bullet (\mathit{thm} \bullet H)) \bullet \mathit{Hb}) \bullet \\
\quad\ \ \ \ \ \ \ \ \mathit{Ha})) \bullet \\
\quad\ \ \ (\mathit{complete\_lattice\_class.Inf\_lower} \cdot \_ \cdot \_ \bullet H \bullet \\
\quad\ \ \ \ \ (\mathit{iffD2} \cdot \_ \cdot \_ \bullet (\mathit{mem\_Collect\_eq} \cdot ?f\ (\bigsqcap\{x.\ ?f\,x \le x\}) \cdot (\lambda a.\ ?f\,a \le a) \bullet (\mathit{thm} \bullet H)) \bullet \\
\quad\ \ \ \ \ \ \ (\mathit{Ha} \cdot ?f\ (\bigsqcap\{x.\ ?f\,x \le x\}) \cdot \bigsqcap\{x.\ ?f\,x \le x\} \bullet \\
\quad\ \ \ \ \ \ \ \ \ (\mathit{complete\_lattice\_class.Inf\_greatest} \cdot \_ \cdot \_ \bullet H \bullet \\
\quad\ \ \ \ \ \ \ \ \ \ \ (\boldsymbol{\lambda} x\ \mathit{Hb}\colon \_. \\
\quad\ \ \ \ \ \ \ \ \ \ \ \ \ \mathit{order\_trans\_rules\_7} \cdot \bigsqcap\{x.\ ?f\,x \le x\} \cdot x \cdot ?f \cdot \_ \bullet (\mathit{thm} \bullet H) \bullet (\mathit{thm} \bullet H) \bullet \\
\quad\ \ \ \ \ \ \ \ \ \ \ \ \ \ (\mathit{complete\_lattice\_class.Inf\_lower} \cdot \_ \cdot \_ \bullet H \bullet \mathit{Hb}) \bullet \\
\quad\ \ \ \ \ \ \ \ \ \ \ \ \ \ (\mathit{iffD1} \cdot \_ \cdot \_ \bullet (\mathit{mem\_Collect\_eq} \cdot x \cdot (\lambda x.\ ?f\,x \le x) \bullet (\mathit{thm} \bullet H)) \bullet \mathit{Hb}) \bullet \\
\quad\ \ \ \ \ \ \ \ \ \ \ \ \ \ \mathit{Ha})))))
\end{array}
$$

Formal proof (2): Isar text

theorem $Knaster\_Tarski$:
  fixes $f :: {}'a{::}complete\_lattice \Rightarrow {}'a$
  assumes $mono$: $\bigwedge x\ y.\ x \le y \Longrightarrow f\ x \le f\ y$
  shows $f\ (\bigsqcap \{x.\ f\ x \le x\}) = \bigsqcap \{x.\ f\ x \le x\}$ (**is** $f\ ?a = ?a$)
proof $-$
  have $f\ ?a \le ?a$ (**is** $\_ \le \bigsqcap ?H$)
  proof $(rule\ Inf\_greatest)$
    fix x **assume** $x \in ?H$
    then have $?a \le x$ **by** $(rule\ Inf\_lower)$
    also from $\langle x \in ?H \rangle$ **have** $f\ \ldots \le x$ **..**
    moreover note $mono$ **finally show** $f\ ?a \le x$ **.**
  qed
  also have $?a \le f\ ?a$
  proof $(rule\ Inf\_lower)$
    from $mono$ **and** $\langle f\ ?a \le ?a \rangle$ **have** $f\ (f\ ?a) \le f\ ?a$ **.**
    then show $f\ ?a \in ?H$ **..**
  qed
  finally show $f\ ?a = ?a$ **.**
qed

Isar language characteristics

Isar: "Intelligible semi-automated reasoning"

- interpreted language of "proof expressions"
  - proof context
  - flow of facts towards goals
  - simple reduction to Isabelle/Pure logic
- language framework
  - highly structured
  - highly extensible: derived commands, proof methods
  - non-computational: language for proofs, not proof procedures

Example proofs patterns: induction and calculation

```
theorem fixes n :: nat shows P n
proof (induct n)
  show P 0  ⟨proof⟩
next
  fix n assume P n
  show P (Suc n)  ⟨proof⟩
qed

notepad
begin
  have a = b  ⟨proof⟩
  also have ... = c  ⟨proof⟩
  also have ... = d  ⟨proof⟩
  finally have a = d .
end
```

Example proof: induction $\times$ calculation

theorem
  fixes $n :: nat$
  shows $(\sum i{=}0..n.\ i) = n * (n + 1)\ div\ 2$
proof $(induct\ n)$
  case 0
  have $(\sum i{=}0..0.\ i) = (0{::}nat)$ **by** $simp$
  also have $\ldots = 0 * (0 + 1)\ div\ 2$ **by** $simp$
  finally show $?case$ **.**
next
  case $(Suc\ n)$
  have $(\sum i{=}0..Suc\ n.\ i) = (\sum i{=}0..n.\ i) + (n + 1)$ **by** $simp$
  also have $\ldots = n * (n + 1)\ div\ 2 + (n + 1)$ **by** $(simp\ add{:}\ Suc.hyps)$
  also have $\ldots = (n * (n + 1) + 2 * (n + 1))\ div\ 2$ **by** $simp$
  also have $\ldots = (Suc\ n * (Suc\ n + 1))\ div\ 2$ **by** $simp$
  finally show $?case$ **.**
qed

The Isar proof language

Notepad for logical entities

notepad
begin

Terms:

let $?f = \lambda x.\ x$ — term binding (abbreviation)
let $\_ + ?b = ?f\ a + b$ — pattern matching
let $?g = ?f\ ?f$ — Hindler-Milner polymorphism

Facts:

note $rules = sym\ refl\ trans$ — collective facts
note $a = rules(2)$ — selection
note $b = this$ — implicit result *this*

end

Logical contexts

Main judgment:

$$\boxed{\Gamma \vdash_\Theta \varphi}$$

- φ: conclusion (rule statement using $\bigwedge/\Longrightarrow$)
- Θ: global theory context

type $\forall\,\overline{\alpha}.\ (\overline{\alpha})c$	polymorphic type constructor
const $c :: \forall\,\overline{\alpha}.\ \tau[\overline{\alpha}]$	polymorphic term constant
axiom a: $\forall\,\overline{\alpha}.\ A[\overline{\alpha}]$	polymorphic proof constant

- Γ: local proof context

type α	fixed type variable
fix $x :: \tau[\alpha]$	fixed term variable
assume a: $A[\alpha,\ x]$	fixed proof variable

Proof context elements (forward reasoning)

notepad
begin
  {
    fix x
    have $B\ x$ $\langle proof \rangle$
  }
  have $\bigwedge x.\ B\ x$ **by** $fact$
end

notepad
begin
  {
    assume A
    have B $\langle proof \rangle$
  }
  have $A \Longrightarrow B$ **by** $fact$
end

Local claims and proofs

Main idea: Pure rules turned into proof schemes

from $facts_1$ **have** $props$ **using** $facts_2$
proof $(initial\_method)$
  $body$
qed $(terminal\_method)$

Solving sub-problems: within $body$

fix $vars$
assume $props$
show $props$ $\langle proof \rangle$

Canonical backwards reasoning

notepad
begin
  have $A \longrightarrow B$
  proof $(rule\ impI)$
    assume A
    show B $\langle proof \rangle$
  qed
end

notepad
begin
  have $\forall x.\ B\ x$
  proof $(rule\ allI)$
    fix x
    show $B\ x$ $\langle proof \rangle$
  qed
end

Note: standard rules can be used implicitly —
by omitting "$(rule\ impI)$" and "$(rule\ allI)$" above.

Example: basic natural deduction

```
notepad
begin
  have A ∧ B ⟶ B ∧ A
  proof
    assume ab: A ∧ B
    show B ∧ A
    proof
      show B using ab by rule
      show A using ab by rule
    qed
  qed
end
```

Atomic proofs

Single-step proofs:

by $\mathit{rule} \quad \equiv \quad$ **..**
by $\mathit{this} \quad \equiv \quad$ **.**

Automated proofs:

by auto
by simp
by blast
by force

Omitted proofs:

sorry $\quad \equiv \quad$ **by** $\mathit{cheating}$

Analyzing atomic proofs

General atomic proof:

by $(initial\_method)$ $(terminal\_method)$

Structured expansion:

proof $(initial\_method)$ **qed** $(terminal\_method)$

Tactical transformation:

apply $(initial\_method)$
apply $(terminal\_method)$
apply $(assumption+)?$
done

Derived proof patterns

Calculational reasoning

$$
\begin{array}{rcl}
\textbf{also}_0 & = & \textbf{note } \mathit{calculation} = \mathit{this} \\
\textbf{also}_{n+1} & = & \textbf{note } \mathit{calculation} = \mathit{trans}\ [\mathit{OF}\ \mathit{calculation}\ \mathit{this}] \\
\textbf{finally} & = & \textbf{also from } \mathit{calculation} \\
\textbf{moreover} & = & \textbf{note } \mathit{calculation} = \mathit{calculation}\ \mathit{this} \\
\textbf{ultimately} & = & \textbf{moreover from } \mathit{calculation}
\end{array}
$$

Example:

notepad
begin
  have $a = b$ $\langle \mathit{proof} \rangle$
  also have $\ldots = c$ $\langle \mathit{proof} \rangle$
  also have $\ldots = d$ $\langle \mathit{proof} \rangle$
  finally have $a = d$ **.**
end

notepad
begin
  have A $\langle \mathit{proof} \rangle$
  moreover have B $\langle \mathit{proof} \rangle$
  moreover have C $\langle \mathit{proof} \rangle$
  ultimately have A **and** B **and** C **.**
end

Note: term "..." abbreviates the argument of the last statement

Induction

using $facts$
proof ($\textit{induct}$ $insts$ *arbitrary:* $vars$ *rule:* $fact$)

Example:

notepad
begin
  fix $n :: nat$ **and** $x :: {}'a$ **have** $P\ n\ x$
  proof $(induct\ n\ arbitrary\!: x)$
    case 0
    show $P\ 0\ x\ \ \langle proof \rangle$
  next
    case $(Suc\ n)$
    from $\langle P\ n\ a\rangle$ **show** $P\ (Suc\ n)\ x\ \ \langle proof \rangle$
  qed
end

Generalized elimination

obtain $\overline{x}$ **where** $\overline{B}\ \overline{x}$ $\langle proof \rangle$ $=$

have $reduction$: $\bigwedge thesis.\ (\bigwedge \overline{x}.\ \overline{B}\ \overline{x} \Longrightarrow thesis) \Longrightarrow thesis$ $\langle proof \rangle$
fix $\overline{x}$ **assume** $\ll eliminate\ reduction \gg$ $\overline{B}\ \overline{x}$

$$\frac{\begin{array}{l}\Gamma \vdash \bigwedge thesis.\ (\bigwedge \overline{x}.\ \overline{B}\ \overline{x} \Longrightarrow thesis) \Longrightarrow thesis \\ \Gamma \cup \overline{B}\ \overline{x} \vdash C\end{array}}{\Gamma \vdash C}\ (eliminate)$$

Example:

notepad
begin
  assume $\exists\, x.\ B\ x$
  then obtain x **where** $B\ x$ **..**
end

notepad
begin
  assume $A \wedge B$
  then obtain A **and** B **..**
end